

## Re: [PATCH] drivers/base: export gpl (un)register\_memory\_notifier

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-02/msg07760.html>

---

- *From:* Dave Hansen <[haveblue@xxxxxxxxxx](mailto:haveblue@xxxxxxxxxx)>
  - *Date:* Fri, 15 Feb 2008 08:55:38 -0800
- 

On Fri, 2008-02-15 at 14:22 +0100, Christoph Raisch wrote:

A translation from kernel to ehea\_bmap space should be fast and predictable (ruling out hashes).  
If a driver doesn't know anything else about the mapping structure, the normal solution in kernel for this type of problem is a multi level look up table like pgd->pud->pmd->pte  
This doesn't sound right to be implemented in a device driver.

We didn't see from the existing code that such a mapping to a contiguous space already exists.  
Maybe we've missed it.

I've been thinking about that, and I don't think you really *\*need\** to keep a comprehensive map like that.

When the memory is in a particular configuration (range of memory present along with unique set of holes) you get a unique ehea\_bmap configuration. That layout is completely predictable.

So, if at any time you want to figure out what the ehea\_bmap address for a particular *\*Linux\** virtual address is, you just need to pretend that you're creating the entire ehea\_bmap, use the same algorithm and figure out host you would have placed things, and use that result.

Now, that's going to be a slow, crappy linear search (but maybe not as slow as recreating the silly thing). So, you might eventually run into some scalability problems with a lot of packets going around. But, I'd be curious if you do in practice.

The other idea is that you create a mapping that is precisely 1:1 with kernel memory. Let's say you have two sections present, 0 and 100. You have a high\_section\_index of 100, and you vmalloc() a 100 entry array.

Re: [PATCH] drivers/base: export gpl (un)register\_memory\_notifier

You need to create a \*CONTIGUOUS\* ehea map? Create one like this:

EHEA\_VADDR->Linux Section

0->0

1->0

2->0

3->0

...

100->100

It's contiguous. Each area points to a valid Linux memory address.

It's also discernable in O(1) to what EHEA address a given Linux address is mapped. You just have a couple of duplicate entries.

-- Dave

--

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>