Subject: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Alexey Dobriyan on Mon, 02 Apr 2007 11:32:05 GMT
View Forum Message <> Reply to Message

They will be used by cpuid driver and powernow-k8 cpufreq driver.

With these changes powernow-k8 driver could run correctly on OpenVZ kernels
with virtual cpus enabled (SCHED_VCPU).

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

```
 arch/i386/lib/Makefile         |    2 -
 arch/i386/lib/cpuid-on-cpu.c   |   67 ++++++++++++++++++++++++++++++++++++++++
 arch/x86_64/lib/Makefile       |    2 -
 arch/x86_64/lib/cpuid-on-cpu.c |    1
 include/asm-i386/processor.h   |   15 +++++++++
 include/asm-x86_64/msr.h       |   13 +++++++
 6 files changed, 98 insertions(+), 2 deletions(-)

--- a/arch/i386/lib/Makefile
+++ b/arch/i386/lib/Makefile
@@ -8,4 +8,4 @@ lib-y = checksum.o delay.o usercopy.o getuser.o putuser.o memcpy.o strstr.o
\

 lib-$(CONFIG_X86_USE_3DNOW) += mmx.o

-obj-$(CONFIG_SMP) += msr-on-cpu.o
+obj-$(CONFIG_SMP) += cpuid-on-cpu.o msr-on-cpu.o
--- a/arch/x86_64/lib/Makefile
+++ b/arch/x86_64/lib/Makefile
@@ -5,7 +5,7 @@
 CFLAGS_csum-partial.o := -funroll-loops

 obj-y := io.o iomap_copy.o
-obj-$(CONFIG_SMP) += msr-on-cpu.o
+obj-$(CONFIG_SMP) += cpuid-on-cpu.o msr-on-cpu.o

 lib-y := csum-partial.o csum-copy.o csum-wrappers.o delay.o \
  usercopy.o getuser.o putuser.o  \
--- a/include/asm-i386/processor.h
+++ b/include/asm-i386/processor.h
@@ -643,6 +643,21 @@ static inline unsigned int cpuid_edx(unsigned int op)
  return edx;
 }

+#ifdef CONFIG_SMP
+void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32 *edx);
```

```
+u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op);
+#else
+static inline void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32
*edx)
+{
+ cpuid(op, eax, ebx, ecx, edx);
+}
+
+static inline u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op)
+{
+ return cpuid_eax(op);
+}
+#endif
+
 /* generic versions from gas */
 #define GENERIC_NOP1 ".byte 0x90\n"
 #define GENERIC_NOP2     ".byte 0x89,0xf6\n"
--- a/include/asm-x86_64/msr.h
+++ b/include/asm-x86_64/msr.h
@@ -163,6 +163,9 @@ static inline unsigned int cpuid_edx(unsigned int op)
 #ifdef CONFIG_SMP
 void rdmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 *l, u32 *h);
 void wrmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 l, u32 h);
+
+void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32 *edx);
+u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op);
 #else  /*  CONFIG_SMP  */
 static inline void rdmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 *l, u32 *h)
 {
@@ -172,6 +175,16 @@ static inline void wrmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 l,
u32 h)
 {
  wrmsr(msr_no, l, h);
 }
+
+static inline void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32
*edx)
+{
+ cpuid(op, eax, ebx, ecx, edx);
+}
+
+static inline u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op)
+{
+ return cpuid_eax(op);
+}
 #endif  /*  CONFIG_SMP  */

 #endif
```

```
--- /dev/null
+++ b/arch/i386/lib/cpuid-on-cpu.c
@@ -0,0 +1,67 @@
+#include <linux/module.h>
+#include <linux/preempt.h>
+#include <linux/smp.h>
+#include <linux/types.h>
+
+struct cpuid_info {
+ u32 op;
+ u32 eax, ebx, ecx, edx;
+};
+
+static void __cpuid_on_cpu(void *info)
+{
+ struct cpuid_info *rv = info;
+
+ cpuid(rv->op, &rv->eax, &rv->ebx, &rv->ecx, &rv->edx);
+}
+
+void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32 *edx)
+{
+ preempt_disable();
+ if (smp_processor_id() == cpu)
+  cpuid(op, eax, ebx, ecx, edx);
+ else {
+  struct cpuid_info rv;
+
+  rv.op = op;
+  smp_call_function_single(cpu, __cpuid_on_cpu, &rv, 0, 1);
+  *eax = rv.eax;
+  *ebx = rv.ebx;
+  *ecx = rv.ecx;
+  *edx = rv.edx;
+ }
+ preempt_enable();
+}
+
+struct cpuid_eax_info {
+ u32 op;
+ u32 eax;
+};
+
+static void __cpuid_eax_on_cpu(void *info)
+{
+ struct cpuid_info *rv = info;
+
+ rv->eax = cpuid_eax(rv->op);
```

```
+}
+
+u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op)
+{
+ u32 ret;
+
+ preempt_disable();
+ if (smp_processor_id() == cpu)
+  ret = cpuid_eax(op);
+ else {
+  struct cpuid_eax_info rv;
+
+  rv.op = op;
+  smp_call_function_single(cpu, __cpuid_eax_on_cpu, &rv, 0, 1);
+  ret = rv.eax;
+ }
+ preempt_enable();
+ return ret;
+}
+
+EXPORT_SYMBOL(cpuid_on_cpu);
+EXPORT_SYMBOL(cpuid_eax_on_cpu);
--- /dev/null
+++ b/arch/x86_64/lib/cpuid-on-cpu.c
@@ -0,0 +1 @@
+#include "../../i386/lib/cpuid-on-cpu.c"
```

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by claudiuc on Mon, 02 Apr 2007 11:33:25 GMT
View Forum Message <> Reply to Message

Aceasta adresa nu mai este folosita. Va rugam trimiteti mail la adresa claudiuc@easymedia.ro.

This e-mail address is not used anymore. Please send further e-mails at claudiuc@easymedia.ro.

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Andi Kleen on Mon, 02 Apr 2007 12:10:29 GMT
View Forum Message <> Reply to Message

On Monday 02 April 2007 13:38, Alexey Dobriyan wrote:
> They will be used by cpuid driver and powernow-k8 cpufreq driver.
>
> With these changes powernow-k8 driver could run correctly on OpenVZ kernels
> with virtual cpus enabled (SCHED_VCPU).

This means openvz has multiple virtual CPU levels? One for cpuid/rdmsr and one
for the rest of the kernel? Both powernow-k8 and cpuid attempt to schedule
to the target CPU so they should already run there. But it is some other CPU,
but when they ask your _on_cpu() functions they suddenly get a "real" CPU?
Where is the difference between these levels of virtualness?

That sounds quite fragile and will likely break often. I just rejected a similar
concept -- virtual nodes and "physical nodes" for similar reasons.

Also it has weird semantics. For example if you have multiple
virtual CPUs mapping to a single CPU then would the powernow-k8 driver
try to set the frequency multiple times on the same physical CPU? That might
go wrong actually because the CPU might not be happy to be poked again
while it is already in a frequency change. Also there is no locking
so in theory two vcpus might try to change frequency in parallel with
probably quite bad effects.

I'm sure there are other scenarios with similar problems. e.g. what
happens with microcode updates etc.?

Before adding any hacks like this I think your vcpu concept
needs to be discussed properly on l-k. For me it doesn't look like it is
something good right now though.

-Andi

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by hpa on Mon, 02 Apr 2007 16:20:16 GMT
View Forum Message <> Reply to Message

Andi Kleen wrote:
> On Monday 02 April 2007 13:38, Alexey Dobriyan wrote:
>> They will be used by cpuid driver and powernow-k8 cpufreq driver.
>>
>> With these changes powernow-k8 driver could run correctly on OpenVZ kernels
>> with virtual cpus enabled (SCHED_VCPU).
>
> This means openvz has multiple virtual CPU levels? One for cpuid/rdmsr and one
> for the rest of the kernel? Both powernow-k8 and cpuid attempt to schedule
> to the target CPU so they should already run there. But it is some other CPU,
> but when they ask your _on_cpu() functions they suddenly get a "real" CPU?
> Where is the difference between these levels of virtualness?
>

The CPUID and MSR drivers do not schedule to the target CPU; instead, on
hardware, they rely on IPI'ing the target processor if it is not the one
that's currently running.

There were a lot of discussion back when about which was the better solution. Alan Cox, in particular, really preferred the interrupt solution as being less likely to cause implicit deadlock.

I do want to add that it's been on my list for some time -- in fact, I keep implementing it half-way and then having other things to do -- to add MSR and CPUID ioctls() that allow the full register file to be set and read back, in order to support architecturally broken MSR and CPUID levels.

-hpa

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Andi Kleen on Mon, 02 Apr 2007 16:55:55 GMT
View Forum Message <> Reply to Message

> The CPUID and MSR drivers do not schedule to the target CPU; instead, on
> hardware,

But powernow-k8 does.

-Andi

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Alexey Dobriyan on Tue, 03 Apr 2007 13:33:31 GMT
View Forum Message <> Reply to Message

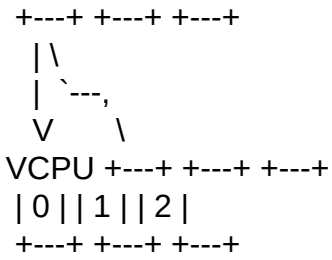
On Mon, Apr 02, 2007 at 02:10:29PM +0200, Andi Kleen wrote:
> On Monday 02 April 2007 13:38, Alexey Dobriyan wrote:
> > They will be used by cpuid driver and powernow-k8 cpufreq driver.
> >
> > With these changes powernow-k8 driver could run correctly on OpenVZ kernels
> > with virtual cpus enabled (SCHED_VCPU).
>
> This means openvz has multiple virtual CPU levels?

Not sure what do you mean.

> One for cpuid/rdmsr and one for the rest of the kernel?

Now I think I do. No, only one VCPU level:

```
 +---+ +---+ +---+
PCPU | 0 | | 1 | | 2 |
```



PCPU chooses VCPU, chooses task from it's runqueue, or becomes idle.

> Both powernow-k8 and cpuid attempt to schedule
> to the target CPU so they should already run there. But it is some other CPU,
> but when they ask your _on_cpu() functions they suddenly get a "real" CPU?
> Where is the difference between these levels of virtualness?

*_on_cpu functions do some work on given physical CPU.
set_cpus_allowed() in openvz operates on VCPU level, so process doing
set_cpus_allowed() still could be scheduled anywhere. Which horribly
breaks cpufreq drivers. To unbreak them, we need a way to do frequency
changing work on given PCPU whose number comes from /sys/*/cpu/cpu*
hierarchy.

> That sounds quite fragile and will likely break often. I just rejected a similar
> concept -- virtual nodes and "physical nodes" for similar reasons.

Care to drop a link, so I could compare them? Don't recall something
like that posted on l-k.

> Also it has weird semantics. For example if you have multiple
> virtual CPUs mapping to a single CPU then would the powernow-k8 driver
> try to set the frequency multiple times on the same physical CPU?

If core cpufreq locking is OK, why would it?

> That might
> go wrong actually because the CPU might not be happy to be poked again
> while it is already in a frequency change. Also there is no locking
> so in theory two vcpus might try to change frequency in parallel with
> probably quite bad effects.
>
> I'm sure there are other scenarios with similar problems. e.g. what
> happens with microcode updates etc.?

apply_microcode() looks small enough to convert it to IPIs, but so far
nobody asked for microcode updates in openvz.

> Before adding any hacks like this I think your vcpu concept
> needs to be discussed properly on l-k. For me it doesn't look like it is

> something good right now though.

Andi, I think it all relies on correctness of core cpufreq locking. Core
cpufreq locking is not changed with these patches so number of bugs
should stay the same.

current way new way
----------- -------

set_cpus_allowed(cpu);
[frequency change part #1] [IPI part #1]
  <=== nasty here ===>
[frequency change part #2] [IPI part #2]
set_cpus_allowed(old)

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Andi Kleen on Tue, 03 Apr 2007 13:42:50 GMT
View Forum Message <> Reply to Message

> > Both powernow-k8 and cpuid attempt to schedule
> > to the target CPU so they should already run there. But it is some other CPU,
> > but when they ask your _on_cpu() functions they suddenly get a "real" CPU?
> > Where is the difference between these levels of virtualness?
>
> *_on_cpu functions do some work on given physical CPU.
> set_cpus_allowed() in openvz operates on VCPU level, so process doing
> set_cpus_allowed() still could be scheduled anywhere.

Ok so you have multple levels.

> > Also it has weird semantics. For example if you have multiple
> > virtual CPUs mapping to a single CPU then would the powernow-k8 driver
> > try to set the frequency multiple times on the same physical CPU?
>
> If core cpufreq locking is OK, why would it?

It won't know about multiple CPUs mapping to a single CPU.

> apply_microcode() looks small enough to convert it to IPIs, but so far
> nobody asked for microcode updates in openvz.

Well if they try it they will probably have problems.

> > Before adding any hacks like this I think your vcpu concept
> > needs to be discussed properly on l-k. For me it doesn't look like it is
> > something good right now though.
>

> Andi, I think it all relies on correctness of core cpufreq locking.

I have my doubts it will cope with you changing all reasonable expected semantics
under it.

-Andi

Subject: Re: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Alexey Dobriyan on Tue, 03 Apr 2007 14:50:13 GMT
View Forum Message <> Reply to Message

On Tue, Apr 03, 2007 at 03:42:50PM +0200, Andi Kleen wrote:
> > > Both powernow-k8 and cpuid attempt to schedule
> > > to the target CPU so they should already run there. But it is some other CPU,
> > > but when they ask your _on_cpu() functions they suddenly get a "real" CPU?
> > > Where is the difference between these levels of virtualness?
> >
> > *_on_cpu functions do some work on given physical CPU.
> > set_cpus_allowed() in openvz operates on VCPU level, so process doing
> > set_cpus_allowed() still could be scheduled anywhere.
>
> Ok so you have multple levels.
>
> > > Also it has weird semantics. For example if you have multiple
> > > virtual CPUs mapping to a single CPU then would the powernow-k8 driver
> > > try to set the frequency multiple times on the same physical CPU?
> >
> > If core cpufreq locking is OK, why would it?
>
> It won't know about multiple CPUs mapping to a single CPU.
>
> > apply_microcode() looks small enough to convert it to IPIs, but so far
> > nobody asked for microcode updates in openvz.
>
> Well if they try it they will probably have problems.
>
> > > Before adding any hacks like this I think your vcpu concept
> > > needs to be discussed properly on l-k. For me it doesn't look like it is
> > > something good right now though.
> >
> > Andi, I think it all relies on correctness of core cpufreq locking.
>
> I have my doubts it will cope with you changing all reasonable expected semantics
> under it.

Synchronization promitives work as expected. Otherwise openvz'd be buried in bugs
all over the map.

Core cpufreq has per-cpu array of rw-semaphores but the index of semaphore one want to down comes from userspace not from number of CPU process is executing virtual or physical.

Probably davej could say something.