Subject: [PATCH 3/3] make 'none' field a flag
Posted by Glauber Costa on Sun, 11 Dec 2011 14:45:38 GMT
View Forum Message <> Reply to Message

There is no reason to have a flags field, and then a separate
bool field just to indicate if 'none' subsystem were explicitly
requested.

Make it a flag

Signed-off-by: Glauber Costa <glommer@parallels.com>
---
 kernel/cgroup.c | 20 ++++++++++++--------
 1 files changed, 12 insertions(+), 8 deletions(-)

```
diff --git a/kernel/cgroup.c b/kernel/cgroup.c
index fa405ee..e700abe 100644
--- a/kernel/cgroup.c
+++ b/kernel/cgroup.c
@@ -232,6 +232,7 @@ inline int cgroup_is_removed(const struct cgroup *cgrp)
 enum {
  ROOT_NOPREFIX, /* mounted subsystems have no named prefix */
  ROOT_CLONE_CHILDREN, /* mounted subsystems starts with clone_children */
+ ROOT_NOSUBSYS, /* explicitly asked for 'none' subsystems */
 };

 static int cgroup_is_releasable(const struct cgroup *cgrp)
@@ -1064,13 +1065,16 @@ struct cgroup_sb_opts {
  unsigned long flags;
  char *release_agent;
  char *name;
- /* User explicitly requested empty subsystem */
- bool none;

  struct cgroupfs_root *new_root;

 };

+static inline int opts_no_subsys(const struct cgroup_sb_opts *opts)
+{
+ return test_bit(ROOT_NOSUBSYS, &opts->flags);
+}
+
 /*
  * Convert a hierarchy specifier into a bitmask of subsystems and flags. Call
  * with cgroup_mutex held to protect the subsys[] array. This function takes
@@ -1098,7 +1102,7 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts
*opts)
```

```
    return -EINVAL;
   if (!strcmp(token, "none")) {
    /* Explicitly have no subsystems */
-   opts->none = true;
+   set_bit(ROOT_NOSUBSYS, &opts->flags);
    continue;
   }
   if (!strcmp(token, "all")) {
@@ -1178,7 +1182,7 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts
*opts)
   * otherwise 'all, 'none' and a subsystem name options were not
   * specified, let's default to 'all'
   */
- if (all_ss || (!all_ss && !one_ss && !opts->none)) {
+ if (all_ss || (!all_ss && !one_ss && !opts_no_subsys(opts))) {
   for (i = 0; i < CGROUP_SUBSYS_COUNT; i++) {
    struct cgroup_subsys *ss = subsys[i];
    if (ss == NULL)
@@ -1202,7 +1206,7 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts
*opts)


  /* Can't specify "none" and some subsystems */
- if (opts->subsys_bits && opts->none)
+ if (opts->subsys_bits && opts_no_subsys(opts))
   return -EINVAL;

  /*
@@ -1370,7 +1374,7 @@ static int cgroup_test_super(struct super_block *sb, void *data)
   * If we asked for subsystems (or explicitly for no
   * subsystems) then they must match
   */
- if ((opts->subsys_bits || opts->none)
+ if ((opts->subsys_bits || opts_no_subsys(opts))
      && (opts->subsys_bits != root->subsys_bits))
   return 0;

@@ -1381,7 +1385,7 @@ static struct cgroupfs_root *cgroup_root_from_opts(struct
cgroup_sb_opts *opts)
 {
  struct cgroupfs_root *root;

- if (!opts->subsys_bits && !opts->none)
+ if (!opts->subsys_bits && !opts_no_subsys(opts))
   return NULL;

  root = kzalloc(sizeof(*root), GFP_KERNEL);
@@ -1426,7 +1430,7 @@ static int cgroup_set_super(struct super_block *sb, void *data)
```

```
  if (!opts->new_root)
   return -EINVAL;

- BUG_ON(!opts->subsys_bits && !opts->none);
+ BUG_ON(!opts->subsys_bits && !opts_no_subsys(opts));

  ret = set_anon_super(sb, NULL);
  if (ret)
--
1.7.6.4
```