Subject: Re: Roadmap for features planed for containers where and Some future features ideas.
Posted by Peter Dolding on Wed, 23 Jul 2008 00:56:46 GMT
View Forum Message <> Reply to Message

From: Kirill A. Shutemov <kirill@shutemov.name>

Changelog:
v4:
 - hierarchy support
 - drop dummy_timer_slack_check()
 - workaround lockdep false (?) positive
 - allow 0 as timer slack value
v3:
 - rework interface
 - s/EXPORT_SYMBOL/EXPORT_SYMBOL_GPL/
v2:
 - fixed with CONFIG_CGROUP_TIMER_SLACK=y
v1:
 - initial revision

Kirill A. Shutemov (2):
  cgroups: export cgroup_iter_{start,next,end}
  cgroups: introduce timer slack subsystem

 include/linux/cgroup_subsys.h | 6 +
 include/linux/init_task.h | 4 +-
 init/Kconfig | 10 ++
 kernel/Makefile | 1 +
 kernel/cgroup.c | 3 +
 kernel/cgroup_timer_slack.c | 262 ++
 kernel/sys.c | 19 ++-
 7 files changed, 298 insertions(+), 7 deletions(-)
 create mode 100644 kernel/cgroup_timer_slack.c

--
1.7.3.5

_____
Containers mailing list
Containers@lists.linux-foundation.org
 https://lists.linux-foundation.org/mailman/listinfo/containe rsFrom: Kirill A. Shutemov <kirill@shutemov.name>

Signed-off-by: Kirill A. Shutemov <kirill@shutemov.name>
---
 kernel/cgroup.c | 3 +++
 1 files changed, 3 insertions(+), 0 deletions(-)

```
diff --git a/kernel/cgroup.c b/kernel/cgroup.c
index b24d702..8234daa 100644
--- a/kernel/cgroup.c
+++ b/kernel/cgroup.c
@@ -2443,6 +2443,7 @@ void cgroup_iter_start(struct cgroup *cgrp, struct cgroup_iter *it)
  it->cg_link = &cgrp->css_sets;
  cgroup_advance_iter(cgrp, it);
 }
+EXPORT_SYMBOL_GPL(cgroup_iter_start);

 struct task_struct *cgroup_iter_next(struct cgroup *cgrp,
      struct cgroup_iter *it)
@@ -2467,11 +2468,13 @@ struct task_struct *cgroup_iter_next(struct cgroup *cgrp,
  }
  return res;
 }
+EXPORT_SYMBOL_GPL(cgroup_iter_next);

 void cgroup_iter_end(struct cgroup *cgrp, struct cgroup_iter *it)
 {
  read_unlock(&css_set_lock);
 }
+EXPORT_SYMBOL_GPL(cgroup_iter_end);

 static inline int started_after_time(struct task_struct *t1,
          struct timespec *time,
--
1.7.3.5
```

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rsFrom: Kirill A. Shutemov <kirill@shutemov.name>

Provides a way of tasks grouping by timer slack value. Introduces per
cgroup max and min timer slack value. When a task attaches to a cgroup,
its timer slack value adjusts (if needed) to fit min-max range.

It also provides a way to set timer slack value for all tasks in the
cgroup at once.

This functionality is useful in mobile devices where certain background
apps are attached to a cgroup and minimum wakeups are desired.

Signed-off-by: Kirill A. Shutemov <kirill@shutemov.name>
Idea-by: Jacob Pan <jacob.jun.pan@linux.intel.com>

```
Signed-off-by: Kirill A. Shutemov <kirill@shutemov.name>
---
 include/linux/cgroup_subsys.h |    6 +
 include/linux/init_task.h     |    4 +-
 init/Kconfig                  |   10 ++
 kernel/Makefile               |    1 +
 kernel/cgroup_timer_slack.c   |  262 ++++++++++++++++++++++++++++++++++++++++++
 kernel/sys.c                  |   19 ++-
 6 files changed, 295 insertions(+), 7 deletions(-)
 create mode 100644 kernel/cgroup_timer_slack.c

diff --git a/include/linux/cgroup_subsys.h b/include/linux/cgroup_subsys.h
index ccefff0..e399228 100644
--- a/include/linux/cgroup_subsys.h
+++ b/include/linux/cgroup_subsys.h
@@ -66,3 +66,9 @@ SUBSYS(blkio)
 #endif

 /* */
+
+#ifdef CONFIG_CGROUP_TIMER_SLACK
+SUBSYS(timer_slack)
+#endif
+
+/* */
diff --git a/include/linux/init_task.h b/include/linux/init_task.h
index caa151f..48eca8f 100644
--- a/include/linux/init_task.h
+++ b/include/linux/init_task.h
@@ -124,6 +124,8 @@ extern struct cred init_cred;
 # define INIT_PERF_EVENTS(tsk)
 #endif

+#define TIMER_SLACK_NS_DEFAULT 50000
+
 /*
  *  INIT_TASK is used to set up the first task table, touch at
  * your own risk!. Base=0, limit=0x1fffff (=2MB)
@@ -177,7 +179,7 @@ extern struct cred init_cred;
  .cpu_timers = INIT_CPU_TIMERS(tsk.cpu_timers),  \
  .fs_excl = ATOMIC_INIT(0),    \
  .pi_lock = __RAW_SPIN_LOCK_UNLOCKED(tsk.pi_lock), \
- .timer_slack_ns = 50000, /* 50 usec default slack */  \
+ .timer_slack_ns = TIMER_SLACK_NS_DEFAULT,   \
  .pids = {       \
   [PIDTYPE_PID]  = INIT_PID_LINK(PIDTYPE_PID),  \
   [PIDTYPE_PGID] = INIT_PID_LINK(PIDTYPE_PGID),  \
diff --git a/init/Kconfig b/init/Kconfig
```

```
index be788c0..6cf465f 100644
--- a/init/Kconfig
+++ b/init/Kconfig
@@ -596,6 +596,16 @@ config CGROUP_FREEZER
    Provides a way to freeze and unfreeze all tasks in a
    cgroup.

+config CGROUP_TIMER_SLACK
+ tristate "Timer slack cgroup subsystem"
+ help
+   Provides a way of tasks grouping by timer slack value.
+   Introduces per cgroup timer slack value which will override
+   the default timer slack value once a task is attached to a
+   cgroup.
+   It's useful in mobile devices where certain background apps
+   are attached to a cgroup and combined wakeups are desired.
+
 config CGROUP_DEVICE
  bool "Device controller for cgroups"
  help
diff --git a/kernel/Makefile b/kernel/Makefile
index 353d3fe..0b60239 100644
--- a/kernel/Makefile
+++ b/kernel/Makefile
@@ -61,6 +61,7 @@ obj-$(CONFIG_BACKTRACE_SELF_TEST) += backtracetest.o
 obj-$(CONFIG_COMPAT) += compat.o
 obj-$(CONFIG_CGROUPS) += cgroup.o
 obj-$(CONFIG_CGROUP_FREEZER) += cgroup_freezer.o
+obj-$(CONFIG_CGROUP_TIMER_SLACK) += cgroup_timer_slack.o
 obj-$(CONFIG_CPUSETS) += cpuset.o
 obj-$(CONFIG_CGROUP_NS) += ns_cgroup.o
 obj-$(CONFIG_UTS_NS) += utsname.o
diff --git a/kernel/cgroup_timer_slack.c b/kernel/cgroup_timer_slack.c
new file mode 100644
index 0000000..affd33a
--- /dev/null
+++ b/kernel/cgroup_timer_slack.c
@@ -0,0 +1,262 @@
+/*
+ * cgroup_timer_slack.c - control group timer slack subsystem
+ *
+ * Copyright Nokia Corparation, 2011
+ * Author: Kirill A. Shutemov
+ *
+ * This program is free software; you can redistribute it and/or modify
+ * it under the terms of the GNU General Public License as published by
+ * the Free Software Foundation; either version 2 of the License, or
+ * (at your option) any later version.
```

```
+ *
+ * This program is distributed in the hope that it will be useful,
+ * but WITHOUT ANY WARRANTY; without even the implied warranty of
+ * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
+ * GNU General Public License for more details.
+ */
+#include <linux/cgroup.h>
+#include <linux/init_task.h>
+#include <linux/module.h>
+#include <linux/slab.h>
+#include <linux/rcupdate.h>
+
+struct cgroup_subsys timer_slack_subsys;
+struct timer_slack_cgroup {
+ struct cgroup_subsys_state css;
+ unsigned long min_slack_ns;
+ unsigned long max_slack_ns;
+};
+
+enum {
+ TIMER_SLACK_MIN,
+ TIMER_SLACK_MAX,
+};
+
+extern int (*timer_slack_check)(struct task_struct *task,
+  unsigned long slack_ns);
+
+static struct timer_slack_cgroup *cgroup_to_tslack_cgroup(struct cgroup *cgroup)
+{
+ struct cgroup_subsys_state *css;
+
+ css = cgroup_subsys_state(cgroup, timer_slack_subsys.subsys_id);
+ return container_of(css, struct timer_slack_cgroup, css);
+}
+
+static int is_timer_slack_allowed(struct timer_slack_cgroup *t
```