Subject: [RFC] Virtualization steps
Posted by dev on Fri, 24 Mar 2006 17:19:59 GMT
View Forum Message <> Reply to Message

Eric, Herbert,

I think it is quite clear, that without some agreement on all these virtualization issues, we won't be able to commit anything good to mainstream. My idea is to gather our efforts to get consensus on most clean parts of code first and commit them one by one.

The proposal is quite simple. We have 4 parties in this conversation (maybe more?): IBM guys, OpenVZ, VServer and Eric Biederman. We discuss the areas which should be considered step by step. Send patches for each area, discuss, come to some agreement and all 4 parties Sign-Off the patch. After that it goes to Andrew/Linus. Worth trying?

So far, (correct me if I'm wrong) we concluded that some people don't want containers as a whole, but want some subsystem namespaces. I suppose for people who care about containers only it doesn't matter, so we can proceed with namespaces, yeah?

So the most easy namespaces to discuss I see:
- utsname
- sys IPC
- network virtualization
- netfilter virtualization

all these were discussed already somehow and looks like there is no fundamental differencies in our approaches (at least OpenVZ and Eric, for sure).

Right now, I suggest to concentrate on first 2 namespaces - utsname and sysvipc. They are small enough and easy. Lets consider them without sysctl/proc issues, as those can be resolved later. I sent the patches for these 2 namespaces to all of you. I really hope for some _good_ critics, so we could work it out quickly.

Thanks,
Kirill

Subject: Re: [RFC] Virtualization steps
Posted by Nick Piggin on Fri, 24 Mar 2006 17:33:01 GMT
View Forum Message <> Reply to Message

Kirill Korotaev wrote:
> Eric, Herbert,

>
> I think it is quite clear, that without some agreement on all these
> virtualization issues, we won't be able to commit anything good to
> mainstream. My idea is to gather our efforts to get consensus on most
> clean parts of code first and commit them one by one.
>
> The proposal is quite simple. We have 4 parties in this conversation
> (maybe more?): IBM guys, OpenVZ, VServer and Eric Biederman. We discuss
> the areas which should be considered step by step. Send patches for each
> area, discuss, come to some agreement and all 4 parties Sign-Off the
> patch. After that it goes to Andrew/Linus. Worth trying?

Oh, after you come to an agreement and start posting patches, can you
also outline why we want this in the kernel (what it does that low
level virtualization doesn't, etc, etc), and how and why you've agreed
to implement it. Basically, some background and a summary of your
discussions for those who can't follow everything. Or is that a faq
item?

Thanks,
Nick

--
SUSE Labs, Novell Inc.
Send instant messages to your online friends http://au.messenger.yahoo.com

---

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Fri, 24 Mar 2006 18:36:18 GMT
View Forum Message <> Reply to Message

Kirill Korotaev <dev@sw.ru> writes:

> Eric, Herbert,
>
> I think it is quite clear, that without some agreement on all these
> virtualization issues, we won't be able to commit anything good to
> mainstream. My idea is to gather our efforts to get consensus on most clean
> parts of code first and commit them one by one.
>
> The proposal is quite simple. We have 4 parties in this conversation (maybe
> more?): IBM guys, OpenVZ, VServer and Eric Biederman. We discuss the areas which
> should be considered step by step. Send patches for each area, discuss, come to
> some agreement and all 4 parties Sign-Off the patch. After that it goes to
> Andrew/Linus. Worth trying?

Yes, this sounds like a path forward that has a reasonable chance of
making progress.

> So far, (correct me if I'm wrong) we concluded that some people don't want
> containers as a whole, but want some subsystem namespaces. I suppose for people
> who care about containers only it doesn't matter, so we can proceed with
> namespaces, yeah?

Yes, I think at one point I have seen all of the major parties receptive
to the concept.

> So the most easy namespaces to discuss I see:
> - utsname
> - sys IPC
> - network virtualization
> - netfilter virtualization

The networking is hard simply because the is so very much of it, and it
is being active developed :)

> all these were discussed already somehow and looks like there is no fundamental
> differencies in our approaches (at least OpenVZ and Eric, for sure).

Yes. I think we agree on what the semantics should be for these parts.
Which should avoid the problem with have with the pid namespace.

> Right now, I suggest to concentrate on first 2 namespaces - utsname and
> sysvipc. They are small enough and easy. Lets consider them without sysctl/proc
> issues, as those can be resolved later. I sent the patches for these 2
> namespaces to all of you. I really hope for some _good_ critics, so we could
> work it out quickly.

Sounds like a plan.

Eric

Subject: Re: [RFC] Virtualization steps
Posted by Dave Hansen on Fri, 24 Mar 2006 19:25:39 GMT
View Forum Message <> Reply to Message

On Sat, 2006-03-25 at 04:33 +1100, Nick Piggin wrote:
> Oh, after you come to an agreement and start posting patches, can you
> also outline why we want this in the kernel (what it does that low
> level virtualization doesn't, etc, etc)

Can you wait for an OLS paper? ;)

I'll summarize it this way: low-level virtualization uses resource
inefficiently.

With this higher-level stuff, you get to share all of the Linux caching,
and can do things like sharing libraries pretty naturally.

They are also much lighter-weight to create and destroy than full
virtual machines. We were planning on doing some performance
comparisons versus some hypervisors like Xen and the ppc64 one to show
scaling with the number of virtualized instances. Creating 100 of these
Linux containers is as easy as a couple of shell scripts, but we still
can't find anybody crazy enough to go create 100 Xen VMs.

Anyway, those are the things that came to my mind first. I'm sure the
others involved have their own motivations.

-- Dave

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Fri, 24 Mar 2006 19:53:55 GMT
View Forum Message <> Reply to Message

Dave Hansen <haveblue@us.ibm.com> writes:

> On Sat, 2006-03-25 at 04:33 +1100, Nick Piggin wrote:
>> Oh, after you come to an agreement and start posting patches, can you
>> also outline why we want this in the kernel (what it does that low
>> level virtualization doesn't, etc, etc)
>
> Can you wait for an OLS paper? ;)
>
> I'll summarize it this way: low-level virtualization uses resource
> inefficiently.
>
> With this higher-level stuff, you get to share all of the Linux caching,
> and can do things like sharing libraries pretty naturally.

Also it is a major enabler for things such as process migration,
between kernels.

> They are also much lighter-weight to create and destroy than full
> virtual machines. We were planning on doing some performance
> comparisons versus some hypervisors like Xen and the ppc64 one to show
> scaling with the number of virtualized instances. Creating 100 of these
> Linux containers is as easy as a couple of shell scripts, but we still
> can't find anybody crazy enough to go create 100 Xen VMs.

One of my favorite test cases is to kill about 100 of them
simultaneously :)

I think on a reasonably beefy dual processor machine I should be able
to get about 1000 of them running all at once.

> Anyway, those are the things that came to my mind first. I'm sure the
> others involved have their own motivations.

The practical aspect is that several groups have found the arguments
compelling enough that they have already done complete
implementations. At which point getting us all to agree on a common
implementation is important. :)

Eric

Subject: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Fri, 24 Mar 2006 21:19:17 GMT
View Forum Message <> Reply to Message

On Fri, Mar 24, 2006 at 08:19:59PM +0300, Kirill Korotaev wrote:
> Eric, Herbert,
>
> I think it is quite clear, that without some agreement on all these
> virtualization issues, we won't be able to commit anything good to
> mainstream. My idea is to gather our efforts to get consensus on most
> clean parts of code first and commit them one by one.
>
> The proposal is quite simple. We have 4 parties in this conversation
> (maybe more?): IBM guys, OpenVZ, VServer and Eric Biederman. We
> discuss the areas which should be considered step by step. Send
> patches for each area, discuss, come to some agreement and all 4
> parties Sign-Off the patch. After that it goes to Andrew/Linus.
> Worth trying?

sounds good to me, as long as we do not consider
the patches 'final' atm .. because I think we should
try to test them with _all_ currently existing solutions
first ... we do not need to bother Andrew with stuff
which doesn't work for the existing and future 'users'.

so IMHO, we should make a kernel branch (Eric or Sam
are probably willing to maintain that), which we keep
in-sync with mainline (not necessarily git, but at
least snapshot wise), where we put all the patches
we agree on, and each party should then adjust the
existing solution to this kernel, so we get some deep
testing in the process, and everybody can see if it
'works' for him or not ...

things where we agree that it 'just works' for everyone
can always be handed upstream, and would probably make
perfect patches for Andrew ...

> So far, (correct me if I'm wrong) we concluded that some people don't
> want containers as a whole, but want some subsystem namespaces. I
> suppose for people who care about containers only it doesn't matter, so
> we can proceed with namespaces, yeah?

yes, the emphasis here should be on lightweight and
modular, so that those folks interested in full featured
containers can just 'assemble' the pieces, while those
desiring service/space isolation pick their subsystems
one by one ...

> So the most easy namespaces to discuss I see:
> - utsname

yes, that's definitely one we can start with, as it seems
that we already have _very_ similar implementations

> - sys IPC

this is something which is also related to limits and
should get special attention with resource sharing,
isolation and control in mind

> - network virtualization

here I see many issues, as for example Linux-VServer
does not necessarily aim for full virtualization, when
simple and performant isolation is sufficient.

don't get me wrong, we are _not_ against network
virtualization per se, but we isolation is just so
much simpler to administrate and often much more
performant, so that it is very interesting for service
separation as well as security applications

just consider the 'typical' service isolation aspect
where you want to have two apaches, separated on two
IPs, but communicating with a single sql database

> - netfilter virtualization

same as for network virtualization, but not really
an issue if it can be 'disabled'

of course, the ideal solution would be some kind
of hybrid, where you can have virtual interfaces as
well as isolated IPs, side-by-side ...

> all these were discussed already somehow and looks like there is no
> fundamental differencies in our approaches (at least OpenVZ and Eric,
> for sure).
>
> Right now, I suggest to concentrate on first 2 namespaces - utsname
> and sysvipc. They are small enough and easy. Lets consider them
> without sysctl/proc issues, as those can be resolved later. I sent the
> patches for these 2 namespaces to all of you. I really hope for some
> _good_ critics, so we could work it out quickly.

will look into them soon ...

best,
Herbert

> Thanks,
> Kirill

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Mon, 27 Mar 2006 18:45:49 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> On Fri, Mar 24, 2006 at 08:19:59PM +0300, Kirill Korotaev wrote:
>> Eric, Herbert,
>>
>> I think it is quite clear, that without some agreement on all these
>> virtualization issues, we won't be able to commit anything good to
>> mainstream. My idea is to gather our efforts to get consensus on most
>> clean parts of code first and commit them one by one.
>>
>> The proposal is quite simple. We have 4 parties in this conversation
>> (maybe more?): IBM guys, OpenVZ, VServer and Eric Biederman. We
>> discuss the areas which should be considered step by step. Send
>> patches for each area, discuss, come to some agreement and all 4
>> parties Sign-Off the patch. After that it goes to Andrew/Linus.
>> Worth trying?
>
> sounds good to me, as long as we do not consider
> the patches 'final' atm .. because I think we should
> try to test them with _all_ currently existing solutions

> first ... we do not need to bother Andrew with stuff
> which doesn't work for the existing and future 'users'.
>
> so IMHO, we should make a kernel branch (Eric or Sam
> are probably willing to maintain that), which we keep
> in-sync with mainline (not necessarily git, but at
> least snapshot wise), where we put all the patches
> we agree on, and each party should then adjust the
> existing solution to this kernel, so we get some deep
> testing in the process, and everybody can see if it
> 'works' for him or not ...

ACK. A collection of patches that we can all agree
on sounds like something worth aiming for.

It looks like Kirill last round of patches can form
a nucleus for that. So far I have seem plenty of technical
objects but no objections to the general direction.

So agreement appears possible.

Eric

Subject: Re: [RFC] Virtualization steps
Posted by Bill Davidsen on Tue, 28 Mar 2006 04:28:12 GMT
View Forum Message <> Reply to Message

Dave Hansen wrote:
> On Sat, 2006-03-25 at 04:33 +1100, Nick Piggin wrote:
>> Oh, after you come to an agreement and start posting patches, can you
>> also outline why we want this in the kernel (what it does that low
>> level virtualization doesn't, etc, etc)
>
> Can you wait for an OLS paper? ;)
>
> I'll summarize it this way: low-level virtualization uses resource
> inefficiently.
>
> With this higher-level stuff, you get to share all of the Linux caching,
> and can do things like sharing libraries pretty naturally.
>
> They are also much lighter-weight to create and destroy than full
> virtual machines. We were planning on doing some performance
> comparisons versus some hypervisors like Xen and the ppc64 one to show
> scaling with the number of virtualized instances. Creating 100 of these
> Linux containers is as easy as a couple of shell scripts, but we still
> can't find anybody crazy enough to go create 100 Xen VMs.

But these require a modified O/S, do they not? Or do I read that
incorrectly? Is this going to be real virtualization able to run any O/S?

Frankly I don't see running 100 VMs as a realistic goal, being able to
run Linux, Windows, Solaris and BEOS unmodified in 4-5 VMs would be far
more useful.
>
> Anyway, those are the things that came to my mind first. I'm sure the
> others involved have their own motivations.
>
> -- Dave
>

Subject: Re: Re: [RFC] Virtualization steps
Posted by kir on Tue, 28 Mar 2006 06:45:10 GMT
View Forum Message <> Reply to Message

Bill Davidsen wrote:

> Dave Hansen wrote:
>
>> On Sat, 2006-03-25 at 04:33 +1100, Nick Piggin wrote:
>>
>>> Oh, after you come to an agreement and start posting patches, can you
>>> also outline why we want this in the kernel (what it does that low
>>> level virtualization doesn't, etc, etc)
>>
>>
>> Can you wait for an OLS paper? ;)
>>
>> I'll summarize it this way: low-level virtualization uses resource
>> inefficiently.
>>
>> With this higher-level stuff, you get to share all of the Linux caching,
>> and can do things like sharing libraries pretty naturally.
>>
>> They are also much lighter-weight to create and destroy than full
>> virtual machines. We were planning on doing some performance
>> comparisons versus some hypervisors like Xen and the ppc64 one to show
>> scaling with the number of virtualized instances. Creating 100 of these
>> Linux containers is as easy as a couple of shell scripts, but we still
>> can't find anybody crazy enough to go create 100 Xen VMs.
>
>
> But these require a modified O/S, do they not? Or do I read that
> incorrectly? Is this going to be real virtualization able to run any O/S?

This type is called OS-level virtualization, or kernel-level
virtualization, or partitioning. Basically it allows to create a
compartments (in OpenVZ we call them VEs -- Virtual Environments) in
which you can run full *unmodified* Linux system (but the kernel itself
-- it is one single kernel common for all compartments). That means that
with this approach you can not run OSs other than Linux, but different
Linux distributions are working just fine.

> Frankly I don't see running 100 VMs as a realistic goal

It is actually not a future goal, but rather a reality. Since os-level
virtualization overhead is very low (1-2 per cent or so), one can run
hundreds of VEs.

Say, on a box with 1GB of RAM OpenVZ [http://openvz.org/] is able to run
about 150 VEs each one having init, apache (serving static content),
sendmail, sshd, cron etc. running. Actually you can run more, but with
the aggressive swapping so performance drops considerably. So it all
mostly depends on RAM, and I'd say that 500+ VEs on a 4GB box should run
just fine. Of course it all depends on what you run inside those VEs.

> , being able to run Linux, Windows, Solaris and BEOS unmodified in 4-5
> VMs would be far more useful.

This is a different story. If you want to run different OSs on the same
box -- use emulation or paravirtualization.

If you are happy to stick to Linux on this box -- use OS-level
virtualization. Aside from the best possible scalability and
performance, the other benefit of this approach is dynamic resource
management -- since there is a single kernel managing all the resources
such as RAM, you can easily tune all those resources runtime. More to
say, you can make one VE use more RAM while nobody else it using it,
leading to much better resource usage. And since there is one single
kernel that manages everything, you could do nice tricks like VE
checkpointing, live migration, etc. etc.

Some more info on topic are available from
http://openvz.org/documentation/tech/

Kir.

>>
>> Anyway, those are the things that came to my mind first. I'm sure the
>> others involved have their own motivations.
>>
>> -- Dave

>>
>

Subject: Re: [RFC] Virtualization steps
Posted by dev on Tue, 28 Mar 2006 08:51:14 GMT
View Forum Message <> Reply to Message

>> so IMHO, we should make a kernel branch (Eric or Sam
>> are probably willing to maintain that), which we keep
>> in-sync with mainline (not necessarily git, but at
>> least snapshot wise), where we put all the patches
>> we agree on, and each party should then adjust the
>> existing solution to this kernel, so we get some deep
>> testing in the process, and everybody can see if it
>> 'works' for him or not ...
>
> ACK. A collection of patches that we can all agree
> on sounds like something worth aiming for.
>
> It looks like Kirill last round of patches can form
> a nucleus for that. So far I have seem plenty of technical
> objects but no objections to the general direction.
yup, I will fix everything and will come with a set of patches for IPC,
so we could select which way is better to do it :)

> So agreement appears possible.
Nice to hear this!

Eric, we have a GIT repo on openvz.org already:
http://git.openvz.org

we will create a separate branch also called -acked, where patches
agreed upon will go.

Thanks,
Kirill

Subject: Re: [RFC] Virtualization steps
Posted by dev on Tue, 28 Mar 2006 09:00:27 GMT
View Forum Message <> Reply to Message

> Frankly I don't see running 100 VMs as a realistic goal, being able to
> run Linux, Windows, Solaris and BEOS unmodified in 4-5 VMs would be far
> more useful.
It is more than realistic. Hosting companies run more than 100 VPSs in

reality. There are also other usefull scenarios. For example, I know the
universities which run VPS for every faculty web site, for every
department, mail server and so on. Why do you think they want to run
only 5VMs on one machine? Much more!

Thanks,
Kirill

Subject: Re: [RFC] Virtualization steps
Posted by dev on Tue, 28 Mar 2006 09:02:08 GMT
View Forum Message <> Reply to Message

> Oh, after you come to an agreement and start posting patches, can you
> also outline why we want this in the kernel (what it does that low
> level virtualization doesn't, etc, etc), and how and why you've agreed
> to implement it. Basically, some background and a summary of your
> discussions for those who can't follow everything. Or is that a faq
> item?
Nick, will be glad to shed some light on it.

First of all, what it does which low level virtualization can't:
- it allows to run 100 containers on 1GB RAM
  (it is called containers, VE - Virtual Environments,
   VPS - Virtual Private Servers).
- it has no much overhead (<1-2%), which is unavoidable with hardware
  virtualization. For example, Xen has >20% overhead on disk I/O.
- it allows to create/deploy VE in less than a minute, VE start/stop
  takes ~1-2 seconds.
- it allows to dynamically change all resource limits/configurations.
  In OpenVZ it is even possible to add/remove virtual CPUs to/from VE.
  It is possible to increase/descrease memory limits on the fly etc.
- it has much more efficient memory usage with single template file
  in a cache if COW-like filesystem is used for VE templates.
- it allows you to access VE files from host easily if needed.
  This helps to make management much more flexible, e.g. you can
  upgrade/repair/fix all you VEs from host, i.e. easy mass management.

OS kernel virtualization
~~~~~~~~~~~~~~~~~~~~~~~~
OS virtualization is a kernel solution, which replaces the usage
of many global variables with context-dependant counterparts. This
allows to have isolated private resources in different contexts.

So VE means essentially context and a set of it's variables/settings,
which include but not limited to, own process tree, files, IPC
resources, IP routing, network devices and such.
~~~~~~~~~~~~~~~~~~~~~~~~

Full virtualization solution consists of:
- virtualization of resources, i.e. private contexts
- resource controls, for limiting contexts
- management tools

Such kind of virtualization solution is implemented in OpenVZ
(http://openvz.org) and Linux-Vserver (http://linux-vserver.org) projects.

Summary of previous discussions on LKML
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
- we agreed upon doing virtualization of each kernel subsystem
  separately, not as a single virtual environment.
- we almost agreed upon calling virtualization of subsystems
  "namespaces".
- we were discussing whether we should have global namespace context,
  like 'current' or bypass context as an argument to all functions
  which require it.
- we didn't agreed on whether we need a config option and ability to
  compile kernel w/o virtual namespaces.

Thansk,
Kirill

Subject: Re: [RFC] Virtualization steps
Posted by Nick Piggin on Tue, 28 Mar 2006 09:15:17 GMT
View Forum Message <> Reply to Message

Kirill Korotaev wrote:
>
> Nick, will be glad to shed some light on it.
>

Thanks very much Kirill.

I don't think I'm qualified to make any decisions about this,
so I don't want to detract from the real discussions, but I
just had a couple more questions:

> First of all, what it does which low level virtualization can't:
> - it allows to run 100 containers on 1GB RAM
>   (it is called containers, VE - Virtual Environments,
>   VPS - Virtual Private Servers).
> - it has no much overhead (<1-2%), which is unavoidable with hardware
>   virtualization. For example, Xen has >20% overhead on disk I/O.

Are any future hardware solutions likely to improve these problems?
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

>
> OS kernel virtualization
> ~~~~~~~~~~~~~~~~~~~~~~~~

Is this considered secure enough that multiple untrusted VEs are run
on production systems?

What kind of users want this, who can't use alternatives like real
VMs?

> Summary of previous discussions on LKML
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Have their been any discussions between the groups pushing this
virtualization, and important kernel developers who are not part of
a virtualization effort? Ie. is there any consensus about the
future of these patches?

Thanks,
Nick

--
SUSE Labs, Novell Inc.
Send instant messages to your online friends http://au.messenger.yahoo.com

---

Subject: Re: [RFC] Virtualization steps
Posted by serue on Tue, 28 Mar 2006 12:53:42 GMT
View Forum Message <> Reply to Message

Quoting Kirill Korotaev (dev@sw.ru):
> >>so IMHO, we should make a kernel branch (Eric or Sam
> >>are probably willing to maintain that), which we keep
> >>in-sync with mainline (not necessarily git, but at
> >>least snapshot wise), where we put all the patches
> >>we agree on, and each party should then adjust the
> >>existing solution to this kernel, so we get some deep
> >>testing in the process, and everybody can see if it
> >>'works' for him or not ...
> >
> >ACK. A collection of patches that we can all agree
> >on sounds like something worth aiming for.
> >
> >It looks like Kirill last round of patches can form
> >a nucleus for that. So far I have seem plenty of technical
> >objects but no objections to the general direction.
> yup, I will fix everything and will come with a set of patches for IPC,

> so we could select which way is better to do it :)
>
> >So agreement appears possible.
> Nice to hear this!
>
> Eric, we have a GIT repo on openvz.org already:
> http://git.openvz.org
>
> we will create a separate branch also called -acked, where patches
> agreed upon will go.

That's ok by me. If a more neutral name/site were preferred, we could use the sf.net set we had finally gotten around to setting up - www.sf.net/projects/lxc (LinuX Containers). Unfortunately that would likely be just a quilt patch repository.

A wiki + git repository would be ideal.

-serge

Subject: Re: [RFC] Virtualization steps
Posted by Bill Davidsen on Tue, 28 Mar 2006 14:38:19 GMT
View Forum Message <> Reply to Message

Kirill Korotaev wrote:

>> Frankly I don't see running 100 VMs as a realistic goal, being able
>> to run Linux, Windows, Solaris and BEOS unmodified in 4-5 VMs would
>> be far more useful.
>
> It is more than realistic. Hosting companies run more than 100 VPSs in
> reality. There are also other usefull scenarios. For example, I know
> the universities which run VPS for every faculty web site, for every
> department, mail server and so on. Why do you think they want to run
> only 5VMs on one machine? Much more!

I made no commont on what "they" might want, I want to make the rack of underutilized Windows, BSD and Solaris servers go away. An approach which doesn't support unmodified guest installs doesn't solve any of my current problems. I didn't say it was in any way not useful, just not of interest to me. What needs I have for Linux environments are answered by jails and/or UML.

--
bill davidsen <davidsen@tmr.com>
  CTO TMR Associates, Inc
  Doing interesting things with small computers since 1979

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Tue, 28 Mar 2006 15:03:34 GMT
View Forum Message <> Reply to Message

Bill Davidsen <davidsen@tmr.com> writes:

> Kirill Korotaev wrote:
>
>>> Frankly I don't see running 100 VMs as a realistic goal, being able to run
>>> Linux, Windows, Solaris and BEOS unmodified in 4-5 VMs would be far more
>>> useful.
>>
>> It is more than realistic. Hosting companies run more than 100 VPSs in
>> reality. There are also other usefull scenarios. For example, I know the
>> universities which run VPS for every faculty web site, for every department,
>> mail server and so on. Why do you think they want to run only 5VMs on one
>> machine? Much more!
>
> I made no commont on what "they" might want, I want to make the rack of
> underutilized Windows, BSD and Solaris servers go away. An approach which
> doesn't support unmodified guest installs doesn't solve any of my current
> problems. I didn't say it was in any way not useful, just not of interest to
> me. What needs I have for Linux environments are answered by jails and/or UML.

So from one perspective that is what we are building. A full featured
jail capable of running an unmodified linux distro. The cost is
simply making a way to use the same names twice for the global
namespaces. UML may use these features to accelerate it's own processes.

Virtualization is really the wrong word to describe what we are building. As
it allows for all kinds of heavy weight implementations, and has an associate
with much heavier things.

At the extreme end where you only have one process in each logical instance
of the kernel, a better name would be a heavy weight process. Where each
such process sees an environment as if it owned the entire machine.

Eric

Subject: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Tue, 28 Mar 2006 15:35:58 GMT
View Forum Message <> Reply to Message

On Tue, Mar 28, 2006 at 07:15:17PM +1000, Nick Piggin wrote:
> Kirill Korotaev wrote:
> >
> >Nick, will be glad to shed some light on it.

> >
>
> Thanks very much Kirill.
>
> I don't think I'm qualified to make any decisions about this,
> so I don't want to detract from the real discussions, but I
> just had a couple more questions:
>
> >First of all, what it does which low level virtualization can't:
> >- it allows to run 100 containers on 1GB RAM
> > (it is called containers, VE - Virtual Environments,
> >   VPS - Virtual Private Servers).
> >- it has no much overhead (<1-2%), which is unavoidable with hardware
> > virtualization. For example, Xen has >20% overhead on disk I/O.
>
> Are any future hardware solutions likely to improve these problems?

not really, but as you know, "640K ought to be enough
for anybody", so maybe future hardware developments will
make shared resources possible (with different kernels)

> >OS kernel virtualization
> >~~~~~~~~~~~~~~~~~~~~~~~~
>
> Is this considered secure enough that multiple untrusted VEs are run
> on production systems?

definitely! there are many, many, hosting providers
using exactly this technology to provide Virutal Private
Servers for their customers, of course, in production

> What kind of users want this, who can't use alternatives like real
> VMs?

well, the same users who do not want to use Bochs for
emulating a PC on a PC, when they can use UML for example,
because it's much faster and easier to use ...

aside from that, Linux-VServer for example, is not only
designed to create complete virtual servers, it also
works for service separation and increasing security for
many applications, like for example:

 - test environments (one guest per distro)
 - service separation (one service per 'container')
 - resource management and accounting

> >Summary of previous discussions on LKML

> >~~~~~~
>
> Have their been any discussions between the groups pushing this
> virtualization, and ...

yes, the discussions are ongoing ... maybe to clarify the
situation for the folks not involved (projects in
alphabetical order):

FreeVPS (Free Virtual Private Server Solution):
==========
[http://www.freevps.com/]
not pushing for inclusion, early Linux-VServer
spinoff, partially maintained but they seem to have
other interrests lately

Alex Lyashkov (FreeVPS kernel maintainer)
[Positive Software Corporation http://www.freevps.com/]

BSD Jail LSM (Linux-Jails security module):
==========
[http://kerneltrap.org/node/3823]

Serge E. Hallyn (Patch/Module maintainer) [IBM]
interested in some kind of mainline solution

Dave Hansen (IBM Linux Technology Center)
interested in virtualization for context/container
migration

Linux-VServer (community project, maintained):
==========
[http://linux-vserver.org/]

Jacques Gelinas (previous VServer maintainer)
not pushing for inclusion

Herbert Poetzl (Linux-VServer kernel maintainer)
not pushing for inclusion, but I want to make damn
sure that there does not come bloat into the kernel
and the mainline effords will be usable for
Linux-VServer and similar ...

Sam Vilain (Refactoring Linux-VServer patches)
[Catalyst http://catalyst.net.nz/]
trying hard to provide a simple/minimalistic version
of Linux-VServer for mainline

many others, not really pushing anything here :)

OpenVZ (open project, maintained, subset of Virtuozzo(tm)):
==========
[http://openvz.org/]

Kir Kolyshkin (OpenVZ maintainer):
[SWsoft http://www.swsoft.com I gues?]
maybe pushing for inclusion ...

Kirill Korotaev (OpenVZ/Virtuozzo kernel developer?)
[SWsoft http://www.swsoft.com]
heavily pushing for inclusion ...

Alexey Kuznetsov (Chief Software Engineer)
[SWsoft http://www.swsoft.com]
not pushing but supporting company interrests

PID Virtualization (kernel branch for inclusion):
==========

Eric W. Biederman (branch developer/maintainer)
[XMission http://xmission.com/]

Virtuozzo(tm) (Commercial solution form SWsoft):
==========
[http://www.virtuozzo.com/]

not involved yet, except via OpenVZ

Stanislav Protassov (Director of Engineering)
[SWsoft http://www.swsoft.com]

A ton of IBM and VZ folks are not listed here, but I
guess you can figure who is who from the email addresses

there are also a bunch of folks from Columbia and
Princeton university interested and/or involved in
kernel level virtualization and context migration.

please extend this list where appropriate, I'm pretty
sure I forgot at least five important/involved persons

> important kernel developers who are not part of a virtualization
> effort?

no idea, probably none for now ...

> Ie. is there any consensus about the future of these patches?

what patches? what future?

HTC,
Herbert

> Thanks,
> Nick
>
> --
> SUSE Labs, Novell Inc.
> Send instant messages to your online friends http://au.messenger.yahoo.com

---

Subject: Re: Re: [RFC] Virtualization steps
Posted by TheWiseOne on Tue, 28 Mar 2006 15:48:56 GMT
View Forum Message <> Reply to Message

Kirill Korotaev wrote:
>> Oh, after you come to an agreement and start posting patches, can you
>> also outline why we want this in the kernel (what it does that low
>> level virtualization doesn't, etc, etc), and how and why you've agreed
>> to implement it. Basically, some background and a summary of your
>> discussions for those who can't follow everything. Or is that a faq
>> item?
> Nick, will be glad to shed some light on it.
>
> First of all, what it does which low level virtualization can't:
> - it allows to run 100 containers on 1GB RAM
>   (it is called containers, VE - Virtual Environments,
>   VPS - Virtual Private Servers).
> - it has no much overhead (<1-2%), which is unavoidable with hardware
>   virtualization. For example, Xen has >20% overhead on disk I/O.

I think the Xen guys would disagree with you on this. Xen claims <3%
overhead on the XenSource site.

Where did you get these figures from? What Xen version did you test?
What was your configuration? Did you have kernel debugging enabled? You
can't just post numbers without the data to back it up, especially when
it conflicts greatly with the Xen developers statements. AFAIK Xen is
well on it's way to inclusion into the mainstream kernel.

Thank you,
Matt Ayres

Subject: Re: [RFC] Virtualization steps
Posted by Nick Piggin on Tue, 28 Mar 2006 15:53:59 GMT
View Forum Message <> Reply to Message

Herbert Poetzl wrote:
> On Tue, Mar 28, 2006 at 07:15:17PM +1000, Nick Piggin wrote:

[...]

Thanks for the clarifications, Herbert.

>>Ie. is there any consensus about the future of these patches?
>
>
> what patches?

One's being thrown around lkml, and future ones being talked about.
Patches ~= changes to kernel.

> what future?

I presume everyone's goal is to get something into the kernel?

--
SUSE Labs, Novell Inc.
Send instant messages to your online friends http://au.messenger.yahoo.com

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Tue, 28 Mar 2006 16:15:56 GMT
View Forum Message <> Reply to Message

Nick Piggin <nickpiggin@yahoo.com.au> writes:

> Kirill Korotaev wrote:
>> Nick, will be glad to shed some light on it.
>>
>
> Thanks very much Kirill.
>
> I don't think I'm qualified to make any decisions about this,
> so I don't want to detract from the real discussions, but I
> just had a couple more questions:
>
>> First of all, what it does which low level virtualization can't:
>> - it allows to run 100 containers on 1GB RAM
>>   (it is called containers, VE - Virtual Environments,
>>   VPS - Virtual Private Servers).

>> - it has no much overhead (<1-2%), which is unavoidable with hardware
>> virtualization. For example, Xen has >20% overhead on disk I/O.
>
> Are any future hardware solutions likely to improve these problems?

This isn't a direct competition, both solutions coincide nicely.

The major efficiency differences are fundamental to the approaches and
can only be solved in software and not hardware. The fundamental efficiency
limits of low level virtualization are not sharing resources between
instances well (think how hard memory hotplug is to solve), the fact
that running a kernel takes at least 1MB for just the kernel, the
fact that no matter how good your hypervisor is there will be some
hardware interface it doesn't virtualize.

Whereas what we are aiming at are just enough modifications to the kernel
to allow multiple instances of user space. We aren't virtualizing anything
that isn't already virtualized in the kernel.

>> OS kernel virtualization
>> ~~~~~~~~~~~~~~~~~~~~~~~~
>
> Is this considered secure enough that multiple untrusted VEs are run
> on production systems?

Kirill or Herbert can give a better answer but that is of the major
points of BSD Jails and their kin is it not?

> What kind of users want this, who can't use alternatives like real
> VMs?

Well that question assumes a lot. The answer that assumes a lot
in the other direction is that adding an additional unnecessary layers
just complicates the problem and slows things down for no reason
while making it so you can't assume the solution is always present.
In addition to doing it in a non-portable way so it is only available
on a few platforms.

I can't even think of a straight answer to the users question.

My users are in the high performance computing realm, and for that
subset it is easy. Xen and it's kin don't virtualize the high
bandwidth low latency communication hardware that is used, and that
may not even be possible. Using a hypervisor in a situation like that
certainly isn't general or easily maintainable. (Think about
what a challenge it has been to get usable infiniband drivers merged).

>> Summary of previous discussions on LKML

>> ~~~~~~
>
> Have their been any discussions between the groups pushing this
> virtualization, and important kernel developers who are not part of
> a virtualization effort? Ie. is there any consensus about the
> future of these patches?

Yes, but just enough to give us hope :)

Unless you count the mount namespace as part of this in which case
pieces are already merged.

The challenging is that writing kernel code that does this is
easy. Writing kernel code that is mergeable and that the different
groups all agree meets their requirements is much harder. It has
taken us until now to have a basic approach that we all agree on.
Now we get to beat each other up over the technical details :)

Eric

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Tue, 28 Mar 2006 16:31:38 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> PID Virtualization (kernel branch for inclusion):
> ==
>
> Eric W. Biederman (branch developer/maintainer)
> [XMission http://xmission.com/]

Actually I work for Linux Networx http://www.lnxi.com
XMission is just my ISP. I find it easier to work from
home. :)

Eric

Subject: Re: Re: [RFC] Virtualization steps
Posted by ebiederm on Tue, 28 Mar 2006 16:42:41 GMT
View Forum Message <> Reply to Message

Matt Ayres <matta@tektonic.net> writes:

> I think the Xen guys would disagree with you on this. Xen claims <3% overhead

> on the XenSource site.
>
> Where did you get these figures from? What Xen version did you test? What was
> your configuration? Did you have kernel debugging enabled? You can't just post
> numbers without the data to back it up, especially when it conflicts greatly
> with the Xen developers statements. AFAIK Xen is well on it's way to inclusion
> into the mainstream kernel.

It doesn't matter. The proof that Xen has more overhead is trivial
Xen does more, and Xen clients don't share resources well.

Nor is this about Xen vs what we are doing. These are different
non conflicting approaches that operating in completely different
ways and solve a different set of problems.

Xen is about multiple kernels.

The alternative is a supped of chroot.

Eric

Subject: Re: Re: [RFC] Virtualization steps
Posted by TheWiseOne on Tue, 28 Mar 2006 17:04:50 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Matt Ayres <matta@tektonic.net> writes:
>
>> I think the Xen guys would disagree with you on this. Xen claims <3% overhead
>> on the XenSource site.
>>
>> Where did you get these figures from? What Xen version did you test? What was
>> your configuration? Did you have kernel debugging enabled? You can't just post
>> numbers without the data to back it up, especially when it conflicts greatly
>> with the Xen developers statements. AFAIK Xen is well on it's way to inclusion
>> into the mainstream kernel.
>
> It doesn't matter. The proof that Xen has more overhead is trivial
> Xen does more, and Xen clients don't share resources well.
>

I understand the difference. It was more about Kirill grabbing numbers
out of the air. I actually think the containers and Xen complement each
other very well. As Xen is now based on 2.6.16 (as are both VServer and
OVZ) it makes sense to run a few Xen domains that then in turn run
containers in some scenarios. As far as the last part, Xen doesn't
share resources at all :)

Thank you,
Matt Ayres

Subject: Re: [RFC] Virtualization steps
Posted by Jeff Dike on Tue, 28 Mar 2006 17:47:55 GMT
View Forum Message <> Reply to Message

On Tue, Mar 28, 2006 at 08:03:34AM -0700, Eric W. Biederman wrote:
> UML may use these features to accelerate it's own processes.

And I'm planning on doing exactly that.

Jeff

Subject: Re: Re: [RFC] Virtualization steps
Posted by Jun OKAJIMA on Tue, 28 Mar 2006 20:26:07 GMT
View Forum Message <> Reply to Message

>
>I'll summarize it this way: low-level virtualization uses resource
>inefficiently.
>
>With this higher-level stuff, you get to share all of the Linux caching,
>and can do things like sharing libraries pretty naturally.
>
>They are also much lighter-weight to create and destroy than full
>virtual machines. We were planning on doing some performance
>comparisons versus some hypervisors like Xen and the ppc64 one to show
>scaling with the number of virtualized instances. Creating 100 of these
>Linux containers is as easy as a couple of shell scripts, but we still
>can't find anybody crazy enough to go create 100 Xen VMs.
>
>Anyway, those are the things that came to my mind first. I'm sure the
>others involved have their own motivations.
>

Some questions.

1. Your point is rignt in some ways, and I agree with you.
   Yes, I currently guess Jail is quite practical than Xen.
   Xen sounds cool, but really practical? I doubt a bit.
   But it would be a narrow thought, maybe.
   How you estimate feature improvement of memory shareing
   on VM (e.g. Xen/VMware)?

I have seen there are many papers about this issue.
If once memory sharing gets much efficient, Xen possibly wins.

2. Folks, how you think about other good points of Xen,
like live migration, or runs solaris, or has suspend/resume or...
No Linux jails have such feature for now, although I dont think
it is impossible with jail.

My current suggestion is,

1. Dont use Xen for running multiple VMs.
2. Use Xen for better admin/operation/deploy... tools.
3. If you need multiple VMs, use jail on Xen.

--- Okajima, Jun. Tokyo, Japan.
http://www.digitalinfra.co.jp/
http://www.colinux.org/
http://www.machboot.com/

Subject: Re: Re: [RFC] Virtualization steps
Posted by kir on Tue, 28 Mar 2006 20:50:05 GMT
View Forum Message <> Reply to Message

Jun OKAJIMA wrote:

>>I'll summarize it this way: low-level virtualization uses resource
>>inefficiently.
>>
>>With this higher-level stuff, you get to share all of the Linux caching,
>>and can do things like sharing libraries pretty naturally.
>>
>>They are also much lighter-weight to create and destroy than full
>>virtual machines. We were planning on doing some performance
>>comparisons versus some hypervisors like Xen and the ppc64 one to show
>>scaling with the number of virtualized instances. Creating 100 of these
>>Linux containers is as easy as a couple of shell scripts, but we still
>>can't find anybody crazy enough to go create 100 Xen VMs.
>>
>>Anyway, those are the things that came to my mind first. I'm sure the
>>others involved have their own motivations.
>>
>>
>>
>
>Some questions.
>

>1. Your point is rignt in some ways, and I agree with you.
> Yes, I currently guess Jail is quite practical than Xen.
> Xen sounds cool, but really practical? I doubt a bit.
> But it would be a narrow thought, maybe.
> How you estimate feature improvement of memory shareing
> on VM (e.g. Xen/VMware)?
> I have seen there are many papers about this issue.
> If once memory sharing gets much efficient, Xen possibly wins.
>
>
This is not just about memory sharing. Dynamic resource management is hardly possible in a model where you have multiple kernels running; all of those kernel were designed to run on a dedicated hardware. As it was pointed out, adding/removing memory from a Xen guest during runtime is tricky.

Finally, multiple-kernels-on-top-of-hypervisor architecture is just more complex and has more overhead then one-kernel-with-many-namespaces.

>2. Folks, how you think about other good points of Xen,
> like live migration, or runs solaris, or has suspend/resume or...
>
>
OpenVZ will have live zero downtime migration and suspend/resume some time next month.

> No Linux jails have such feature for now, although I dont think
> it is impossible with jail.
>
>
>My current suggestion is,
>
>1. Dont use Xen for running multiple VMs.
>2. Use Xen for better admin/operation/deploy... tools.
>
>
This point is controversial. Tools are tools -- they can be made to support Xen, Linux VServer, UML, OpenVZ, VMware -- or even all of them!

But anyway, speaking of tools and better admin operations, what it takes to create a Xen domain (I mean create all those files needed to run a new Xen domain), and how much time it takes? Say, in OpenVZ creation of a VE (Virtual Environment) is a matter of unpacking a ~100MB tarball and copying 1K config file -- which essentially means one can create a VE in a minute. Linux-VServer should be pretty much the same.

Another concern is, yes, manageability. In OpenVZ model the host system can easily access all the VPSs' files, making, say, a mass software

update a reality. You can easily change some settings in 100+ VEs very
easy. In systems based on Xen and, say, VMware one should log in into
each system, one by one, to administer them, which is not unlike the
'separate physical server' model.

>3. If you need multiple VMs, use jail on Xen.
>
>
Indeed, a mixed approach is very interesting. You can run OpenVZ or
Linux-VServer in a Xen domain, that makes a lot of sense.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Jun OKAJIMA on Tue, 28 Mar 2006 21:35:02 GMT
View Forum Message <> Reply to Message

>
>>2. Folks, how you think about other good points of Xen,
>>   like live migration, or runs solaris, or has suspend/resume or...
>>
>>
>OpenVZ will have live zero downtime migration and suspend/resume some
>time next month.
>

COOL!!!!

>>
>>1. Dont use Xen for running multiple VMs.
>>2. Use Xen for better admin/operation/deploy... tools.
>>
>>
>This point is controversial. Tools are tools -- they can be made to
>support Xen, Linux VServer, UML, OpenVZ, VMware -- or even all of them!
>
>But anyway, speaking of tools and better admin operations, what it takes
>to create a Xen domain (I mean create all those files needed to run a
>new Xen domain), and how much time it takes? Say, in OpenVZ creation of
>a VE (Virtual Environment) is a matter of unpacking a ~100MB tarball and
>copying 1K config file -- which essentially means one can create a VE in
>a minute. Linux-VServer should be pretty much the same.
>
>Another concern is, yes, manageability. In OpenVZ model the host system
>can easily access all the VPSs' files, making, say, a mass software
>update a reality. You can easily change some settings in 100+ VEs very
>easy. In systems based on Xen and, say, VMware one should log in into
>each system, one by one, to administer them, which is not unlike the
>'separate physical server' model.

>
>>3. If you need multiple VMs, use jail on Xen.
>>
>>
>Indeed, a mixed approach is very interesting. You can run OpenVZ or
>Linux-VServer in a Xen domain, that makes a lot of sense.
>
>

Sorry for making misunderstanding.
What I wanted to say with "2" (use Xen as a tool) is, probably same as
what you are guessing now.
I mean, you make a server like this.
1. Install jailed Linux(OpenVZ/Vserver/or..) on Xen
2. make only one domU. and many VMs on this domU with jail.
3. runs many (more than 100 or...) VMs with jail, not with Xen.
4. but, for example, you want to migrate to another PC,
   use Xen live migration.
The fourth point would help administration tasks easier. This is the point
where I mentioned about "better tool".
There is other usage of Xen as admin tool. For example, if you need device
driver (e.g. new iSCSI H/W driver or gigabit ether or...) of 2.6 kernel, but
no need to use any other 2.6 funcs, keep guest OS (domU) as 2.4, and make
dom0 as 2.6 Xen. This also helps admin tasks.
Probably, the biggest problem for now is, Xen patch conflicts with
Vserver/OpenVZ patch.

--- Okajima, Jun. Tokyo, Japan.

Subject: Re: Re: [RFC] Virtualization steps
Posted by ebiederm on Tue, 28 Mar 2006 21:51:05 GMT
View Forum Message <> Reply to Message

Jun OKAJIMA <okajima@digitalinfra.co.jp> writes:

> Probably, the biggest problem for now is, Xen patch conflicts with
> Vserver/OpenVZ patch.

The implementations are significantly different enough that I don't
see Xen and any jail patch really conflicting. There might be some
trivial conflicts in /proc but even that seems unlikely.

Eric

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Tue, 28 Mar 2006 21:58:23 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

>> - network virtualization
>
> here I see many issues, as for example Linux-VServer
> does not necessarily aim for full virtualization, when
> simple and performant isolation is sufficient.

The current technique employed by vserver is implementable
in a security module today. We are implementing each of
these pieces as a separate namespace. So actually using
any one of them is optional. So implementing your current
method of network isolation in a security module should be straight
forward.

Eric

Subject: Re: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Tue, 28 Mar 2006 21:59:29 GMT
View Forum Message <> Reply to Message

On Tue, 2006-03-28 at 10:45 +0400, Kir Kolyshkin wrote:
> It is actually not a future goal, but rather a reality. Since os-level
> virtualization overhead is very low (1-2 per cent or so), one can run
> hundreds of VEs.

Huh? You managed to measure it!? Or do you just mean "negligible" by
"1-2 per cent" ? :-)

Sam.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Kir Kolyshkin on Tue, 28 Mar 2006 22:24:09 GMT
View Forum Message <> Reply to Message

Sam Vilain wrote:

>On Tue, 2006-03-28 at 10:45 +0400, Kir Kolyshkin wrote:
>
>
>>It is actually not a future goal, but rather a reality. Since os-level
>>virtualization overhead is very low (1-2 per cent or so), one can run

>>hundreds of VEs.
>>
>>
>
>Huh? You managed to measure it!? Or do you just mean "negligible" by
>"1-2 per cent" ? :-)
>
>
We run different tests to measure OpenVZ/Virtuozzo overhead, as we do
care much for that stuff. I do not remember all the gory details at the
moment, but I gave the correct numbers: "1-2 per cent or so".

There are things such as networking (OpenVZ's venet device) overhead, a
fair cpu scheduler overhead, something else.

Why do you think it can not be measured? It either can be, or it is too
low to be measured reliably (a fraction of a per cent or so).

Regards,
  Kir.

Subject: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Tue, 28 Mar 2006 22:50:56 GMT
View Forum Message <> Reply to Message

On Tue, 2006-03-28 at 12:51 +0400, Kirill Korotaev wrote:
> we will create a separate branch also called -acked, where patches
> agreed upon will go.

No need. Just use Acked-By: comments.

Also, can I give some more feedback on the way you publish your patches:

1. git's replication uses the notion of a forward-only commit list.
   So, if you change patches or rebase them then you have to rewind
   the base point - which in pure git terms means create a new head.
   So, you should use the convention of putting some identifier - a
   date, or a version number - in each head.

2. Why do you have a seperate repository for your normal openvz and the
   -ms trees? You can just you different heads.

3. Apache was doing something weird to the HEAD symlink in your
   repository. (mind you, if you adopt notion 1., this becomes
   irrelevant :-))

Otherwise, it's a great thing to see your patches published via git!

I can't recommend Stacked Git more highly for performing the 'winding'
of the patch stack necessary for revising patches. Google for "stgit".

Sam.

Subject: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Tue, 28 Mar 2006 23:07:34 GMT
View Forum Message <> Reply to Message

On Tue, 2006-03-28 at 09:41 -0500, Bill Davidsen wrote:
> > It is more than realistic. Hosting companies run more than 100 VPSs in
> > reality. There are also other usefull scenarios. For example, I know
> > the universities which run VPS for every faculty web site, for every
> > department, mail server and so on. Why do you think they want to run
> > only 5VMs on one machine? Much more!
>
> I made no commont on what "they" might want, I want to make the rack of
> underutilized Windows, BSD and Solaris servers go away. An approach
> which doesn't support unmodified guest installs doesn't solve any of my
> current problems. I didn't say it was in any way not useful, just not of
> interest to me. What needs I have for Linux environments are answered by
> jails and/or UML.

We are talking about adding jail technology, also known as containers on
Solaris and vserver/openvz on Linux, to the mainline kernel.

So, you are obviously interested!

Because of course, you can take an unmodified filesystem of the guest
and assuming the kernels are compatible run them without changes. I
find this consolidation approach indispensible.

Sam.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Tue, 28 Mar 2006 23:17:58 GMT
View Forum Message <> Reply to Message

On Tue, 2006-03-28 at 14:51 -0700, Eric W. Biederman wrote:
> Jun OKAJIMA <okajima@digitalinfra.co.jp> writes:
>
> > Probably, the biggest problem for now is, Xen patch conflicts with
> > Vserver/OpenVZ patch.
>

> The implementations are significantly different enough that I don't
> see Xen and any jail patch really conflicting. There might be some
> trivial conflicts in /proc but even that seems unlikely.

This has been done before,

http://list.linux-vserver.org/archive/vserver/msg10235.html

Sam.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Tue, 28 Mar 2006 23:28:13 GMT
View Forum Message <> Reply to Message

On Wed, 2006-03-29 at 02:24 +0400, Kir Kolyshkin wrote:
> >Huh? You managed to measure it!? Or do you just mean "negligible" by
> >"1-2 per cent" ? :-)
> We run different tests to measure OpenVZ/Virtuozzo overhead, as we do
> care much for that stuff. I do not remember all the gory details at the
> moment, but I gave the correct numbers: "1-2 per cent or so".
>
> There are things such as networking (OpenVZ's venet device) overhead, a
> fair cpu scheduler overhead, something else.
>
> Why do you think it can not be measured? It either can be, or it is too
> low to be measured reliably (a fraction of a per cent or so).

Well, for instance the fair CPU scheduling overhead is so tiny it may as
well not be there in the VServer patch. It's just a per-vserver TBF
that feeds back into the priority (and hence timeslice length) of the
process. ie, you get "CPU tokens" which deplete as processes in your
vserver run and you either get a boost or a penalty depending on the
level of the tokens in the bucket. This doesn't provide guarantees, but
works well for many typical workloads. And once Herbert fixed the SMP
cacheline problems in my code ;) it was pretty much full speed. That
is, until you want it to sacrifice overall performance for enforcing
limits.

How does your fair scheduler work? Do you just keep a runqueue for each
vps?

To be honest, I've never needed to determine whether its overhead is 1%
or 0.01%, it would just be a meaningless benchmark anyway :-). I know
it's "good enough for me".

Sam.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Kirill Korotaev on Wed, 29 Mar 2006 00:55:01 GMT
View Forum Message <> Reply to Message

> Kirill Korotaev wrote:
>>> Oh, after you come to an agreement and start posting patches, can you
>>> also outline why we want this in the kernel (what it does that low
>>> level virtualization doesn't, etc, etc), and how and why you've agreed
>>> to implement it. Basically, some background and a summary of your
>>> discussions for those who can't follow everything. Or is that a faq
>>> item?
>> Nick, will be glad to shed some light on it.
>>
>> First of all, what it does which low level virtualization can't:
>> - it allows to run 100 containers on 1GB RAM
>>   (it is called containers, VE - Virtual Environments,
>>   VPS - Virtual Private Servers).
>> - it has no much overhead (<1-2%), which is unavoidable with hardware
>>   virtualization. For example, Xen has >20% overhead on disk I/O.
>
> I think the Xen guys would disagree with you on this. Xen claims <3%
> overhead on the XenSource site.
>
> Where did you get these figures from? What Xen version did you test?
> What was your configuration? Did you have kernel debugging enabled? You
> can't just post numbers without the data to back it up, especially when
> it conflicts greatly with the Xen developers statements. AFAIK Xen is
> well on it's way to inclusion into the mainstream kernel.
I have no exact numbers in the hands as I'm in another country right now.
But! We tested Xen not long ago with iozone test suite and it gave
~20-30% disk I/O overhead. Recently we were testing CPU scheduler and
EDF scheduler gave me 33% overhead on some very simple loads with almost
busy loops inside VMs. It also was not providing any good fairness on
2CPU SMP system to my suprise. You can object to me, but better simply
retest it if interested yourself. There were other tests as well, which
reported very different overheads on Xen 3. I suppose Xen guys do such
measurements themself, no?
And I'm sure, they are constantly improving it, they are doing a good
work on it.

Thanks,
Kirill

Subject: Re: [RFC] Virtualization steps
Posted by dev on Wed, 29 Mar 2006 01:39:00 GMT
View Forum Message <> Reply to Message

Nick,

>> First of all, what it does which low level virtualization can't:
>> - it allows to run 100 containers on 1GB RAM
>> (it is called containers, VE - Virtual Environments,
>> VPS - Virtual Private Servers).
>> - it has no much overhead (<1-2%), which is unavoidable with hardware
>> virtualization. For example, Xen has >20% overhead on disk I/O.
>
> Are any future hardware solutions likely to improve these problems?
Probably you are aware of VT-i/VT-x technologies and planned virtualized
MMU and I/O MMU from Intel and AMD.
These features should improve the performance somehow, but there is
still a limit for decreasing the overhead, since at least disk, network,
video and such devices should be emulated.

>> OS kernel virtualization
>> ~~~~~~~~~~~~~~~~~~~~~~~~
>
> Is this considered secure enough that multiple untrusted VEs are run
> on production systems?
it is secure enough. What makes it secure? In general:
- virtualization, which makes resources private
- resource control, which makes VE to be limited with its usages
In more technical details virtualization projects make user access (and
capabilities) checks stricter. Moreover, OpenVZ is using "denied by
default" approach to make sure it is secure and VE users are not allowed
something else.

Also, about 2-3 month ago we had a security review of OpenVZ project
made by Solar Designer. So, in general such virtualization approach
should be not less secure than VM-like one. VM core code is bigger and
there is enough chances for bugs there.

> What kind of users want this, who can't use alternatives like real
> VMs?
Many companies, just can't share their names. But in general no
enterprise and hosting companies need to run different OSes on the same
machine. For them it is quite natural to use N machines for Linux and M
for Windows. And since VEs are much more lightweight and easier to work
with, they like it very much.

Just for example, OpenVZ core is running more than 300,000 VEs worldwide.

Thanks,
Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by Kirill Korotaev on Wed, 29 Mar 2006 09:13:14 GMT
View Forum Message <> Reply to Message

Sam,

>> Why do you think it can not be measured? It either can be, or it is too
>> low to be measured reliably (a fraction of a per cent or so).
>
> Well, for instance the fair CPU scheduling overhead is so tiny it may as
> well not be there in the VServer patch. It's just a per-vserver TBF
> that feeds back into the priority (and hence timeslice length) of the
> process. ie, you get "CPU tokens" which deplete as processes in your
> vserver run and you either get a boost or a penalty depending on the
> level of the tokens in the bucket. This doesn't provide guarantees, but
> works well for many typical workloads.
I wonder what is the value of it if it doesn't do guarantees or QoS?
In our experiments with it we failed to observe any fairness. So I
suppose the only goal of this is too make sure that maliscuios user want
consume all the CPU power, right?

> How does your fair scheduler work? Do you just keep a runqueue for each
> vps?
we keep num_online_cpus runqueues per VPS.
Fairs scheduler is some kind of SFQ like algorithm which selects VPS to
be scheduled, than standart linux scheduler selects a process in a VPS
runqueues to run.

> To be honest, I've never needed to determine whether its overhead is 1%
> or 0.01%, it would just be a meaningless benchmark anyway :-). I know
> it's "good enough for me".
Sure! We feel the same, but people like numbers :)

Thanks,
Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Wed, 29 Mar 2006 11:08:49 GMT
View Forum Message <> Reply to Message

On Wed, 2006-03-29 at 13:13 +0400, Kirill Korotaev wrote:
> > Well, for instance the fair CPU scheduling overhead is so tiny it may as
> > well not be there in the VServer patch. It's just a per-vserver TBF
> > that feeds back into the priority (and hence timeslice length) of the
> > process. ie, you get "CPU tokens" which deplete as processes in your
> > vserver run and you either get a boost or a penalty depending on the
> > level of the tokens in the bucket. This doesn't provide guarantees, but

> > works well for many typical workloads.
> I wonder what is the value of it if it doesn't do guarantees or QoS?

It still does "QoS". The TBF has a "fill rate", which is basically N tokens per M jiffies. Then you just set the size of the "bucket", and the prio bonus given is between -5 (when bucket is full) and +15 (when bucket is empty). The normal -10 to +10 'interactive' prio bonus is reduced to -5 to +5 to compensate.

In other words, it's like a global 'nice' across all of the processes in the vserver.

So, these characteristics do provide some level of guarantees, but not all that people expect. eg, people want to say "cap usage at 5%", but as designed the scheduler does not ever prevent runnable processes from running if the CPUs have nothing better to do, so they think the scheduler is broken. It is also possible with a fork bomb (assuming the absence of appropriate ulimits) that you start enough processes that you don't care that they are all effectively nice +19.

Herbert later made it add some of these guarantees, but I believe there is a performance impact of some kind.

> In our experiments with it we failed to observe any fairness.

Well, it does not aim to be 'fair', it aims to be useful for allocating CPU to vservers. ie, if you allocate X% of the CPU in the system to a vserver, and it uses more, then try to make it use less via priority penalties - and give others shortchanged or not using the CPU very much performance bonuses. That's all.

So, if you under- or over-book CPU allocation, it doesn't work. The idea was that monitoring it could be shipped out to userland. I just wanted something flexible enough to allow virtually any policy to be put into place without wasting too many cycles.

> > How does your fair scheduler work? Do you just keep a runqueue for each
> > vps?
> we keep num_online_cpus runqueues per VPS.

Right. I considered that approach but just couldn't be bothered implementing it, so went with the TBF because it worked and was lightweight.

> Fairs scheduler is some kind of SFQ like algorithm which selects VPS to
> be scheduled, than standart linux scheduler selects a process in a VPS
> runqueues to run.

Right.

> > To be honest, I've never needed to determine whether its overhead is 1%
> > or 0.01%, it would just be a meaningless benchmark anyway :-). I know
> > it's "good enough for me".
> Sure! We feel the same, but people like numbers :)

Sometimes the answer has to be "mu".

Sam.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Wed, 29 Mar 2006 13:45:24 GMT
View Forum Message <> Reply to Message

On Wed, Mar 29, 2006 at 01:13:14PM +0400, Kirill Korotaev wrote:
> Sam,
>
> >>Why do you think it can not be measured? It either can be, or it is too
> >>low to be measured reliably (a fraction of a per cent or so).
> >
> >Well, for instance the fair CPU scheduling overhead is so tiny it may as
> >well not be there in the VServer patch. It's just a per-vserver TBF
> >that feeds back into the priority (and hence timeslice length) of the
> >process. ie, you get "CPU tokens" which deplete as processes in your
> >vserver run and you either get a boost or a penalty depending on the
> >level of the tokens in the bucket. This doesn't provide guarantees, but
> >works well for many typical workloads.

> I wonder what is the value of it if it doesn't do guarantees or QoS?
> In our experiments with it we failed to observe any fairness.

probably a misconfiguration on your side ...

> So I suppose the only goal of this is too make sure that maliscuios
> user want consume all the CPU power, right?

the currently used scheduler extensions do much
more than that, basically all kinds of scenarios
can be satisfied with it, at almost no overhead

> >How does your fair scheduler work?
> >Do you just keep a runqueue for each vps?
> we keep num_online_cpus runqueues per VPS.

> Fairs scheduler is some kind of SFQ like algorithm which selects VPS
> to be scheduled, than standart linux scheduler selects a process in a

> VPS runqueues to run.
>
> >To be honest, I've never needed to determine whether its overhead is 1%
> >or 0.01%, it would just be a meaningless benchmark anyway :-). I know
> >it's "good enough for me".

> Sure! We feel the same, but people like numbers :)

well, do you have numbers?

best,
Herbert

> Thanks,
> Kirill

Subject: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Wed, 29 Mar 2006 13:47:58 GMT
View Forum Message <> Reply to Message

On Wed, Mar 29, 2006 at 05:39:00AM +0400, Kirill Korotaev wrote:
> Nick,
>
> >>First of all, what it does which low level virtualization can't:
> >>- it allows to run 100 containers on 1GB RAM
> >> (it is called containers, VE - Virtual Environments,
> >> VPS - Virtual Private Servers).
> >>- it has no much overhead (<1-2%), which is unavoidable with hardware
> >> virtualization. For example, Xen has >20% overhead on disk I/O.
> >
> >Are any future hardware solutions likely to improve these problems?
> Probably you are aware of VT-i/VT-x technologies and planned virtualized
> MMU and I/O MMU from Intel and AMD.
> These features should improve the performance somehow, but there is
> still a limit for decreasing the overhead, since at least disk, network,
> video and such devices should be emulated.
>
> >>OS kernel virtualization
> >>~~~~~~~~~~~~~~~~~~~~~~~~
> >
> >Is this considered secure enough that multiple untrusted VEs are run
> >on production systems?
> it is secure enough. What makes it secure? In general:
> - virtualization, which makes resources private
> - resource control, which makes VE to be limited with its usages
> In more technical details virtualization projects make user access (and
> capabilities) checks stricter. Moreover, OpenVZ is using "denied by

> default" approach to make sure it is secure and VE users are not allowed
> something else.
>
> Also, about 2-3 month ago we had a security review of OpenVZ project
> made by Solar Designer. So, in general such virtualization approach
> should be not less secure than VM-like one. VM core code is bigger and
> there is enough chances for bugs there.
>
> >What kind of users want this, who can't use alternatives like real
> >VMs?
> Many companies, just can't share their names. But in general no
> enterprise and hosting companies need to run different OSes on the same
> machine. For them it is quite natural to use N machines for Linux and M
> for Windows. And since VEs are much more lightweight and easier to work
> with, they like it very much.
>
> Just for example, OpenVZ core is running more than 300,000 VEs worldwide.

not bad, how did you get to those numbers?
and, more important, how many of those are actually OpenVZ?
(compared to Virtuozzo(tm))

best,
Herbert

> Thanks,
> Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by dev on Wed, 29 Mar 2006 14:47:58 GMT
View Forum Message <> Reply to Message

>> I wonder what is the value of it if it doesn't do guarantees or QoS?
>> In our experiments with it we failed to observe any fairness.
>
> probably a misconfiguration on your side ...
maybe you can provide some instructions on which kernel version to use
and how to setup the following scenario:
2CPU box. 3 VPSs which should run with 1:2:3 ratio of CPU usage.

> well, do you have numbers?
just run the above scenario with one busy loop inside each VPS. I was
not able to observe 1:2:3 cpu distribution. Other scenarios also didn't
showed my any fairness. The results were different. Sometimes 1:1:2,
sometimes others.

Thanks,

Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Wed, 29 Mar 2006 17:29:01 GMT
View Forum Message <> Reply to Message

On Wed, Mar 29, 2006 at 06:47:58PM +0400, Kirill Korotaev wrote:
> >>I wonder what is the value of it if it doesn't do guarantees or QoS?
> >>In our experiments with it we failed to observe any fairness.
> >
> >probably a misconfiguration on your side ...
> maybe you can provide some instructions on which kernel version to use
> and how to setup the following scenario: 2CPU box. 3 VPSs which should
> run with 1:2:3 ratio of CPU usage.

that is quite simple, you enable the Hard CPU Scheduler
and select the Idle Time Skip, then you set the following
token bucket values depending on what your mean with
'should run with 1:2:3 ratio of CPU usage':

a) a guaranteed maximum of 16.7%, 33.3% and 50.0%

b) a fair sharing according to 1:2:3

c) a guaranteed minimum of 16.7%, 33.3% and 50.0%
   with a fair sharing of 1:2:3 for the rest ...

for all cases you would set:
(adjust according to you reserve/boost likings)

    VPS1,2,3: tokens_min = 50, tokens_max = 500
         interval = interval2 = 6

a) VPS1: rate = 1, hard, noidleskip
    VPS2: rate = 2, hard, noidleskip
    VPS3: rate = 3, hard, noidleskip

b) VPS1: rate2 = 1, hard, idleskip
    VPS2: rate2 = 2, hard, idleskip
    VPS3: rate2 = 3, hard, idleskip

c) VPS1: rate = rate2 = 1, hard, idleskip
    VPS2: rate = rate2 = 2, hard, idleskip
    VPS3: rate = rate2 = 3, hard, idleskip

of course, adjusting rate/interval while keeping

the ratio might help you depending on the guest load
(i.e. more batch load type or mor interactive stuff)

of course, you can do those adjustments per CPU so, if
you for example want to assign one CPU to the third
guest, you can do that easily too ...

> >well, do you have numbers?
> just run the above scenario with one busy loop inside each VPS. I was
> not able to observe 1:2:3 cpu distribution. Other scenarios also didn't
> showed my any fairness. The results were different. Sometimes 1:1:2,
> sometimes others.

what was your setup?

best,
Herbert

> Thanks,
> Kirill

Subject: Re: [RFC] Virtualization steps
Posted by Dave Hansen on Wed, 29 Mar 2006 20:30:34 GMT
View Forum Message <> Reply to Message

On Tue, 2006-03-28 at 12:51 +0400, Kirill Korotaev wrote:
> Eric, we have a GIT repo on openvz.org already:
> http://git.openvz.org

Git is great for getting patches and lots of updates out, but I'm not
sure it is idea for what we're trying to do. We'll need things reviewed
at each step, especially because we're going to be touching so much
common code.

I'd guess set of quilt (or patch-utils) patches is probably best,
especially if we're trying to get stuff into -mm first.

-- Dave

Subject: Re: [RFC] Virtualization steps
Posted by ebiederm on Wed, 29 Mar 2006 20:47:20 GMT
View Forum Message <> Reply to Message

Dave Hansen <haveblue@us.ibm.com> writes:

> On Tue, 2006-03-28 at 12:51 +0400, Kirill Korotaev wrote:
>> Eric, we have a GIT repo on openvz.org already:
>> http://git.openvz.org
>
> Git is great for getting patches and lots of updates out, but I'm not
> sure it is idea for what we're trying to do. We'll need things reviewed
> at each step, especially because we're going to be touching so much
> common code.
>
> I'd guess set of quilt (or patch-utils) patches is probably best,
> especially if we're trying to get stuff into -mm first.

Git is as good at holding patches as quilt. It isn't quite as
good at working with them as quilt but in the long term that is
fixable.

The important point is that we get a collection of patches that
we can all agree to, and that we publish it.

At this point it sounds like each group will happily publish the
patches, and that might not be a bad double check, on agreement.

Then we have someone send them to Andrew. Or we have a quilt or
a git tree that Andrew knows he can pull from.

But we do need lots of review so distribution to Andrew and the other
kernel developers as plain patches appears to be the healthy choice.
I'm going to go bury my head in the sand and finish my OLS paper now.

Eric

Subject: Re: [RFC] Virtualization steps
Posted by Bill Davidsen on Wed, 29 Mar 2006 20:56:38 GMT
View Forum Message <> Reply to Message

Sam Vilain wrote:
> On Tue, 2006-03-28 at 09:41 -0500, Bill Davidsen wrote:
>>> It is more than realistic. Hosting companies run more than 100 VPSs in
>>> reality. There are also other usefull scenarios. For example, I know
>>> the universities which run VPS for every faculty web site, for every
>>> department, mail server and so on. Why do you think they want to run
>>> only 5VMs on one machine? Much more!
>> I made no commont on what "they" might want, I want to make the rack of
>> underutilized Windows, BSD and Solaris servers go away. An approach
>> which doesn't support unmodified guest installs doesn't solve any of my
>> current problems. I didn't say it was in any way not useful, just not of

>> interest to me. What needs I have for Linux environments are answered by
>> jails and/or UML.
>
> We are talking about adding jail technology, also known as containers on
> Solaris and vserver/openvz on Linux, to the mainline kernel.
>
> So, you are obviously interested!
>
> Because of course, you can take an unmodified filesystem of the guest
> and assuming the kernels are compatible run them without changes. I
> find this consolidation approach indispensible.
>
The only way to assume kernels are compatible is to run the same distro.
Because vendor kernels are sure not compatible, even running a
kernel.org kernel on Fedora (for instance) reveals the the utilities are
also tweaked to expect the kernel changes, and you wind up with a system
which feels like wearing someone else's hat. It's stable but little
things just don't work right.

--
    -bill davidsen (davidsen@tmr.com)
"The secret to procrastination is to put things off until the
  last possible moment - but no longer" -me

Subject: Re: [RFC] Virtualization steps
Posted by Bill Davidsen on Wed, 29 Mar 2006 21:37:47 GMT
View Forum Message <> Reply to Message

Herbert Poetzl wrote:

>>> Summary of previous discussions on LKML
>>> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
>> Have their been any discussions between the groups pushing this
>> virtualization, and ...
>
> yes, the discussions are ongoing ... maybe to clarify the
> situation for the folks not involved (projects in
> alphabetical order):
>
Thank you! Nice to have a scorecard.
--
    -bill davidsen (davidsen@tmr.com)
"The secret to procrastination is to put things off until the
  last possible moment - but no longer" -me

Subject: Re: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Wed, 29 Mar 2006 21:37:52 GMT
View Forum Message <> Reply to Message

On Wed, 2006-03-29 at 18:47 +0400, Kirill Korotaev wrote:
> >> I wonder what is the value of it if it doesn't do guarantees or QoS?
> >> In our experiments with it we failed to observe any fairness.
> >
> > probably a misconfiguration on your side ...
> maybe you can provide some instructions on which kernel version to use
> and how to setup the following scenario:
> 2CPU box. 3 VPSs which should run with 1:2:3 ratio of CPU usage.

Ok, I'll call those three VPSes fast, faster and fastest.

"fast"   : fill rate 1, interval 3
"faster" : fill rate 2, interval 3
"fastest" : fill rate 3, interval 3

That all adds up to a fill rate of 6 with an interval of 3, but that is
right because with two processors you have 2 tokens to allocate per
jiffie. Also set the bucket size to something of the order of HZ.

You can watch the processes within each vserver's priority jump up and
down with `vtop' during testing. Also you should be able to watch the
vserver's bucket fill and empty in /proc/virtual/XXX/sched (IIRC)

> > well, do you have numbers?
> just run the above scenario with one busy loop inside each VPS. I was
> not able to observe 1:2:3 cpu distribution. Other scenarios also didn't
> showed my any fairness. The results were different. Sometimes 1:1:2,
> sometimes others.

I mentioned this earlier, but for the sake of the archives I'll repeat -
if you are running with any of the buckets on empty, the scheduler is
imbalanced and therefore not going to provide the exact distribution you
asked for.

However with a single busy loop in each vserver I'd expect the above to
yield roughly 100% for fastest, 66% for faster and 33% for fast, within
5 seconds or so of starting those processes (assuming you set a bucket
size of HZ).

Sam.

Subject: Re: [RFC] Virtualization steps
Posted by Sam Vilain on Wed, 29 Mar 2006 22:44:47 GMT

Dave Hansen wrote:

>On Tue, 2006-03-28 at 12:51 +0400, Kirill Korotaev wrote:
>
>
>>Eric, we have a GIT repo on openvz.org already:
>>http://git.openvz.org
>>
>>
>
>Git is great for getting patches and lots of updates out, but I'm not
>sure it is idea for what we're trying to do. We'll need things reviewed
>at each step, especially because we're going to be touching so much
>common code.
>
>I'd guess set of quilt (or patch-utils) patches is probably best,
>especially if we're trying to get stuff into -mm first.
>
>

The apparent problem is that the git commit history on a branch cannot be unwound. However, that is fine - just make another branch and put your new sequence of commits there.

Tools exist that allow you to wind and unwind the commit history arbitrarily to revise patches before they are published on a branch that you don't want to just delete. For instance:

stacked git

  http://www.procode.org/stgit/

or patchy git

   http://www.spearce.org/2006/02/pg-version-0111-released.html

are examples of such tools.

I recommend starting with stacked git, it really is nice.

Sam.

Subject: Re: [RFC] Virtualization steps
Posted by dev on Thu, 30 Mar 2006 13:51:36 GMT

ok. This is also easier for us, as it is a usual way of doing things in
OpenVZ. Will see...

> On Tue, 2006-03-28 at 12:51 +0400, Kirill Korotaev wrote:
>> Eric, we have a GIT repo on openvz.org already:
>> http://git.openvz.org
>
> Git is great for getting patches and lots of updates out, but I'm not
> sure it is idea for what we're trying to do. We'll need things reviewed
> at each step, especially because we're going to be touching so much
> common code.
>
> I'd guess set of quilt (or patch-utils) patches is probably best,
> especially if we're trying to get stuff into -mm first.
>
> -- Dave
>
>

Subject: Re: Re: [RFC] Virtualization steps
Posted by dev on Wed, 12 Apr 2006 08:22:11 GMT
View Forum Message <> Reply to Message

Sam,

> Ok, I'll call those three VPSes fast, faster and fastest.
>
> "fast"   : fill rate 1, interval 3
> "faster" : fill rate 2, interval 3
> "fastest" : fill rate 3, interval 3
>
> That all adds up to a fill rate of 6 with an interval of 3, but that is
> right because with two processors you have 2 tokens to allocate per
> jiffie. Also set the bucket size to something of the order of HZ.
>
> You can watch the processes within each vserver's priority jump up and
> down with `vtop' during testing. Also you should be able to watch the
> vserver's bucket fill and empty in /proc/virtual/XXX/sched (IIRC)
>
> I mentioned this earlier, but for the sake of the archives I'll repeat -
> if you are running with any of the buckets on empty, the scheduler is
> imbalanced and therefore not going to provide the exact distribution you
> asked for.
>
> However with a single busy loop in each vserver I'd expect the above to
> yield roughly 100% for fastest, 66% for faster and 33% for fast, within
> 5 seconds or so of starting those processes (assuming you set a bucket

> size of HZ).

Sam, what we observe is the situation, when Linux cpu scheduler spreads
2 tasks on 1st CPU and 1 task on the 2nd CPU. Std linux scheduler
doesn't do any rebalancing after that, so no plays with tokens make the
spread to be 3:2:1, since the lowest priority process gets a full 2nd
CPU (100% instead of 33% of CPU).

Where is my mistake? Can you provide a configuration where we could test
or the instuctions on how to avoid this?

Thanks,
Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Thu, 13 Apr 2006 01:05:06 GMT
View Forum Message <> Reply to Message

On Wed, Apr 12, 2006 at 12:28:56PM +0400, Kirill Korotaev wrote:
> Sam,
>
> >Ok, I'll call those three VPSes fast, faster and fastest.
> >
> >"fast"   : fill rate 1, interval 3
> >"faster" : fill rate 2, interval 3
> >"fastest" : fill rate 3, interval 3
> >
> >That all adds up to a fill rate of 6 with an interval of 3, but that is
> >right because with two processors you have 2 tokens to allocate per
> >jiffie. Also set the bucket size to something of the order of HZ.
> >
> >You can watch the processes within each vserver's priority jump up and
> >down with `vtop' during testing. Also you should be able to watch the
> >vserver's bucket fill and empty in /proc/virtual/XXX/sched (IIRC)
> >
> >I mentioned this earlier, but for the sake of the archives I'll repeat -
> >if you are running with any of the buckets on empty, the scheduler is
> >imbalanced and therefore not going to provide the exact distribution you
> >asked for.
> >
> >However with a single busy loop in each vserver I'd expect the above to
> >yield roughly 100% for fastest, 66% for faster and 33% for fast, within
> >5 seconds or so of starting those processes (assuming you set a bucket
> >size of HZ).
>
> Sam, what we observe is the situation, when Linux cpu scheduler spreads
> 2 tasks on 1st CPU and 1 task on the 2nd CPU. Std linux scheduler

> doesn't do any rebalancing after that, so no plays with tokens make the
> spread to be 3:2:1, since the lowest priority process gets a full 2nd
> CPU (100% instead of 33% of CPU).
>
> Where is my mistake? Can you provide a configuration where we could test
> or the instuctions on how to avoid this?

well, your mistake seems to be that you probably haven't
tested this yet, because with the following (simple)
setups I seem to get what you consider impossible
(of course, not as precise as your scheduler does it)

```
vcontext --create --xid 100 ./cpuhog -n 1 100 &
vcontext --create --xid 200 ./cpuhog -n 1 200 &
vcontext --create --xid 300 ./cpuhog -n 1 300 &

vsched --xid 100 --fill-rate 1 --interval 6
vsched --xid 200 --fill-rate 2 --interval 6
vsched --xid 300 --fill-rate 3 --interval 6

vattribute --xid 100 --flag sched_hard
vattribute --xid 200 --flag sched_hard
vattribute --xid 300 --flag sched_hard


  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
   39 root      25   0  1304  248  200 R   74  0.1   0:46.16 ./cpuhog -n 1 300
   38 root      25   0  1308  252  200 H   53  0.1   0:34.06 ./cpuhog -n 1 200
   37 root      25   0  1308  252  200 H   28  0.1   0:19.53 ./cpuhog -n 1 100
   46 root       0   0  1804  912  736 R    1  0.4   0:02.14 top -cid 20
```

and here the other way round:

```
vsched --xid 100 --fill-rate 3 --interval 6
vsched --xid 200 --fill-rate 2 --interval 6
vsched --xid 300 --fill-rate 1 --interval 6

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
   36 root      25   0  1304  248  200 R   75  0.1   0:58.41 ./cpuhog -n 1 100
   37 root      25   0  1308  252  200 H   54  0.1   0:42.77 ./cpuhog -n 1 200
   38 root      25   0  1308  252  200 R   29  0.1   0:25.30 ./cpuhog -n 1 300
   45 root       0   0  1804  912  736 R    1  0.4   0:02.26 top -cid 20
```

note that this was done on a virtual dual cpu
machine (QEMU 8.0) with 2.6.16-vs2.1.1-rc16 and
that there were roughly 25% idle time, which I'm

unable to explain atm ...

feel free to jump on that fact, but I consider
it unimportant for now ...

best,
Herbert

> Thanks,
> Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by Kirill Korotaev on Thu, 13 Apr 2006 06:45:22 GMT
View Forum Message <> Reply to Message

Herbert,

Thanks a lot for the details, I will give it a try once again. Looks
like fairness in this scenario simply requires sched_hard settings.

Herbert... I don't know why you've decided that my goal is to prove that
your scheduler is bad or not precise. My goal is simply to investigate
different approaches and make some measurements. I suppose you can
benefit from such a volunteer, don't you think so? Anyway, thanks again
and don't be cycled on the idea that OpenVZ are so cruel bad guys :)

Thanks,
Kirill

> well, your mistake seems to be that you probably haven't
> tested this yet, because with the following (simple)
> setups I seem to get what you consider impossible
> (of course, not as precise as your scheduler does it)
>
>
> vcontext --create --xid 100 ./cpuhog -n 1 100 &
> vcontext --create --xid 200 ./cpuhog -n 1 200 &
> vcontext --create --xid 300 ./cpuhog -n 1 300 &
>
> vsched --xid 100 --fill-rate 1 --interval 6
> vsched --xid 200 --fill-rate 2 --interval 6
> vsched --xid 300 --fill-rate 3 --interval 6
>
> vattribute --xid 100 --flag sched_hard
> vattribute --xid 200 --flag sched_hard
> vattribute --xid 300 --flag sched_hard
>

>
> PID USER   PR NI VIRT RES SHR S %CPU %MEM   TIME+ COMMAND
>   39 root   25   0 1304 248 200 R   74 0.1   0:46.16 ./cpuhog -n 1 300
>   38 root   25   0 1308 252 200 H   53 0.1   0:34.06 ./cpuhog -n 1 200
>   37 root   25   0 1308 252 200 H   28 0.1   0:19.53 ./cpuhog -n 1 100
>   46 root   0   0 1804 912 736 R   1 0.4   0:02.14 top -cid 20
>
> and here the other way round:
>
> vsched --xid 100 --fill-rate 3 --interval 6
> vsched --xid 200 --fill-rate 2 --interval 6
> vsched --xid 300 --fill-rate 1 --interval 6
>
> PID USER   PR NI VIRT RES SHR S %CPU %MEM   TIME+ COMMAND
>   36 root   25   0 1304 248 200 R   75 0.1   0:58.41 ./cpuhog -n 1 100
>   37 root   25   0 1308 252 200 H   54 0.1   0:42.77 ./cpuhog -n 1 200
>   38 root   25   0 1308 252 200 R   29 0.1   0:25.30 ./cpuhog -n 1 300
>   45 root   0   0 1804 912 736 R   1 0.4   0:02.26 top -cid 20
>
>
> note that this was done on a virtual dual cpu
> machine (QEMU 8.0) with 2.6.16-vs2.1.1-rc16 and
> that there were roughly 25% idle time, which I'm
> unable to explain atm ...
>
> feel free to jump on that fact, but I consider
> it unimportant for now ...
>
> best,
> Herbert
>
>
>>Thanks,
>>Kirill
>
>

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Thu, 13 Apr 2006 13:42:39 GMT
View Forum Message <> Reply to Message

On Thu, Apr 13, 2006 at 10:52:19AM +0400, Kirill Korotaev wrote:
> Herbert,
>
> Thanks a lot for the details, I will give it a try once again. Looks
> like fairness in this scenario simply requires sched_hard settings.

hmm, not precisely, it's a cpu limit you described
and that is what this configuration does, for fair
scheduling you need to activate the indle skip and
configure it in a similar way ...

> Herbert... I don't know why you've decided that my goal is to prove
> that your scheduler is bad or not precise. My goal is simply to
> investigate different approaches and make some measurements.

fair enough ...

> I suppose you can benefit from such a volunteer, don't you think so?

well, if the 'results' and 'methods' will be made
public, I can, until now all I got was something
along the lines:

 "Linux-VServer is not stable! WE (swsoft?) have
  a secret but essential test suite running two
  weeks to confirm that OUR kernels ARE stable,
  and Linux-VServer will never pass those tests,
  but of course, we can't tell you what kind of
  tests or what results we got"

which doesn't help me anything and which, to be
honest, does not sound very friendly either ...

> Anyway, thanks again and don't be cycled on the idea that OpenVZ are
> so cruel bad guys :)

but what about the Virtuozzo(tm) guys? :)
I'm really trying not to generalize here ...

best,
Herbert

> Thanks,
> Kirill
>
> >well, your mistake seems to be that you probably haven't
> >tested this yet, because with the following (simple)
> >setups I seem to get what you consider impossible
> >(of course, not as precise as your scheduler does it)
> >
> >
> >vcontext --create --xid 100 ./cpuhog -n 1 100 &
> >vcontext --create --xid 200 ./cpuhog -n 1 200 &
> >vcontext --create --xid 300 ./cpuhog -n 1 300 &

> >
> >vsched --xid 100 --fill-rate 1 --interval 6
> >vsched --xid 200 --fill-rate 2 --interval 6
> >vsched --xid 300 --fill-rate 3 --interval 6
> >
> >vattribute --xid 100 --flag sched_hard
> >vattribute --xid 200 --flag sched_hard
> >vattribute --xid 300 --flag sched_hard
> >
> >
> > PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
> > 39 root 25 0 1304 248 200 R 74 0.1 0:46.16 ./cpuhog -n 1
> > 300 38 root 25 0 1308 252 200 H 53 0.1 0:34.06 ./cpuhog
> > -n 1 200 37 root 25 0 1308 252 200 H 28 0.1 0:19.53
> > ./cpuhog -n 1 100 46 root 0 0 1804 912 736 R 1 0.4
> > 0:02.14 top -cid 20
> >and here the other way round:
> >
> >vsched --xid 100 --fill-rate 3 --interval 6
> >vsched --xid 200 --fill-rate 2 --interval 6
> >vsched --xid 300 --fill-rate 1 --interval 6
> >
> > PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
> > 36 root 25 0 1304 248 200 R 75 0.1 0:58.41 ./cpuhog -n 1
> > 100 37 root 25 0 1308 252 200 H 54 0.1 0:42.77 ./cpuhog
> > -n 1 200 38 root 25 0 1308 252 200 R 29 0.1 0:25.30
> > ./cpuhog -n 1 300 45 root 0 0 1804 912 736 R 1 0.4
> > 0:02.26 top -cid 20
> >
> >note that this was done on a virtual dual cpu
> >machine (QEMU 8.0) with 2.6.16-vs2.1.1-rc16 and
> >that there were roughly 25% idle time, which I'm
> >unable to explain atm ...
> >
> >feel free to jump on that fact, but I consider
> >it unimportant for now ...
> >
> >best,
> >Herbert
> >
> >
> >>Thanks,
> >>Kirill
> >
> >
>

Subject: Re: Re: [RFC] Virtualization steps
Posted by Cedric Le Goater on Thu, 13 Apr 2006 21:33:13 GMT
View Forum Message <> Reply to Message

Herbert Poetzl wrote:

> well, if the 'results' and 'methods' will be made
> public, I can, until now all I got was something
> along the lines:
>
> "Linux-VServer is not stable! WE (swsoft?) have
>   a secret but essential test suite running two
>   weeks to confirm that OUR kernels ARE stable,
>   and Linux-VServer will never pass those tests,
>   but of course, we can't tell you what kind of
>   tests or what results we got"
>
> which doesn't help me anything and which, to be
> honest, does not sound very friendly either ...

Recently, we've been running tests and benchmarks in different
virtualization environments : openvz, vserver, vserver in a minimal context
and also Xen as a reference in the virtual machine world.

We ran the usual benchmarks, dbench, tbench, lmbench, kernerl build, on the
native kernel, on the patched kernel and in each virtualized environment.
We also did some scalability tests to see how each solution behaved. And
finally, some tests on live migration. We didn't do much on network nor on
resource management behavior.

We'd like to continue in an open way. But first, we want to make sure we
have the right tests, benchmarks, tools, versions, configuration, tuning,
etc, before publishing any results :) We have some materials already but
before proposing we would like to have your comments and advices on what we
should or shouldn't use.

Thanks for doing such a great job on lightweight containers,

C.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Thu, 13 Apr 2006 22:45:33 GMT
View Forum Message <> Reply to Message

On Thu, Apr 13, 2006 at 11:33:13PM +0200, Cedric Le Goater wrote:
> Herbert Poetzl wrote:
>

> > well, if the 'results' and 'methods' will be made
> > public, I can, until now all I got was something
> > along the lines:
> >
> > "Linux-VServer is not stable! WE (swsoft?) have
> >   a secret but essential test suite running two
> >   weeks to confirm that OUR kernels ARE stable,
> >   and Linux-VServer will never pass those tests,
> >   but of course, we can't tell you what kind of
> >   tests or what results we got"
> >
> > which doesn't help me anything and which, to be
> > honest, does not sound very friendly either ...
>
> Recently, we've been running tests and benchmarks in different
> virtualization environments : openvz, vserver, vserver in a minimal
> context and also Xen as a reference in the virtual machine world.
>
> We ran the usual benchmarks, dbench, tbench, lmbench, kernerl build,
> on the native kernel, on the patched kernel and in each virtualized
> environment. We also did some scalability tests to see how each
> solution behaved. And finally, some tests on live migration. We didn't
> do much on network nor on resource management behavior.

I would be really interested in getting comparisons
between vanilla kernels and linux-vserver patched
versions, especially vs2.1.1 and vs2.0.2 on the
same test setup with a minimum difference in config

I doubt that you can really compare across the
existing virtualization technologies, as it really
depends on the setup and hardware

> We'd like to continue in an open way. But first, we want to make sure
> we have the right tests, benchmarks, tools, versions, configuration,
> tuning, etc, before publishing any results :) We have some materials
> already but before proposing we would like to have your comments and
> advices on what we should or shouldn't use.

In my experience it is extremely hard to do 'proper'
comparisons, because the slightest change of the
environment can cause big differences ...

here as example, a kernel build (-j99) on 2.6.16
on a test host, with and without a chroot:

without:

451.03user 26.27system 2:00.38elapsed 396%CPU
449.39user 26.21system 1:59.95elapsed 396%CPU
447.40user 25.86system 1:59.79elapsed 395%CPU

now with:

490.77user 24.45system 2:13.35elapsed 386%CPU
489.69user 24.50system 2:12.60elapsed 387%CPU
490.41user 24.99system 2:12.22elapsed 389%CPU

now is chroot() that imperformant? no, but the change
in /tmp being on a partition vs. tmpfs makes quite
some difference here

even moving from one partition to another will give
measurable difference here, all within a small margin

an interesting aspect is the gain (or loss) you have
when you start several guests basically doing the
same thing (and sharing the same files, etc)

> Thanks for doing such a great job on lightweight containers,

you're welcome!

best,
Herbert

> C.

Subject: Re: Re: [RFC] Virtualization steps
Posted by kir on Thu, 13 Apr 2006 22:51:32 GMT
View Forum Message <> Reply to Message

Cedric Le Goater wrote:

> Recently, we've been running tests and benchmarks in different
>
>virtualization environments : openvz, vserver, vserver in a minimal context
>and also Xen as a reference in the virtual machine world.
>
>We ran the usual benchmarks, dbench, tbench, lmbench, kernerl build, on the
>native kernel, on the patched kernel and in each virtualized environment.
>We also did some scalability tests to see how each solution behaved. And
>finally, some tests on live migration. We didn't do much on network nor on
>resource management behavior.
>

>We'd like to continue in an open way. But first, we want to make sure we
>have the right tests, benchmarks, tools, versions, configuration, tuning,
>etc, before publishing any results :) We have some materials already but
>before proposing we would like to have your comments and advices on what we
>should or shouldn't use.
>
>Thanks for doing such a great job on lightweight containers,
>
>C.
>
>
Cedrik,

You made my day, I am really happy to hear that! Such testing and benchmarking should be done by an independent third party, and IBM fits that requirement just fine. It all makes much sense for everybody who's involved.

If it will be opened (not just results, but also the processes and tools), and all the projects will be able to contribute and help, that would be just great. We do a lot of testing in-house, and will be happy to contribute to such an independent testing/benchmarking project.

Speaking of live migration, we in OpenVZ plan to release our implementation as soon as next week.

Regards,
Kir.

Subject: Re: Re: [RFC] Virtualization steps
Posted by dev on Fri, 14 Apr 2006 07:35:03 GMT
View Forum Message <> Reply to Message

> I would be really interested in getting comparisons
> between vanilla kernels and linux-vserver patched
> versions, especially vs2.1.1 and vs2.0.2 on the
> same test setup with a minimum difference in config
>
> I doubt that you can really compare across the
> existing virtualization technologies, as it really
> depends on the setup and hardware
and kernel .config's :)
for example, I'm pretty sure, OVZ smp kernel is not the same as any of prebuilt vserver kernels.

> In my experience it is extremely hard to do 'proper'
> comparisons, because the slightest change of the

> environment can cause big differences ...
>
> here as example, a kernel build (-j99) on 2.6.16
> on a test host, with and without a chroot:
>
> without:
>
> 451.03user 26.27system 2:00.38elapsed 396%CPU
> 449.39user 26.21system 1:59.95elapsed 396%CPU
> 447.40user 25.86system 1:59.79elapsed 395%CPU
>
> now with:
>
> 490.77user 24.45system 2:13.35elapsed 386%CPU
> 489.69user 24.50system 2:12.60elapsed 387%CPU
> 490.41user 24.99system 2:12.22elapsed 389%CPU
>
> now is chroot() that imperformant? no, but the change
> in /tmp being on a partition vs. tmpfs makes quite
> some difference here

filesystem performance also very much depends on disk layout.
If you use different partitions of the same disk for Xen, vserver and OVZ,
one of them will be quickest while others can be significantly slower
and slower :/

Thanks,
Kirill

Subject: Re: Re: [RFC] Virtualization steps
Posted by Cedric Le Goater on Fri, 14 Apr 2006 09:56:21 GMT
View Forum Message <> Reply to Message

Bonjour !

Herbert Poetzl wrote:

> I would be really interested in getting comparisons
> between vanilla kernels and linux-vserver patched
> versions, especially vs2.1.1 and vs2.0.2 on the
> same test setup with a minimum difference in config

We did the tests last month and used the stable version : vs2.0.2rc9 on a
2.6.15.4. Using benchmarks like dbench, tbench, lmbench, the vserver patch
has no impact, vserver overhead in a context is hardly measurable (<3%),
same results for a debian sarge running in a vserver.

It is pretty difficult to follow everyone patches. This makes the

comparisons difficult so we chose to normalize all the results with the native kernel results. But in a way, this is good because the goal of these tests isn't to compare technologies but to measure their overhead and stability. And at the end, we don't care if openvz is faster than vserver, we want containers in the linux kernel to be fast and stable, one day :)

> I doubt that you can really compare across the
> existing virtualization technologies, as it really
> depends on the setup and hardware

I agree these are very different technologies but from a user point of view, they provide a similar service. So, it is interesting to see what are the drawbacks and the benefits of each solution. You want fault containment and strict isolation, here's the price. You want performance, here's another.

Anyway, there's already enough focus on the virtual machines so we should focus only on lightweight containers.

>> We'd like to continue in an open way. But first, we want to make sure
>> we have the right tests, benchmarks, tools, versions, configuration,
>> tuning, etc, before publishing any results :) We have some materials
>> already but before proposing we would like to have your comments and
>> advices on what we should or shouldn't use.
>
> In my experience it is extremely hard to do 'proper'
> comparisons, because the slightest change of the
> environment can cause big differences ...
>
> here as example, a kernel build (-j99) on 2.6.16
> on a test host, with and without a chroot:
>
> without:
>
> 451.03user 26.27system 2:00.38elapsed 396%CPU
> 449.39user 26.21system 1:59.95elapsed 396%CPU
> 447.40user 25.86system 1:59.79elapsed 395%CPU
>
> now with:
>
> 490.77user 24.45system 2:13.35elapsed 386%CPU
> 489.69user 24.50system 2:12.60elapsed 387%CPU
> 490.41user 24.99system 2:12.22elapsed 389%CPU
>
> now is chroot() that imperformant? no, but the change
> in /tmp being on a partition vs. tmpfs makes quite
> some difference here
>
> even moving from one partition to another will give

> measurable difference here, all within a small margin

very interesting thanks.

> an interesting aspect is the gain (or loss) you have
> when you start several guests basically doing the
> same thing (and sharing the same files, etc)

we have these in the pipe also, we called them scalability test: trying to
run as much containers as possible and see how performance drops (when the
kernel survives the test :)

ok, now i guess we want to make some kind of test plan.

C.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Cedric Le Goater on Fri, 14 Apr 2006 10:08:05 GMT
View Forum Message <> Reply to Message

Bonjour !

Kir Kolyshkin wrote:

> You made my day, I am really happy to hear that! Such testing and
> benchmarking should be done by an independent third party, and IBM fits
> that requirement just fine. It all makes much sense for everybody who's
> involved.
>
> If it will be opened (not just results, but also the processes and
> tools), and all the projects will be able to contribute and help, that
> would be just great. We do a lot of testing in-house, and will be happy
> to contribute to such an independent testing/benchmarking project.

What we have in mind is something like http://test.kernel.org/ for each
patch set. I guess we will start humbly at the beginning :)

Initially, the idea was to test the patch series we've been sending on
lkml. But as we've been running tests on existing solutions, openvz,
vserver, and our own prototype, we thought that extending to all was
interesting and fair.

The goal is to promote lightweight containers in the linux kernel, so this
needs to be open.

> Speaking of live migration, we in OpenVZ plan to release our
> implementation as soon as next week.

We've been working on that topic for a long time, we are very interested in
seeing what you've acheived ! Migration tests is also an interesting topic
we could add with time to the containers tests.

thanks,

C.

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Sat, 15 Apr 2006 19:29:11 GMT
View Forum Message <> Reply to Message

On Fri, Apr 14, 2006 at 11:56:21AM +0200, Cedric Le Goater wrote:
> Bonjour !
>
> Herbert Poetzl wrote:
>
> > I would be really interested in getting comparisons
> > between vanilla kernels and linux-vserver patched
> > versions, especially vs2.1.1 and vs2.0.2 on the
> > same test setup with a minimum difference in config
>
> We did the tests last month and used the stable version : vs2.0.2rc9
> on a 2.6.15.4. Using benchmarks like dbench, tbench, lmbench, the
> vserver patch has no impact, vserver overhead in a context is hardly
> measurable (<3%), same results for a debian sarge running in a
> vserver.

with 2.1.1-rc16 they are not supposed to be measurable
at all, so if you measure any difference here, please
let me know about it, as I consider it an issue :)

> It is pretty difficult to follow everyone patches. This makes the
> comparisons difficult so we chose to normalize all the results with
> the native kernel results. But in a way, this is good because the goal
> of these tests isn't to compare technologies but to measure their
> overhead and stability. And at the end, we don't care if openvz is
> faster than vserver, we want containers in the linux kernel to be fast
> and stable, one day :)

I'm completely with you here ...

> > I doubt that you can really compare across the
> > existing virtualization technologies, as it really
> > depends on the setup and hardware
>

> I agree these are very different technologies but from a user point
> of view, they provide a similar service. So, it is interesting to see
> what are the drawbacks and the benefits of each solution. You want
> fault containment and strict isolation, here's the price. You want
> performance, here's another.

precisely, taht's why there are different projects
and different aims ...

> Anyway, there's already enough focus on the virtual machines so we
> should focus only on lightweight containers.
>
> >> We'd like to continue in an open way. But first, we want to
> >> make sure we have the right tests, benchmarks, tools, versions,
> >> configuration, tuning, etc, before publishing any results :) We
> >> have some materials already but before proposing we would like to
> >> have your comments and advices on what we should or shouldn't use.
> > >
> > > In my experience it is extremely hard to do 'proper'
> > > comparisons, because the slightest change of the
> > > environment can cause big differences ...
> > >
> > > here as example, a kernel build (-j99) on 2.6.16
> > > on a test host, with and without a chroot:
> > >
> > > without:
> > >
> > > 451.03user 26.27system 2:00.38elapsed 396%CPU
> > > 449.39user 26.21system 1:59.95elapsed 396%CPU
> > > 447.40user 25.86system 1:59.79elapsed 395%CPU
> > >
> > > now with:
> > >
> > > 490.77user 24.45system 2:13.35elapsed 386%CPU
> > > 489.69user 24.50system 2:12.60elapsed 387%CPU
> > > 490.41user 24.99system 2:12.22elapsed 389%CPU
> > >
> > > now is chroot() that imperformant? no, but the change
> > > in /tmp being on a partition vs. tmpfs makes quite
> > > some difference here
> > >
> > > even moving from one partition to another will give
> > > measurable difference here, all within a small margin
>
> very interesting thanks.
>
> > an interesting aspect is the gain (or loss) you have
> > when you start several guests basically doing the

> > same thing (and sharing the same files, etc)
>
> we have these in the pipe also, we called them scalability test:
> trying to run as much containers as possible and see how performance
> drops (when the kernel survives the test :)

yes, might want to check with and without unification
here too, as I think you can reach more than 100% native
speed in the multi guest scenario with that :)

> ok, now i guess we want to make some kind of test plan.

sounds good, please keep me posted ...

best,
Herbert

> C.
> -
> To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
> Please read the FAQ at http://www.tux.org/lkml/

Subject: Re: Re: [RFC] Virtualization steps
Posted by Herbert Poetzl on Sat, 15 Apr 2006 19:31:34 GMT
View Forum Message <> Reply to Message

On Fri, Apr 14, 2006 at 12:08:05PM +0200, Cedric Le Goater wrote:
> Bonjour !
>
> Kir Kolyshkin wrote:
>
> > You made my day, I am really happy to hear that! Such testing and
> > benchmarking should be done by an independent third party, and
> > IBM fits that requirement just fine. It all makes much sense for
> > everybody who's involved.
> >
> > If it will be opened (not just results, but also the processes and
> > tools), and all the projects will be able to contribute and help,
> > that would be just great. We do a lot of testing in-house, and will
> > be happy to contribute to such an independent testing/benchmarking
> > project.
>
> What we have in mind is something like http://test.kernel.org/ for
> each patch set. I guess we will start humbly at the beginning :)
>

> Initially, the idea was to test the patch series we've been sending on
> lkml. But as we've been running tests on existing solutions, openvz,
> vserver, and our own prototype, we thought that extending to all was
> interesting and fair.

would be really great if you could extend that to something
like the PLM where folks (like linux-vserver and openvz) can
test their patches against mainline kernels in a fairly
automated way ...

I guess that would be some initial work, but could improve
many other patches (not only those related to virtualization)

best,
Herbert

> The goal is to promote lightweight containers in the linux kernel, so
> this needs to be open.
>
> > Speaking of live migration, we in OpenVZ plan to release our
> > implementation as soon as next week.
>
> We've been working on that topic for a long time, we are very
> interested in seeing what you've acheived ! Migration tests is also an
> interesting topic we could add with time to the containers tests.
>
> thanks,
>
> C.
> -
> To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
> Please read the FAQ at http://www.tux.org/lkml/