Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Andrew Morton on Fri, 18 Aug 2006 16:42:48 GMT
View Forum Message <> Reply to Message

On Fri, 18 Aug 2006 07:45:48 -0700
Dave Hansen <haveblue@us.ibm.com> wrote:

> On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> >
> > A) Have separate memory management for each container,
> >   with separate buddy allocator, lru lists, page replacement mechanism.
> >   That implies a considerable overhead, and the main challenge there
> >   is sharing of pages between these separate memory managers.
>
> Hold on here for just a sec...
>
> It is quite possible to do memory management aimed at one container
> while that container's memory still participates in the main VM.
>
> There is overhead here, as the LRU scanning mechanisms get less
> efficient, but I'd rather pay a penalty at LRU scanning time than divide
> up the VM, or coarsely start failing allocations.
>

I have this mad idea that you can divide a 128GB machine up into 256 fake
NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
So that 4.5GB job would be placed within an exclusive cpuset which has nine
"mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
no kernel changes needed.

Unfortunately this is not testable because numa=fake=256 doesn't come even
vaguely close to working. Am trying to get that fixed.

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Dave Hansen on Fri, 18 Aug 2006 17:29:16 GMT
View Forum Message <> Reply to Message

On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> I have this mad idea that you can divide a 128GB machine up into 256 fake
> NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> So that 4.5GB job would be placed within an exclusive cpuset which has nine
> "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> no kernel changes needed.

Is this similar to Mel Gorman's zone-based anti-fragmentation approach?
I thought he was discouraged from pursuing that at the VM summit.

-- Dave

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Andrew Morton on Fri, 18 Aug 2006 17:38:20 GMT
View Forum Message <> Reply to Message

On Fri, 18 Aug 2006 10:29:16 -0700
Dave Hansen <haveblue@us.ibm.com> wrote:

> On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> > I have this mad idea that you can divide a 128GB machine up into 256 fake
> > NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> > So that 4.5GB job would be placed within an exclusive cpuset which has nine
> > "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> > no kernel changes needed.
>
> Is this similar to Mel Gorman's zone-based anti-fragmentation approach?

I don't think so - it's using zones, but for a quite different thing.

> I thought he was discouraged from pursuing that at the VM summit.

That seemed to a be a 49%/51% call, iirc.

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Rohit Seth on Fri, 18 Aug 2006 17:59:06 GMT
View Forum Message <> Reply to Message

On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> On Fri, 18 Aug 2006 07:45:48 -0700
> Dave Hansen <haveblue@us.ibm.com> wrote:
>
> > On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> > >
> > > A) Have separate memory management for each container,
> > >   with separate buddy allocator, lru lists, page replacement mechanism.
> > >   That implies a considerable overhead, and the main challenge there
> > >   is sharing of pages between these separate memory managers.
> >
> > Hold on here for just a sec...
> >
> > It is quite possible to do memory management aimed at one container
> > while that container's memory still participates in the main VM.
> >
> > There is overhead here, as the LRU scanning mechanisms get less

> > efficient, but I'd rather pay a penalty at LRU scanning time than divide
> > up the VM, or coarsely start failing allocations.
> >
>
> I have this mad idea that you can divide a 128GB machine up into 256 fake
> NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> So that 4.5GB job would be placed within an exclusive cpuset which has nine
> "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> no kernel changes needed.
>
Sounds like an interesting idea. Will have to depend on something like
memory hot-plug to get the things move around...

-rohit

> Unfortunately this is not testable because numa=fake=256 doesn't come even
> vaguely close to working. Am trying to get that fixed.

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Paul Jackson on Fri, 18 Aug 2006 18:09:19 GMT
View Forum Message <> Reply to Message

Andrew wrote:
> "mems" (what are these called?)

I call them "Memory Nodes", or "nodes" for short when the qualifier
"memory" is clear from the context.

I was just reading up on FB-DIMM memory, and notice that each DIMM on
a channel has a different latency. That sure looks like the defining
characteristic of NUMA memory to my way of thinking - non-uniform memory.

So for extra credit if your fake numa nodes pan out, it would be cute if
the fake nodes could be defined so as to respect these latency
differences - memory in different nodes if at different positions along
a memory channel. Then a sysadmin could put their most latency
sensitive jobs on the DIMMs closest to the CPUs.

--

I won't rest till it's the best ...
Programmer, Linux Scalability
Paul Jackson <pj@sgi.com> 1.925.600.0401

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Chandra Seetharaman on Fri, 18 Aug 2006 18:17:45 GMT

On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> On Fri, 18 Aug 2006 07:45:48 -0700
> Dave Hansen <haveblue@us.ibm.com> wrote:
>
> > On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> > >
> > > A) Have separate memory management for each container,
> > >   with separate buddy allocator, lru lists, page replacement mechanism.
> > >   That implies a considerable overhead, and the main challenge there
> > >   is sharing of pages between these separate memory managers.
> >
> > Hold on here for just a sec...
> >
> > It is quite possible to do memory management aimed at one container
> > while that container's memory still participates in the main VM.
> >
> > There is overhead here, as the LRU scanning mechanisms get less
> > efficient, but I'd rather pay a penalty at LRU scanning time than divide
> > up the VM, or coarsely start failing allocations.
> >
>
> I have this mad idea that you can divide a 128GB machine up into 256 fake
> NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> So that 4.5GB job would be placed within an exclusive cpuset which has nine
> "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> no kernel changes needed.

In this model memory and container are tightly coupled, hence memory
might be unused/wasted in one container/resource group", while a
different group is hitting its limit too often.

In order to minimize this effect, resource controllers should be
providing both minimum and maximum amount of resources available for a
resource group.

>
> Unfortunately this is not testable because numa=fake=256 doesn't come even
> vaguely close to working. Am trying to get that fixed.
>
>

--

---------- ----------
Chandra Seetharaman | Be careful what you choose....
- sekharan@us.ibm.com | .....you may get it.
---------- ----------

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Andrew Morton on Fri, 18 Aug 2006 18:18:16 GMT
View Forum Message <> Reply to Message

On Fri, 18 Aug 2006 10:59:06 -0700
Rohit Seth <rohitseth@google.com> wrote:

> On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> > On Fri, 18 Aug 2006 07:45:48 -0700
> > Dave Hansen <haveblue@us.ibm.com> wrote:
> >
> > > On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> > > >
> > > > A) Have separate memory management for each container,
> > > > with separate buddy allocator, lru lists, page replacement mechanism.
> > > > That implies a considerable overhead, and the main challenge there
> > > > is sharing of pages between these separate memory managers.
> > >
> > > Hold on here for just a sec...
> > >
> > > It is quite possible to do memory management aimed at one container
> > > while that container's memory still participates in the main VM.
> > >
> > > There is overhead here, as the LRU scanning mechanisms get less
> > > efficient, but I'd rather pay a penalty at LRU scanning time than divide
> > > up the VM, or coarsely start failing allocations.
> > >
> >
> > I have this mad idea that you can divide a 128GB machine up into 256 fake
> > NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> > So that 4.5GB job would be placed within an exclusive cpuset which has nine
> > "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> > no kernel changes needed.
> >
> Sounds like an interesting idea. Will have to depend on something like
> memory hot-plug to get the things move around...
>

mmm, hadn't thought that far ahead. One could manually resize such a
contained with sys_move_pages(). Or just sit and wait: normal page

allocation and reclaim activity would eventually resize the job to the new
set of mems.

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Chandra Seetharaman on Fri, 18 Aug 2006 18:27:34 GMT
View Forum Message <> Reply to Message

On Fri, 2006-08-18 at 11:17 -0700, Chandra Seetharaman wrote:
> On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> > On Fri, 18 Aug 2006 07:45:48 -0700
> > Dave Hansen <haveblue@us.ibm.com> wrote:
> >
> > > On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> > > >
> > > > A) Have separate memory management for each container,
> > > >   with separate buddy allocator, lru lists, page replacement mechanism.
> > > >   That implies a considerable overhead, and the main challenge there
> > > >   is sharing of pages between these separate memory managers.
> > >
> > > Hold on here for just a sec...
> > >
> > > It is quite possible to do memory management aimed at one container
> > > while that container's memory still participates in the main VM.
> > >
> > > There is overhead here, as the LRU scanning mechanisms get less
> > > efficient, but I'd rather pay a penalty at LRU scanning time than divide
> > > up the VM, or coarsely start failing allocations.
> > >
> >
> > I have this mad idea that you can divide a 128GB machine up into 256 fake
> > NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> > So that 4.5GB job would be placed within an exclusive cpuset which has nine
> > "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> > no kernel changes needed.
>
> In this model memory and container are tightly coupled, hence memory
> might be unused/wasted in one container/resource group", while a
> different group is hitting its limit too often.
>
> In order to minimize this effect, resource controllers should be
> providing both minimum and maximum amount of resources available for a
> resource group.

Forgot to mention... There is a set of patches submitted by KUROSAWA
Takahiro that implements this by creating pseudo zones (It has the same
limitation though). http://marc.theaimsgroup.com/?l=ckrm-
tech&m=113867467006531&w=2

>
> >
> > Unfortunately this is not testable because numa=fake=256 doesn't come even
> > vaguely close to working. Am trying to get that fixed.
> >
> > -- -------------
> > Using Tomcat but need to do more? Need to support web services, security?
> > Get stuff done quickly with pre-integrated technology to make your job easier
> > Download IBM WebSphere Application Server v.1.0.1 based on Apache Geronimo
> > http://sel.as-us.falkag.net/sel?cmd=lnk&kid=120709&b id=263057&dat=121642
> > _____
> > ckrm-tech mailing list
> > https://lists.sourceforge.net/lists/listinfo/ckrm-tech
--

 -- ----------
   Chandra Seetharaman | Be careful what you choose....
           - sekharan@us.ibm.com | .....you may get it.
 -- ----------

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Magnus Damm on Mon, 21 Aug 2006 02:38:40 GMT
View Forum Message <> Reply to Message

On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> On Fri, 18 Aug 2006 07:45:48 -0700
> Dave Hansen <haveblue@us.ibm.com> wrote:
>
> > On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> > >
> > > A) Have separate memory management for each container,
> > >   with separate buddy allocator, lru lists, page replacement mechanism.
> > >   That implies a considerable overhead, and the main challenge there
> > >   is sharing of pages between these separate memory managers.
> >
> > Hold on here for just a sec...
> >
> > It is quite possible to do memory management aimed at one container
> > while that container's memory still participates in the main VM.
> >
> > There is overhead here, as the LRU scanning mechanisms get less
> > efficient, but I'd rather pay a penalty at LRU scanning time than divide
> > up the VM, or coarsely start failing allocations.
> >
>
> I have this mad idea that you can divide a 128GB machine up into 256 fake

> NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> So that 4.5GB job would be placed within an exclusive cpuset which has nine
> "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> no kernel changes needed.
>
> Unfortunately this is not testable because numa=fake=256 doesn't come even
> vaguely close to working. Am trying to get that fixed.

You may be looking for the NUMA emulation patches posted here:

http://marc.theaimsgroup.com/?l=linux-mm&m=1128065875018 84&w=2

There is a slightly updated x86_64 version here too:

http://marc.theaimsgroup.com/?l=linux-mm&m=1131613865203 42&w=2

/ magnus

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Magnus Damm on Mon, 21 Aug 2006 02:42:58 GMT
View Forum Message <> Reply to Message

On Fri, 2006-08-18 at 10:59 -0700, Rohit Seth wrote:
> On Fri, 2006-08-18 at 09:42 -0700, Andrew Morton wrote:
> > On Fri, 18 Aug 2006 07:45:48 -0700
> > Dave Hansen <haveblue@us.ibm.com> wrote:
> >
> > > On Fri, 2006-08-18 at 12:08 +0400, Andrey Savochkin wrote:
> > > >
> > > > A) Have separate memory management for each container,
> > > >   with separate buddy allocator, lru lists, page replacement mechanism.
> > > >   That implies a considerable overhead, and the main challenge there
> > > >   is sharing of pages between these separate memory managers.
> > >
> > > Hold on here for just a sec...
> > >
> > > It is quite possible to do memory management aimed at one container
> > > while that container's memory still participates in the main VM.
> > >
> > > There is overhead here, as the LRU scanning mechanisms get less
> > > efficient, but I'd rather pay a penalty at LRU scanning time than divide
> > > up the VM, or coarsely start failing allocations.
> > >
> >
> > I have this mad idea that you can divide a 128GB machine up into 256 fake
> > NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> > So that 4.5GB job would be placed within an exclusive cpuset which has nine

> > "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> > no kernel changes needed.
> >
> Sounds like an interesting idea. Will have to depend on something like
> memory hot-plug to get the things move around...

Yeah, moving things around: The pzone memory resource controller
introduces dynamically sized zones if I remember correctly.

http://www.opensubscriber.com/message/ckrm-tech@lists.source forge.net/3133911.html

/ magnus

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Andi Kleen on Mon, 21 Aug 2006 07:48:40 GMT
View Forum Message <> Reply to Message

> You may be looking for the NUMA emulation patches posted here:
>
> http://marc.theaimsgroup.com/?l=linux-mm&m=1128065875018 84&w=2
>
> There is a slightly updated x86_64 version here too:
>
> http://marc.theaimsgroup.com/?l=linux-mm&m=1131613865203 42&w=2

Hmm, I must have missed that version. Seems like a improvement. Best you
resubmit it, although I'll probably only take it after the .19 merge

-Andi

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Magnus Damm on Mon, 21 Aug 2006 08:42:41 GMT
View Forum Message <> Reply to Message

On Mon, 2006-08-21 at 09:48 +0200, Andi Kleen wrote:
> > You may be looking for the NUMA emulation patches posted here:
> >
> > http://marc.theaimsgroup.com/?l=linux-mm&m=1128065875018 84&w=2
> >
> > There is a slightly updated x86_64 version here too:
> >
> > http://marc.theaimsgroup.com/?l=linux-mm&m=1131613865203 42&w=2
>
> Hmm, I must have missed that version. Seems like a improvement. Best you

> resubmit it, although I'll probably only take it after the .19 merge

No problem. The second URL pointed to a x86_64 version where I tried to
break out code to make some kind of generic NUMA emulation layer. At
that time no one seemed interested in that strategy as a simple resource
control solution so I gave that up.

For x86_64 I think it's only worth mucking around with the code if
people believe that it is the right way to go for in-kernel resource
control.

The x86_64 patches above include code to divide each real NUMA node into
several smaller emulated nodes, but that is kind of pointless if people
only use it for non-resource control purposes, ie just to play with
CPUSETS and NUMA on non-NUMA hardware. For simple purposes like that I
think the existing NUMA emulation code for x86_64 works perfectly well.

I still think that i386 users would benefit from NUMA emulation though.
If you want me to up-port the i386-specific code just let me know.

Thanks,

/ magnus

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Andi Kleen on Mon, 21 Aug 2006 09:03:45 GMT
View Forum Message <> Reply to Message

On Monday 21 August 2006 10:42, Magnus Damm wrote:

> No problem. The second URL pointed to a x86_64 version where I tried to
> break out code to make some kind of generic NUMA emulation layer. At
> that time no one seemed interested in that strategy as a simple resource
> control solution so I gave that up.
>
> For x86_64 I think it's only worth mucking around with the code if
> people believe that it is the right way to go for in-kernel resource
> control.

Does it by chance fix the existing code? Andrew has been complaining
(and I could reproduce) that numa=fake=16 makes it triple fault at boot.
The theory was that it didn't like empty nodes which can happen this way.
I unfortunately didn't have time to look into it closely so far.

> The x86_64 patches above include code to divide each real NUMA node into
> several smaller emulated nodes, but that is kind of pointless if people
> only use it for non-resource control purposes, ie just to play with

> CPUSETS and NUMA on non-NUMA hardware. For simple purposes like that I
> think the existing NUMA emulation code for x86_64 works perfectly well.
>
> I still think that i386 users would benefit from NUMA emulation though.
> If you want me to up-port the i386-specific code just let me know.

I personally have my doubts about 32bit NUMA -- it will always have
ZONE_NORMAL only on a single node, which limits it very much.
But ok I guess it might be useful to somebody.

-Andi

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Magnus Damm on Mon, 21 Aug 2006 09:18:08 GMT
View Forum Message <> Reply to Message

On Mon, 2006-08-21 at 11:03 +0200, Andi Kleen wrote:
> On Monday 21 August 2006 10:42, Magnus Damm wrote:
>
> > No problem. The second URL pointed to a x86_64 version where I tried to
> > break out code to make some kind of generic NUMA emulation layer. At
> > that time no one seemed interested in that strategy as a simple resource
> > control solution so I gave that up.
> >
> > For x86_64 I think it's only worth mucking around with the code if
> > people believe that it is the right way to go for in-kernel resource
> > control.
>
> Does it by chance fix the existing code? Andrew has been complaining
> (and I could reproduce) that numa=fake=16 makes it triple fault at boot.
> The theory was that it didn't like empty nodes which can happen this way.
> I unfortunately didn't have time to look into it closely so far.

The code does rearrange how the boundaries are calculated, and it may
happen to fix that specific problem. I'll try to find some time later
this week to have a look at it.

> > The x86_64 patches above include code to divide each real NUMA node into
> > several smaller emulated nodes, but that is kind of pointless if people
> > only use it for non-resource control purposes, ie just to play with
> > CPUSETS and NUMA on non-NUMA hardware. For simple purposes like that I
> > think the existing NUMA emulation code for x86_64 works perfectly well.
> >
> > I still think that i386 users would benefit from NUMA emulation though.
> > If you want me to up-port the i386-specific code just let me know.
>
> I personally have my doubts about 32bit NUMA -- it will always have

> ZONE_NORMAL only on a single node, which limits it very much.
> But ok I guess it might be useful to somebody.

Very true. I was mainly thinking about the i386 code as a simple way for
people to play with NUMA and CPUSETS.

/ magnus

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by dev on Mon, 21 Aug 2006 13:33:19 GMT
View Forum Message <> Reply to Message

Andrew Morton wrote:
> I have this mad idea that you can divide a 128GB machine up into 256 fake
> NUMA nodes, then you use each "node" as a 512MB unit of memory allocation.
> So that 4.5GB job would be placed within an exclusive cpuset which has nine
> "mems" (what are these called?) and voila: the job has a hard 4.5GB limit,
> no kernel changes needed.
this doesn't allow memory overcommitment, does it?

Kirill

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by Paul Jackson on Mon, 21 Aug 2006 17:51:06 GMT
View Forum Message <> Reply to Message

> this doesn't allow memory overcommitment, does it?

Uh - no - I don't think so. You can over commit
the memory of a task in a small cpuset just as well
as you can a task in a big cpuset or even one in the
top cpuset covering the entire system.

Perhaps I didn't understand your point.

--
I won't rest till it's the best ...
Programmer, Linux Scalability
Paul Jackson <pj@sgi.com> 1.925.600.0401

Subject: Re: [ckrm-tech] [PATCH 4/7] UBC: syscalls (user interface)
Posted by dev on Tue, 22 Aug 2006 08:50:14 GMT
View Forum Message <> Reply to Message

Paul Jackson wrote:
>>this doesn't allow memory overcommitment, does it?
>
>
> Uh - no - I don't think so. You can over commit
> the memory of a task in a small cpuset just as well
> as you can a task in a big cpuset or even one in the
> top cpuset covering the entire system.
>
> Perhaps I didn't understand your point.
My point was that when you have lots of containers
their summary memory limit can be much higher then available RAM.
This allows bursts of memory usage for containers, since
it is very unlikely for all of them to consume the memory
simulatenously. E.g. hosters usually oversell memory
say 2 times on the node.

So the question was whether it is possbile to overcommit memory
with NUMA emulation?

Thanks,
Kirill