Subject: [PATCH 09/10] CGroup API files: Move "releasable" to cgroup_debug subsystem
Posted by Paul Menage on Sat, 23 Feb 2008 22:47:34 GMT
View Forum Message <> Reply to Message

The "releasable" control file provided by the cgroup framework exports
the state of a per-cgroup flag that's related to the notify-on-release
feature. This isn't really generally useful, unless you're trying to
debug this particular feature of cgroups.

This patch moves the "releasable" file to the cgroup_debug subsystem.

Signed-off-by: Paul Menage <menage@google.com>

---
 include/linux/cgroup.h | 11 ++++++++++
 kernel/cgroup.c | 23 ----------------------
 kernel/cgroup_debug.c | 12 ++++++++++-
 3 files changed, 22 insertions(+), 24 deletions(-)

Index: cgroup-2.6.25-rc2-mm1/include/linux/cgroup.h
===
--- cgroup-2.6.25-rc2-mm1.orig/include/linux/cgroup.h
+++ cgroup-2.6.25-rc2-mm1/include/linux/cgroup.h
@@ -88,6 +88,17 @@ static inline void css_put(struct cgroup
   __css_put(css);
 }

+/* bits in struct cgroup flags field */
+enum {
+ /* Control Group is dead */
+ CGRP_REMOVED,
+ /* Control Group has previously had a child cgroup or a task,
+ * but no longer (only if CGRP_NOTIFY_ON_RELEASE is set) */
+ CGRP_RELEASABLE,
+ /* Control Group requires release notifications to userspace */
+ CGRP_NOTIFY_ON_RELEASE,
+};
+
 struct cgroup {
  unsigned long flags; /* "unsigned long" so bitops work */

Index: cgroup-2.6.25-rc2-mm1/kernel/cgroup.c
===
--- cgroup-2.6.25-rc2-mm1.orig/kernel/cgroup.c
+++ cgroup-2.6.25-rc2-mm1/kernel/cgroup.c
@@ -119,17 +119,6 @@ static int root_count;
  */

```
 static int need_forkexit_callback;

-/* bits in struct cgroup flags field */
-enum {
- /* Control Group is dead */
- CGRP_REMOVED,
- /* Control Group has previously had a child cgroup or a task,
-  * but no longer (only if CGRP_NOTIFY_ON_RELEASE is set) */
- CGRP_RELEASABLE,
- /* Control Group requires release notifications to userspace */
- CGRP_NOTIFY_ON_RELEASE,
-};
-
 /* convenient tests for these bits */
 inline int cgroup_is_removed(const struct cgroup *cgrp)
 {
@@ -1299,7 +1288,6 @@ enum cgroup_filetype {
  FILE_DIR,
  FILE_TASKLIST,
  FILE_NOTIFY_ON_RELEASE,
- FILE_RELEASABLE,
  FILE_RELEASE_AGENT,
 };

@@ -2169,11 +2157,6 @@ static u64 cgroup_read_notify_on_release
  return notify_on_release(cgrp);
 }

-static u64 cgroup_read_releasable(struct cgroup *cgrp, struct cftype *cft)
-{
- return test_bit(CGRP_RELEASABLE, &cgrp->flags);
-}
-
 /*
  * for the common functions, 'private' gives the type of file
  */
@@ -2193,12 +2176,6 @@ static struct cftype files[] = {
   .write = cgroup_common_file_write,
   .private = FILE_NOTIFY_ON_RELEASE,
  },
-
- {
-  .name = "releasable",
-  .read_u64 = cgroup_read_releasable,
-  .private = FILE_RELEASABLE,
- }
 };
```

```
 static struct cftype cft_release_agent = {
Index: cgroup-2.6.25-rc2-mm1/kernel/cgroup_debug.c
===================================================================
--- cgroup-2.6.25-rc2-mm1.orig/kernel/cgroup_debug.c
+++ cgroup-2.6.25-rc2-mm1/kernel/cgroup_debug.c
@@ -1,5 +1,5 @@
 /*
- * kernel/ccontainer_debug.c - Example cgroup subsystem that
+ * kernel/cgroup_debug.c - Example cgroup subsystem that
  * exposes debug info
  *
  * Copyright (C) Google Inc, 2007
@@ -62,6 +62,11 @@ static u64 current_css_set_refcount_read
  return count;
 }

+static u64 releasable_read(struct cgroup *cgrp, struct cftype *cft)
+{
+ return test_bit(CGRP_RELEASABLE, &cgrp->flags);
+}
+
 static struct cftype files[] =  {
  {
   .name = "cgroup_refcount",
@@ -81,6 +86,11 @@ static struct cftype files[] =  {
   .name = "current_css_set_refcount",
   .read_u64 = current_css_set_refcount_read,
  },
+
+ {
+  .name = "releasable",
+  .read_u64 = releasable_read,
+ }
 };

 static int debug_populate(struct cgroup_subsys *ss, struct cgroup *cont)

--
```