Subject: [PATCH 1/6] IPC namespace core
Posted by Kirill Korotaev on Fri, 09 Jun 2006 15:01:38 GMT
View Forum Message <> Reply to Message

This patch implements core IPC namespace changes:
- ipc_namespace structure
- new config option CONFIG_IPC_NS
- adds CLONE_NEWIPC flag
- unshare support

Signed-Off-By: Pavel Emelianov <xemul@openvz.org>
Signed-Off-By: Kirill Korotaev <dev@openvz.org>

```
--- ./include/linux/init_task.h.ipcns 2006-06-06 14:47:58.000000000 +0400
+++ ./include/linux/init_task.h 2006-06-08 14:28:23.000000000 +0400
@@ -73,6 +73,7 @@ extern struct nsproxy init_nsproxy;
  .count  = ATOMIC_INIT(1),    \
  .nslock  = SPIN_LOCK_UNLOCKED,    \
  .uts_ns  = &init_uts_ns,     \
+ .ipc_ns  = &init_ipc_ns,     \
  .namespace = NULL,      \
 }

--- ./include/linux/ipc.h.ipcns 2006-04-21 11:59:36.000000000 +0400
+++ ./include/linux/ipc.h 2006-06-08 15:43:43.000000000 +0400
@@ -2,6 +2,7 @@
 #define _LINUX_IPC_H

 #include <linux/types.h>
+#include <linux/kref.h>

 #define IPC_PRIVATE ((__kernel_key_t) 0)

@@ -68,6 +69,41 @@ struct kern_ipc_perm
  void  *security;
 };

+struct ipc_ids;
+struct ipc_namespace {
+ struct kref kref;
+ struct ipc_ids *ids[3];
+
+ int  sem_ctls[4];
+ int  used_sems;
+
+ int  msg_ctlmax;
+ int  msg_ctlmnb;
```

```
+ int  msg_ctlmni;
+
+ size_t  shm_ctlmax;
+ size_t  shm_ctlall;
+ int  shm_ctlmni;
+ int  shm_tot;
+};
+
+extern struct ipc_namespace init_ipc_ns;
+extern void free_ipc_ns(struct kref *kref);
+extern int copy_ipcs(unsigned long flags, struct task_struct *tsk);
+extern int unshare_ipcs(unsigned long flags, struct ipc_namespace **ns);
+
+static inline struct ipc_namespace *get_ipc_ns(struct ipc_namespace *ns)
+{
+ if (ns)
+  kref_get(&ns->kref);
+ return ns;
+}
+
+static inline void put_ipc_ns(struct ipc_namespace *ns)
+{
+ kref_put(&ns->kref, free_ipc_ns);
+}
+
 #endif /* __KERNEL__ */

 #endif /* _LINUX_IPC_H */
--- ./include/linux/nsproxy.h.ipcns 2006-06-06 14:47:58.000000000 +0400
+++ ./include/linux/nsproxy.h 2006-06-08 15:28:02.000000000 +0400
@@ -6,6 +6,7 @@

 struct namespace;
 struct uts_namespace;
+struct ipc_namespace;

 /*
  * A structure to contain pointers to all per-process
@@ -23,6 +24,7 @@ struct nsproxy {
  atomic_t count;
  spinlock_t nslock;
  struct uts_namespace *uts_ns;
+ struct ipc_namespace *ipc_ns;
  struct namespace *namespace;
 };
 extern struct nsproxy init_nsproxy;
--- ./include/linux/sched.h.ipcns 2006-06-06 14:47:58.000000000 +0400
+++ ./include/linux/sched.h 2006-06-08 14:28:23.000000000 +0400
```

```
@@ -25,6 +25,7 @@
 #define CLONE_CHILD_SETTID 0x01000000 /* set the TID in the child */
 #define CLONE_STOPPED  0x02000000 /* Start in stopped state */
 #define CLONE_NEWUTS  0x04000000 /* New utsname group? */
+#define CLONE_NEWIPC  0x08000000 /* New ipcs */

 /*
  * Scheduling policies
--- ./init/Kconfig.ipcns 2006-06-06 14:47:58.000000000 +0400
+++ ./init/Kconfig 2006-06-09 14:18:09.000000000 +0400
@@ -137,6 +137,15 @@ config SYSVIPC
    section 6.4 of the Linux Programmer's Guide, available from
    <http://www.tldp.org/guides.html>.

+config IPC_NS
+ bool "IPC Namespaces"
+ depends on SYSVIPC
+ default n
+ help
+   Support ipc namespaces.  This allows containers, i.e. virtual
+   environments, to use ipc namespaces to provide different ipc
+   objects for different servers.  If unsure, say N.
+
 config POSIX_MQUEUE
  bool "POSIX Message Queues"
  depends on NET && EXPERIMENTAL
--- ./kernel/fork.c.ipcns 2006-06-06 14:47:58.000000000 +0400
+++ ./kernel/fork.c 2006-06-08 15:31:03.000000000 +0400
@@ -1592,6 +1592,7 @@ asmlinkage long sys_unshare(unsigned lon
  struct sem_undo_list *new_ulist = NULL;
  struct nsproxy *new_nsproxy = NULL, *old_nsproxy = NULL;
  struct uts_namespace *uts, *new_uts = NULL;
+ struct ipc_namespace *ipc, *new_ipc = NULL;

  check_unshare_flags(&unshare_flags);

@@ -1617,18 +1618,20 @@ asmlinkage long sys_unshare(unsigned lon
    goto bad_unshare_cleanup_fd;
  if ((err = unshare_utsname(unshare_flags, &new_uts)))
    goto bad_unshare_cleanup_semundo;
+ if ((err = unshare_ipcs(unshare_flags, &new_ipc)))
+  goto bad_unshare_cleanup_uts;

  if (new_ns || new_uts) {
    old_nsproxy = current->nsproxy;
    new_nsproxy = dup_namespaces(old_nsproxy);
    if (!new_nsproxy) {
     err = -ENOMEM;
```

```
-   goto bad_unshare_cleanup_uts;
+   goto bad_unshare_cleanup_ipc;
    }
   }

   if (new_fs || new_ns || new_sigh || new_mm || new_fd || new_ulist ||
-     new_uts) {
+     new_uts || new_ipc) {

    task_lock(current);

@@ -1676,12 +1679,22 @@ asmlinkage long sys_unshare(unsigned lon
     new_uts = uts;
    }

+   if (new_ipc) {
+    ipc = current->nsproxy->ipc_ns;
+    current->nsproxy->ipc_ns = new_ipc;
+    new_ipc = ipc;
+   }
+
    task_unlock(current);
   }

   if (new_nsproxy)
    put_nsproxy(new_nsproxy);

+bad_unshare_cleanup_ipc:
+ if (new_ipc)
+   put_ipc_ns(new_ipc);
+
  bad_unshare_cleanup_uts:
   if (new_uts)
    put_uts_ns(new_uts);
--- ./kernel/nsproxy.c.ipcns 2006-06-06 14:47:59.000000000 +0400
+++ ./kernel/nsproxy.c 2006-06-09 14:22:31.000000000 +0400
@@ -7,6 +7,10 @@
   *  modify it under the terms of the GNU General Public License as
   *  published by the Free Software Foundation, version 2 of the
   *  License.
+ *
+ *  Jun 2006 - namespaces support
+ *             OpenVZ, SWsoft Inc.
+ *             Pavel Emelianov <xemul@openvz.org>
   */

  #include <linux/module.h>
@@ -59,6 +63,8 @@ struct nsproxy *dup_namespaces(struct ns
```

```
    get_namespace(ns->namespace);
   if (ns->uts_ns)
    get_uts_ns(ns->uts_ns);
+  if (ns->ipc_ns)
+   get_ipc_ns(ns->ipc_ns);
  }

  return ns;
@@ -79,7 +85,7 @@ int copy_namespaces(int flags, struct ta

  get_nsproxy(old_ns);

- if (!(flags & (CLONE_NEWNS | CLONE_NEWUTS)))
+ if (!(flags & (CLONE_NEWNS | CLONE_NEWUTS | CLONE_NEWIPC)))
   return 0;

  new_ns = clone_namespaces(old_ns);
@@ -91,24 +97,31 @@ int copy_namespaces(int flags, struct ta
  tsk->nsproxy = new_ns;

  err = copy_namespace(flags, tsk);
- if (err) {
-  tsk->nsproxy = old_ns;
-  put_nsproxy(new_ns);
-  goto out;
- }
+ if (err)
+  goto out_ns;

  err = copy_utsname(flags, tsk);
- if (err) {
-  if (new_ns->namespace)
-   put_namespace(new_ns->namespace);
-  tsk->nsproxy = old_ns;
-  put_nsproxy(new_ns);
-  goto out;
- }
+ if (err)
+  goto out_uts;
+
+ err = copy_ipcs(flags, tsk);
+ if (err)
+  goto out_ipc;

 out:
  put_nsproxy(old_ns);
  return err;
+
```

```
+out_ipc:
+ if (new_ns->uts_ns)
+  put_uts_ns(new_ns->uts_ns);
+out_uts:
+ if (new_ns->namespace)
+  put_namespace(new_ns->namespace);
+out_ns:
+ tsk->nsproxy = old_ns;
+ put_nsproxy(new_ns);
+ goto out;
 }

 void free_nsproxy(struct nsproxy *ns)
@@ -117,5 +130,7 @@ void free_nsproxy(struct nsproxy *ns)
    put_namespace(ns->namespace);
   if (ns->uts_ns)
    put_uts_ns(ns->uts_ns);
+  if (ns->ipc_ns)
+   put_ipc_ns(ns->ipc_ns);
   kfree(ns);
 }
```