

---

Subject: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses  
Posted by [xemul](#) on Tue, 17 Apr 2007 11:45:18 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Evgeny Kravtsunov <emkravts@openvz.org>

compare\_ether\_addr() implicitly requires that the addresses  
passed are 2-bytes aligned in memory.

This is not true for br\_stp\_change\_bridge\_id() and  
br\_stp\_recalculate\_bridge\_id() in which one of the addresses  
is unsigned char \*, and thus may not be 2-bytes aligned.

Signed-off-by: Evgeny Kravtsunov <emkravts@openvz.org>

Signed-off-by: Kirill Korotaev <dev@openvz.org>

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

---

```
--- a/net/bridge/br_stp_if.c 2006-09-20 07:42:06.000000000 +0400
+++ b/net/bridge/br_stp_if.c 2007-04-13 12:28:08.000000000 +0400
@@ -124,7 +124,9 @@ void br_stp_disable_port(struct net_brid
/* called under bridge lock */
void br_stp_change_bridge_id(struct net_bridge *br, const unsigned char *addr)
{
- unsigned char oldaddr[6];
+ /* should be aligned on 2 bytes for compare_ether_addr() */
+ unsigned short oldaddr_aligned[ETH_ALEN >> 1];
+ unsigned char *oldaddr = (unsigned char *)oldaddr_aligned;
  struct net_bridge_port *p;
  int wasroot;

@@ -149,11 +151,14 @@ void br_stp_change_bridge_id(struct net_
  br_become_root_bridge(br);
}

-static const unsigned char br_mac_zero[6];
+/* should be aligned on 2 bytes for compare_ether_addr() */
+static const unsigned short br_mac_zero_aligned[ETH_ALEN >> 1];

/* called under bridge lock */
void br_stp_recalculate_bridge_id(struct net_bridge *br)
{
+ const unsigned char *br_mac_zero =
+ (const unsigned char *)br_mac_zero_aligned;
  const unsigned char *addr = br_mac_zero;
  struct net_bridge_port *p;
```

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [davem](#) on Tue, 17 Apr 2007 19:31:35 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Pavel Emelianov <xemul@sw.ru>

Date: Tue, 17 Apr 2007 15:49:30 +0400

> From: Evgeny Kravtsunov <emkravts@openvz.org>  
>  
> compare\_ether\_addr() implicitly requires that the addresses  
> passed are 2-bytes aligned in memory.  
>  
> This is not true for br\_stp\_change\_bridge\_id() and  
> br\_stp\_recalculate\_bridge\_id() in which one of the addresses  
> is unsigned char \*, and thus may not be 2-bytes aligned.  
>  
> Signed-off-by: Evgeny Kravtsunov <emkravts@openvz.org>  
> Signed-off-by: Kirill Korotaev <dev@openvz.org>  
> Signed-off-by: Pavel Emelianov <xemul@openvz.org>

Patch applied, thank you.

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Stephen Hemminger](#) on Tue, 17 Apr 2007 19:55:41 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

David Miller wrote:

> From: Pavel Emelianov <xemul@sw.ru>  
> Date: Tue, 17 Apr 2007 15:49:30 +0400  
>  
>  
>> From: Evgeny Kravtsunov <emkravts@openvz.org>  
>>  
>> compare\_ether\_addr() implicitly requires that the addresses  
>> passed are 2-bytes aligned in memory.  
>>  
>> This is not true for br\_stp\_change\_bridge\_id() and  
>> br\_stp\_recalculate\_bridge\_id() in which one of the addresses  
>> is unsigned char \*, and thus may not be 2-bytes aligned.  
>>  
>> Signed-off-by: Evgeny Kravtsunov <emkravts@openvz.org>  
>> Signed-off-by: Kirill Korotaev <dev@openvz.org>  
>> Signed-off-by: Pavel Emelianov <xemul@openvz.org>  
>>  
>

> Patch applied, thank you.  
>  
I had a better way, I'll fix.

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [davem](#) on Tue, 17 Apr 2007 20:09:29 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Stephen Hemminger <[shemminger@linux-foundation.org](mailto:shemminger@linux-foundation.org)>

Date: Tue, 17 Apr 2007 12:55:41 -0700

> David Miller wrote:  
> > From: Pavel Emelianov <[xemul@sw.ru](mailto:xemul@sw.ru)>  
> > Date: Tue, 17 Apr 2007 15:49:30 +0400  
> >  
> >  
> >> From: Evgeny Kravtsunov <[emkravts@openvz.org](mailto:emkravts@openvz.org)>  
> >>  
> >> compare\_ether\_addr() implicitly requires that the addresses  
> >> passed are 2-bytes aligned in memory.  
> >>  
> >> This is not true for br\_stp\_change\_bridge\_id() and  
> >> br\_stp\_recalculate\_bridge\_id() in which one of the addresses  
> >> is unsigned char \*, and thus may not be 2-bytes aligned.  
> >>  
> >> Signed-off-by: Evgeny Kravtsunov <[emkravts@openvz.org](mailto:emkravts@openvz.org)>  
> >> Signed-off-by: Kirill Korotaev <[dev@openvz.org](mailto:dev@openvz.org)>  
> >> Signed-off-by: Pavel Emelianov <[xemul@openvz.org](mailto:xemul@openvz.org)>  
> >>  
> >  
> > Patch applied, thank you.  
> >  
> I had a better way, I'll fix.

You better hurry I want today's fixes I applied out to Linus this afternoon.

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Stephen Hemminger](#) on Tue, 17 Apr 2007 20:37:23 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

The previous patch relied on the bridge id being aligned by the compiler (which happens as a side effect). So please use

this instead.

compare\_ether\_addr() implicitly requires that the addresses passed are 2-bytes aligned in memory.

This is not true for br\_stp\_change\_bridge\_id() and br\_stp\_recalculate\_bridge\_id() in which one of the addresses is unsigned char \*, and thus may not be 2-bytes aligned.

Signed-off-by: Evgeny Kravtsunov <emkravts@openvz.org>

Signed-off-by: Kirill Korotaev <dev@openvz.org>

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

Signed-off-by: Stephen Hemminger <shemminger@linux-foundation.org>

---

```
--- linux-2.6.orig/net/bridge/br_stp_if.c 2007-04-16
20:03:04.000000000 -0700 +++ linux-2.6/net/bridge/br_stp_if.c
2007-04-17 13:25:52.000000000 -0700 @@ -126,23 +126,22 @@
/* called under bridge lock */
void br_stp_change_bridge_id(struct net_bridge *br, const unsigned
char *addr) {
- unsigned char oldaddr[6];
+ bridge_id old_id;
  struct net_bridge_port *p;
  int wasroot;

  wasroot = br_is_root_bridge(br);

- memcpy(oldaddr, br->bridge_id.addr, ETH_ALEN);
+ old_id = br->bridge_id;
  memcpy(br->bridge_id.addr, addr, ETH_ALEN);
  memcpy(br->dev->dev_addr, addr, ETH_ALEN);

  list_for_each_entry(p, &br->port_list, list) {
- if (!compare_ether_addr(p->designated_bridge.addr,
oldaddr))
+ if (!compare_ether_addr(p->designated_bridge.addr,
old_id.addr)) memcpy(p->designated_bridge.addr, addr, ETH_ALEN);

- if (!compare_ether_addr(p->designated_root.addr,
oldaddr))
+ if (!compare_ether_addr(p->designated_root.addr,
old_id.addr)) memcpy(p->designated_root.addr, addr, ETH_ALEN);
-
  }

  br_configuration_update(br);
```

```

@@ -151,19 +150,17 @@
    br_become_root_bridge(br);
}

-static const unsigned char br_mac_zero[6];
-
/* called under bridge lock */
void br_stp_recalculate_bridge_id(struct net_bridge *br)
{
- const unsigned char *addr = br_mac_zero;
+ static const bridge_id id_zero;
+ const unsigned char *addr = id_zero.addr;
    struct net_bridge_port *p;

    list_for_each_entry(p, &br->port_list, list) {
- if (addr == br_mac_zero ||
+ if (addr == id_zero.addr ||
        memcmp(p->dev->dev_addr, addr, ETH_ALEN) < 0)
        addr = p->dev->dev_addr;
-
    }

    if (compare_ether_addr(br->bridge_id.addr, addr))
--- linux-2.6.orig/net/bridge/br_private.h 2007-04-17
13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br_private.h
2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@
{
    unsigned char prio[2];
    unsigned char addr[6];
-};
+} __attribute__((aligned(8)));

struct mac_addr
{

```

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [davem](#) on Tue, 17 Apr 2007 21:09:41 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Stephen Hemminger <[shemminger@linux-foundation.org](mailto:shemminger@linux-foundation.org)>

Date: Tue, 17 Apr 2007 13:37:23 -0700

> The previous patch relied on the bridge id being aligned by  
> the compiler (which happens as a side effect). So please use  
> this instead.  
>

> compare\_ether\_addr() implicitly requires that the addresses  
> passed are 2-bytes aligned in memory.  
>  
> This is not true for br\_stp\_change\_bridge\_id() and  
> br\_stp\_recalculate\_bridge\_id() in which one of the addresses  
> is unsigned char \*, and thus may not be 2-bytes aligned.  
>  
> Signed-off-by: Evgeny Kravtsunov <emkravts@openvz.org>  
> Signed-off-by: Kirill Korotaev <dev@openvz.org>  
> Signed-off-by: Pavel Emelianov <xemul@openvz.org>  
> Signed-off-by: Stephen Hemminger <shemminger@linux-foundation.org>

bridge\_id would be aligned by luck, because it is composed of char's  
there is no explicit reason it should be aligned on at least an  
unsigned short boundary.

I like the other patch much better, it provided explicit alignment and  
is guaranteed to get rid of the problem.

Indeed, I wrote a test program on 32-bit Sparc to validate this:

```
struct bridge_id {
    unsigned char a[6];
    unsigned char b[6];
};

extern void bar(unsigned char *, unsigned char *);

void foo(void)
{
    unsigned char a;
    struct bridge_id b;

    bar(&b.a[0], &a);
}
```

foo() gets compiled like this:

```
foo:
    save %sp, -120, %sp
    add %fp, -21, %o0
    call bar, 0
    add %fp, -9, %o1
    jmp %i7+8
    restore
```

See? The bridge\_id (passed in via %o0) is on an odd byte boundary  
on the stack.

So your patch isn't fixing the bug at all.

I'm going to apply the original patch, because that one will actually fix the problem and was actually tested on a system that saw the problem.

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Eric Dumazet](#) on Tue, 17 Apr 2007 21:24:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

```
> From: Stephen Hemminger <shemminger@linux-foundation.org>
> Date: Tue, 17 Apr 2007 13:37:23 -0700
>
>> The previous patch relied on the bridge id being aligned by
>> the compiler (which happens as a side effect). So please use
>> this instead.
>>
>> compare_ether_addr() implicitly requires that the addresses
>> passed are 2-bytes aligned in memory.
>>
>> This is not true for br_stp_change_bridge_id() and
>> br_stp_recalculate_bridge_id() in which one of the addresses
>> is unsigned char *, and thus may not be 2-bytes aligned.
>>
>> Signed-off-by: Evgeny Kravtsunov <emkravts@openvz.org>
>> Signed-off-by: Kirill Korotaev <dev@openvz.org>
>> Signed-off-by: Pavel Emelianov <xemul@openvz.org>
>> Signed-off-by: Stephen Hemminger <shemminger@linux-foundation.org>
>
> bridge_id would be aligned by luck, because it is composed of char's
> there is no explicit reason it should be aligned on at least an
> unsigned short boundary.
>
> I like the other patch much better, it provided explicit alignment and
> is guaranteed to get rid of the problem.
>
> Indeed, I wrote a test program on 32-bit Sparc to validate this:
>
> struct bridge_id {
> unsigned char a[6];
> unsigned char b[6];
> };
>
> extern void bar(unsigned char *, unsigned char *);
```

```

>
> void foo(void)
> {
>   unsigned char a;
>   struct bridge_id b;
>
>   bar(&b.a[0], &a);
> }
>
> foo() gets compiled like this:
>
> foo:
>   save %sp, -120, %sp
>   add %fp, -21, %o0
>   call bar, 0
>   add %fp, -9, %o1
>   jmp %i7+8
>   restore
>
> See? The bridge_id (passed in via %o0) is on an odd byte boundary
> on the stack.
>
> So your patch isn't fixing the bug at all.
>
> I'm going to apply the original patch, because that one will
> actually fix the problem and was actually tested on a system
> that saw the problem.

```

I suspect you missed part of Stephen patch :

(maybe some mailer problem...)

```

--- linux-2.6.orig/net/bridge/br_private.h 2007-04-17
13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br_private.h
2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@
{
    unsigned char prio[2];
    unsigned char addr[6];
-};
+} __attribute__((aligned(8)));

    struct mac_addr
    {

```

---



---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [davem](#) on Tue, 17 Apr 2007 21:27:42 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Eric Dumazet <dada1@cosmosbay.com>

Date: Tue, 17 Apr 2007 23:24:36 +0200

> I suspect you missed part of Stephen patch :

>

> (maybe some mailer problem...)

My bad, sorry :(

I pushed the other version of the fix to Linus just now.

I'll put Stephen's version in if he sends me a fixup relative to that.

Sorry again.

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [xemul](#) on Wed, 18 Apr 2007 06:39:20 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

[snip]

```
> --- linux-2.6.orig/net/bridge/br_private.h 2007-04-17
> 13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br_private.h
> 2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@
> {
>  unsigned char prio[2];
>  unsigned char addr[6];
> -};
> +} __attribute__((aligned(8)));
```

Why "8"? Mustn't it be "16"? Address is to be 2-bytes aligned...

```
> struct mac_addr
> {
> -
> To unsubscribe from this list: send the line "unsubscribe netdev" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
>
```

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet

addresses

Posted by [davem](#) on Wed, 18 Apr 2007 08:28:04 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Pavel Emelianov <xemul@sw.ru>

Date: Wed, 18 Apr 2007 10:43:56 +0400

> [snip]

>

> > --- linux-2.6.orig/net/bridge/br\_private.h 2007-04-17

> > 13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br\_private.h

> > 2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@

> > {

> > unsigned char prio[2];

> > unsigned char addr[6];

> > -};

> > +} \_\_attribute\_\_((aligned(8)));

>

> Why "8"? Mustn't it be "16"? Address is to be 2-bytes aligned...

Actually it could be made "2", the aligned() attribute is in bytes, not bits.

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [xemul](#) on Wed, 18 Apr 2007 08:32:46 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

David Miller wrote:

> From: Pavel Emelianov <xemul@sw.ru>

> Date: Wed, 18 Apr 2007 10:43:56 +0400

>

>> [snip]

>>

>>> --- linux-2.6.orig/net/bridge/br\_private.h 2007-04-17

>>> 13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br\_private.h

>>> 2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@

>>> {

>>> unsigned char prio[2];

>>> unsigned char addr[6];

>>> -};

>>> +} \_\_attribute\_\_((aligned(8)));

>> Why "8"? Mustn't it be "16"? Address is to be 2-bytes aligned...

>

> Actually it could be made "2", the aligned() attribute is

> in bytes, not bits.

Indeed :) My bad :( Thank you...

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Stephen Hemminger](#) on Wed, 18 Apr 2007 14:44:39 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

On Wed, 18 Apr 2007 01:28:04 -0700 (PDT)

David Miller <davem@davemloft.net> wrote:

```
> From: Pavel Emelianov <xemul@sw.ru>
> Date: Wed, 18 Apr 2007 10:43:56 +0400
>
> > [snip]
> >
> > > --- linux-2.6.orig/net/bridge/br_private.h 2007-04-17
> > > 13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br_private.h
> > > 2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@
> > > {
> > > unsigned char prio[2];
> > > unsigned char addr[6];
> > > -};
> > > +} __attribute__((aligned(8)));
> >
> > Why "8"? Mustn't it be "16"? Address is to be 2-bytes aligned...
>
> Actually it could be made "2", the aligned() attribute is
> in bytes, not bits.
```

It could be 2 but 8 might allow a compiler on a 64 bit platform to be smarter in comparisons and assignments. For 2.6.22, I'll make a nicer version similar to ktime\_t.

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [davem](#) on Wed, 18 Apr 2007 20:04:22 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Stephen Hemminger <shemminger@linux-foundation.org>

Date: Wed, 18 Apr 2007 07:44:39 -0700

```
> On Wed, 18 Apr 2007 01:28:04 -0700 (PDT)
> David Miller <davem@davemloft.net> wrote:
>
> > From: Pavel Emelianov <xemul@sw.ru>
```

```

> > Date: Wed, 18 Apr 2007 10:43:56 +0400
> >
> > > [snip]
> > >
> > > > --- linux-2.6.orig/net/bridge/br_private.h 2007-04-17
> > > > 13:26:48.000000000 -0700 +++ linux-2.6/net/bridge/br_private.h
> > > > 2007-04-17 13:30:29.000000000 -0700 @@ -36,7 +36,7 @@
> > > > {
> > > > unsigned char prio[2];
> > > > unsigned char addr[6];
> > > > -};
> > > > +} __attribute__((aligned(8)));
> > >
> > > Why "8"? Mustn't it be "16"? Address is to be 2-bytes aligned...
> >
> > Actually it could be made "2", the aligned() attribute is
> > in bytes, not bits.
>
> It could be 2 but 8 might allow a compiler on a 64 bit platform
> to be smarter in comparisons and assignments.

```

Absolutely.

Although I don't think gcc does anything fancy since we don't use memcmp(). It's a tradeoff, we'd like to use unsigned long comparisons when both objects are aligned correctly but we also don't want it to use any more than one potentially mispredicted branch.

We could add some alignment tests to the ethernet address comparison code, but it's probably more trouble than it's worth.

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Eric Dumazet](#) on Thu, 19 Apr 2007 14:14:23 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

On Wed, 18 Apr 2007 13:04:22 -0700 (PDT)  
David Miller <davem@davemloft.net> wrote:

```

>
> Although I don't think gcc does anything fancy since we don't
> use memcmp(). It's a tradeoff, we'd like to use unsigned long
> comparisons when both objects are aligned correctly but we also
> don't want it to use any more than one potentially mispredicted
> branch.

```

Again, memcmp() \*cannot\* be optimized, because its semantic is to compare bytes.

memcpy() can take into account alignment if known at compile time, not memcmp()

<http://lists.openwall.net/netdev/2007/03/13/31>

>  
> We could add some alignment tests to the ethernet address  
> comparison code, but it's probably more trouble than it's  
> worth.  
> -  
> To unsubscribe from this list: send the line "unsubscribe netdev" in  
> the body of a message to majordomo@vger.kernel.org  
> More majordomo info at <http://vger.kernel.org/majordomo-info.html>  
>  
--  
Eric Dumazet <dada1@cosmosbay.com>

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Stephen Hemminger](#) on Thu, 19 Apr 2007 18:18:57 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

On Thu, 19 Apr 2007 16:14:23 +0200

Eric Dumazet <dada1@cosmosbay.com> wrote:

> On Wed, 18 Apr 2007 13:04:22 -0700 (PDT)  
> David Miller <davem@davemloft.net> wrote:  
>  
> >  
> > Although I don't think gcc does anything fancy since we don't  
> > use memcmp(). It's a tradeoff, we'd like to use unsigned long  
> > comparisons when both objects are aligned correctly but we also  
> > don't want it to use any more than one potentially mispredicted  
> > branch.  
>  
> Again, memcmp() \*cannot\* be optimized, because its semantic is to compare bytes.  
>  
> memcpy() can take into account alignment if known at compile time, not memcmp()  
>  
> <http://lists.openwall.net/netdev/2007/03/13/31>

It can if we order bytes in the bridge id properly. See ktime\_t for example.

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [davem](#) on Thu, 19 Apr 2007 20:01:01 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Eric Dumazet <dada1@cosmosbay.com>

Date: Thu, 19 Apr 2007 16:14:23 +0200

> On Wed, 18 Apr 2007 13:04:22 -0700 (PDT)

> David Miller <davem@davemloft.net> wrote:

>

> >

> > Although I don't think gcc does anything fancy since we don't

> > use memcmp(). It's a tradeoff, we'd like to use unsigned long

> > comparisons when both objects are aligned correctly but we also

> > don't want it to use any more than one potentially mispredicted

> > branch.

>

> Again, memcmp() \*cannot\* be optimized, because its semantic is to compare bytes.

>

> memcpy() can take into account alignment if known at compile time, not memcmp()

>

> <http://lists.openwall.net/netdev/2007/03/13/31>

I was prehaps thinking about strlen() where I know several implementations work a word at a time even though it is a byte-based operation:

-----  
#define LO\_MAGIC 0x01010101

#define HI\_MAGIC 0x80808080

...  
sethi %hi(HI\_MAGIC), %o4

...  
or %o4, %lo(HI\_MAGIC), %o3

...  
sethi %hi(LO\_MAGIC), %o4

...  
or %o4, %lo(LO\_MAGIC), %o2

...  
8:  
ld [%o0], %o5

2:  
sub %o5, %o2, %o4  
andcc %o4, %o3, %g0  
be,pt %icc, 8b  
add %o0, 4, %o0

-----

I figured some similar trick could be done with strcmp() and memcmp().

---

---

Subject: Re: [BRIDGE] Unaligned access on IA64 when comparing ethernet addresses

Posted by [Eric Dumazet](#) on Thu, 19 Apr 2007 20:29:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

```
> From: Eric Dumazet <dada1@cosmosbay.com>
> Date: Thu, 19 Apr 2007 16:14:23 +0200
>
>> On Wed, 18 Apr 2007 13:04:22 -0700 (PDT)
>> David Miller <davem@davemloft.net> wrote:
>>
>>> Although I don't think gcc does anything fancy since we don't
>>> use memcmp(). It's a tradeoff, we'd like to use unsigned long
>>> comparisons when both objects are aligned correctly but we also
>>> don't want it to use any more than one potentially mispredicted
>>> branch.
>> Again, memcmp() *cannot* be optimized, because its semantic is to compare bytes.
>>
>> memcpy() can take into account alignment if known at compile time, not memcmp()
>>
>> http://lists.openwall.net/netdev/2007/03/13/31
>
> I was prehaps thinking about strlen() where I know several
> implementations work a word at a time even though it is
> a byte-based operation:
>
> -----
> #define LO_MAGIC 0x01010101
> #define HI_MAGIC 0x80808080
> ...
> sethi %hi(HI_MAGIC), %o4
> ...
> or %o4, %lo(HI_MAGIC), %o3
> ...
> sethi %hi(LO_MAGIC), %o4
> ...
> or %o4, %lo(LO_MAGIC), %o2
> ...
> 8:
> ld [%o0], %o5
> 2:
> sub %o5, %o2, %o4
> andcc %o4, %o3, %g0
```

```

> be,pt %icc, 8b
> add %o0, 4, %o0
> -----
>
> I figured some similar trick could be done with strcmp() and
> memcmp().
>
>

```

Hum, I was refering to IA64 (or the more spreaded x86 arches), that is little endian AFAIK.

On big endian machines, a compiler can indeed perform some word tricks for memcmp() if it knows at compile time both pointers are word aligned.

PowerPc example (xlc compiler)

```

int func(const unsigned int *a, const unsigned int *b)
{
    return memcmp(a, b, 6);
}

```

```

.func:                                # 0x00000000 (H.10.NO_SYMBOL)
    l    r5,0(r3)
    l    r0,0(r4)
    cmp   0,r5,r0
    bc    BO_IF_NOT,CR0_EQ,___L2c
    lhz   r3,4(r3)
    lhz   r0,4(r4)
    sf    r0,r0,r3
    sfze  r3,r0
    a     r0,r3,r0
    aze   r3,r0
    bcr   BO_ALWAYS,CR0_LT
___L2c:                                # 0x0000002c (H.10.NO_SYMBOL+0x2c)
    sf    r0,r0,r5
    sfze  r3,r0
    a     r0,r3,r0
    aze   r3,r0
    bcr   BO_ALWAYS,CR0_LT

```

But to compare 6 bytes, known to be aligned to even addresses, current code is just fine and portable. We *could* use arch/endian specific tricks to save one or two cycles, but who really wants that ?