Subject: Re: [PATCH] Send quota messages via netlink
Posted by akpm on Wed, 29 Aug 2007 04:13:35 GMT
View Forum Message <> Reply to Message

On Tue, 28 Aug 2007 16:13:18 +0200 Jan Kara <jack@suse.cz> wrote:

```
>   Hello,
>
>   I'm sending rediffed patch implementing sending of quota messages via netlink
> interface (some rationale in patch description). I've already posted it to
> LKML some time ago and there were no objections, so I guess it's fine to put
> it to -mm. Andrew, would you be so kind? Thanks.
>   Userspace deamon reading the messages from the kernel and sending them to
> dbus and/or user console is also written (it's part of quota-tools). The
> only remaining problem is there are a few changes needed to libnl needed for
> the userspace daemon. They were basically acked by the maintainer but it
> seems he has not merged the patches yet. So this will take a bit more time.
>
```

So it's a new kernel->userspace interface.

But we have no description of the interface :(

```
> +/* Send warning to userspace about user which exceeded quota */
> +static void send_warning(const struct dquot *dquot, const char warntype)
> +{
> + static unsigned long seq;
> + struct sk_buff *skb;
> + void *msg_head;
> + int ret;
> +
> + skb = genlmsg_new(QUOTA_NL_MSG_SIZE, GFP_NOFS);
> + if (!skb) {
> +  printk(KERN_ERR
> +    "VFS: Not enough memory to send quota warning.\n");
> +  return;
> + }
> + msg_head = genlmsg_put(skb, 0, seq++, &quota_genl_family, 0, QUOTA_NL_C_WARNING);
> + if (!msg_head) {
> +  printk(KERN_ERR
> +    "VFS: Cannot store netlink header in quota warning.\n");
> +  goto err_out;
> + }
> + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
> + if (ret)
> +  goto attr_err_out;
> + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
> + if (ret)
> +  goto attr_err_out;
> + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
> + if (ret)
> +  goto attr_err_out;
> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
> +  MAJOR(dquot->dq_sb->s_dev));
> + if (ret)
> +  goto attr_err_out;
> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
> +  MINOR(dquot->dq_sb->s_dev));
> + if (ret)
> +  goto attr_err_out;
> + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
> + if (ret)
> +  goto attr_err_out;
> + genlmsg_end(skb, msg_head);
> +
> + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
> + if (ret < 0 && ret != -ESRCH)
> +  printk(KERN_ERR
> +   "VFS: Failed to send notification message: %d\n", ret);
> + return;
> +attr_err_out:
> + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
> +err_out:
> + kfree_skb(skb);
> +}
> +#endif
```

This is it. Normally netlink payloads are represented as a struct. How come this one is built-by-hand?

It doesn't appear to be versioned. Should it be?

Does it have (or need) reserved-set-to-zero space for expansion? Again, hard to tell..

I guess it's OK to send a major and minor out of the kernel like this. What's it for? To represent a filesytem? I wonder if there's a more modern and useful way of describing the fs. Path to mountpoint or something?

I suspect the namespace virtualisation guys would be interested in a new interface which is sending current->user->uid up to userspace. uids are per-namespace now. What are the implications? (cc's added)

Is it worth adding a comment explaining why GFP_NOFS is used here?

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Send quota messages via netlink
Posted by davem on Wed, 29 Aug 2007 04:54:45 GMT

View Forum Message <> Reply to Message

From: Andrew Morton <akpm@linux-foundation.org>
Date: Tue, 28 Aug 2007 21:13:35 -0700

> This is it. Normally netlink payloads are represented as a struct. How
> come this one is built-by-hand?

He is using attributes, which is perfect and arbitrarily
extensible with zero backwards compatability concerns.

If he wants to provide a new attribute, he just adds it
without any issues.

When new attributes are added, older apps simply ignore the attributes
they don't understand.

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Send quota messages via netlink
Posted by ebiederm on Wed, 29 Aug 2007 05:41:42 GMT

View Forum Message <> Reply to Message

Andrew Morton <akpm@linux-foundation.org> writes:

```
> On Tue, 28 Aug 2007 16:13:18 +0200 Jan Kara <jack@suse.cz> wrote:
>
>>   Hello,
>>
>> I'm sending rediffed patch implementing sending of quota messages via netlink
>> interface (some rationale in patch description). I've already posted it to
>> LKML some time ago and there were no objections, so I guess it's fine to put
>> it to -mm. Andrew, would you be so kind? Thanks.
>>   Userspace deamon reading the messages from the kernel and sending them to
>> dbus and/or user console is also written (it's part of quota-tools). The
>> only remaining problem is there are a few changes needed to libnl needed for
>> the userspace daemon. They were basically acked by the maintainer but it
>> seems he has not merged the patches yet. So this will take a bit more time.
>>
>
> So it's a new kernel->userspace interface.
>
> But we have no description of the interface :(
>
>> +/* Send warning to userspace about user which exceeded quota */
>> +static void send_warning(const struct dquot *dquot, const char warntype)
>> +{
>> + static unsigned long seq;
>> + struct sk_buff *skb;
>> + void *msg_head;
>> + int ret;
>> +
>> + skb = genlmsg_new(QUOTA_NL_MSG_SIZE, GFP_NOFS);
>> + if (!skb) {
>> +  printk(KERN_ERR
>> +    "VFS: Not enough memory to send quota warning.\n");
>> +  return;
>> + }
>> + msg_head = genlmsg_put(skb, 0, seq++, &quota_genl_family, 0,
> QUOTA_NL_C_WARNING);
>> + if (!msg_head) {
>> +  printk(KERN_ERR
>> +    "VFS: Cannot store netlink header in quota warning.\n");
>> +  goto err_out;
>> + }
>> + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
>> +  MAJOR(dquot->dq_sb->s_dev));
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
>> +  MINOR(dquot->dq_sb->s_dev));
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
>> + if (ret)
>> +  goto attr_err_out;
>> + genlmsg_end(skb, msg_head);
>> +
>> + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
>> + if (ret < 0 && ret != -ESRCH)
>> +  printk(KERN_ERR
>> +   "VFS: Failed to send notification message: %d\n", ret);
>> + return;
>> +attr_err_out:
>> + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
>> +err_out:
>> + kfree_skb(skb);
>> +}
>> +#endif
>
```

> This is it. Normally netlink payloads are represented as a struct. How
> come this one is built-by-hand?

No netlink fields (unless I'm confused) are represented as a struct,
not the entire netlink payload.

> It doesn't appear to be versioned. Should it be?

Well. If it is using netlink properly each field should have a tag.
So it should not need to be versioned, because each field is strictly
controlled.

> Does it have (or need) reserved-set-to-zero space for expansion? Again,
> hard to tell..

Not if netlink is used properly. Just another nested tag.

> I guess it's OK to send a major and minor out of the kernel like this.
> What's it for? To represent a filesytem? I wonder if there's a more
> modern and useful way of describing the fs. Path to mountpoint or
> something?

Or perhaps the string the fs was mounted with.

> I suspect the namespace virtualisation guys would be interested in a new
> interface which is sending current->user->uid up to userspace. uids are
> per-namespace now. What are the implications? (cc's added)

That we definitely would be. Although the user namespaces is rather
strongly incomplete at the moment.

> Is it worth adding a comment explaining why GFP_NOFS is used here?

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Balbir Singh on Wed, 29 Aug 2007 06:30:07 GMT
View Forum Message <> Reply to Message

Andrew Morton wrote:
> On Tue, 28 Aug 2007 16:13:18 +0200 Jan Kara <jack@suse.cz> wrote:
>
>>   Hello,
>>
>>   I'm sending rediffed patch implementing sending of quota messages via netlink
>> interface (some rationale in patch description). I've already posted it to
>> LKML some time ago and there were no objections, so I guess it's fine to put
>> it to -mm. Andrew, would you be so kind? Thanks.
>>   Userspace deamon reading the messages from the kernel and sending them to
>> dbus and/or user console is also written (it's part of quota-tools). The
>> only remaining problem is there are a few changes needed to libnl needed for
>> the userspace daemon. They were basically acked by the maintainer but it
>> seems he has not merged the patches yet. So this will take a bit more time.
>>
>
> So it's a new kernel->userspace interface.
>
> But we have no description of the interface :(
>

And could we have some description of the context under which all the message
exchanges take place. When are these messages sent out -- what event
is the user space notified of?

```
>> +/* Send warning to userspace about user which exceeded quota */
>> +static void send_warning(const struct dquot *dquot, const char warntype)
>> +{
>> + static unsigned long seq;
>> + struct sk_buff *skb;
>> + void *msg_head;
>> + int ret;
>> +
>> + skb = genlmsg_new(QUOTA_NL_MSG_SIZE, GFP_NOFS);
>> + if (!skb) {
>> +  printk(KERN_ERR
>> +    "VFS: Not enough memory to send quota warning.\n");
>> +  return;
>> + }
>> + msg_head = genlmsg_put(skb, 0, seq++, &quota_genl_family, 0,
QUOTA_NL_C_WARNING);
>> + if (!msg_head) {
>> +  printk(KERN_ERR
>> +    "VFS: Cannot store netlink header in quota warning.\n");
>> +  goto err_out;
```

One problem, we've been is losing notifications. It does not happen for us
due to the cpumask interface (which allows us to have parallel sockets
for each cpu or a set of cpus). How frequent are your notifications?

```
>> + }
>> + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
>> +  MAJOR(dquot->dq_sb->s_dev));
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
>> +  MINOR(dquot->dq_sb->s_dev));
>> + if (ret)
>> +  goto attr_err_out;
>> + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
>> + if (ret)
>> +  goto attr_err_out;
>> + genlmsg_end(skb, msg_head);
>> +
```

Have you looked at ensuring that the data structure works across 32 bit
and 64 bit systems (in terms of binary compatibility)? That's usually
a nice to have feature.

```
>> + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
>> + if (ret < 0 && ret != -ESRCH)
>> +  printk(KERN_ERR
>> +   "VFS: Failed to send notification message: %d\n", ret);
>> + return;
>> +attr_err_out:
>> + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
>> +err_out:
```

```
>> + kfree_skb(skb);
>> +}
>> +#endif
>
> This is it.  Normally netlink payloads are represented as a struct.  How
> come this one is built-by-hand?
>
> It doesn't appear to be versioned.  Should it be?
>
```

Yes, versioning is always nice and genetlink supports it.

```
> Does it have (or need) reserved-set-to-zero space for expansion?  Again,
> hard to tell..
>
> I guess it's OK to send a major and minor out of the kernel like this.
> What's it for?  To represent a filesytem?  I wonder if there's a more
> modern and useful way of describing the fs.  Path to mountpoint or
> something?
>
> I suspect the namespace virtualisation guys would be interested in a new
> interface which is sending current->user->uid up to userspace.  uids are
> per-namespace now.  What are the implications?  (cc's added)
>
```

The memory controller or VM would also be interested in notifications
of OOM. At OLS this year interest was shown in getting OOM notifications
and allow the user space a chance to handle the notification and take
action (especially for containers). We already have containerstats for
containers (which I was planning to reuse), but I was told that we would
be interested in user space OOM notifications in general.

```
> Is it worth adding a comment explaining why GFP_NOFS is used here?
>
>
```

--
 Warm Regards,
 Balbir Singh
 Linux Technology Center
 IBM, ISTL

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Jan Kara on Wed, 29 Aug 2007 12:26:47 GMT
View Forum Message <> Reply to Message

On Tue 28-08-07 21:13:35, Andrew Morton wrote:
> On Tue, 28 Aug 2007 16:13:18 +0200 Jan Kara <jack@suse.cz> wrote:
>
> >   Hello,
> >
> >   I'm sending rediffed patch implementing sending of quota messages via netlink
> > interface (some rationale in patch description). I've already posted it to
> > LKML some time ago and there were no objections, so I guess it's fine to put
> > it to -mm. Andrew, would you be so kind? Thanks.
> >   Userspace deamon reading the messages from the kernel and sending them to
> > dbus and/or user console is also written (it's part of quota-tools). The
> > only remaining problem is there are a few changes needed to libnl needed for
> > the userspace daemon. They were basically acked by the maintainer but it
> > seems he has not merged the patches yet. So this will take a bit more time.
> >
>
> So it's a new kernel->userspace interface.
>
> But we have no description of the interface :(

  Oops, forgotten about it. I'll write one. Do we have some standard place
where to document such interfaces? I could create some file in
Documentation/filesystems/ but that seems a bit superfluous...

```
> > +/* Send warning to userspace about user which exceeded quota */
> > +static void send_warning(const struct dquot *dquot, const char warntype)
> > +{
> > + static unsigned long seq;
> > + struct sk_buff *skb;
> > + void *msg_head;
> > + int ret;
> > +
> > + skb = genlmsg_new(QUOTA_NL_MSG_SIZE, GFP_NOFS);
> > + if (!skb) {
> > +  printk(KERN_ERR
> > +    "VFS: Not enough memory to send quota warning.\n");
> > +  return;
> > + }
> > + msg_head = genlmsg_put(skb, 0, seq++, &quota_genl_family, 0,
QUOTA_NL_C_WARNING);
> > + if (!msg_head) {
> > +  printk(KERN_ERR
> > +    "VFS: Cannot store netlink header in quota warning.\n");
> > +  goto err_out;
> > + }
> > + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
> > + if (ret)
> > +  goto attr_err_out;
> > + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
> > + if (ret)
> > +  goto attr_err_out;
> > + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
> > + if (ret)
> > +  goto attr_err_out;
> > + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
> > +  MAJOR(dquot->dq_sb->s_dev));
> > + if (ret)
> > +  goto attr_err_out;
> > + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
> > +  MINOR(dquot->dq_sb->s_dev));
> > + if (ret)
> > +  goto attr_err_out;
> > + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
> > + if (ret)
> > +  goto attr_err_out;
> > + genlmsg_end(skb, msg_head);
> > +
> > + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
> > + if (ret < 0 && ret != -ESRCH)
> > +  printk(KERN_ERR
> > +   "VFS: Failed to send notification message: %d\n", ret);
> > + return;
> > +attr_err_out:
> > + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
> > +err_out:
> > + kfree_skb(skb);
> > +}
> > +#endif
>
```

> This is it. Normally netlink payloads are represented as a struct. How
> come this one is built-by-hand?

I use "generic netlink", which is in fact a layer built on top of netlink. As far as I've read it's documentation, creating a message argument by argument is the preferred way. As David writes, this way we can add new arguments without worries about backward compatibility, alignment issues or such things.

> It doesn't appear to be versioned. Should it be?

We don't need a version for future additions. Also each attribute sent has its identifier (e.g. QUOTA_NL_A_CAUSED_ID) and userspace checks these identifiers and unknown attributes are ignored. But in case we would like to remove some attribute, versioning would be probably useful so that userspace won't break silently... So I'll add it.

> Does it have (or need) reserved-set-to-zero space for expansion? Again,
> hard to tell..
  No, we don't need it as I wrote above.

> I guess it's OK to send a major and minor out of the kernel like this.
> What's it for? To represent a filesytem? I wonder if there's a more
> modern and useful way of describing the fs. Path to mountpoint or
> something?
  I also find major/minor pair a bit old-fashioned. But the identifying it
by a mountpoint is problematic - quota does not care about namespaces and
such and so it works with superblocks. It's not trivial to get a mountpoint
from a superblock (and generally it's frown upon, isn't it?). Also if a
filesystem is mounted on several places, we have to pick one (OK, userspace
has to do this choice anyway when displaying the message but still...).

> I suspect the namespace virtualisation guys would be interested in a new
> interface which is sending current->user->uid up to userspace. uids are
> per-namespace now. What are the implications? (cc's added)
  I know there's something going on in this area but I don't know any
details. If somebody has some advice what should be passed into userspace
so that user/group can be idenitified, it is welcome.

> Is it worth adding a comment explaining why GFP_NOFS is used here?
  Probably yes. Added.

  Thanks for all your comments.

        Honza
--
Jan Kara <jack@suse.cz>
SuSE CR Labs

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Jan Kara on Wed, 29 Aug 2007 12:46:15 GMT

View Forum Message <> Reply to Message

On Wed 29-08-07 12:00:07, Balbir Singh wrote:
> Andrew Morton wrote:
> > On Tue, 28 Aug 2007 16:13:18 +0200 Jan Kara <jack@suse.cz> wrote:
> >>   I'm sending rediffed patch implementing sending of quota messages via netlink
> >> interface (some rationale in patch description). I've already posted it to
> >> LKML some time ago and there were no objections, so I guess it's fine to put
> >> it to -mm. Andrew, would you be so kind? Thanks.

> >>   Userspace deamon reading the messages from the kernel and sending them to
> >> dbus and/or user console is also written (it's part of quota-tools). The
> >> only remaining problem is there are a few changes needed to libnl needed for
> >> the userspace daemon. They were basically acked by the maintainer but it
> >> seems he has not merged the patches yet. So this will take a bit more time.
> >>
> >
> > So it's a new kernel->userspace interface.
> >
> > But we have no description of the interface :(
> >
>
> And could we have some description of the context under which all the message
> exchanges take place. When are these messages sent out -- what event
> is the user space notified of?

The user is notified about either exceeding his quota softlimit or
reaching hardlimit. If you are interested in more details, please ask.

```
> >> +/* Send warning to userspace about user which exceeded quota */
> >> +static void send_warning(const struct dquot *dquot, const char warntype)
> >> +{
> >> + static unsigned long seq;
> >> + struct sk_buff *skb;
> >> + void *msg_head;
> >> + int ret;
> >> +
> >> + skb = genlmsg_new(QUOTA_NL_MSG_SIZE, GFP_NOFS);
> >> + if (!skb) {
> >> +  printk(KERN_ERR
> >> +    "VFS: Not enough memory to send quota warning.\n");
> >> +  return;
> >> + }
> >> + msg_head = genlmsg_put(skb, 0, seq++, &quota_genl_family, 0,
QUOTA_NL_C_WARNING);
> >> + if (!msg_head) {
> >> +  printk(KERN_ERR
> >> +    "VFS: Cannot store netlink header in quota warning.\n");
> >> +  goto err_out;
>
```

> One problem, we've been is losing notifications. It does not happen for us
> due to the cpumask interface (which allows us to have parallel sockets
> for each cpu or a set of cpus). How frequent are your notifications?

Quite infrequent... Users won't exceed their quotas too often :).

```
> >> + }
> >> + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
> >> + if (ret)
> >> +  goto attr_err_out;
> >> + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
> >> + if (ret)
> >> +  goto attr_err_out;
> >> + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
> >> + if (ret)
> >> +  goto attr_err_out;
> >> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
> >> +  MAJOR(dquot->dq_sb->s_dev));
> >> + if (ret)
> >> +  goto attr_err_out;
> >> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
> >> +  MINOR(dquot->dq_sb->s_dev));
> >> + if (ret)
> >> +  goto attr_err_out;
> >> + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
> >> + if (ret)
> >> +  goto attr_err_out;
> >> + genlmsg_end(skb, msg_head);
> >> +
>
```

> Have you looked at ensuring that the data structure works across 32 bit
> and 64 bit systems (in terms of binary compatibility)? That's usually
> a nice to have feature.

Generic netlink should take care of this - arguments are typed so it
knows how much bits numbers have. So this should be no issue. Are there any
other problems that you have in mind?

```
> >> + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
> >> + if (ret < 0 && ret != -ESRCH)
> >> +  printk(KERN_ERR
> >> +   "VFS: Failed to send notification message: %d\n", ret);
> >> + return;
> >> +attr_err_out:
> >> + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
> >> +err_out:
> >> + kfree_skb(skb);
> >> +}
> >> +#endif
> >
```

> > This is it. Normally netlink payloads are represented as a struct. How
> > come this one is built-by-hand?
> >
> > It doesn't appear to be versioned. Should it be?
> >
>
> Yes, versioning is always nice and genetlink supports it.
>
> > Does it have (or need) reserved-set-to-zero space for expansion? Again,

> > hard to tell..
> >
> > I guess it's OK to send a major and minor out of the kernel like this.
> > What's it for? To represent a filesytem? I wonder if there's a more
> > modern and useful way of describing the fs. Path to mountpoint or
> > something?
> >
> > I suspect the namespace virtualisation guys would be interested in a new
> > interface which is sending current->user->uid up to userspace. uids are
> > per-namespace now. What are the implications? (cc's added)
>
> The memory controller or VM would also be interested in notifications
> of OOM. At OLS this year interest was shown in getting OOM notifications
> and allow the user space a chance to handle the notification and take
> action (especially for containers). We already have containerstats for
> containers (which I was planning to reuse), but I was told that we would
> be interested in user space OOM notifications in general.

Generic netlink can be used to pass this information (although in OOM
situation, it may be a bit hairy to get the network stack working...). But
I guess it's not related to my patch.

Honza

--
Jan Kara <jack@suse.cz>
SuSE CR Labs

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Balbir Singh on Fri, 31 Aug 2007 06:59:53 GMT

View Forum Message <> Reply to Message

Jan Kara wrote:

```
>>>> + }
>>>> + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
>>>> + if (ret)
>>>> +  goto attr_err_out;
>>>> + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
>>>> + if (ret)
>>>> +  goto attr_err_out;
>>>> + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
>>>> + if (ret)
>>>> +  goto attr_err_out;
>>>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
>>>> +  MAJOR(dquot->dq_sb->s_dev));
```

```
>>>> + if (ret)
>>>> +  goto attr_err_out;
>>>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
>>>> +  MINOR(dquot->dq_sb->s_dev));
>>>> + if (ret)
>>>> +  goto attr_err_out;
>>>> + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
>>>> + if (ret)
>>>> +  goto attr_err_out;
>>>> + genlmsg_end(skb, msg_head);
>>>> +
```

>> Have you looked at ensuring that the data structure works across 32 bit
>> and 64 bit systems (in terms of binary compatibility)? That's usually
>> a nice to have feature.
>   Generic netlink should take care of this - arguments are typed so it
> knows how much bits numbers have. So this should be no issue. Are there any
> other problems that you have in mind?
>

Yes, but apart from that, if I remember Jamal Hadi's initial comments
on taskstats, he recommended that we align everything to 64 bit so
that the data is well aligned for 64 bit systems. You could also consider
creating a data structure, document it's members, align them and use
that to send out the data.

```
>>>> + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
>>>> + if (ret < 0 && ret != -ESRCH)
>>>> +  printk(KERN_ERR
>>>> +   "VFS: Failed to send notification message: %d\n", ret);
>>>> + return;
>>>> +attr_err_out:
>>>> + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
>>>> +err_out:
>>>> + kfree_skb(skb);
>>>> +}
>>>> +#endif
```

>>> This is it. Normally netlink payloads are represented as a struct. How
>>> come this one is built-by-hand?
>>>
>>> It doesn't appear to be versioned. Should it be?
>>>
>> Yes, versioning is always nice and genetlink supports it.
>>

It would nice for you to use the versioning feature.

>> The memory controller or VM would also be interested in notifications
>> of OOM. At OLS this year interest was shown in getting OOM notifications

>> and allow the user space a chance to handle the notification and take
>> action (especially for containers). We already have containerstats for
>> containers (which I was planning to reuse), but I was told that we would
>> be interested in user space OOM notifications in general.

> Generic netlink can be used to pass this information (although in OOM
> situation, it may be a bit hairy to get the network stack working...). But
> I guess it's not related to my patch.

We could have a pre-allocated buffer stored at startup and use that for
OOM notification. In the case of container OOM, we are likely to have
free global memory. Working towards an infrastructure so that anybody can
build on top of it and sending notifications on interesting events becomes
easier would be nice. We can reuse code that way and add fewer bugs :-)

--
 Warm Regards,
 Balbir Singh
 Linux Technology Center
 IBM, ISTL

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Jan Kara on Mon, 03 Sep 2007 10:18:54 GMT

View Forum Message <> Reply to Message

On Fri 31-08-07 12:29:53, Balbir Singh wrote:
> Jan Kara wrote:

```
> >>>> + }
> >>>> + ret = nla_put_u32(skb, QUOTA_NL_A_QTYPE, dquot->dq_type);
> >>>> + if (ret)
> >>>> +  goto attr_err_out;
> >>>> + ret = nla_put_u64(skb, QUOTA_NL_A_EXCESS_ID, dquot->dq_id);
> >>>> + if (ret)
> >>>> +  goto attr_err_out;
> >>>> + ret = nla_put_u32(skb, QUOTA_NL_A_WARNING, warntype);
> >>>> + if (ret)
> >>>> +  goto attr_err_out;
> >>>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MAJOR,
> >>>> +  MAJOR(dquot->dq_sb->s_dev));
> >>>> + if (ret)
> >>>> +  goto attr_err_out;
> >>>> + ret = nla_put_u32(skb, QUOTA_NL_A_DEV_MINOR,
```

> >>>> + MINOR(dquot->dq_sb->s_dev));
> >>>> + if (ret)
> >>>> + goto attr_err_out;
> >>>> + ret = nla_put_u64(skb, QUOTA_NL_A_CAUSED_ID, current->user->uid);
> >>>> + if (ret)
> >>>> + goto attr_err_out;
> >>>> + genlmsg_end(skb, msg_head);
> >>>> +
> >> Have you looked at ensuring that the data structure works across 32 bit
> >> and 64 bit systems (in terms of binary compatibility)? That's usually
> >> a nice to have feature.
> >   Generic netlink should take care of this - arguments are typed so it
> > knows how much bits numbers have. So this should be no issue. Are there any
> > other problems that you have in mind?
> >
> Yes, but apart from that, if I remember Jamal Hadi's initial comments
> on taskstats, he recommended that we align everything to 64 bit so
> that the data is well aligned for 64 bit systems. You could also consider
  But each attribute is just one number (either 32 or 64 bit) so there's
not much to align. Also each attribute has its netlink header so alignment
is anyway hard to predict. Finally, this is by no means performance
critical - average system using quotas may get say 1 notification per user
per month?

> creating a data structure, document it's members, align them and use
> that to send out the data.
  I don't like sending one structure - by doing that you loose the
flexibility of netlink attributes...

> >>>> + ret = genlmsg_multicast(skb, 0, quota_genl_family.id, GFP_NOFS);
> >>>> + if (ret < 0 && ret != -ESRCH)
> >>>> + printk(KERN_ERR
> >>>> +   "VFS: Failed to send notification message: %d\n", ret);
> >>>> + return;
> >>>> +attr_err_out:
> >>>> + printk(KERN_ERR "VFS: Failed to compose quota message: %d\n", ret);
> >>>> +err_out:
> >>>> + kfree_skb(skb);
> >>>> +}
> >>>> +#endif
> >>> This is it. Normally netlink payloads are represented as a struct. How
> >>> come this one is built-by-hand?
> >>>
> >>> It doesn't appear to be versioned. Should it be?
> >>>
> >> Yes, versioning is always nice and genetlink supports it.
> >>
> It would nice for you to use the versioning feature.

How does generic netlink support versioning? I have not found this
feature. Looking into Documentation/accounting/taskstats.txt it seems that
taskstats are versioning only the structure taskstats itself but not the
buch of attributes as a whole...

> >> The memory controller or VM would also be interested in notifications
> >> of OOM. At OLS this year interest was shown in getting OOM notifications
> >> and allow the user space a chance to handle the notification and take
> >> action (especially for containers). We already have containerstats for
> >> containers (which I was planning to reuse), but I was told that we would
> >> be interested in user space OOM notifications in general.
>
> > Generic netlink can be used to pass this information (although in OOM
> > situation, it may be a bit hairy to get the network stack working...). But
> > I guess it's not related to my patch.
>
> We could have a pre-allocated buffer stored at startup and use that for
> OOM notification. In the case of container OOM, we are likely to have
> free global memory. Working towards an infrastructure so that anybody can
> build on top of it and sending notifications on interesting events becomes
> easier would be nice. We can reuse code that way and add fewer bugs :-)

Yes, but generic netlink itself is such an infrastructure, isn't it? It
is about 70 lines of code to implement notification for quota subsystem so
it's really simple...

Honza

--
Jan Kara <jack@suse.cz>
SuSE CR Labs