Subject: Re: [RFC][PATCH 1/2] Virtualization of UTS
Posted by ebiederm on Fri, 24 Mar 2006 19:09:01 GMT
View Forum Message <> Reply to Message

Kirill Korotaev <dev@sw.ru> writes:

> This patch introduces utsname namespace in system, which allows to have
> different utsnames on the host.
> Introduces config option CONFIG_UTS_NS and uts_namespace structure for this.

Ok. It looks like we need to resolve the sysctl issues before we merge
either patch, into the stable kernel.

We also need to discuss the system call interface, as without one
the functionality is unusable :)

>
http://git.openvz.org/?p=linux-2.6-openvz-ms;a=commitdiff;h=216bb5e42c7eef7f1ed361244a60b1496e8bdf63
>
> Signed-Off-By: Pavel Emelianov <xemul@openvz.org>
> Signed-Off-By: Kirill Korotaev <dev@openvz.org>
>
> Kirill
> --- a/include/linux/init_task.h
> +++ b/include/linux/init_task.h
> @@ -3,6 +3,7 @@
>
> #include <linux/file.h>
> #include <linux/rcupdate.h>
> +#include <linux/utsname.h>
>
> #define INIT_FDTABLE \
> { \
> @@ -72,6 +73,12 @@
>
> extern struct group_info init_groups;
>
> +#ifdef CONFIG_UTS_NS
> +#define INIT_UTS_NS .uts_ns = &init_uts_ns,
> +#else
> +#define INIT_UTS_NS
> +#endif
> +
> /*
> * INIT_TASK is used to set up the first task table, touch at
> * your own risk!. Base=0, limit=0x1fffff (=2MB)
> @@ -121,6 +128,7 @@ extern struct group_info init_groups;

```
>   .journal_info = NULL,      \
>   .cpu_timers = INIT_CPU_TIMERS(tsk.cpu_timers),  \
>   .fs_excl = ATOMIC_INIT(0),    \
> + INIT_UTS_NS       \
>  }
>
>
> --- a/include/linux/sched.h
> +++ b/include/linux/sched.h
> @@ -688,6 +688,7 @@ static inline void prefetch_stack(struct
>
>  struct audit_context;  /* See audit.c */
>  struct mempolicy;
> +struct uts_namespace;
>
>  struct task_struct {
>   volatile long state; /* -1 unrunnable, 0 runnable, >0 stopped */
> @@ -802,6 +803,9 @@ struct task_struct {
>   struct files_struct *files;
>  /* namespace */
>   struct namespace *namespace;
> +#ifdef CONFIG_UTS_NS
> + struct uts_namespace *uts_ns;
> +#endif
>  /* signal handlers */
>   struct signal_struct *signal;
>   struct sighand_struct *sighand;
> --- a/include/linux/utsname.h
> +++ b/include/linux/utsname.h
> @@ -30,7 +30,36 @@ struct new_utsname {
>   char domainname[65];
>  };
>
> +#ifdef CONFIG_UTS_NS
> +#include <asm/atomic.h>
> +
> +struct uts_namespace {
> + atomic_t cnt;
> + struct new_utsname name;
> +};
> +
> +extern struct uts_namespace *create_uts_ns(void);
> +extern struct uts_namespace *clone_uts_ns(void);
> +extern void free_uts_ns(struct uts_namespace *ns);
> +
> +static inline void get_uts_ns(struct uts_namespace *ns)
> +{
> + atomic_inc(&ns->cnt);
```

```
> +}
> +
> +static inline void put_uts_ns(struct uts_namespace *ns)
> +{
> + if (atomic_dec_and_test(&ns->cnt))
> +  free_uts_ns(ns);
> +}
> +
> +#define system_utsname (current->uts_ns->name)
> +extern struct uts_namespace init_uts_ns;
> +#else
> +#define get_uts_ns(ns) do { } while (0)
> +#define put_uts_ns(ns) do { } while (0)
>  extern struct new_utsname system_utsname;
> +#endif
>
>  extern struct rw_semaphore uts_sem;
>  #endif
> --- a/init/version.c
> +++ b/init/version.c
> @@ -17,6 +17,51 @@
>
>  int version_string(LINUX_VERSION_CODE);
>
> +#ifdef CONFIG_UTS_NS
> +struct uts_namespace init_uts_ns = {
> + .cnt = ATOMIC_INIT(1),
> + .name = {
> +  .sysname = UTS_SYSNAME,
> +  .nodename = UTS_NODENAME,
> +  .release = UTS_RELEASE,
> +  .version = UTS_VERSION,
> +  .machine = UTS_MACHINE,
> +  .domainname = UTS_DOMAINNAME,
> + },
> +};

> +struct uts_namespace *create_uts_ns(void)
> +{
> + struct uts_namespace *ns;
> +
> + ns = kmalloc(sizeof(struct uts_namespace), GFP_KERNEL);
> + if (ns == NULL)
> +  return NULL;
> +
> + memset(&ns->name, 0, sizeof(ns->name));
> + atomic_set(&ns->cnt, 1);
> + return ns;
```

```
> +}
```

Setting name to 0. Seems very wrong.

```
> +struct uts_namespace *clone_uts_ns(void)
> +{
> + struct uts_namespace *ns, *cur;
> +
> + ns = kmalloc(sizeof(struct uts_namespace), GFP_KERNEL);
> + if (ns == NULL)
> +  return NULL;
> +
> + cur = current->uts_ns;
> + memcpy(&ns->name, &cur->name, sizeof(cur->name));
> + atomic_set(&ns->cnt, 1);
> + return ns;
> +}
```

Having both create_uts_ns and clone_uts_ns is redundant.

There is a reasonable argument for copying resetting
nodename and domainname to UTS_NODENAME, and UTS_DOMAINNAME,
when we create a new instance. As we almost certainly do not
want to continue to use the old ones and we need some value
in there. Plus some user space tools special case the default
UTS_NODENAME for setting the name from DHCP.

We can also do this from user space so it is not a huge deal
either way. We certainly need to copy the other fields
and not initialize them from the defaults because there is
kernel initialization code that modifies the defaults.

```
> +void free_uts_ns(struct uts_namespace *ns)
> +{
> + kfree(ns);
> +}
> +#else
>  struct new_utsname system_utsname = {
>   .sysname = UTS_SYSNAME,
>   .nodename = UTS_NODENAME,
> @@ -27,6 +72,7 @@ struct new_utsname system_utsname = {
>  };
>
>  EXPORT_SYMBOL(system_utsname);
> +#endif
```

I am a little concerned about the maintenance issues with
placing having two identical initializers from system_utsname.

```
>  const char linux_banner[] =
>   "Linux version " UTS_RELEASE " (" LINUX_COMPILE_BY "@"
> --- a/kernel/exit.c
> +++ b/kernel/exit.c
> @@ -107,6 +107,7 @@ repeat:
>   spin_unlock(&p->proc_lock);
>   proc_pid_flush(proc_dentry);
>   release_thread(p);
> + put_uts_ns(p->uts_ns);
>   put_task_struct(p);
>
>   p = leader;
> --- a/kernel/fork.c
> +++ b/kernel/fork.c
> @@ -1192,6 +1192,7 @@ static task_t *copy_process(unsigned lon
>   }
>   attach_pid(p, PIDTYPE_TGID, p->tgid);
>   attach_pid(p, PIDTYPE_PID, p->pid);
> + get_uts_ns(p->uts_ns);
>
>   nr_threads++;
>   total_forks++;
```

We don't need the tasklist_lock around get_uts_ns.
Please move it earlier.

```
> --- a/kernel/sysctl.c
> +++ b/kernel/sysctl.c
> @@ -229,12 +229,18 @@ static ctl_table root_table[] = {
>   { .ctl_name = 0 }
>  };
```

Since we can disable CONFIG_SYSCTL we probably want to leave
this part for another patch. Although incremental progress
is probably ok.

```
> +#ifdef CONFIG_UTS_NS
> +#define sysctl_system_utsname (init_uts_ns.name)
> +#else
> +#define sysctl_system_utsname (system_utsname)
> +#endif
> +
>  static ctl_table kern_table[] = {
>   {
>    .ctl_name = KERN_OSTYPE,
>    .procname = "ostype",
> -  .data  = system_utsname.sysname,
```

```
> -  .maxlen  = sizeof(system_utsname.sysname),
> +  .data  = sysctl_system_utsname.sysname,
> +  .maxlen  = sizeof(sysctl_system_utsname.sysname),
>    .mode  = 0444,
>    .proc_handler = &proc_doutsstring,
>    .strategy = &sysctl_string,
> @@ -242,8 +248,8 @@ static ctl_table kern_table[] = {
>   {
>    .ctl_name = KERN_OSRELEASE,
>    .procname = "osrelease",
> -  .data  = system_utsname.release,
> -  .maxlen  = sizeof(system_utsname.release),
> +  .data  = sysctl_system_utsname.release,
> +  .maxlen  = sizeof(sysctl_system_utsname.release),
>    .mode  = 0444,
>    .proc_handler = &proc_doutsstring,
>    .strategy = &sysctl_string,
> @@ -251,8 +257,8 @@ static ctl_table kern_table[] = {
>   {
>    .ctl_name = KERN_VERSION,
>    .procname = "version",
> -  .data  = system_utsname.version,
> -  .maxlen  = sizeof(system_utsname.version),
> +  .data  = sysctl_system_utsname.version,
> +  .maxlen  = sizeof(sysctl_system_utsname.version),
>    .mode  = 0444,
>    .proc_handler = &proc_doutsstring,
>    .strategy = &sysctl_string,
> @@ -260,8 +266,8 @@ static ctl_table kern_table[] = {
>   {
>    .ctl_name = KERN_NODENAME,
>    .procname = "hostname",
> -  .data  = system_utsname.nodename,
> -  .maxlen  = sizeof(system_utsname.nodename),
> +  .data  = sysctl_system_utsname.nodename,
> +  .maxlen  = sizeof(sysctl_system_utsname.nodename),
>    .mode  = 0644,
>    .proc_handler = &proc_doutsstring,
>    .strategy = &sysctl_string,
> @@ -269,8 +275,8 @@ static ctl_table kern_table[] = {
>   {
>    .ctl_name = KERN_DOMAINNAME,
>    .procname = "domainname",
> -  .data  = system_utsname.domainname,
> -  .maxlen  = sizeof(system_utsname.domainname),
> +  .data  = sysctl_system_utsname.domainname,
> +  .maxlen  = sizeof(sysctl_system_utsname.domainname),
>    .mode  = 0644,
```

> .proc_handler = &proc_doutsstring,
> .strategy = &sysctl_string,