Subject: [RFC][PATCH 2/7] CGroup API: Add cgroup map data type
Posted by Paul Menage on Fri, 15 Feb 2008 20:44:20 GMT
View Forum Message <> Reply to Message

Adds a new type of supported control file representation, a map from
strings to u64 values.

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

---
 include/linux/cgroup.h | 19 +++++++++++++++++
 kernel/cgroup.c | 61 ++-
 2 files changed, 79 insertions(+), 1 deletion(-)

Index: cgroupmap-2.6.24-mm1/include/linux/cgroup.h
===
--- cgroupmap-2.6.24-mm1.orig/include/linux/cgroup.h
+++ cgroupmap-2.6.24-mm1/include/linux/cgroup.h
@@ -191,6 +191,17 @@ enum cgroup_file_type {
  CGROUP_FILE_VOID,
  CGROUP_FILE_U64,
  CGROUP_FILE_STRING,
+ CGROUP_FILE_MAP,
+};
+
+/*
+ * cgroup_map_cb is an abstract callback API for reporting map-valued
+ * control files
+ */
+
+struct cgroup_map_cb {
+ int (*fill)(struct cgroup_map_cb *cb, const char *key, u64 value);
+ void *state;
 };

 #define MAX_CFTYPE_NAME 64
@@ -215,6 +226,14 @@ struct cftype {
   * single integer. Use it in place of read()
   */
  u64 (*read_uint) (struct cgroup *cont, struct cftype *cft);
+ /*
+ * read_map() is used for defining a map of key/value
+ * pairs. It should call cb->fill(cb, key, value) for each
+ * entry.
+ */
+ int (*read_map) (struct cgroup *cont, struct cftype *cft,
+   struct cgroup_map_cb *cb);
+

```
  ssize_t (*write) (struct cgroup *cont, struct cftype *cft,
      struct file *file,
      const char __user *buf, size_t nbytes, loff_t *ppos);
Index: cgroupmap-2.6.24-mm1/kernel/cgroup.c
===================================================================
--- cgroupmap-2.6.24-mm1.orig/kernel/cgroup.c
+++ cgroupmap-2.6.24-mm1/kernel/cgroup.c
@@ -1488,6 +1488,46 @@ static ssize_t cgroup_file_read(struct f
  return -EINVAL;
 }

+/*
+ * seqfile ops/methods for returning structured data. Currently just
+ * supports string->u64 maps, but can be extended in future.
+ */
+
+struct cgroup_seqfile_state {
+ struct cftype *cft;
+ struct cgroup *cgroup;
+};
+
+static int cgroup_map_add(struct cgroup_map_cb *cb, const char *key, u64 value)
+{
+ struct seq_file *sf = cb->state;
+ return seq_printf(sf, "%s: %llu\n", key, value);
+}
+
+static int cgroup_seqfile_show(struct seq_file *m, void *arg)
+{
+ struct cgroup_seqfile_state *state = m->private;
+ struct cftype *cft = state->cft;
+ struct cgroup_map_cb cb = {
+  .fill = cgroup_map_add,
+  .state = m,
+ };
+ if (cft->read_map) {
+  return cft->read_map(state->cgroup, cft, &cb);
+ } else {
+  BUG();
+ }
+}
+
+int cgroup_seqfile_release(struct inode *inode, struct file *file)
+{
+ struct seq_file *seq = file->private_data;
+ kfree(seq->private);
+ return single_release(inode, file);
+}
```

```
+
+static struct file_operations cgroup_seqfile_operations;
+
 static int cgroup_file_open(struct inode *inode, struct file *file)
 {
  int err;
@@ -1500,7 +1540,18 @@ static int cgroup_file_open(struct inode
  cft = __d_cft(file->f_dentry);
  if (!cft)
   return -ENODEV;
- if (cft->open)
+ if (cft->read_map) {
+  struct cgroup_seqfile_state *state =
+   kzalloc(sizeof(*state), GFP_USER);
+  if (!state)
+   return -ENOMEM;
+  state->cft = cft;
+  state->cgroup = __d_cgrp(file->f_dentry->d_parent);
+  file->f_op = &cgroup_seqfile_operations;
+  err = single_open(file, cgroup_seqfile_show, state);
+  if (err < 0)
+   kfree(state);
+ } else if (cft->open)
   err = cft->open(inode, file);
  else
   err = 0;
@@ -1539,6 +1590,12 @@ static struct file_operations cgroup_fil
  .release = cgroup_file_release,
 };

+static struct file_operations cgroup_seqfile_operations = {
+ .read = seq_read,
+ .llseek = seq_lseek,
+ .release = cgroup_seqfile_release,
+};
+
 static struct inode_operations cgroup_dir_inode_operations = {
  .lookup = simple_lookup,
  .mkdir = cgroup_mkdir,
@@ -2206,6 +2263,8 @@ static int cgroup_api_show(struct seq_fi
  if (type == CGROUP_FILE_UNKNOWN) {
   if (cft->read_uint)
    type = CGROUP_FILE_U64;
+  else if (cft->read_map)
+   type = CGROUP_FILE_MAP;
   else if (cft->read)
    type = CGROUP_FILE_STRING;
   else if (!cft->open)
```

--

_____

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