Subject: Re: [PATCH 3/8] CGroup Files: Move the release_agent file to use typed handlers
Posted by akpm on Tue, 24 Jun 2008 23:23:25 GMT
View Forum Message <> Reply to Message

On Fri, 20 Jun 2008 16:44:01 -0700
menage@google.com wrote:

> Adds cgroup_release_agent_write() and cgroup_release_agent_show()
> methods to handle writing/reading the path to a cgroup hierarchy's
> release agent. As a result, cgroup_common_file_read() is now unnecessary.
>
> As part of the change, a previously-tolerated race in
> cgroup_release_agent() is avoided by copying the current
> release_agent_path prior to calling call_usermode_helper().
>
> Signed-off-by: Paul Menage <menage@google.com>
>
> ---
> include/linux/cgroup.h | 2
> kernel/cgroup.c | 125 ++++++++++++++++++++++++++----------
> 2 files changed, 59 insertions(+), 68 deletions(-)
>
> Index: cws-2.6.26-rc5-mm3/kernel/cgroup.c
> ==========
> --- cws-2.6.26-rc5-mm3.orig/kernel/cgroup.c
> +++ cws-2.6.26-rc5-mm3/kernel/cgroup.c
> @@ -89,11 +89,7 @@ struct cgroupfs_root {
> /* Hierarchy-specific flags */
> unsigned long flags;
>
> - /* The path to use for release notifications. No locking
> - * between setting and use - so if userspace updates this
> - * while child cgroups exist, you could miss a
> - * notification. We ensure that it's always a valid
> - * NUL-terminated string */
> + /* The path to use for release notifications. */
> char release_agent_path[PATH_MAX];
> };
>
> @@ -1329,6 +1325,45 @@ enum cgroup_filetype {
> FILE_RELEASE_AGENT,
> };
>
> +/**
> + * cgroup_lock_live_group - take cgroup_mutex and check that cgrp is alive.
> + * @cgrp: the cgroup to be checked for liveness
> + *

```
> + * Returns true (with lock held) on success, or false (with no lock
> + * held) on failure.
> + */
> +int cgroup_lock_live_group(struct cgroup *cgrp)
> +{
> + mutex_lock(&cgroup_mutex);
> + if (cgroup_is_removed(cgrp)) {
> +  mutex_unlock(&cgroup_mutex);
> +  return false;
> + }
> + return true;
> +}
```

I think that if we're going to do this it would be nice to add a symmetrical cgroup_unlock_live_group()?

Because code like this:

```
> + if (!cgroup_lock_live_group(cgrp))
> +  return -ENODEV;
> + strcpy(cgrp->root->release_agent_path, buffer);
> + mutex_unlock(&cgroup_mutex);
```

is a bit WTFish, no? it forces each caller of cgroup_lock_live_group() to know about cgroup_lock_live_group() internals.

That would be kind of OKayish if this code was closely localised, but...

```
> --- cws-2.6.26-rc5-mm3.orig/include/linux/cgroup.h
> +++ cws-2.6.26-rc5-mm3/include/linux/cgroup.h
> @@ -295,6 +295,8 @@ int cgroup_add_files(struct cgroup *cgrp
>
>  int cgroup_is_removed(const struct cgroup *cgrp);
>
> +int cgroup_lock_live_group(struct cgroup *cgrp);
> +
>  int cgroup_path(const struct cgroup *cgrp, char *buf, int buflen);
>
>  int cgroup_task_count(const struct cgroup *cgrp);
>
```

I assume this gets used in another .c file in a later patch.

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