## Subject: Re: [RFC][PATCH 2/2] Virtualization of IPC Posted by Dave Hansen on Fri, 24 Mar 2006 19:13:20 GMT

View Forum Message <> Reply to Message

On Fri, 2006-03-24 at 20:35 +0300, Kirill Korotaev wrote:

- > This patch introduces IPC namespaces, which allow to create isolated IPC
- > users or containers.
- > Introduces CONFIG\_IPC\_NS and ipc\_namespace structure.
- > It also uses current->ipc ns as a pointer to current namespace, which
- > reduces places where additional argument to functions should be added.

In three words, I think this has "too many #ifdefs".

The non-containerized or namespaced case should probably just be one, static namespace variable that gets wrapped up in some nice #ifdefed hlper functions.

For instance, instead of this:

```
+#ifdef CONFIG_IPC_NS
+#define msg_ids (*(current->ipc_ns->msg_ids))
+#endif

Have

#ifdef CONFIG_IPC_NS
static inline struct ipc_namespace *current_ipc_ns(void)
{
    return current->ipc_ns;
}
#else
static inline struct ipc_namespace *current_ipc_ns(void)
{
    return &static_ipc_ns;
}
#endif
```

And use current\_ipc\_ns()->msg\_ids. I can't imagine that gcc can't figure that out and turn it back into effectively the same thing.

I really dislike the idea of replacing nice variables with macros that add indirection. They really might fool people. Putting a function there is much nicer.

Why avoid to passing these things around as function arguments? Doesn't that make it more explicit what is going on, and where the indirection is occurring? Does it also make refcounting and lifetime issues easier to manage?

## BTW, Did you see my version of this?

-- Dave