
Subject: [PATCH 9/9] Containers (V9): Simple debug info subsystem

Posted by [Paul Menage](#) on Fri, 27 Apr 2007 10:46:16 GMT

[View Forum Message](#) <> [Reply to Message](#)

This example subsystem exports debugging information as an aid to diagnosing refcount leaks, etc, in the container framework.

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

```
include/linux/container_subsys.h | 4 +
init/Kconfig                     | 10 ++++
kernel/Makefile                  | 1
kernel/container_debug.c         | 89 +++++
4 files changed, 104 insertions(+)
```

Index: container-2.6.21-rc7-mm1/include/linux/container_subsys.h

=====

--- container-2.6.21-rc7-mm1.orig/include/linux/container_subsys.h

+++ container-2.6.21-rc7-mm1/include/linux/container_subsys.h

@@ -19,4 +19,8 @@ SUBSYS(cpuset)

/* */

+#ifdef CONFIG_CONTAINER_DEBUG

+SUBSYS(debug)

+#endif

+

/* */

Index: container-2.6.21-rc7-mm1/init/Kconfig

=====

--- container-2.6.21-rc7-mm1.orig/init/Kconfig

+++ container-2.6.21-rc7-mm1/init/Kconfig

@@ -291,6 +291,16 @@ config IKCONFIG_PROC

config CONTAINERS

bool

+config CONTAINER_DEBUG

+ bool "Example debug container subsystem"

+ select CONTAINERS

+ help

+ This option enables a simple container subsystem that

+ exports useful debugging information about the containers

+ framework

+

+ Say N if unsure

+

config CPUSETS

bool "Cpuset support"

depends on SMP

Index: container-2.6.21-rc7-mm1/kernel/container_debug.c

```
=====
--- /dev/null
+++ container-2.6.21-rc7-mm1/kernel/container_debug.c
@@ -0,0 +1,89 @@
+/*
+ * kernel/ccontainer_debug.c - Example container subsystem that
+ * exposes debug info
+ *
+ * Copyright (C) Google Inc, 2007
+ *
+ * Developed by Paul Menage (menage@google.com)
+ */
+
+#include <linux/container.h>
+#include <linux/fs.h>
+
+static int debug_create(struct container_subsys *ss, struct container *cont)
+{
+ struct container_subsys_state *css = kzalloc(sizeof(*css), GFP_KERNEL);
+ if (!css)
+ return -ENOMEM;
+ cont->subsys[debug_subsys_id] = css;
+ return 0;
+}
+
+static void debug_destroy(struct container_subsys *ss, struct container *cont)
+{
+ kfree(cont->subsys[debug_subsys_id]);
+}
+
+static u64 container_refcount_read(struct container *cont, struct cftype *cft)
+{
+ return atomic_read(&cont->count);
+}
+
+static u64 taskcount_read(struct container *cont, struct cftype *cft)
+{
+ u64 count;
+ container_lock();
+ count = container_task_count(cont);
+ container_unlock();
+ return count;
+}
+
```

```

+static u64 current_css_group_read(struct container *cont, struct cftype *cft)
+{
+ return (u64) current->containers;
+}
+
+static u64 current_css_group_refcount_read(struct container *cont,
+      struct cftype *cft)
+{
+ u64 count;
+ rcu_read_lock();
+ count = atomic_read(&current->containers->ref.refcount);
+ rcu_read_unlock();
+ return count;
+}
+
+static struct cftype files[] = {
+ {
+ .name = "debug.container_refcount",
+ .read_uint = container_refcount_read,
+ },
+ {
+ .name = "debug.taskcount",
+ .read_uint = taskcount_read,
+ },
+ {
+ .name = "debug.current_css_group",
+ .read_uint = current_css_group_read,
+ },
+ {
+ .name = "debug.current_css_group_refcount",
+ .read_uint = current_css_group_refcount_read,
+ },
+};
+
+static int debug_populate(struct container_subsys *ss, struct container *cont)
+{
+ return container_add_files(cont, files, ARRAY_SIZE(files));
+}
+
+struct container_subsys debug_subsys = {
+ .name = "debug",
+ .create = debug_create,
+ .destroy = debug_destroy,
+ .populate = debug_populate,
+ .subsys_id = debug_subsys_id,
+};

```

Index: container-2.6.21-rc7-mm1/kernel/Makefile

=====

--- container-2.6.21-rc7-mm1.orig/kernel/Makefile

+++ container-2.6.21-rc7-mm1/kernel/Makefile

@@ -37,6 +37,7 @@ obj-\$(CONFIG_BSD_PROCESS_ACCT) += acct.o

obj-\$(CONFIG_KEXEC) += kexec.o

obj-\$(CONFIG_COMPAT) += compat.o

obj-\$(CONFIG_CONTAINERS) += container.o

+obj-\$(CONFIG_CONTAINER_DEBUG) += container_debug.o

obj-\$(CONFIG_CPUSETS) += cpuset.o

obj-\$(CONFIG_CONTAINER_CPUACCT) += cpu_acct.o

obj-\$(CONFIG_IKCONFIG) += configs.o

--