Subject: Re: [RFC][PATCH 1/5] Virtualization/containers: startup Posted by Linus Torvalds on Fri, 03 Feb 2006 17:15:19 GMT

View Forum Message <> Reply to Message

On Fri, 3 Feb 2006, Kirill Korotaev wrote:

>

- > This patch introduces some abstract container/VPS kernel structure and tiny
- > amount of operations on it.

Please don't use things like "vps_t".

It's a _mistake_ to use typedef for structures and pointers. When you see a

vps_t a;

in the source, what does it mean?

In contrast, if it says

struct virtual container *a;

you can actually tell what "a" is.

Lots of people think that typedefs "help readability". Not so. They are useful only for

(a) totally opaque objects (where the typedef is actively used to _hide_ what the object is).

Example: "pte_t" etc opaque objects that you can only access using the proper accessor functions.

NOTE! Opaqueness and "accessor functions" are not good in themselves. The reason we have them for things like pte_t etc is that there really is absolutely zero portably accessible information there.

(b) Clear integer types, where the abstraction _helps_ avoid confusion whether it is "int" or "long".

u8/u16/u32 are perfectly fine typedefs.

NOTE! Again - there needs to be a _reason_ for this. If something is "unsigned long", then there's no reason to do

typedef long myflags_t;

but if there is a clear reason for why it under certain circumstances

might be an "unsigned int" and under other configurations might be "unsigned long", then by all means go ahead and use a typedef.

(c) when you use sparse to literally create a _new_ type for type-checking.

Maybe there are other cases too, but the rule should basically be to NEVER EVER use a typedef unless you can clearly match one of those rules.

In general, a pointer, or a struct that has elements that can reasonably be directly accessed should _never_ be a typedef.

Linus