Subject: Re: [PATCH 06/11] sysfs: Implement sysfs tagged directory support.
Posted by Tejun Heo on Sun, 29 Jun 2008 03:51:37 GMT
View Forum Message <> Reply to Message

Hello, Eric.

Eric W. Biederman wrote:
> Tejun thank you for the review, and my apologies for the delayed
> reply.

Me being the king of delays, no need for apologies. :-)

>> As before, I can't bring myself to like this interface. Is computing
>> tags dynamically really necessary? Can't we do the followings?
>
> It isn't so much computing tags dynamically but rather it is reading them
> from where they are stored.

It's still dynamic from sysfs's POV and I think that will make
maintenance more difficult.

>> tag = sysfs_allocate_tag(s);
>> sysfs_enable_tag(kobj (or sd), tag);
>> sysfs_sb_show_tag(sb, tag);
>>
>> Where tags are allocated using ida and each sb has bitmap of enabled
>> tags so that sysfs ops can simply use something like the following to
>> test whether it's enabled.
>>
>> bool sysfs_tag_enabled(sb, tag)
>> {
>> return sysfs_info(sb)->tag_map & (1 << tag);
>> }
>
>
> Youch that seems limiting. The expectation is that we could have
> as many as 100 different containers in use on a single system at one
> time. So 100 apparent copies of the network stack.

100 netns would mean 100 bits and 100 different views of them would mean
100 sb's where each sb would need bitmap larger than 100 bits. I don't
think there would be a scalability problem. Am I missing something?

> There is also a second dimension here we multiplex different
> directories based on different sets of tags. One directory based
> on user namespaces another on the network namespaces.

No matter which criteria is used to select ns, it should end up being

mapped to a set of tags (here, ida allocated numbers). Unless tags can change dynamically, there shouldn't be functional difference.

> The tags in practice are just pointers to the namespace pointers.
>
> So while we could use the ida technique to specify which set of tags
> we are talking about for a directory it isn't sufficient.

I failed to follow here. Can you please elaborate a bit? If you can describe a simple example to me, it would be much appreciated.

> The question sysfs_tag_enabled(sb, tag) makes no sense to me.
> Especially in the context of needed a sysfs_sb_show_tag(sb, tag);
>
> The current structure is because of all of the darn fool races and
> magic that sysfs does. We have to say for a given directory: Your
> contents will always be tagged, and only those that one tag that
> matches what was captured by the superblock when sysfs is mounted
> will be shown.

sysfs_tag_enabled() was meant to test whether a directory which is tagged should be shown under the current sb.

>> Tags which can change dynamically seems too confusing to me and it
>> makes things difficult to verify as it's unclear how those tags are
>> gonna to change.
>
> We have a fundamental issue that we have to handle, and it sounds like
> you are proposing something that will not handle it.
>
> - network devices can move between namespaces.
> - network devices have driver specific sysfs attributes hanging off of them.
>
> So we have to move the network devices and their sysfs attributes
> between namespaces, and I implemented that in kobject_rename,
> sysfs_rename path.
>
> The tags on a kobject can only change during a rename operation.
> So when the change happens is well defined. Further there is a
> set of functions: sysfs_creation_tag, sysfs_removal_tag,
> sysfs_lookup_tag, sysfs_dirent_tag which makes it clear what we
> are doing.
>
> If you really don't like how the tags are managed we need to talk
> about how we store the tags on kobjects and on the super block.
>
> Registering a set of tags could easily make the sb_tag function
> obsolete, and that is one small piece of code so it is no big deal.

>
> struct sysfs_tag_type_operations {
> const void *(*mount_tag)(void);
> const void *(*kobject_tag)(struct kobject *kobj);
> };
>
> Then we could do:
> struct sysfs_sbtag_operations *tag_type_ops[MAX_TAG_TYPES];
>
> And sysfs_tag_info could become.
> struct sysfs_tag_info {
> void *tag[MAX_TAG_TYPES];
> };
>
> During subsystem initialization we could call
> tag_type = sysfs_allocate_tag_type();
>
> Just after the subsystem creates a directory.
> sysfs_enable_tagging(kobj/sd, tag_type);
>
> Then anytime we currently call sb_tag during lookup we can instead
> just look at sysfs_info(sb)->tag[tag_type] and compare that with
> sd->s_tag.tag.

What you described is pretty much what I'm talking about. The only difference is whether to use caller-provided pointer as tag or an ida-allocated integer. The last sentence of the above paragraph is basically sys_tag_enabled() function (maybe misnamed).

The main reason why I'm whining about this so much is because I think tag should be something abstracted inside sysfs proper. It's something which affects very internal operation of sysfs and I really want to keep the implementation details inside sysfs. Spreading implementation over kobject and sysfs didn't turn out too pretty after all.

Thank you.

--
tejun