Subject: Re: Re: [PATCH 1/2] iptables 32bit compat layer
Posted by dim on Tue, 21 Feb 2006 09:04:49 GMT
View Forum Message <> Reply to Message

On Monday 20 February 2006 18:55, Arnd Bergmann wrote:
> On Monday 20 February 2006 09:10, Mishin Dmitry wrote:

> > 16:16:02.000000000 +0300 +++

> > +0300 struct xt_match

> > @@ -118,6 +125,10 @@ struct xt_match

> > +#ifdef CONFIG_COMPAT

> > convert); +#endif

>
> Is CONFIG_COMPAT the right conditional here? If the code is only used
> for architectures that have different aligments, it should not need be
> compiled in for the other architectures.
So, I'll define ARCH_HAS_FUNNY_64_ALIGNMENT in x86_64 and ia64 code and will
check it, as Andi suggested.

>
> > @@ -154,6 +165,10 @@ struct xt_target

> > +#ifdef CONFIG_COMPAT

> > convert); +#endif

> > @@ -233,6 +248,34 @@ extern void xt_proto_fini(int af);

> > +#ifdef CONFIG_COMPAT
> > +#include <net/compat.h>

> > +
> > +/* FIXME: this works only on 32 bit tasks
> > + * need to change whole approach in order to calculate align as function
> > of + * current task alignment */
> > +
> > +struct compat_xt_counters
> > +{

> > +};
>
> Hmm, maybe we should have something like
>
> typedef u64 __attribute__((aligned(4))) compat_u64;
>
> in order to get the right alignment on the architectures
> where it makes a difference. Do all compiler versions
> get that right?
good point. I don't know this and that's why tried to avoid use of 'aligned'
attribute.

>
> > ---

> > 16:06:41.000000000 +0300 +++

> > +0300 @@ -364,5 +365,62 @@ extern unsigned int ipt_do_table(struct

> > +
> > +#ifdef CONFIG_COMPAT
> > +#include <net/compat.h>
> > +
> > +struct compat_ipt_getinfo
> > +{

> > +};
>
> This structure looks like it does not need any
> conversions. You should probably just use
> struct ipt_getinfo then.
I just saw compat_uint_t use in net/compat.c and thought, that it is a good
style to use it. Does anybody know arch, where sizeof(compat_uint_t) != 4?

>
> > +
> > +struct compat_ipt_entry_match
> > +{

> > +};
> > +
> > +struct compat_ipt_entry_target
> > +{

> > +};
>
> Dito
Disagree, ipt_entry_match and ipt_entry_target contain pointers which make
their alignment equal 8 byte on 64bits architectures.

>

> > +
> > +extern int ipt_match_align_compat(void *match, void **dstptr,

> > +extern int ipt_target_align_compat(void *target, void **dstptr,

> > +
> > +#endif /* CONFIG_COMPAT */

> > @@ -23,6 +23,14 @@ struct compat_cmsghdr {

> > +#if defined(CONFIG_X86_64)
> > +#define is_current_32bits() (current_thread_info()->flags & _TIF_IA32)
> > +#elif defined(CONFIG_IA64)
> > +#define is_current_32bits() (IS_IA32_PROCESS(ia64_task_regs(current)))
> > +#else

> > +#endif
> > +
>
> This definition looks very wrong to me. For x86_64, the right thing to
> check should be TS_COMPAT, no _TIF_IA32, since you can also call the 64 bit
> syscall entry point from a i386 task running on x86_64. For most other
> architectures, is_current_32bits returns something that is not reflected in
> the name. I would e.g. expect the function to return '1' on i386 and the
> correct task state on other compat platforms, instead of a bogus '0'.
>
> There have been long discussions about the inclusions of the
> 'is_compat_task' macro. Let's at least not define a second function that
> does almost the same but gets it wrong.
>
> I would much rather have either an extra 'compat' argument to to
> sock_setsockopt and proto_ops->setsockopt than to spread the use
> of is_compat_task further.
Another weak place in my code. is_compat_task() approach has one advantage -
it doesn't require a lot of current code modifications.
>

> > @@ -308,107 +308,6 @@ void scm_detach_fds_compat(struct msghdr

> > - * For now, we assume that the compatibility and native version

> > - */
> > -struct compat_ipt_replace {

> > -};
>
> Is the FIXME above the only reason that the code needs to be changed?
> What is the reason that you did not just address this in the
> compat_sys_setsockopt implementation?
Code above doesn't work. iptables with version >= 1.3 does alignment checks as
well as kernel does. So, we can't simply put entries with 8 bytes alignment
to userspace or with 4 bytes alignment to kernel - we need translate them
entry by entry. So, I tried to do this the most correct way - that userspace
will hide its alignment from kernel and vice versa, with not only
SET_REPLACE, but also GET_INFO, GET_ENTRIES and SET_COUNTERS translation.
First implementation was exactly in compat_sys_setsockopt, but David asked me
to do this in netfilter code itself.

>
> Arnd <><
>
--
Thanks,
Dmitry.