Subject: Re: [PATCH 06/11] sysfs: Implement sysfs tagged directory support.
Posted by Tejun Heo on Wed, 02 Jul 2008 03:24:42 GMT
View Forum Message <> Reply to Message

Hello,

Eric W. Biederman wrote:
> What we are implementing is not, a sb with a set of tags that are displayed,
> but directories with a single tag that is displayed. The sb just happens
> to hold the state for the directories.
>
> A directory displaying only a single tag is an necessary constraint for
> a large number of reasons.

Okay, that isn't exactly the impression I get but... well. Let's see.

>>> And array allows the lookup of the tag I am looking for before
>>> I search for the sd. An bitmap requires me to compare each entry.
>> How so? sysfs_sb->bitmap which contains enough bits for all the defined
>> tags and determining whether a sd should be shown or not is as simple as
>> single test_bit.
>
> Yes. The compare happens to be test_bit.
>
> With a bitmap you must visit each dirent with a given name and see if
> it has a tag that is displayed.
>
> With an array you can lookup the tag aprori and can potentially do a
> hash table lookup or a tree lookup and are not required to visit each
> entry.

A few things...

1. The lookup is currently done linearly and is fast enough for now.
Also, most lookup ops are cached by vfs layer. I'm not sure how
probable it is that we're gonna need hash or tree based sd lookup.

2. I don't think it's gonna be too difficult to speed up bitmap based
lookup. It would require a bit more intelligence but there's no
fundamental restriction. Just organizing the tree by tag first would
give us the same order of magnitude lookup given that the tags are used
the same way.

>> What I'm feeling unease about is the extra level of abstraction added by
>> tag types. A sd is given a tag. A sb shows a set of tags. The most
>> straight forward to implement that is to give sd a tag and test the tag
>> against sb's set of tags. The type is added because pointer tag
>> requires sequential matching which is usually best to avoid. It's

>> nothing fundamental. It's an extra baggage.
>
> That is just one important aspect of it. We need a way to describe
> which tag a sb,directory pair displays. It is a fundamental concept.

For netns, yes. I just think it would be better if the sysfs mechanism
to support that concept is more generic especially because it doesn't
seem too difficult to make it that way.

>>> Cause you to view an the tags as dynamic?
>> The thing is that I don't really see why there's tagged_dir_ops at all.
>
> We need callbacks for interfacing with the kobject layer, and for
> selecting our set of tags at mount time. Not tagged_dir_ops so much
> as tagged_type_ops.

The kobject op seems a bit strange way to interface to me. For mount,
yeah, we'll need a hook somewhere or pass it via mount option maybe.

>> What's needed is tagged sd's and sb's which can show subset of those
>> tags, so adding callback ops for tags just doesn't make much sense to
>> me. The interface should ideally be...
>
>> 1. alloc/release tag
>   Agreed.
>
>> 2. set / change / remove tag on sd
>   Essentially agreed.
>
>   Create an sd with a tag, change the tag on a sd.
>   Having an untagged sd in a directory that requires tags should
>   not be allowed.
>
>> 3. enable / disable tag on a sb
>   Disagree that is too flexible. Tags on a sb need to be
>   unchanging or else we get vfs layer issues.

Yeah, this really should be something which can't change once it's mounted.

>   Further the abstraction is logically exactly one tag on a
>   (sb,directory) pair.

I'm not so sure here. As a policy, maybe but I don't really see a
fundamental reason that the mechanism should enforce this.

>   The operations needed are.
>   - Select the set of tags on a sb (at mount time)
>   This requires we call a set of callbacks. [My mount_sb callback]

>
>   - release a tag (which implies removing all tagged entries and
>   removing the sb reference)
>
> 4. Interface with the kobject layer.
>   kobject_add calls sysfs_create_dir
>   kboject_rename calls sysfs_rename_dir
>   kobject_del calls sysfs_remove_dir
>
>   For the first two operations we need a helper function to go from a
>   kobject to a tag.

Why not just add a parameter to sysfs_create_dir()? It's just twisted.

>   For the second two operations we need to go from a kobject to a sd.
>
>> This has been my opinion from the beginning. Unless the tags need to be
>> changed dynamically on demand (which I hope is not the case), there just
>> is plainly no reason to have callbacks for tags.
>
> We don't need callbacks to poll to see if the tags on a sd have
> changed.
>
> We need helper functions for interfacing with the rest of the kernel.

Yes, that's why I view it as strange. These can be done in forward way
(by passing in mount options and/or arguments) but it's done by first
going into the sysfs and then calling back out to outer layer.

Thanks.

--
tejun