Subject: [RFC][PATCH 0/5] uts namespaces: Introduction
Posted by serue on Fri, 07 Apr 2006 18:36:00 GMT
View Forum Message <> Reply to Message

Introduce utsname namespaces. Instead of a single system_utsname
containing hostname domainname etc, a process can request it's
copy of the uts info to be cloned. The data will be copied from
it's original, but any further changes will not be seen by processes
which are not it's children, and vice versa.

This is useful, for instance, for vserver/openvz, which can now clone
a new uts namespace for each new virtual server.

This patchset is based on Kirill Korotaev's Mar 24 submission, taking
comments (in particular from James Morris and Eric Biederman) into
account.

Some performance results are attached. I was mainly curious whether
it would be worth putting the task_struct->uts_ns pointer inside
a #ifdef CONFIG_UTS_NS. The result show that leaving it in when
CONFIG_UTS_NS=n has negligable performance impact, so that is the
approach this patch takes.

-serge

Performance testing was done on a 2-cpu hyperthreaded
x86 box with 16G ram. The following tests were run:
dbench (20 times, four clients, on reiser fs non-isolated partition)
tbench (20 times, 5 connections)
kernbench (20 times)
reaim (20 times ranging from 1 to 15 users)

They were run on 2.6.17-rc1:
pristine
patched, but with !CONFIG_UTS_NS ("disabled")
patched with CONFIG_UTS_NS=y ("enabled")

All results are presented as means +/- 95% confidence interval.

Dbench results:
pristine: 387.080727 +/- 9.344585
patched disabled: 389.524364 +/- 9.574921
patched enabled: 370.155600 +/- 30.127808

Tbench results:
pristine: 388.940100 +/- 18.095104
patched disabled: 389.173700 +/- 23.658035
patched enabled: 394.333200 +/- 25.813393

Kernbench results:
pristine: 70.317500 +/- 0.210833
patched, disabled: 70.860000 +/- 0.179292
patched, enabled: 70.346500 +/- 0.184784

Reaim results:
pristine:

Nclients	Mean	95% CI
1	106080.000000	11327.896029
3	236057.142000	18205.544810
5	247867.136000	23536.800062
7	265370.000000	21284.335743
9	262969.936000	18225.497529
11	278256.000000	6230.342816
13	284288.016000	8924.589388
15	286987.170000	7881.034658

patched, disabled:

Nclients	Mean	95% CI
1	105400.000000	8739.978241
3	229500.000000	0.000000
5	252325.176667	16685.663423
7	265125.000000	6747.777319
9	271258.645000	11715.635212
11	280662.608333	7775.229351
13	277719.706667	8173.390359
15	278515.421667	10963.211450

patched, enabled:

Nclients	Mean	95% CI
1	102000.000000	0.000000
3	224400.000000	14159.870036
5	242963.288000	40529.490781
7	255150.000000	8745.802081
9	270154.284000	8918.863136
11	283134.260000	12239.361252
13	288497.540000	11336.550964
15	280022.728000	8804.882369

Subject: [RFC][PATCH 1/5] uts namespaces: Implement utsname namespaces
Posted by serue on Fri, 07 Apr 2006 18:36:00 GMT
View Forum Message <> Reply to Message

This patch defines the uts namespace and some manipulators.
Adds the uts namespace to task_struct, and initializes a
system-wide init namespace which will continue to be used when

it makes sense.

Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>
---
 include/linux/init_task.h | 2 +
 include/linux/sched.h | 2 +
 include/linux/utsname.h | 40 ++++++++++++++++++++++++-
 init/Kconfig | 8 +++++
 init/version.c | 70 ++-----
 kernel/exit.c | 2 +
 kernel/fork.c | 9 +++++-
 7 files changed, 122 insertions(+), 11 deletions(-)

14c326d603d88d9ed40a1ddafbf23fc3da68a645
diff --git a/include/linux/init_task.h b/include/linux/init_task.h
index 41ecbb8..21b1751 100644
--- 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 \
 { \
@@ -123,6 +124,7 @@ extern struct group_info init_groups;
  .journal_info = NULL, \
  .cpu_timers = INIT_CPU_TIMERS(tsk.cpu_timers), \
  .fs_excl = ATOMIC_INIT(0), \
+ .uts_ns = &init_uts_ns, \
 }

diff --git a/include/linux/sched.h b/include/linux/sched.h
index 541f482..97c7990 100644
--- a/include/linux/sched.h
+++ b/include/linux/sched.h
@@ -684,6 +684,7 @@ static inline void prefetch_stack(struct

 struct audit_context; /* See audit.c */
 struct mempolicy;
+struct uts_namespace;

 enum sleep_type {
  SLEEP_NORMAL,
@@ -807,6 +808,7 @@ struct task_struct {
  struct files_struct *files;

```
 /* namespace */
  struct namespace *namespace;
+ struct uts_namespace *uts_ns;
 /* signal handlers */
  struct signal_struct *signal;
  struct sighand_struct *sighand;
diff --git a/include/linux/utsname.h b/include/linux/utsname.h
index 13e1da0..cc28ac5 100644
--- a/include/linux/utsname.h
+++ b/include/linux/utsname.h
@@ -1,5 +1,8 @@
 #ifndef _LINUX_UTSNAME_H
 #define _LINUX_UTSNAME_H
+#include <linux/sched.h>
+#include <linux/kref.h>
+#include <asm/atomic.h>

 #define __OLD_UTS_LEN 8

@@ -30,7 +33,42 @@ struct new_utsname {
  char domainname[65];
 };

-extern struct new_utsname system_utsname;
+struct uts_namespace {
+ struct kref kref;
+ struct new_utsname name;
+};
+extern struct uts_namespace init_uts_ns;
+
+#ifdef CONFIG_UTS_NS
+
+extern struct uts_namespace *clone_uts_ns(struct uts_namespace *old_ns);
+extern struct uts_namespace *unshare_uts_ns(void);
+extern void free_uts_ns(struct kref *kref);
+
+static inline void get_uts_ns(struct uts_namespace *ns)
+{
+ kref_get(&ns->kref);
+}
+
+static inline void put_uts_ns(struct uts_namespace *ns)
+{
+ kref_put(&ns->kref, free_uts_ns);
+}
+
+#else
+static inline void get_uts_ns(struct uts_namespace *ns)
```

```
+{
+}
+static inline void put_uts_ns(struct uts_namespace *ns)
+{
+}
+#endif
+
+static inline struct new_utsname *utsname(void)
+{
+ return &current->uts_ns->name;
+}
+

 extern struct rw_semaphore uts_sem;
 #endif
diff --git a/init/Kconfig b/init/Kconfig
index 3b36a1d..8460e5a 100644
--- a/init/Kconfig
+++ b/init/Kconfig
@@ -166,6 +166,14 @@ config SYSCTL
    building a kernel for install/rescue disks or your system is very
    limited in memory.

+config UTS_NS
+ bool "UTS Namespaces"
+ default n
+ help
+   Support uts namespaces.  This allows containers, i.e.
+   vservers, to use uts namespaces to provide different
+   uts info for different servers.  If unsure, say N.
+
 config AUDIT
  bool "Auditing support"
  depends on NET
diff --git a/init/version.c b/init/version.c
index 3ddc3ce..e128d72 100644
--- a/init/version.c
+++ b/init/version.c
@@ -11,22 +11,76 @@
 #include <linux/uts.h>
 #include <linux/utsname.h>
 #include <linux/version.h>
+#include <linux/sched.h>

 #define version(a) Version_ ## a
 #define version_string(a) version(a)

 int version_string(LINUX_VERSION_CODE);
```

```
-struct new_utsname system_utsname = {
- .sysname = UTS_SYSNAME,
- .nodename = UTS_NODENAME,
- .release = UTS_RELEASE,
- .version = UTS_VERSION,
- .machine = UTS_MACHINE,
- .domainname = UTS_DOMAINNAME,
+struct uts_namespace init_uts_ns = {
+ .kref = {
+  .refcount = ATOMIC_INIT(2),
+ },
+ .name = {
+  .sysname = UTS_SYSNAME,
+  .nodename = UTS_NODENAME,
+  .release = UTS_RELEASE,
+  .version = UTS_VERSION,
+  .machine = UTS_MACHINE,
+  .domainname = UTS_DOMAINNAME,
+ },
 };

-EXPORT_SYMBOL(system_utsname);
+#ifdef CONFIG_UTS_NS
+/*
+ * Clone a new ns copying an original utsname, setting refcount to 1
+ * @old_ns: namespace to clone
+ * Return NULL on error (failure to kmalloc), new ns otherwise
+ */
+struct uts_namespace *clone_uts_ns(struct uts_namespace *old_ns)
+{
+ struct uts_namespace *ns;
+
+ ns = kmalloc(sizeof(struct uts_namespace), GFP_KERNEL);
+ if (ns) {
+  memcpy(&ns->name, &old_ns->name, sizeof(ns->name));
+  kref_init(&ns->kref);
+ }
+ return ns;
+}
+
+/*
+ * unshare the current process' utsname namespace.  Changes
+ * to the utsname of this process won't be seen by parent, and
+ * vice versa
+ *
+ * Return NULL on error (failure to kmalloc), new ns otherwise
+ *
```

```
+ * TODO: decide where this should be locked  (depends on how/where
+ * we decide to use this)
+ */
+struct uts_namespace *unshare_uts_ns(void)
+{
+ struct uts_namespace *old_ns = current->uts_ns;
+ struct uts_namespace *new_ns = clone_uts_ns(old_ns);
+ if (new_ns) {
+  current->uts_ns = new_ns;
+  put_uts_ns(old_ns);
+ }
+ return new_ns;
+}
+EXPORT_SYMBOL(unshare_uts_ns);
+
+void free_uts_ns(struct kref *kref)
+{
+ struct uts_namespace *ns;
+
+ ns = container_of(kref, struct uts_namespace, kref);
+ kfree(ns);
+}
+EXPORT_SYMBOL(free_uts_ns);
+#endif

 const char linux_banner[] =
  "Linux version " UTS_RELEASE " (" LINUX_COMPILE_BY "@"
diff --git a/kernel/exit.c b/kernel/exit.c
index 6c2eeb8..97c5405 100644
--- a/kernel/exit.c
+++ b/kernel/exit.c
@@ -34,6 +34,7 @@
 #include <linux/mutex.h>
 #include <linux/futex.h>
 #include <linux/compat.h>
+#include <linux/utsname.h>

 #include <asm/uaccess.h>
 #include <asm/unistd.h>
@@ -173,6 +174,7 @@ repeat:
  spin_unlock(&p->proc_lock);
  proc_pid_flush(proc_dentry);
  release_thread(p);
+ put_uts_ns(p->uts_ns);
  call_rcu(&p->rcu, delayed_put_task_struct);

  p = leader;
diff --git a/kernel/fork.c b/kernel/fork.c
```

```
index 3384eb8..62e4479 100644
--- a/kernel/fork.c
+++ b/kernel/fork.c
@@ -44,6 +44,7 @@
 #include <linux/rmap.h>
 #include <linux/acct.h>
 #include <linux/cn_proc.h>
+#include <linux/utsname.h>

 #include <asm/pgtable.h>
 #include <asm/pgalloc.h>
@@ -1119,6 +1120,8 @@ static task_t *copy_process(unsigned lon
  /* Perform scheduler related setup. Assign this task to a CPU. */
  sched_fork(p, clone_flags);

+ get_uts_ns(p->uts_ns);
+
  /* Need tasklist lock for parent etc handling! */
  write_lock_irq(&tasklist_lock);

@@ -1158,7 +1161,7 @@ static task_t *copy_process(unsigned lon
   spin_unlock(&current->sighand->siglock);
   write_unlock_irq(&tasklist_lock);
   retval = -ERESTARTNOINTR;
-  goto bad_fork_cleanup_namespace;
+  goto bad_fork_cleanup_utsns;
  }

  if (clone_flags & CLONE_THREAD) {
@@ -1171,7 +1174,7 @@ static task_t *copy_process(unsigned lon
    spin_unlock(&current->sighand->siglock);
    write_unlock_irq(&tasklist_lock);
    retval = -EAGAIN;
-   goto bad_fork_cleanup_namespace;
+   goto bad_fork_cleanup_utsns;
   }

   p->group_leader = current->group_leader;
@@ -1223,6 +1226,8 @@ static task_t *copy_process(unsigned lon
  proc_fork_connector(p);
  return p;

+bad_fork_cleanup_utsns:
+ put_uts_ns(p->uts_ns);
 bad_fork_cleanup_namespace:
  exit_namespace(p);
 bad_fork_cleanup_keys:
--
```

Subject: [RFC][PATCH 5/5] uts namespaces: Enable UTS namespaces debugging
Posted by serue on Fri, 07 Apr 2006 18:36:00 GMT
View Forum Message <> Reply to Message

Adds a /uts directory in debugfs which exposes the current UTS
namespace. If a file in this directory is changed, it will
unshare and modify the UTS namespace of the current process.

Clearly this is purely for testing purposes. Testing namespaces
in this fashion allows us to postpone concensus on a namespace
unsharing mechanism.

Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>
---
fs/debugfs/Makefile | 2 -
fs/debugfs/uts.c | 170 ++
lib/Kconfig.debug | 11 +++
3 files changed, 182 insertions(+), 1 deletions(-)
create mode 100644 fs/debugfs/uts.c

67f3dc966381313f31ee3643183161a8c230199b
diff --git a/fs/debugfs/Makefile b/fs/debugfs/Makefile
index 840c456..e5df8b9 100644
--- a/fs/debugfs/Makefile
+++ b/fs/debugfs/Makefile
@@ -1,4 +1,4 @@
debugfs-objs := inode.o file.o

obj-$(CONFIG_DEBUG_FS) += debugfs.o
-
+obj-$(CONFIG_DEBUG_UTS_NS) += uts.o
diff --git a/fs/debugfs/uts.c b/fs/debugfs/uts.c
new file mode 100644
index 0000000..45b29af
--- /dev/null
+++ b/fs/debugfs/uts.c
@@ -0,0 +1,170 @@
+/*
+ * uts.c - adds a uts/ directory to debug UTS namespaces
+ *
+ * Copyright (C) 2006 IBM
+ *
+ * Author: Cedric Le Goater <clg@fr.ibm.com>
+ *
+ * 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, version 2 of the
+ * License.
+ */
+
+#include <linux/module.h>
+#include <linux/kernel.h>
+#include <linux/pagemap.h>
+#include <linux/debugfs.h>
+#include <linux/utsname.h>
+
+static struct dentry *uts_dentry;
+static struct dentry *uts_dentry_sysname;
+static struct dentry *uts_dentry_nodename;
+static struct dentry *uts_dentry_release;
+static struct dentry *uts_dentry_version;
+static struct dentry *uts_dentry_machine;
+static struct dentry *uts_dentry_domainname;
+
+static inline char* uts_buffer(struct dentry *dentry)
+{
+ if (dentry == uts_dentry_sysname)
+  return utsname()->sysname;
+ else if (dentry == uts_dentry_nodename)
+  return utsname()->nodename;
+ else if (dentry == uts_dentry_release)
+  return utsname()->release;
+ else if (dentry == uts_dentry_version)
+  return utsname()->version;
+ else if (dentry == uts_dentry_machine)
+  return utsname()->machine;
+ else if (dentry == uts_dentry_domainname)
+  return utsname()->domainname;
+ else
+  return NULL;
+}
+
+static ssize_t uts_read_file(struct file *file, char __user *user_buf,
+        size_t count, loff_t *ppos)
+{
+ size_t len;
+ char* buf;
+
+ if (*ppos < 0)
+  return -EINVAL;
+ if (*ppos >= count)
+  return 0;
+
```

```
+ buf = uts_buffer(file->f_dentry);
+ if (!buf)
+  return -ENOENT;
+
+ len = strlen(buf);
+ if (len > count)
+  len = count;
+ if (len)
+  if (copy_to_user(user_buf, buf, len))
+   return -EFAULT;
+ if (len < count) {
+  if (put_user('\n', ((char __user *) user_buf) + len))
+   return -EFAULT;
+  len++;
+ }
+
+ *ppos += count;
+ return count;
+}
+
+
+static ssize_t uts_write_file(struct file * file, const char __user * user_buf,
+         size_t count, loff_t *ppos)
+
+{
+ size_t len;
+ const char __user *p;
+ char c;
+ char* buf;
+
+ if (!unshare_uts_ns())
+  return -ENOMEM;
+
+ buf = uts_buffer(file->f_dentry);
+ if (!buf)
+  return -ENOENT;
+
+ len = 0;
+ p = user_buf;
+ while (len < count) {
+  if (get_user(c, p++))
+   return -EFAULT;
+  if (c == 0 || c == '\n')
+   break;
+  len++;
+ }
+
+ if (len >= __NEW_UTS_LEN)
```

```
+  len = __NEW_UTS_LEN - 1;
+
+ if (copy_from_user(buf, user_buf, len))
+  return -EFAULT;
+
+ buf[len] = 0;
+
+ *ppos += count;
+ return count;
+}
+
+static int uts_open(struct inode *inode, struct file *file)
+{
+ return 0;
+}
+
+static struct file_operations uts_file_operations = {
+ .read =  uts_read_file,
+ .write = uts_write_file,
+ .open =  uts_open,
+};
+
+static int __init uts_init(void)
+{
+ uts_dentry = debugfs_create_dir("uts", NULL);
+
+ uts_dentry_sysname = debugfs_create_file("sysname", 0666,
+       uts_dentry, NULL,
+       &uts_file_operations);
+ uts_dentry_nodename = debugfs_create_file("nodename", 0666,
+        uts_dentry, NULL,
+        &uts_file_operations);
+ uts_dentry_release = debugfs_create_file("release", 0666,
+       uts_dentry, NULL,
+       &uts_file_operations);
+ uts_dentry_version = debugfs_create_file("version", 0666,
+       uts_dentry, NULL,
+       &uts_file_operations);
+ uts_dentry_machine = debugfs_create_file("machine", 0666,
+       uts_dentry, NULL,
+       &uts_file_operations);
+ uts_dentry_domainname = debugfs_create_file("domainname", 0666,
+          uts_dentry, NULL,
+          &uts_file_operations);
+ return 0;
+}
+
+static void __exit uts_exit(void)
```

```
+{
+ debugfs_remove(uts_dentry_sysname);
+ debugfs_remove(uts_dentry_nodename);
+ debugfs_remove(uts_dentry_release);
+ debugfs_remove(uts_dentry_version);
+ debugfs_remove(uts_dentry_machine);
+ debugfs_remove(uts_dentry_domainname);
+ debugfs_remove(uts_dentry);
+}
+
+module_init(uts_init);
+module_exit(uts_exit);
+
+
+MODULE_DESCRIPTION("UTS namespace debugfs");
+MODULE_AUTHOR("Cedric Le Goater <clg@fr.ibm.com>");
+MODULE_LICENSE("GPL");
+
diff --git a/lib/Kconfig.debug b/lib/Kconfig.debug
index d57fd91..5ed09b8 100644
--- a/lib/Kconfig.debug
+++ b/lib/Kconfig.debug
@@ -223,3 +223,14 @@ config RCU_TORTURE_TEST
   at boot time (you probably don't).
   Say M if you want the RCU torture tests to build as a module.
   Say N if you are unsure.
+
+config DEBUG_UTS_NS
+ tristate "UTS Namespaces debugging"
+ depends on UTS_NS
+ select DEBUG_FS
+ default n
+ help
+   This option provides a kernel module that adds a uts/ directory
+   in debugfs which can be used to unshare and modify the uts namespace
+   of the current process and children. If unsure, say N.
+
--
1.2.4
```

Subject: [RFC][PATCH 2/5] uts namespaces: Switch to using uts namespaces
Posted by serue on Fri, 07 Apr 2006 18:36:00 GMT

View Forum Message <> Reply to Message

Replace references to system_utsname to the per-process uts namespace
where appropriate. This includes things like uname.

Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>
---
 arch/alpha/kernel/osf_sys.c | 24 ++++++++++++------------
 arch/i386/kernel/sys_i386.c | 12 ++++++------
 arch/ia64/sn/kernel/sn2/sn_hwperf.c | 2 +-
 arch/m32r/kernel/sys_m32r.c | 2 +-
 arch/mips/kernel/linux32.c | 2 +-
 arch/mips/kernel/syscall.c | 18 +++++++++---------
 arch/mips/kernel/sysirix.c | 12 ++++++------
 arch/parisc/hpux/sys_hpux.c | 22 ++++++++++++----------
 arch/powerpc/kernel/syscalls.c | 14 +++++++-------
 arch/sh/kernel/sys_sh.c | 2 +-
 arch/sh64/kernel/sys_sh64.c | 2 +-
 arch/sparc/kernel/sys_sparc.c | 4 ++--
 arch/sparc/kernel/sys_sunos.c | 10 +++++-----
 arch/sparc64/kernel/sys_sparc.c | 4 ++--
 arch/sparc64/kernel/sys_sunos32.c | 10 +++++-----
 arch/sparc64/solaris/misc.c | 6 +++---
 arch/um/drivers/mconsole_kern.c | 6 +++---
 arch/um/kernel/syscall_kern.c | 12 ++++++------
 arch/um/sys-x86_64/syscalls.c | 2 +-
 arch/x86_64/ia32/sys_ia32.c | 10 +++++-----
 arch/x86_64/kernel/sys_x86_64.c | 2 +-
 arch/xtensa/kernel/syscalls.c | 2 +-
 drivers/char/random.c | 4 ++--
 fs/cifs/connect.c | 28 ++++++++++++++--------------
 fs/exec.c | 2 +-
 fs/lockd/clntproc.c | 4 ++--
 fs/lockd/mon.c | 2 +-
 fs/lockd/svclock.c | 2 +-
 fs/lockd/xdr.c | 2 +-
 fs/nfs/nfsroot.c | 2 +-
 include/linux/lockd/lockd.h | 2 +-
 kernel/sys.c | 14 +++++++-------
 32 files changed, 121 insertions(+), 121 deletions(-)

92a8cf13a78415ed5ec9068698b5039ddcc00210
diff --git a/arch/alpha/kernel/osf_sys.c b/arch/alpha/kernel/osf_sys.c
index 31afe3d..b793b96 100644
--- a/arch/alpha/kernel/osf_sys.c
+++ b/arch/alpha/kernel/osf_sys.c
@@ -402,15 +402,15 @@ osf_utsname(char __user *name)

  down_read(&uts_sem);
  error = -EFAULT;
- if (copy_to_user(name + 0, system_utsname.sysname, 32))
+ if (copy_to_user(name + 0, utsname()->sysname, 32))
   goto out;

```
- if (copy_to_user(name + 32, system_utsname.nodename, 32))
+ if (copy_to_user(name + 32, utsname()->nodename, 32))
   goto out;
- if (copy_to_user(name + 64, system_utsname.release, 32))
+ if (copy_to_user(name + 64, utsname()->release, 32))
   goto out;
- if (copy_to_user(name + 96, system_utsname.version, 32))
+ if (copy_to_user(name + 96, utsname()->version, 32))
   goto out;
- if (copy_to_user(name + 128, system_utsname.machine, 32))
+ if (copy_to_user(name + 128, utsname()->machine, 32))
   goto out;

  error = 0;
@@ -449,8 +449,8 @@ osf_getdomainname(char __user *name, int

  down_read(&uts_sem);
  for (i = 0; i < len; ++i) {
-  __put_user(system_utsname.domainname[i], name + i);
-  if (system_utsname.domainname[i] == '\0')
+  __put_user(utsname()->domainname[i], name + i);
+  if (utsname()->domainname[i] == '\0')
     break;
  }
  up_read(&uts_sem);
@@ -608,11 +608,11 @@ asmlinkage long
 osf_sysinfo(int command, char __user *buf, long count)
 {
  static char * sysinfo_table[] = {
-  system_utsname.sysname,
-  system_utsname.nodename,
-  system_utsname.release,
-  system_utsname.version,
-  system_utsname.machine,
+  utsname()->sysname,
+  utsname()->nodename,
+  utsname()->release,
+  utsname()->version,
+  utsname()->machine,
   "alpha", /* instruction set architecture */
   "dummy", /* hardware serial number */
   "dummy", /* hardware manufacturer */
diff --git a/arch/i386/kernel/sys_i386.c b/arch/i386/kernel/sys_i386.c
index 8fdb1fb..4af731d 100644
--- a/arch/i386/kernel/sys_i386.c
+++ b/arch/i386/kernel/sys_i386.c
@@ -210,7 +210,7 @@ asmlinkage int sys_uname(struct old_utsn
  if (!name)
```

```
   return -EFAULT;
  down_read(&uts_sem);
- err=copy_to_user(name, &system_utsname, sizeof (*name));
+ err=copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  return err?-EFAULT:0;
 }
@@ -226,15 +226,15 @@ asmlinkage int sys_olduname(struct oldol

    down_read(&uts_sem);

- error = __copy_to_user(&name->sysname,&system_utsname.sysname,__OLD_UTS_LEN);
+ error = __copy_to_user(&name->sysname,&utsname()->sysname,__OLD_UTS_LEN);
  error |= __put_user(0,name->sysname+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->nodename,&system_utsname.nodename,__OLD_UTS_LEN);
+ error |= __copy_to_user(&name->nodename,&utsname()->nodename,__OLD_UTS_LEN);
  error |= __put_user(0,name->nodename+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->release,&system_utsname.release,__OLD_UTS_LEN);
+ error |= __copy_to_user(&name->release,&utsname()->release,__OLD_UTS_LEN);
  error |= __put_user(0,name->release+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->version,&system_utsname.version,__OLD_UTS_LEN);
+ error |= __copy_to_user(&name->version,&utsname()->version,__OLD_UTS_LEN);
  error |= __put_user(0,name->version+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->machine,&system_utsname.machine,__OLD_UTS_LEN);
+ error |= __copy_to_user(&name->machine,&utsname()->machine,__OLD_UTS_LEN);
  error |= __put_user(0,name->machine+__OLD_UTS_LEN);

  up_read(&uts_sem);
diff --git a/arch/ia64/sn/kernel/sn2/sn_hwperf.c b/arch/ia64/sn/kernel/sn2/sn_hwperf.c
index d917afa..a0632a9 100644
--- a/arch/ia64/sn/kernel/sn2/sn_hwperf.c
+++ b/arch/ia64/sn/kernel/sn2/sn_hwperf.c
@@ -420,7 +420,7 @@ static int sn_topology_show(struct seq_f
    "coherency_domain %d, "
    "region_size %d\n",

-   partid, system_utsname.nodename,
+   partid, utsname()->nodename,
    shubtype ? "shub2" : "shub1",
    (u64)nasid_mask << nasid_shift, nasid_msb, nasid_shift,
    system_size, sharing_size, coher, region_size);
diff --git a/arch/m32r/kernel/sys_m32r.c b/arch/m32r/kernel/sys_m32r.c
index 670cb49..11412c0 100644
--- a/arch/m32r/kernel/sys_m32r.c
+++ b/arch/m32r/kernel/sys_m32r.c
@@ -206,7 +206,7 @@ asmlinkage int sys_uname(struct old_utsn
  if (!name)
   return -EFAULT;
```

```
  down_read(&uts_sem);
- err=copy_to_user(name, &system_utsname, sizeof (*name));
+ err=copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  return err?-EFAULT:0;
 }
diff --git a/arch/mips/kernel/linux32.c b/arch/mips/kernel/linux32.c
index 3f40c37..b9b702f 100644
--- a/arch/mips/kernel/linux32.c
+++ b/arch/mips/kernel/linux32.c
@@ -1100,7 +1100,7 @@ asmlinkage long sys32_newuname(struct ne
  int ret = 0;

  down_read(&uts_sem);
- if (copy_to_user(name,&system_utsname,sizeof *name))
+ if (copy_to_user(name,utsname(),sizeof *name))
   ret = -EFAULT;
  up_read(&uts_sem);

diff --git a/arch/mips/kernel/syscall.c b/arch/mips/kernel/syscall.c
index 2aeaa2f..8b13d57 100644
--- a/arch/mips/kernel/syscall.c
+++ b/arch/mips/kernel/syscall.c
@@ -232,7 +232,7 @@ out:
  */
 asmlinkage int sys_uname(struct old_utsname __user * name)
 {
- if (name && !copy_to_user(name, &system_utsname, sizeof (*name)))
+ if (name && !copy_to_user(name, utsname(), sizeof (*name)))
   return 0;
  return -EFAULT;
 }
@@ -249,15 +249,15 @@ asmlinkage int sys_olduname(struct oldol
  if (!access_ok(VERIFY_WRITE,name,sizeof(struct oldold_utsname)))
   return -EFAULT;

- error = __copy_to_user(&name->sysname,&system_utsname.sysname,__OLD_UTS_LEN);
+ error = __copy_to_user(&name->sysname,&utsname()->sysname,__OLD_UTS_LEN);
  error -= __put_user(0,name->sysname+__OLD_UTS_LEN);
- error -= __copy_to_user(&name->nodename,&system_utsname.nodename,__OLD_UTS_LEN);
+ error -= __copy_to_user(&name->nodename,&utsname()->nodename,__OLD_UTS_LEN);
  error -= __put_user(0,name->nodename+__OLD_UTS_LEN);
- error -= __copy_to_user(&name->release,&system_utsname.release,__OLD_UTS_LEN);
+ error -= __copy_to_user(&name->release,&utsname()->release,__OLD_UTS_LEN);
  error -= __put_user(0,name->release+__OLD_UTS_LEN);
- error -= __copy_to_user(&name->version,&system_utsname.version,__OLD_UTS_LEN);
+ error -= __copy_to_user(&name->version,&utsname()->version,__OLD_UTS_LEN);
  error -= __put_user(0,name->version+__OLD_UTS_LEN);
```

```
- error -= __copy_to_user(&name->machine,&system_utsname.machine,__OLD_UTS_LEN);
+ error -= __copy_to_user(&name->machine,&utsname()->machine,__OLD_UTS_LEN);
  error = __put_user(0,name->machine+__OLD_UTS_LEN);
  error = error ? -EFAULT : 0;

@@ -293,10 +293,10 @@ asmlinkage int _sys_sysmips(int cmd, lon
    return -EFAULT;

   down_write(&uts_sem);
-  strncpy(system_utsname.nodename, nodename, len);
+  strncpy(utsname()->nodename, nodename, len);
   nodename[__NEW_UTS_LEN] = '\0';
-  strlcpy(system_utsname.nodename, nodename,
-          sizeof(system_utsname.nodename));
+  strlcpy(utsname()->nodename, nodename,
+          sizeof(utsname()->nodename));
   up_write(&uts_sem);
   return 0;
  }
diff --git a/arch/mips/kernel/sysirix.c b/arch/mips/kernel/sysirix.c
index 5407b78..1b4e7e7 100644
--- a/arch/mips/kernel/sysirix.c
+++ b/arch/mips/kernel/sysirix.c
@@ -884,7 +884,7 @@ asmlinkage int irix_getdomainname(char _
  down_read(&uts_sem);
  if (len > __NEW_UTS_LEN)
   len = __NEW_UTS_LEN;
- err = copy_to_user(name, system_utsname.domainname, len) ? -EFAULT : 0;
+ err = copy_to_user(name, utsname()->domainname, len) ? -EFAULT : 0;
  up_read(&uts_sem);

  return err;
@@ -1127,11 +1127,11 @@ struct iuname {
 asmlinkage int irix_uname(struct iuname __user *buf)
 {
  down_read(&uts_sem);
- if (copy_from_user(system_utsname.sysname, buf->sysname, 65)
-     || copy_from_user(system_utsname.nodename, buf->nodename, 65)
-     || copy_from_user(system_utsname.release, buf->release, 65)
-     || copy_from_user(system_utsname.version, buf->version, 65)
-     || copy_from_user(system_utsname.machine, buf->machine, 65)) {
+ if (copy_from_user(utsname()->sysname, buf->sysname, 65)
+     || copy_from_user(utsname()->nodename, buf->nodename, 65)
+     || copy_from_user(utsname()->release, buf->release, 65)
+     || copy_from_user(utsname()->version, buf->version, 65)
+     || copy_from_user(utsname()->machine, buf->machine, 65)) {
   return -EFAULT;
  }
```

```
  up_read(&uts_sem);
diff --git a/arch/parisc/hpux/sys_hpux.c b/arch/parisc/hpux/sys_hpux.c
index 05273cc..9fc2c08 100644
--- a/arch/parisc/hpux/sys_hpux.c
+++ b/arch/parisc/hpux/sys_hpux.c
@@ -266,15 +266,15 @@ static int hpux_uname(struct hpux_utsnam

  down_read(&uts_sem);

- error = __copy_to_user(&name->sysname,&system_utsname.sysname,HPUX_UTSLEN-1);
+ error = __copy_to_user(&name->sysname,&utsname()->sysname,HPUX_UTSLEN-1);
  error |= __put_user(0,name->sysname+HPUX_UTSLEN-1);
- error |= __copy_to_user(&name->nodename,&system_utsname.nodename,HPUX_UTSLEN-1);
+ error |= __copy_to_user(&name->nodename,&utsname()->nodename,HPUX_UTSLEN-1);
  error |= __put_user(0,name->nodename+HPUX_UTSLEN-1);
- error |= __copy_to_user(&name->release,&system_utsname.release,HPUX_UTSLEN-1);
+ error |= __copy_to_user(&name->release,&utsname()->release,HPUX_UTSLEN-1);
  error |= __put_user(0,name->release+HPUX_UTSLEN-1);
- error |= __copy_to_user(&name->version,&system_utsname.version,HPUX_UTSLEN-1);
+ error |= __copy_to_user(&name->version,&utsname()->version,HPUX_UTSLEN-1);
  error |= __put_user(0,name->version+HPUX_UTSLEN-1);
- error |= __copy_to_user(&name->machine,&system_utsname.machine,HPUX_UTSLEN-1);
+ error |= __copy_to_user(&name->machine,&utsname()->machine,HPUX_UTSLEN-1);
  error |= __put_user(0,name->machine+HPUX_UTSLEN-1);

  up_read(&uts_sem);
@@ -373,8 +373,8 @@ int hpux_utssys(char *ubuf, int n, int t
   /*  TODO:  print a warning about using this?  */
   down_write(&uts_sem);
   error = -EFAULT;
-  if (!copy_from_user(system_utsname.sysname, ubuf, len)) {
-   system_utsname.sysname[len] = 0;
+  if (!copy_from_user(utsname()->sysname, ubuf, len)) {
+   utsname()->sysname[len] = 0;
    error = 0;
   }
   up_write(&uts_sem);
@@ -400,8 +400,8 @@ int hpux_utssys(char *ubuf, int n, int t
   /*  TODO:  print a warning about this?  */
   down_write(&uts_sem);
   error = -EFAULT;
-  if (!copy_from_user(system_utsname.release, ubuf, len)) {
-   system_utsname.release[len] = 0;
+  if (!copy_from_user(utsname()->release, ubuf, len)) {
+   utsname()->release[len] = 0;
    error = 0;
   }
   up_write(&uts_sem);
```

```
@@ -422,13 +422,13 @@ int hpux_getdomainname(char *name, int l

   down_read(&uts_sem);

- nlen = strlen(system_utsname.domainname) + 1;
+ nlen = strlen(utsname()->domainname) + 1;

  if (nlen < len)
   len = nlen;
  if(len > __NEW_UTS_LEN)
   goto done;
- if(copy_to_user(name, system_utsname.domainname, len))
+ if(copy_to_user(name, utsname()->domainname, len))
   goto done;
  err = 0;
 done:
diff --git a/arch/powerpc/kernel/syscalls.c b/arch/powerpc/kernel/syscalls.c
index 9b69d99..d358866 100644
--- a/arch/powerpc/kernel/syscalls.c
+++ b/arch/powerpc/kernel/syscalls.c
@@ -260,7 +260,7 @@ long ppc_newuname(struct new_utsname __u
  int err = 0;

  down_read(&uts_sem);
- if (copy_to_user(name, &system_utsname, sizeof(*name)))
+ if (copy_to_user(name, utsname(), sizeof(*name)))
   err = -EFAULT;
  up_read(&uts_sem);
  if (!err)
@@ -273,7 +273,7 @@ int sys_uname(struct old_utsname __user
  int err = 0;

  down_read(&uts_sem);
- if (copy_to_user(name, &system_utsname, sizeof(*name)))
+ if (copy_to_user(name, utsname(), sizeof(*name)))
   err = -EFAULT;
  up_read(&uts_sem);
  if (!err)
@@ -289,19 +289,19 @@ int sys_olduname(struct oldold_utsname _
   return -EFAULT;

  down_read(&uts_sem);
- error = __copy_to_user(&name->sysname, &system_utsname.sysname,
+ error = __copy_to_user(&name->sysname, &utsname()->sysname,
          __OLD_UTS_LEN);
  error |= __put_user(0, name->sysname + __OLD_UTS_LEN);
- error |= __copy_to_user(&name->nodename, &system_utsname.nodename,
+ error |= __copy_to_user(&name->nodename, &utsname()->nodename,
```

```
     __OLD_UTS_LEN);
  error |= __put_user(0, name->nodename + __OLD_UTS_LEN);
- error |= __copy_to_user(&name->release, &system_utsname.release,
+ error |= __copy_to_user(&name->release, &utsname()->release,
     __OLD_UTS_LEN);
  error |= __put_user(0, name->release + __OLD_UTS_LEN);
- error |= __copy_to_user(&name->version, &system_utsname.version,
+ error |= __copy_to_user(&name->version, &utsname()->version,
     __OLD_UTS_LEN);
  error |= __put_user(0, name->version + __OLD_UTS_LEN);
- error |= __copy_to_user(&name->machine, &system_utsname.machine,
+ error |= __copy_to_user(&name->machine, &utsname()->machine,
     __OLD_UTS_LEN);
  error |= override_machine(name->machine);
  up_read(&uts_sem);
diff --git a/arch/sh/kernel/sys_sh.c b/arch/sh/kernel/sys_sh.c
index 917b2f3..e4966b2 100644
--- a/arch/sh/kernel/sys_sh.c
+++ b/arch/sh/kernel/sys_sh.c
@@ -267,7 +267,7 @@ asmlinkage int sys_uname(struct old_utsn
  if (!name)
   return -EFAULT;
  down_read(&uts_sem);
- err=copy_to_user(name, &system_utsname, sizeof (*name));
+ err=copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  return err?-EFAULT:0;
 }
diff --git a/arch/sh64/kernel/sys_sh64.c b/arch/sh64/kernel/sys_sh64.c
index 58ff7d5..a8dc88c 100644
--- a/arch/sh64/kernel/sys_sh64.c
+++ b/arch/sh64/kernel/sys_sh64.c
@@ -279,7 +279,7 @@ asmlinkage int sys_uname(struct old_utsn
  if (!name)
   return -EFAULT;
  down_read(&uts_sem);
- err=copy_to_user(name, &system_utsname, sizeof (*name));
+ err=copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  return err?-EFAULT:0;
 }
diff --git a/arch/sparc/kernel/sys_sparc.c b/arch/sparc/kernel/sys_sparc.c
index 0cdfc9d..c8ad73c 100644
--- a/arch/sparc/kernel/sys_sparc.c
+++ b/arch/sparc/kernel/sys_sparc.c
@@ -470,13 +470,13 @@ asmlinkage int sys_getdomainname(char __

   down_read(&uts_sem);
```

```
- nlen = strlen(system_utsname.domainname) + 1;
+ nlen = strlen(utsname()->domainname) + 1;

  if (nlen < len)
   len = nlen;
  if (len > __NEW_UTS_LEN)
   goto done;
- if (copy_to_user(name, system_utsname.domainname, len))
+ if (copy_to_user(name, utsname()->domainname, len))
   goto done;
  err = 0;
 done:
diff --git a/arch/sparc/kernel/sys_sunos.c b/arch/sparc/kernel/sys_sunos.c
index 288de27..9f9206f 100644
--- a/arch/sparc/kernel/sys_sunos.c
+++ b/arch/sparc/kernel/sys_sunos.c
@@ -483,13 +483,13 @@ asmlinkage int sunos_uname(struct sunos_
 {
  int ret;
  down_read(&uts_sem);
- ret = copy_to_user(&name->sname[0], &system_utsname.sysname[0], sizeof(name->sname) -
1);
+ ret = copy_to_user(&name->sname[0], &utsname()->sysname[0], sizeof(name->sname) - 1);
  if (!ret) {
-  ret |= __copy_to_user(&name->nname[0], &system_utsname.nodename[0],
sizeof(name->nname) - 1);
+  ret |= __copy_to_user(&name->nname[0], &utsname()->nodename[0], sizeof(name->nname) -
1);
   ret |= __put_user('\0', &name->nname[8]);
-  ret |= __copy_to_user(&name->rel[0], &system_utsname.release[0], sizeof(name->rel) - 1);
-  ret |= __copy_to_user(&name->ver[0], &system_utsname.version[0], sizeof(name->ver) - 1);
-  ret |= __copy_to_user(&name->mach[0], &system_utsname.machine[0], sizeof(name->mach) -
1);
+  ret |= __copy_to_user(&name->rel[0], &utsname()->release[0], sizeof(name->rel) - 1);
+  ret |= __copy_to_user(&name->ver[0], &utsname()->version[0], sizeof(name->ver) - 1);
+  ret |= __copy_to_user(&name->mach[0], &utsname()->machine[0], sizeof(name->mach) - 1);
  }
  up_read(&uts_sem);
  return ret ? -EFAULT : 0;
diff --git a/arch/sparc64/kernel/sys_sparc.c b/arch/sparc64/kernel/sys_sparc.c
index 7a86913..0453bd2 100644
--- a/arch/sparc64/kernel/sys_sparc.c
+++ b/arch/sparc64/kernel/sys_sparc.c
@@ -707,13 +707,13 @@ asmlinkage long sys_getdomainname(char _

   down_read(&uts_sem);
```

```
- nlen = strlen(system_utsname.domainname) + 1;
+ nlen = strlen(utsname()->domainname) + 1;

         if (nlen < len)
                 len = nlen;
  if (len > __NEW_UTS_LEN)
   goto done;
- if (copy_to_user(name, system_utsname.domainname, len))
+ if (copy_to_user(name, utsname()->domainname, len))
   goto done;
  err = 0;
 done:
diff --git a/arch/sparc64/kernel/sys_sunos32.c b/arch/sparc64/kernel/sys_sunos32.c
index ae5b32f..ba98c47 100644
--- a/arch/sparc64/kernel/sys_sunos32.c
+++ b/arch/sparc64/kernel/sys_sunos32.c
@@ -439,16 +439,16 @@ asmlinkage int sunos_uname(struct sunos_
  int ret;

  down_read(&uts_sem);
- ret = copy_to_user(&name->sname[0], &system_utsname.sysname[0],
+ ret = copy_to_user(&name->sname[0], &utsname()->sysname[0],
       sizeof(name->sname) - 1);
- ret |= copy_to_user(&name->nname[0], &system_utsname.nodename[0],
+ ret |= copy_to_user(&name->nname[0], &utsname()->nodename[0],
        sizeof(name->nname) - 1);
  ret |= put_user('\0', &name->nname[8]);
- ret |= copy_to_user(&name->rel[0], &system_utsname.release[0],
+ ret |= copy_to_user(&name->rel[0], &utsname()->release[0],
        sizeof(name->rel) - 1);
- ret |= copy_to_user(&name->ver[0], &system_utsname.version[0],
+ ret |= copy_to_user(&name->ver[0], &utsname()->version[0],
        sizeof(name->ver) - 1);
- ret |= copy_to_user(&name->mach[0], &system_utsname.machine[0],
+ ret |= copy_to_user(&name->mach[0], &utsname()->machine[0],
        sizeof(name->mach) - 1);
  up_read(&uts_sem);
  return (ret ? -EFAULT : 0);
diff --git a/arch/sparc64/solaris/misc.c b/arch/sparc64/solaris/misc.c
index 5284996..5d0162a 100644
--- a/arch/sparc64/solaris/misc.c
+++ b/arch/sparc64/solaris/misc.c
@@ -239,7 +239,7 @@ asmlinkage int solaris_utssys(u32 buf, u
   /* Let's cheat */
   err  = set_utsfield(v->sysname, "SunOS", 1, 0);
   down_read(&uts_sem);
-  err |= set_utsfield(v->nodename, system_utsname.nodename,
+  err |= set_utsfield(v->nodename, utsname()->nodename,
```

```
         1, 1);
   up_read(&uts_sem);
   err |= set_utsfield(v->release, "2.6", 0, 0);
@@ -263,7 +263,7 @@ asmlinkage int solaris_utsname(u32 buf)
  /* Why should we not lie a bit? */
  down_read(&uts_sem);
  err  = set_utsfield(v->sysname, "SunOS", 0, 0);
- err |= set_utsfield(v->nodename, system_utsname.nodename, 1, 1);
+ err |= set_utsfield(v->nodename, utsname()->nodename, 1, 1);
  err |= set_utsfield(v->release, "5.6", 0, 0);
  err |= set_utsfield(v->version, "Generic", 0, 0);
  err |= set_utsfield(v->machine, machine(), 0, 0);
@@ -295,7 +295,7 @@ asmlinkage int solaris_sysinfo(int cmd,
  case SI_HOSTNAME:
   r = buffer + 256;
   down_read(&uts_sem);
-  for (p = system_utsname.nodename, q = buffer;
+  for (p = utsname()->nodename, q = buffer;
        q < r && *p && *p != '.'; *q++ = *p++);
   up_read(&uts_sem);
   *q = 0;
diff --git a/arch/um/drivers/mconsole_kern.c b/arch/um/drivers/mconsole_kern.c
index 28e3760..5f87323 100644
--- a/arch/um/drivers/mconsole_kern.c
+++ b/arch/um/drivers/mconsole_kern.c
@@ -106,9 +106,9 @@ void mconsole_version(struct mc_request
 {
  char version[256];

- sprintf(version, "%s %s %s %s %s", system_utsname.sysname,
-  system_utsname.nodename, system_utsname.release,
-  system_utsname.version, system_utsname.machine);
+ sprintf(version, "%s %s %s %s %s", utsname()->sysname,
+  utsname()->nodename, utsname()->release,
+  utsname()->version, utsname()->machine);
  mconsole_reply(req, version, 0, 0);
 }

diff --git a/arch/um/kernel/syscall_kern.c b/arch/um/kernel/syscall_kern.c
index 37d3978..d90e9ed 100644
--- a/arch/um/kernel/syscall_kern.c
+++ b/arch/um/kernel/syscall_kern.c
@@ -110,7 +110,7 @@ long sys_uname(struct old_utsname __user
  if (!name)
   return -EFAULT;
  down_read(&uts_sem);
- err=copy_to_user(name, &system_utsname, sizeof (*name));
+ err=copy_to_user(name, utsname(), sizeof (*name));
```

```
  up_read(&uts_sem);
  return err?-EFAULT:0;
 }
@@ -126,19 +126,19 @@ long sys_olduname(struct oldold_utsname

    down_read(&uts_sem);

- error = __copy_to_user(&name->sysname,&system_utsname.sysname,
+ error = __copy_to_user(&name->sysname,&utsname()->sysname,
           __OLD_UTS_LEN);
  error |= __put_user(0,name->sysname+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->nodename,&system_utsname.nodename,
+ error |= __copy_to_user(&name->nodename,&utsname()->nodename,
     __OLD_UTS_LEN);
  error |= __put_user(0,name->nodename+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->release,&system_utsname.release,
+ error |= __copy_to_user(&name->release,&utsname()->release,
     __OLD_UTS_LEN);
  error |= __put_user(0,name->release+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->version,&system_utsname.version,
+ error |= __copy_to_user(&name->version,&utsname()->version,
     __OLD_UTS_LEN);
  error |= __put_user(0,name->version+__OLD_UTS_LEN);
- error |= __copy_to_user(&name->machine,&system_utsname.machine,
+ error |= __copy_to_user(&name->machine,&utsname()->machine,
     __OLD_UTS_LEN);
  error |= __put_user(0,name->machine+__OLD_UTS_LEN);

diff --git a/arch/um/sys-x86_64/syscalls.c b/arch/um/sys-x86_64/syscalls.c
index 6acee5c..3ad014e 100644
--- a/arch/um/sys-x86_64/syscalls.c
+++ b/arch/um/sys-x86_64/syscalls.c
@@ -21,7 +21,7 @@ asmlinkage long sys_uname64(struct new_u
 {
  int err;
  down_read(&uts_sem);
- err = copy_to_user(name, &system_utsname, sizeof (*name));
+ err = copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  if (personality(current->personality) == PER_LINUX32)
   err |= copy_to_user(&name->machine, "i686", 5);
diff --git a/arch/x86_64/ia32/sys_ia32.c b/arch/x86_64/ia32/sys_ia32.c
index f182b20..6e0a19d 100644
--- a/arch/x86_64/ia32/sys_ia32.c
+++ b/arch/x86_64/ia32/sys_ia32.c
@@ -801,13 +801,13 @@ asmlinkage long sys32_olduname(struct ol

    down_read(&uts_sem);
```

```
- error = __copy_to_user(&name->sysname,&system_utsname.sysname,__OLD_UTS_LEN);
+ error = __copy_to_user(&name->sysname,&utsname()->sysname,__OLD_UTS_LEN);
   __put_user(0,name->sysname+__OLD_UTS_LEN);
-  __copy_to_user(&name->nodename,&system_utsname.nodename,__OLD_UTS_LEN);
+  __copy_to_user(&name->nodename,&utsname()->nodename,__OLD_UTS_LEN);
   __put_user(0,name->nodename+__OLD_UTS_LEN);
-  __copy_to_user(&name->release,&system_utsname.release,__OLD_UTS_LEN);
+  __copy_to_user(&name->release,&utsname()->release,__OLD_UTS_LEN);
   __put_user(0,name->release+__OLD_UTS_LEN);
-  __copy_to_user(&name->version,&system_utsname.version,__OLD_UTS_LEN);
+  __copy_to_user(&name->version,&utsname()->version,__OLD_UTS_LEN);
   __put_user(0,name->version+__OLD_UTS_LEN);
   {
    char *arch = "x86_64";
@@ -830,7 +830,7 @@ long sys32_uname(struct old_utsname __us
  if (!name)
   return -EFAULT;
  down_read(&uts_sem);
- err=copy_to_user(name, &system_utsname, sizeof (*name));
+ err=copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  if (personality(current->personality) == PER_LINUX32)
   err |= copy_to_user(&name->machine, "i686", 5);
diff --git a/arch/x86_64/kernel/sys_x86_64.c b/arch/x86_64/kernel/sys_x86_64.c
index 6449ea8..76bf7c2 100644
--- a/arch/x86_64/kernel/sys_x86_64.c
+++ b/arch/x86_64/kernel/sys_x86_64.c
@@ -148,7 +148,7 @@ asmlinkage long sys_uname(struct new_uts
 {
  int err;
  down_read(&uts_sem);
- err = copy_to_user(name, &system_utsname, sizeof (*name));
+ err = copy_to_user(name, utsname(), sizeof (*name));
  up_read(&uts_sem);
  if (personality(current->personality) == PER_LINUX32)
   err |= copy_to_user(&name->machine, "i686", 5);
diff --git a/arch/xtensa/kernel/syscalls.c b/arch/xtensa/kernel/syscalls.c
index f20c649..30060c1 100644
--- a/arch/xtensa/kernel/syscalls.c
+++ b/arch/xtensa/kernel/syscalls.c
@@ -129,7 +129,7 @@ out:

 int sys_uname(struct old_utsname * name)
 {
- if (name && !copy_to_user(name, &system_utsname, sizeof (*name)))
+ if (name && !copy_to_user(name, utsname(), sizeof (*name)))
   return 0;
```

```
  return -EFAULT;
 }
diff --git a/drivers/char/random.c b/drivers/char/random.c
index 86be04b..ec4c11d 100644
--- a/drivers/char/random.c
+++ b/drivers/char/random.c
@@ -888,8 +888,8 @@ static void init_std_data(struct entropy

  do_gettimeofday(&tv);
  add_entropy_words(r, (__u32 *)&tv, sizeof(tv)/4);
- add_entropy_words(r, (__u32 *)&system_utsname,
-     sizeof(system_utsname)/4);
+ add_entropy_words(r, (__u32 *)utsname(),
+     sizeof(*(utsname()))/4);
 }

 static int __init rand_initialize(void)
diff --git a/fs/cifs/connect.c b/fs/cifs/connect.c
index 0b86d5c..852ff41 100644
--- a/fs/cifs/connect.c
+++ b/fs/cifs/connect.c
@@ -765,12 +765,12 @@ cifs_parse_mount_options(char *options,
  separator[1] = 0;

  memset(vol->source_rfc1001_name,0x20,15);
- for(i=0;i < strnlen(system_utsname.nodename,15);i++) {
+ for(i=0;i < strnlen(utsname()->nodename,15);i++) {
   /* does not have to be a perfect mapping since the field is
   informational, only used for servers that do not support
   port 445 and it can be overridden at mount time */
   vol->source_rfc1001_name[i] =
-   toupper(system_utsname.nodename[i]);
+   toupper(utsname()->nodename[i]);
  }
  vol->source_rfc1001_name[15] = 0;
  /* null target name indicates to use *SMBSERVR default called name
@@ -2077,7 +2077,7 @@ CIFSSessSetup(unsigned int xid, struct c
       32, nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bytes_returned =
-      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release,
+      cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release,
       32, nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bcc_ptr += 2;
@@ -2104,8 +2104,8 @@ CIFSSessSetup(unsigned int xid, struct c
   }
   strcpy(bcc_ptr, "Linux version ");
```

```
   bcc_ptr += strlen("Linux version ");
-  strcpy(bcc_ptr, system_utsname.release);
-  bcc_ptr += strlen(system_utsname.release) + 1;
+  strcpy(bcc_ptr, utsname()->release);
+  bcc_ptr += strlen(utsname()->release) + 1;
   strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
   bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
  }
@@ -2346,7 +2346,7 @@ CIFSSpnegoSessSetup(unsigned int xid, st
       32, nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bytes_returned =
-      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release, 32,
+       cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release, 32,
       nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bcc_ptr += 2;
@@ -2371,8 +2371,8 @@ CIFSSpnegoSessSetup(unsigned int xid, st
   }
   strcpy(bcc_ptr, "Linux version ");
   bcc_ptr += strlen("Linux version ");
-  strcpy(bcc_ptr, system_utsname.release);
-  bcc_ptr += strlen(system_utsname.release) + 1;
+  strcpy(bcc_ptr, utsname()->release);
+  bcc_ptr += strlen(utsname()->release) + 1;
   strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
   bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
  }
@@ -2622,7 +2622,7 @@ CIFSNTLMSSPNegotiateSessSetup(unsigned i
       32, nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bytes_returned =
-      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release, 32,
+       cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release, 32,
       nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bcc_ptr += 2; /* null terminate Linux version */
@@ -2639,8 +2639,8 @@ CIFSNTLMSSPNegotiateSessSetup(unsigned i
  } else {  /* ASCII */
   strcpy(bcc_ptr, "Linux version ");
   bcc_ptr += strlen("Linux version ");
-  strcpy(bcc_ptr, system_utsname.release);
-  bcc_ptr += strlen(system_utsname.release) + 1;
+  strcpy(bcc_ptr, utsname()->release);
+  bcc_ptr += strlen(utsname()->release) + 1;
   strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
   bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
   bcc_ptr++; /* empty domain field */
```

```
@@ -3001,7 +3001,7 @@ CIFSNTLMSSPAuthSessSetup(unsigned int xi
       32, nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bytes_returned =
-      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release, 32,
+       cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release, 32,
       nls_codepage);
   bcc_ptr += 2 * bytes_returned;
   bcc_ptr += 2; /* null term version string */
@@ -3053,8 +3053,8 @@ CIFSNTLMSSPAuthSessSetup(unsigned int xi

   strcpy(bcc_ptr, "Linux version ");
   bcc_ptr += strlen("Linux version ");
-  strcpy(bcc_ptr, system_utsname.release);
-  bcc_ptr += strlen(system_utsname.release) + 1;
+  strcpy(bcc_ptr, utsname()->release);
+  bcc_ptr += strlen(utsname()->release) + 1;
   strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
   bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
   bcc_ptr++; /* null domain */
diff --git a/fs/exec.c b/fs/exec.c
index 0291a68..d881479 100644
--- a/fs/exec.c
+++ b/fs/exec.c
@@ -1347,7 +1347,7 @@ static void format_corename(char *corena
    case 'h':
     down_read(&uts_sem);
     rc = snprintf(out_ptr, out_end - out_ptr,
-           "%s", system_utsname.nodename);
+           "%s", utsname()->nodename);
     up_read(&uts_sem);
     if (rc > out_end - out_ptr)
      goto out;
diff --git a/fs/lockd/clntproc.c b/fs/lockd/clntproc.c
index f96e381..915e596 100644
--- a/fs/lockd/clntproc.c
+++ b/fs/lockd/clntproc.c
@@ -130,11 +130,11 @@ static void nlmclnt_setlockargs(struct n
  nlmclnt_next_cookie(&argp->cookie);
  argp->state   = nsm_local_state;
  memcpy(&lock->fh, NFS_FH(fl->fl_file->f_dentry->d_inode), sizeof(struct nfs_fh));
- lock->caller  = system_utsname.nodename;
+ lock->caller  = utsname()->nodename;
  lock->oh.data = req->a_owner;
  lock->oh.len  = snprintf(req->a_owner, sizeof(req->a_owner), "%u@%s",
     (unsigned int)fl->fl_u.nfs_fl.owner->pid,
-    system_utsname.nodename);
+    utsname()->nodename);
```

```
  lock->svid = fl->fl_u.nfs_fl.owner->pid;
  lock->fl.fl_start = fl->fl_start;
  lock->fl.fl_end = fl->fl_end;
diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
index 3fc683f..547aaa3 100644
--- a/fs/lockd/mon.c
+++ b/fs/lockd/mon.c
@@ -152,7 +152,7 @@ xdr_encode_common(struct rpc_rqst *rqstp
   */
  sprintf(buffer, "%u.%u.%u.%u", NIPQUAD(argp->addr));
  if (!(p = xdr_encode_string(p, buffer))
-  || !(p = xdr_encode_string(p, system_utsname.nodename)))
+  || !(p = xdr_encode_string(p, utsname()->nodename)))
   return ERR_PTR(-EIO);
  *p++ = htonl(argp->prog);
  *p++ = htonl(argp->vers);
diff --git a/fs/lockd/svclock.c b/fs/lockd/svclock.c
index d2b66ba..61b4791 100644
--- a/fs/lockd/svclock.c
+++ b/fs/lockd/svclock.c
@@ -326,7 +326,7 @@ static int nlmsvc_setgrantargs(struct nl
 {
  locks_copy_lock(&call->a_args.lock.fl, &lock->fl);
  memcpy(&call->a_args.lock.fh, &lock->fh, sizeof(call->a_args.lock.fh));
- call->a_args.lock.caller = system_utsname.nodename;
+ call->a_args.lock.caller = utsname()->nodename;
  call->a_args.lock.oh.len = lock->oh.len;

  /* set default data area */
diff --git a/fs/lockd/xdr.c b/fs/lockd/xdr.c
index f22a376..4eec051 100644
--- a/fs/lockd/xdr.c
+++ b/fs/lockd/xdr.c
@@ -516,7 +516,7 @@ nlmclt_decode_res(struct rpc_rqst *req,
  */
 #define NLM_void_sz  0
 #define NLM_cookie_sz  1+XDR_QUADLEN(NLM_MAXCOOKIELEN)
-#define NLM_caller_sz  1+XDR_QUADLEN(sizeof(system_utsname.nodename))
+#define NLM_caller_sz  1+XDR_QUADLEN(sizeof(utsname()->nodename))
 #define NLM_netobj_sz  1+XDR_QUADLEN(XDR_MAX_NETOBJ)
 /* #define NLM_owner_sz  1+XDR_QUADLEN(NLM_MAXOWNER) */
 #define NLM_fhandle_sz  1+XDR_QUADLEN(NFS2_FHSIZE)
diff --git a/fs/nfs/nfsroot.c b/fs/nfs/nfsroot.c
index c0a754e..1d656a6 100644
--- a/fs/nfs/nfsroot.c
+++ b/fs/nfs/nfsroot.c
@@ -312,7 +312,7 @@ static int __init root_nfs_name(char *na
  /* Override them by options set on kernel command-line */
```

```
  root_nfs_parse(name, buf);

- cp = system_utsname.nodename;
+ cp = utsname()->nodename;
  if (strlen(buf) + strlen(cp) > NFS_MAXPATHLEN) {
   printk(KERN_ERR "Root-NFS: Pathname for remote directory too long.\n");
   return -1;
diff --git a/include/linux/lockd/lockd.h b/include/linux/lockd/lockd.h
index 995f89d..ac15b87 100644
--- a/include/linux/lockd/lockd.h
+++ b/include/linux/lockd/lockd.h
@@ -80,7 +80,7 @@ struct nlm_wait;
 /*
  * Memory chunk for NLM client RPC request.
  */
-#define NLMCLNT_OHSIZE  (sizeof(system_utsname.nodename)+10)
+#define NLMCLNT_OHSIZE  (sizeof(utsname()->nodename)+10)
 struct nlm_rqst {
  unsigned int  a_flags; /* initial RPC task flags */
  struct nlm_host * a_host;  /* host handle */
diff --git a/kernel/sys.c b/kernel/sys.c
index 0b6ec0e..bcaa48e 100644
--- a/kernel/sys.c
+++ b/kernel/sys.c
@@ -1671,7 +1671,7 @@ asmlinkage long sys_newuname(struct new_
  int errno = 0;

  down_read(&uts_sem);
- if (copy_to_user(name,&system_utsname,sizeof *name))
+ if (copy_to_user(name,utsname(),sizeof *name))
   errno = -EFAULT;
  up_read(&uts_sem);
  return errno;
@@ -1689,8 +1689,8 @@ asmlinkage long sys_sethostname(char __u
  down_write(&uts_sem);
  errno = -EFAULT;
  if (!copy_from_user(tmp, name, len)) {
-  memcpy(system_utsname.nodename, tmp, len);
-  system_utsname.nodename[len] = 0;
+  memcpy(utsname()->nodename, tmp, len);
+  utsname()->nodename[len] = 0;
   errno = 0;
  }
  up_write(&uts_sem);
@@ -1706,11 +1706,11 @@ asmlinkage long sys_gethostname(char __u
  if (len < 0)
   return -EINVAL;
  down_read(&uts_sem);
```

```
- i = 1 + strlen(system_utsname.nodename);
+ i = 1 + strlen(utsname()->nodename);
  if (i > len)
   i = len;
  errno = 0;
- if (copy_to_user(name, system_utsname.nodename, i))
+ if (copy_to_user(name, utsname()->nodename, i))
   errno = -EFAULT;
  up_read(&uts_sem);
  return errno;
@@ -1735,8 +1735,8 @@ asmlinkage long sys_setdomainname(char _
  down_write(&uts_sem);
  errno = -EFAULT;
  if (!copy_from_user(tmp, name, len)) {
-  memcpy(system_utsname.domainname, tmp, len);
-  system_utsname.domainname[len] = 0;
+  memcpy(utsname()->domainname, tmp, len);
+  utsname()->domainname[len] = 0;
   errno = 0;
  }
  up_write(&uts_sem);
--
1.2.4
```

Subject: [RFC][PATCH 3/5] uts namespaces: Use init uts_namespace when appropriate

Posted by serue on Fri, 07 Apr 2006 18:36:00 GMT

View Forum Message <> Reply to Message

In some places, particularly drivers and __init code, the init uts namespace is the appropriate one to use. This patch replaces those with a direct reference to init_uts_ns.name. Note that we can drop this patch and simply
do #define system_utsname (init_uts_ns.name)
however by using this patch we make explicit, for the sake of review, those places where we do and do not use the utsname namespace.

Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>

```
---
 arch/arm/kernel/setup.c                |    2 +-
 arch/arm26/kernel/setup.c              |    2 +-
 arch/cris/kernel/setup.c               |    2 +-
 arch/i386/kernel/process.c             |    6 +++---
 arch/i386/kernel/traps.c               |    6 +++---
 arch/powerpc/kernel/process.c          |    2 +-
 arch/powerpc/kernel/setup_64.c         |    2 +-
 arch/powerpc/platforms/pseries/setup.c |    2 +-
 arch/sh/kernel/setup.c                 |    2 +-
```

```
 arch/um/kernel/um_arch.c                |    6 +++---
 arch/um/sys-x86_64/sysrq.c              |    2 +-
 arch/x86_64/kernel/process.c            |    6 +++---
 drivers/infiniband/hw/ipath/ipath_verbs.c |    2 +-
 drivers/parisc/led.c                    |    2 +-
 drivers/scsi/lpfc/lpfc_ct.c             |    8 ++++----
 drivers/usb/core/hcd.c                  |    4 ++--
 drivers/usb/gadget/ether.c              |    2 +-
 drivers/usb/gadget/file_storage.c       |    2 +-
 drivers/usb/gadget/serial.c             |    2 +-
 drivers/usb/gadget/zero.c               |    2 +-
 include/asm-i386/bugs.h                 |    2 +-
 include/asm-i386/elf.h                  |    2 +-
 include/asm-sh/bugs.h                   |    2 +-
 kernel/power/snapshot.c                 |   10 +++++-----
 net/ipv4/ipconfig.c                     |   16 ++++++++--------
 net/sunrpc/clnt.c                       |    4 ++--
 sound/core/info_oss.c                   |   10 +++++-----
 27 files changed, 55 insertions(+), 55 deletions(-)

ef54ab30a75ae83207b385090d3f1ff6c912f0d5
diff --git a/arch/arm/kernel/setup.c b/arch/arm/kernel/setup.c
index 4375284..647c516 100644
--- a/arch/arm/kernel/setup.c
+++ b/arch/arm/kernel/setup.c
@@ -319,7 +319,7 @@ static void __init setup_processor(void)
        cpu_name, processor_id, (int)processor_id & 15,
        proc_arch[cpu_architecture()]);

- sprintf(system_utsname.machine, "%s%c", list->arch_name, ENDIANNESS);
+ sprintf(init_uts_ns.name.machine, "%s%c", list->arch_name, ENDIANNESS);
  sprintf(elf_platform, "%s%c", list->elf_name, ENDIANNESS);
  elf_hwcap = list->elf_hwcap;

diff --git a/arch/arm26/kernel/setup.c b/arch/arm26/kernel/setup.c
index 4eb329e..6564e73 100644
--- a/arch/arm26/kernel/setup.c
+++ b/arch/arm26/kernel/setup.c
@@ -144,7 +144,7 @@ static void __init setup_processor(void)

  dump_cpu_info();

- sprintf(system_utsname.machine, "%s", list->arch_name);
+ sprintf(init_uts_ns.name.machine, "%s", list->arch_name);
  sprintf(elf_platform, "%s", list->elf_name);
  elf_hwcap = list->elf_hwcap;

diff --git a/arch/cris/kernel/setup.c b/arch/cris/kernel/setup.c
```

```
index 619a6ee..f9a29a4 100644
--- a/arch/cris/kernel/setup.c
+++ b/arch/cris/kernel/setup.c
@@ -161,7 +161,7 @@ setup_arch(char **cmdline_p)
  show_etrax_copyright();

  /* Setup utsname */
- strcpy(system_utsname.machine, cris_machine_name);
+ strcpy(init_uts_ns.name.machine, cris_machine_name);
 }

 static void *c_start(struct seq_file *m, loff_t *pos)
diff --git a/arch/i386/kernel/process.c b/arch/i386/kernel/process.c
index 6259afe..89fac4c 100644
--- a/arch/i386/kernel/process.c
+++ b/arch/i386/kernel/process.c
@@ -297,9 +297,9 @@ void show_regs(struct pt_regs * regs)
  if (user_mode_vm(regs))
   printk(" ESP: %04x:%08lx",0xffff & regs->xss,regs->esp);
  printk(" EFLAGS: %08lx    %s  (%s %.*s)\n",
-        regs->eflags, print_tainted(), system_utsname.release,
-        (int)strcspn(system_utsname.version, " "),
-        system_utsname.version);
+        regs->eflags, print_tainted(), init_uts_ns.name.release,
+        (int)strcspn(init_uts_ns.name.version, " "),
+        init_uts_ns.name.version);
  printk("EAX: %08lx EBX: %08lx ECX: %08lx EDX: %08lx\n",
   regs->eax,regs->ebx,regs->ecx,regs->edx);
  printk("ESI: %08lx EDI: %08lx EBP: %08lx",
diff --git a/arch/i386/kernel/traps.c b/arch/i386/kernel/traps.c
index e385279..addff65 100644
--- a/arch/i386/kernel/traps.c
+++ b/arch/i386/kernel/traps.c
@@ -260,9 +260,9 @@ void show_registers(struct pt_regs *regs
  printk(KERN_EMERG "CPU:    %d\nEIP:    %04x:[<%08lx>]    %s VLI\n"
    "EFLAGS: %08lx   (%s %.*s) \n",
   smp_processor_id(), 0xffff & regs->xcs, regs->eip,
-  print_tainted(), regs->eflags, system_utsname.release,
-  (int)strcspn(system_utsname.version, " "),
-  system_utsname.version);
+  print_tainted(), regs->eflags, init_uts_ns.name.release,
+  (int)strcspn(init_uts_ns.name.version, " "),
+  init_uts_ns.name.version);
  print_symbol(KERN_EMERG "EIP is at %s\n", regs->eip);
  printk(KERN_EMERG "eax: %08lx   ebx: %08lx   ecx: %08lx   edx: %08lx\n",
   regs->eax, regs->ebx, regs->ecx, regs->edx);
diff --git a/arch/powerpc/kernel/process.c b/arch/powerpc/kernel/process.c
index 2dd47d2..7c21450 100644
```

```
--- a/arch/powerpc/kernel/process.c
+++ b/arch/powerpc/kernel/process.c
@@ -425,7 +425,7 @@ void show_regs(struct pt_regs * regs)
  printk("NIP: "REG" LR: "REG" CTR: "REG"\n",
         regs->nip, regs->link, regs->ctr);
  printk("REGS: %p TRAP: %04lx   %s  (%s)\n",
-        regs, regs->trap, print_tainted(), system_utsname.release);
+        regs, regs->trap, print_tainted(), init_uts_ns.name.release);
  printk("MSR: "REG" ", regs->msr);
  printbits(regs->msr, msr_bits);
  printk("  CR: %08lX  XER: %08lX\n", regs->ccr, regs->xer);
diff --git a/arch/powerpc/kernel/setup_64.c b/arch/powerpc/kernel/setup_64.c
index 13e91c4..26f0477 100644
--- a/arch/powerpc/kernel/setup_64.c
+++ b/arch/powerpc/kernel/setup_64.c
@@ -435,7 +435,7 @@ void __init setup_system(void)
  smp_release_cpus();
 #endif

- printk("Starting Linux PPC64 %s\n", system_utsname.version);
+ printk("Starting Linux PPC64 %s\n", init_uts_ns.name.version);

  printk("-----------------------------------------------------\n ");
  printk("ppc64_pft_size                = 0x%lx\n", ppc64_pft_size);
diff --git a/arch/powerpc/platforms/pseries/setup.c b/arch/powerpc/platforms/pseries/setup.c
index 5eb55ef..9df0d0e 100644
--- a/arch/powerpc/platforms/pseries/setup.c
+++ b/arch/powerpc/platforms/pseries/setup.c
@@ -255,7 +255,7 @@ static int __init pSeries_init_panel(voi
 {
  /* Manually leave the kernel version on the panel. */
  ppc_md.progress("Linux ppc64\n", 0);
- ppc_md.progress(system_utsname.version, 0);
+ ppc_md.progress(init_uts_ns.name.version, 0);

  return 0;
 }
diff --git a/arch/sh/kernel/setup.c b/arch/sh/kernel/setup.c
index bb229ef..fab811b 100644
--- a/arch/sh/kernel/setup.c
+++ b/arch/sh/kernel/setup.c
@@ -481,7 +481,7 @@ static int show_cpuinfo(struct seq_file
   seq_printf(m, "machine\t\t: %s\n", get_system_type());

  seq_printf(m, "processor\t: %d\n", cpu);
- seq_printf(m, "cpu family\t: %s\n", system_utsname.machine);
+ seq_printf(m, "cpu family\t: %s\n", init_uts_ns.name.machine);
  seq_printf(m, "cpu type\t: %s\n", get_cpu_subtype());
```

```
  show_cpuflags(m);
diff --git a/arch/um/kernel/um_arch.c b/arch/um/kernel/um_arch.c
index 7d51dd7..4caad31 100644
--- a/arch/um/kernel/um_arch.c
+++ b/arch/um/kernel/um_arch.c
@@ -167,7 +167,7 @@ static char *usage_string =

 static int __init uml_version_setup(char *line, int *add)
 {
- printf("%s\n", system_utsname.release);
+ printf("%s\n", init_uts_ns.name.release);
  exit(0);

  return 0;
@@ -278,7 +278,7 @@ static int __init Usage(char *line, int
 {
   const char **p;

- printf(usage_string, system_utsname.release);
+ printf(usage_string, init_uts_ns.name.release);
   p = &__uml_help_start;
   while (p < &__uml_help_end) {
    printf("%s", *p);
@@ -400,7 +400,7 @@ int linux_main(int argc, char **argv)
  /* Reserve up to 4M after the current brk */
  uml_reserved = ROUND_4M(brk_start) + (1 << 22);

- setup_machinename(system_utsname.machine);
+ setup_machinename(init_uts_ns.name.machine);

 #ifdef CONFIG_CMDLINE_ON_HOST
  argv1_begin = argv[1];
diff --git a/arch/um/sys-x86_64/sysrq.c b/arch/um/sys-x86_64/sysrq.c
index d0a25af..49a549a 100644
--- a/arch/um/sys-x86_64/sysrq.c
+++ b/arch/um/sys-x86_64/sysrq.c
@@ -16,7 +16,7 @@ void __show_regs(struct pt_regs * regs)
  printk("\n");
  print_modules();
  printk("Pid: %d, comm: %.20s %s %s\n",
-        current->pid, current->comm, print_tainted(), system_utsname.release);
+        current->pid, current->comm, print_tainted(), init_uts_ns.name.release);
  printk("RIP: %04lx:[<%016lx>] ", PT_REGS_CS(regs) & 0xffff,
         PT_REGS_RIP(regs));
  printk("\nRSP: %016lx  EFLAGS: %08lx\n", PT_REGS_RSP(regs),
diff --git a/arch/x86_64/kernel/process.c b/arch/x86_64/kernel/process.c
index 70dd8e5..f79a080 100644
```

```
--- a/arch/x86_64/kernel/process.c
+++ b/arch/x86_64/kernel/process.c
@@ -292,9 +292,9 @@ void __show_regs(struct pt_regs * regs)
  print_modules();
  printk("Pid: %d, comm: %.20s %s %s %.*s\n",
   current->pid, current->comm, print_tainted(),
-  system_utsname.release,
-  (int)strcspn(system_utsname.version, " "),
-  system_utsname.version);
+  init_uts_ns.name.release,
+  (int)strcspn(init_uts_ns.name.version, " "),
+  init_uts_ns.name.version);
  printk("RIP: %04lx:[<%016lx>] ", regs->cs & 0xffff, regs->rip);
  printk_address(regs->rip);
  printk("\nRSP: %04lx:%016lx  EFLAGS: %08lx\n", regs->ss, regs->rsp,
diff --git a/drivers/infiniband/hw/ipath/ipath_verbs.c b/drivers/infiniband/hw/ipath/ipath_verbs.c
index 9f27fd3..d5479a8 100644
--- a/drivers/infiniband/hw/ipath/ipath_verbs.c
+++ b/drivers/infiniband/hw/ipath/ipath_verbs.c
@@ -1048,7 +1048,7 @@ static void *ipath_register_ib_device(in
  dev->process_mad = ipath_process_mad;

  snprintf(dev->node_desc, sizeof(dev->node_desc),
-   IPATH_IDSTR " %s kernel_SMA", system_utsname.nodename);
+   IPATH_IDSTR " %s kernel_SMA", init_uts_ns.name.nodename);

  ret = ib_register_device(dev);
  if (ret)
diff --git a/drivers/parisc/led.c b/drivers/parisc/led.c
index 298f2dd..c05e169 100644
--- a/drivers/parisc/led.c
+++ b/drivers/parisc/led.c
@@ -684,7 +684,7 @@ int __init led_init(void)
  int ret;

  snprintf(lcd_text_default, sizeof(lcd_text_default),
-  "Linux %s", system_utsname.release);
+  "Linux %s", init_uts_ns.name.release);

  /* Work around the buggy PDC of KittyHawk-machines */
  switch (CPU_HVERSION) {
diff --git a/drivers/scsi/lpfc/lpfc_ct.c b/drivers/scsi/lpfc/lpfc_ct.c
index b65ee57..ef05e16 100644
--- a/drivers/scsi/lpfc/lpfc_ct.c
+++ b/drivers/scsi/lpfc/lpfc_ct.c
@@ -961,8 +961,8 @@ lpfc_fdmi_cmd(struct lpfc_hba * phba, st
    ae = (ATTRIBUTE_ENTRY *) ((uint8_t *) rh + size);
    ae->ad.bits.AttrType = be16_to_cpu(OS_NAME_VERSION);
```

```
    sprintf(ae->un.OsNameVersion, "%s %s %s",
-    system_utsname.sysname, system_utsname.release,
-    system_utsname.version);
+    init_uts_ns.name.sysname, init_uts_ns.name.release,
+    init_uts_ns.name.version);
    len = strlen(ae->un.OsNameVersion);
    len += (len & 3) ? (4 - (len & 3)) : 4;
    ae->ad.bits.AttrLen = be16_to_cpu(FOURBYTES + len);
@@ -1080,7 +1080,7 @@ lpfc_fdmi_cmd(struct lpfc_hba * phba, st
         size);
     ae->ad.bits.AttrType = be16_to_cpu(HOST_NAME);
     sprintf(ae->un.HostName, "%s",
-     system_utsname.nodename);
+     init_uts_ns.name.nodename);
     len = strlen(ae->un.HostName);
     len += (len & 3) ? (4 - (len & 3)) : 4;
     ae->ad.bits.AttrLen =
@@ -1168,7 +1168,7 @@ lpfc_fdmi_tmo_handler(struct lpfc_hba *p

  ndlp = lpfc_findnode_did(phba, NLP_SEARCH_ALL, FDMI_DID);
  if (ndlp) {
-  if (system_utsname.nodename[0] != '\0') {
+  if (init_uts_ns.name.nodename[0] != '\0') {
    lpfc_fdmi_cmd(phba, ndlp, SLI_MGMT_DHBA);
   } else {
    mod_timer(&phba->fc_fdmitmo, jiffies + HZ * 60);
diff --git a/drivers/usb/core/hcd.c b/drivers/usb/core/hcd.c
index fbd938d..b979b16 100644
--- a/drivers/usb/core/hcd.c
+++ b/drivers/usb/core/hcd.c
@@ -318,8 +318,8 @@ static int rh_string (

   // id 3 == vendor description
  } else if (id == 3) {
-  snprintf (buf, sizeof buf, "%s %s %s", system_utsname.sysname,
-   system_utsname.release, hcd->driver->description);
+  snprintf (buf, sizeof buf, "%s %s %s", init_uts_ns.name.sysname,
+   init_uts_ns.name.release, hcd->driver->description);

  // unsupported IDs --> "protocol stall"
  } else
diff --git a/drivers/usb/gadget/ether.c b/drivers/usb/gadget/ether.c
index c3d8e5c..e49359d 100644
--- a/drivers/usb/gadget/ether.c
+++ b/drivers/usb/gadget/ether.c
@@ -2242,7 +2242,7 @@ eth_bind (struct usb_gadget *gadget)
   return -ENODEV;
  }
```

```
  snprintf (manufacturer, sizeof manufacturer, "%s %s/%s",
-  system_utsname.sysname, system_utsname.release,
+  init_uts_ns.name.sysname, init_uts_ns.name.release,
   gadget->name);

  /* If there's an RNDIS configuration, that's what Windows wants to
diff --git a/drivers/usb/gadget/file_storage.c b/drivers/usb/gadget/file_storage.c
index cf3be29..81ffe03 100644
--- a/drivers/usb/gadget/file_storage.c
+++ b/drivers/usb/gadget/file_storage.c
@@ -3965,7 +3965,7 @@ static int __init fsg_bind(struct usb_ga
  usb_gadget_set_selfpowered(gadget);

  snprintf(manufacturer, sizeof manufacturer, "%s %s with %s",
-   system_utsname.sysname, system_utsname.release,
+   init_uts_ns.name.sysname, init_uts_ns.name.release,
    gadget->name);

  /* On a real device, serial[] would be loaded from permanent
diff --git a/drivers/usb/gadget/serial.c b/drivers/usb/gadget/serial.c
index b992546..1063159 100644
--- a/drivers/usb/gadget/serial.c
+++ b/drivers/usb/gadget/serial.c
@@ -1496,7 +1496,7 @@ static int __init gs_bind(struct usb_gad
   return -ENOMEM;

  snprintf(manufacturer, sizeof(manufacturer), "%s %s with %s",
-  system_utsname.sysname, system_utsname.release,
+  init_uts_ns.name.sysname, init_uts_ns.name.release,
   gadget->name);

  memset(dev, 0, sizeof(struct gs_dev));
diff --git a/drivers/usb/gadget/zero.c b/drivers/usb/gadget/zero.c
index 51424f6..5aa1bd4 100644
--- a/drivers/usb/gadget/zero.c
+++ b/drivers/usb/gadget/zero.c
@@ -1240,7 +1240,7 @@ autoconf_fail:
   EP_OUT_NAME, EP_IN_NAME);

  snprintf (manufacturer, sizeof manufacturer, "%s %s with %s",
-  system_utsname.sysname, system_utsname.release,
+  init_uts_ns.name.sysname, init_uts_ns.name.release,
   gadget->name);

  return 0;
diff --git a/include/asm-i386/bugs.h b/include/asm-i386/bugs.h
index 50233e0..ce0386e 100644
--- a/include/asm-i386/bugs.h
```

```
+++ b/include/asm-i386/bugs.h
@@ -190,6 +190,6 @@ static void __init check_bugs(void)
  check_fpu();
  check_hlt();
  check_popad();
- system_utsname.machine[1] = '0' + (boot_cpu_data.x86 > 6 ? 6 : boot_cpu_data.x86);
+ init_uts_ns.name.machine[1] = '0' + (boot_cpu_data.x86 > 6 ? 6 : boot_cpu_data.x86);
  alternative_instructions();
 }
diff --git a/include/asm-i386/elf.h b/include/asm-i386/elf.h
index 4153d80..53c2829 100644
--- a/include/asm-i386/elf.h
+++ b/include/asm-i386/elf.h
@@ -108,7 +108,7 @@ typedef struct user_fxsr_struct elf_fpxr
    For the moment, we have only optimizations for the Intel generations,
    but that could change... */

-#define ELF_PLATFORM  (system_utsname.machine)
+#define ELF_PLATFORM  (init_uts_ns.name.machine)

 #ifdef __KERNEL__
 #define SET_PERSONALITY(ex, ibcs2) do { } while (0)
diff --git a/include/asm-sh/bugs.h b/include/asm-sh/bugs.h
index a6de3d0..0a1648d 100644
--- a/include/asm-sh/bugs.h
+++ b/include/asm-sh/bugs.h
@@ -18,7 +18,7 @@ static void __init check_bugs(void)
 {
  extern char *get_cpu_subtype(void);
  extern unsigned long loops_per_jiffy;
- char *p= &system_utsname.machine[2]; /* "sh" */
+ char *p= &init_uts_ns.name.machine[2]; /* "sh" */

  cpu_data->loops_per_jiffy = loops_per_jiffy;

diff --git a/kernel/power/snapshot.c b/kernel/power/snapshot.c
index c5863d0..14b3f24 100644
--- a/kernel/power/snapshot.c
+++ b/kernel/power/snapshot.c
@@ -523,7 +523,7 @@ static void init_header(struct swsusp_in
  memset(info, 0, sizeof(struct swsusp_info));
  info->version_code = LINUX_VERSION_CODE;
  info->num_physpages = num_physpages;
- memcpy(&info->uts, &system_utsname, sizeof(system_utsname));
+ memcpy(&info->uts, &init_uts_ns.name, sizeof(init_uts_ns.name));
  info->cpus = num_online_cpus();
  info->image_pages = nr_copy_pages;
  info->pages = nr_copy_pages + nr_meta_pages + 1;
```

```
@@ -662,13 +662,13 @@ static int check_header(struct swsusp_in
   reason = "kernel version";
  if (info->num_physpages != num_physpages)
   reason = "memory size";
- if (strcmp(info->uts.sysname,system_utsname.sysname))
+ if (strcmp(info->uts.sysname,init_uts_ns.name.sysname))
   reason = "system type";
- if (strcmp(info->uts.release,system_utsname.release))
+ if (strcmp(info->uts.release,init_uts_ns.name.release))
   reason = "kernel release";
- if (strcmp(info->uts.version,system_utsname.version))
+ if (strcmp(info->uts.version,init_uts_ns.name.version))
   reason = "version";
- if (strcmp(info->uts.machine,system_utsname.machine))
+ if (strcmp(info->uts.machine,init_uts_ns.name.machine))
   reason = "machine";
  if (reason) {
   printk(KERN_ERR "swsusp: Resume mismatch: %s\n", reason);
diff --git a/net/ipv4/ipconfig.c b/net/ipv4/ipconfig.c
index cb8a92f..b00635f 100644
--- a/net/ipv4/ipconfig.c
+++ b/net/ipv4/ipconfig.c
@@ -367,7 +367,7 @@ static int __init ic_defaults(void)
   */

  if (!ic_host_name_set)
-  sprintf(system_utsname.nodename, "%u.%u.%u.%u", NIPQUAD(ic_myaddr));
+  sprintf(init_uts_ns.name.nodename, "%u.%u.%u.%u", NIPQUAD(ic_myaddr));

  if (root_server_addr == INADDR_NONE)
   root_server_addr = ic_servaddr;
@@ -806,7 +806,7 @@ static void __init ic_do_bootp_ext(u8 *e
    }
    break;
   case 12: /* Host name */
-   ic_bootp_string(system_utsname.nodename, ext+1, *ext, __NEW_UTS_LEN);
+   ic_bootp_string(init_uts_ns.name.nodename, ext+1, *ext, __NEW_UTS_LEN);
    ic_host_name_set = 1;
    break;
   case 15: /* Domain name (DNS) */
@@ -817,7 +817,7 @@ static void __init ic_do_bootp_ext(u8 *e
     ic_bootp_string(root_server_path, ext+1, *ext, sizeof(root_server_path));
    break;
   case 40: /* NIS Domain name (_not_ DNS) */
-   ic_bootp_string(system_utsname.domainname, ext+1, *ext, __NEW_UTS_LEN);
+   ic_bootp_string(init_uts_ns.name.domainname, ext+1, *ext, __NEW_UTS_LEN);
    break;
  }
```

```
 }
@@ -1369,7 +1369,7 @@ static int __init ip_auto_config(void)
  printk(", mask=%u.%u.%u.%u", NIPQUAD(ic_netmask));
  printk(", gw=%u.%u.%u.%u", NIPQUAD(ic_gateway));
  printk(",\n     host=%s, domain=%s, nis-domain=%s",
-        system_utsname.nodename, ic_domain, system_utsname.domainname);
+        init_uts_ns.name.nodename, ic_domain, init_uts_ns.name.domainname);
  printk(",\n     bootserver=%u.%u.%u.%u", NIPQUAD(ic_servaddr));
  printk(", rootserver=%u.%u.%u.%u", NIPQUAD(root_server_addr));
  printk(", rootpath=%s", root_server_path);
@@ -1479,11 +1479,11 @@ static int __init ip_auto_config_setup(c
     case 4:
      if ((dp = strchr(ip, '.'))) {
       *dp++ = '\0';
-      strlcpy(system_utsname.domainname, dp,
-       sizeof(system_utsname.domainname));
+      strlcpy(init_uts_ns.name.domainname, dp,
+       sizeof(init_uts_ns.name.domainname));
      }
-     strlcpy(system_utsname.nodename, ip,
-      sizeof(system_utsname.nodename));
+     strlcpy(init_uts_ns.name.nodename, ip,
+      sizeof(init_uts_ns.name.nodename));
      ic_host_name_set = 1;
      break;
     case 5:
diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c
index aa8965e..c21f28b 100644
--- a/net/sunrpc/clnt.c
+++ b/net/sunrpc/clnt.c
@@ -176,10 +176,10 @@ rpc_new_client(struct rpc_xprt *xprt, ch
  }

  /* save the nodename */
- clnt->cl_nodelen = strlen(system_utsname.nodename);
+ clnt->cl_nodelen = strlen(init_uts_ns.name.nodename);
  if (clnt->cl_nodelen > UNX_MAXNODENAME)
   clnt->cl_nodelen = UNX_MAXNODENAME;
- memcpy(clnt->cl_nodename, system_utsname.nodename, clnt->cl_nodelen);
+ memcpy(clnt->cl_nodename, init_uts_ns.name.nodename, clnt->cl_nodelen);
  return clnt;

 out_no_auth:
diff --git a/sound/core/info_oss.c b/sound/core/info_oss.c
index f9ce854..dcb665a 100644
--- a/sound/core/info_oss.c
+++ b/sound/core/info_oss.c
@@ -94,11 +94,11 @@ static void snd_sndstat_proc_read(struct
```

```
 {
  snd_iprintf(buffer, "Sound Driver:3.8.1a-980706 (ALSA v" CONFIG_SND_VERSION " emulation
code)\n");
  snd_iprintf(buffer, "Kernel: %s %s %s %s %s\n",
-       system_utsname.sysname,
-       system_utsname.nodename,
-       system_utsname.release,
-       system_utsname.version,
-       system_utsname.machine);
+       init_uts_ns.name.sysname,
+       init_uts_ns.name.nodename,
+       init_uts_ns.name.release,
+       init_uts_ns.name.version,
+       init_uts_ns.name.machine);
  snd_iprintf(buffer, "Config options: 0\n");
  snd_iprintf(buffer, "\nInstalled drivers: \n");
  snd_iprintf(buffer, "Type 10: ALSA emulation\n");
--
1.2.4
```

Subject: Re: [RFC][PATCH 0/5] uts namespaces: Introduction
Posted by ebiederm on Fri, 07 Apr 2006 19:06:09 GMT
View Forum Message <> Reply to Message

"Serge E. Hallyn" <serue@us.ibm.com> writes:

> Introduce utsname namespaces. Instead of a single system_utsname
> containing hostname domainname etc, a process can request it's
> copy of the uts info to be cloned. The data will be copied from
> it's original, but any further changes will not be seen by processes
> which are not it's children, and vice versa.
>
> This is useful, for instance, for vserver/openvz, which can now clone
> a new uts namespace for each new virtual server.
>
> This patchset is based on Kirill Korotaev's Mar 24 submission, taking
> comments (in particular from James Morris and Eric Biederman) into
> account.
>
> Some performance results are attached. I was mainly curious whether
> it would be worth putting the task_struct->uts_ns pointer inside
> a #ifdef CONFIG_UTS_NS. The result show that leaving it in when
> CONFIG_UTS_NS=n has negligable performance impact, so that is the
> approach this patch takes.

Ok. This looks like the best version so far.

I like the utsname() function thing to shorten the
idiom of current->uts_ns->name.

We probably want to introduce utsname() and an init_utsname()
before any of the other changes, and then perform the substitutions,
before we actually change the code so the patchset can make it
through a git-bisect. This will also allows for something
that can be put in compat-mac.h for backports of anything that
cares.

Eric

Subject: Re: [RFC][PATCH 1/5] uts namespaces: Implement utsname namespaces
Posted by Sam Ravnborg on Fri, 07 Apr 2006 19:13:59 GMT
View Forum Message <> Reply to Message

On Fri, Apr 07, 2006 at 01:36:00PM -0500, Serge E. Hallyn wrote:
> This patch defines the uts namespace and some manipulators.
> Adds the uts namespace to task_struct, and initializes a
> system-wide init namespace which will continue to be used when
> it makes sense.
It also kills system_utsname so you left the kernel uncompileable.
Can you kill it later?

> diff --git a/include/linux/utsname.h b/include/linux/utsname.h
> index 13e1da0..cc28ac5 100644
> --- a/include/linux/utsname.h
> +++ b/include/linux/utsname.h
> @@ -1,5 +1,8 @@
> #ifndef _LINUX_UTSNAME_H
> #define _LINUX_UTSNAME_H
You can kill this include
> +#include <linux/sched.h>

if you move this static inline to sched.h
 +
> +static inline struct new_utsname *utsname(void)
> +{
> + return &current->uts_ns->name;
> +}
And since it operates on &current that may make sense.

 Sam

Subject: Re: [RFC][PATCH 2/5] uts namespaces: Switch to using uts namespaces

Posted by Sam Ravnborg on Fri, 07 Apr 2006 19:17:42 GMT
View Forum Message <> Reply to Message

On Fri, Apr 07, 2006 at 01:36:00PM -0500, Serge E. Hallyn wrote:
> Replace references to system_utsname to the per-process uts namespace
> where appropriate. This includes things like uname.
If you define helpers that operates on system_utsname and then apply
this patch the kernel will still compile, and only later you can
introduce the new stuff.

Sam

Subject: Re: [RFC][PATCH 1/5] uts namespaces: Implement utsname namespaces
Posted by serue on Fri, 07 Apr 2006 19:20:52 GMT
View Forum Message <> Reply to Message

Quoting Sam Ravnborg (sam@ravnborg.org):
> On Fri, Apr 07, 2006 at 01:36:00PM -0500, Serge E. Hallyn wrote:
> > This patch defines the uts namespace and some manipulators.
> > Adds the uts namespace to task_struct, and initializes a
> > system-wide init namespace which will continue to be used when
> > it makes sense.
> It also kills system_utsname so you left the kernel uncompileable.
> Can you kill it later?

I can insert a #define system_utsname (init_uts_ns.name) in patch 1
and nuke it at patch 3.

-serge

Subject: Re: [RFC][PATCH 2/5] uts namespaces: Switch to using uts namespaces
Posted by serue on Fri, 07 Apr 2006 19:25:31 GMT
View Forum Message <> Reply to Message

Quoting Sam Ravnborg (sam@ravnborg.org):
> On Fri, Apr 07, 2006 at 01:36:00PM -0500, Serge E. Hallyn wrote:
> > Replace references to system_utsname to the per-process uts namespace
> > where appropriate. This includes things like uname.
> If you define helpers that operates on system_utsname and then apply
> this patch the kernel will still compile, and only later you can
> introduce the new stuff.

Ok, will try structuring the patches that way.

thanks,
-serge

Subject: Re: [RFC][PATCH 0/5] uts namespaces: Introduction
Posted by serue on Fri, 07 Apr 2006 19:28:00 GMT
View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> "Serge E. Hallyn" <serue@us.ibm.com> writes:
>
> > Introduce utsname namespaces. Instead of a single system_utsname
> > containing hostname domainname etc, a process can request it's
> > copy of the uts info to be cloned. The data will be copied from
> > it's original, but any further changes will not be seen by processes
> > which are not it's children, and vice versa.
> >
> > This is useful, for instance, for vserver/openvz, which can now clone
> > a new uts namespace for each new virtual server.
> >
> > This patchset is based on Kirill Korotaev's Mar 24 submission, taking
> > comments (in particular from James Morris and Eric Biederman) into
> > account.
> >
> > Some performance results are attached. I was mainly curious whether
> > it would be worth putting the task_struct->uts_ns pointer inside
> > a #ifdef CONFIG_UTS_NS. The result show that leaving it in when
> > CONFIG_UTS_NS=n has negligable performance impact, so that is the
> > approach this patch takes.
>
> Ok. This looks like the best version so far.
>
> I like the utsname() function thing to shorten the
> idiom of current->uts_ns->name.
>
> We probably want to introduce utsname() and an init_utsname()
> before any of the other changes, and then perform the substitutions,

This is the same as what Sam is saying, right? Just making sure I
understand.

> before we actually change the code so the patchset can make it
> through a git-bisect. This will also allows for something

Ok, I've finally got the rest of git doing my bidding, I'll go read
up on git-bisect.

thanks for the comments,
-serge

Subject: Re: [RFC][PATCH 0/5] uts namespaces: Introduction

"Serge E. Hallyn" <serue@us.ibm.com> writes:

> Quoting Eric W. Biederman (ebiederm@xmission.com):
>> "Serge E. Hallyn" <serue@us.ibm.com> writes:
>>
>> > Introduce utsname namespaces. Instead of a single system_utsname
>> > containing hostname domainname etc, a process can request it's
>> > copy of the uts info to be cloned. The data will be copied from
>> > it's original, but any further changes will not be seen by processes
>> > which are not it's children, and vice versa.
>> >
>> > This is useful, for instance, for vserver/openvz, which can now clone
>> > a new uts namespace for each new virtual server.
>> >
>> > This patchset is based on Kirill Korotaev's Mar 24 submission, taking
>> > comments (in particular from James Morris and Eric Biederman) into
>> > account.
>> >
>> > Some performance results are attached. I was mainly curious whether
>> > it would be worth putting the task_struct->uts_ns pointer inside
>> > a #ifdef CONFIG_UTS_NS. The result show that leaving it in when
>> > CONFIG_UTS_NS=n has negligable performance impact, so that is the
>> > approach this patch takes.
>>
>> Ok. This looks like the best version so far.
>>
>> I like the utsname() function thing to shorten the
>> idiom of current->uts_ns->name.
>>
>> We probably want to introduce utsname() and an init_utsname()
>> before any of the other changes, and then perform the substitutions,
>
> This is the same as what Sam is saying, right? Just making sure I
> understand.

Yes.

>> before we actually change the code so the patchset can make it
>> through a git-bisect. This will also allows for something
>
> Ok, I've finally got the rest of git doing my bidding, I'll go read
> up on git-bisect.

Basically git-bisect is an automated binary search through patches
to help find bugs. If you ever can't compile at an intermediate
patch git-bisect and other people walking through the patches

looking for bugs won't like it.

It's not mandatory that you never break anything in a patchset,
but it is much friendlier when you can avoid breakage.

Eric

Subject: Re: [RFC][PATCH 1/5] uts namespaces: Implement utsname namespaces
Posted by serue on Fri, 07 Apr 2006 19:39:37 GMT
View Forum Message <> Reply to Message

Quoting Sam Ravnborg (sam@ravnborg.org):
> > diff --git a/include/linux/utsname.h b/include/linux/utsname.h
> > index 13e1da0..cc28ac5 100644
> > --- a/include/linux/utsname.h
> > +++ b/include/linux/utsname.h
> > @@ -1,5 +1,8 @@
> > #ifndef _LINUX_UTSNAME_H
> > #define _LINUX_UTSNAME_H
> You can kill this include
> > +#include <linux/sched.h>
>
> if you move this static inline to sched.h
> +
> > +static inline struct new_utsname *utsname(void)
> > +{
> > + return &current->uts_ns->name;
> > +}
> And since it operates on &current that may make sense.

I had it there originally. Don't mind moving it back if that
seems more appropriate, but of course then we'll need
to #include <linux/utsname.h> in sched.h, since we need to
know struct uts_ns to get uts_ns->name.

So is moving it to sched.h the way to go?

thanks,
-serge

Subject: Re: [RFC][PATCH 1/5] uts namespaces: Implement utsname namespaces
Posted by James Morris on Fri, 07 Apr 2006 20:47:44 GMT
View Forum Message <> Reply to Message

On Fri, 7 Apr 2006, Serge E. Hallyn wrote:

> +EXPORT_SYMBOL(unshare_uts_ns);
> +EXPORT_SYMBOL(free_uts_ns);

Why not EXPORT_SYMBOL_GPL?

What do you expect the user api to look like, a syscall?

Probably need to think about LSM hooks for creating and updating the
namespaces.

- James
--
James Morris
<jmorris@namei.org>

Subject: Re: [RFC][PATCH 1/5] uts namespaces: Implement utsname namespaces
Posted by serue on Fri, 07 Apr 2006 22:13:34 GMT
View Forum Message <> Reply to Message

Quoting James Morris (jmorris@namei.org):
> On Fri, 7 Apr 2006, Serge E. Hallyn wrote:
>
>
> > > +EXPORT_SYMBOL(unshare_uts_ns);
> > > +EXPORT_SYMBOL(free_uts_ns);
>
> Why not EXPORT_SYMBOL_GPL?

Actually come to think of it they don't need to be exported.

I will move the exports to the last, debugging, patch.

> What do you expect the user api to look like, a syscall?

This remains to be determined, and this patchset purposely doesn't
address it. AFAIU, the two most likely options are extending clone and
unshare, and using new syscalls. Whatever is decided for the other
namespaces, this should use.

With this patchset (minus the last patch for debugging) uts namespaces
are supported, but processes can't clone their uts namespace yet.

> Probably need to think about LSM hooks for creating and updating the
> namespaces.

True, that is something that needs to be discussed when the topic
of how to implement unsharing comes up again.

thanks,
-serge

Subject: Re: [RFC][PATCH 2/5] uts namespaces: Switch to using uts namespaces
Posted by dev on Tue, 11 Apr 2006 12:19:54 GMT
View Forum Message <> Reply to Message

Serge,

BTW, have you noticed that NFS is using utsname for internal processes
and in general case this makes NFS ns to be coupled with uts ns?

Kirill

> Replace references to system_utsname to the per-process uts namespace
> where appropriate. This includes things like uname.
>
> Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>
> ---
> arch/alpha/kernel/osf_sys.c | 24 ++++++++++++------------
> arch/i386/kernel/sys_i386.c | 12 ++++++------
> arch/ia64/sn/kernel/sn2/sn_hwperf.c | 2 +-
> arch/m32r/kernel/sys_m32r.c | 2 +-
> arch/mips/kernel/linux32.c | 2 +-
> arch/mips/kernel/syscall.c | 18 +++++++++---------
> arch/mips/kernel/sysirix.c | 12 ++++++------
> arch/parisc/hpux/sys_hpux.c | 22 ++++++++++++++++++++++
> arch/powerpc/kernel/syscalls.c | 14 ++++++++++++++
> arch/sh/kernel/sys_sh.c | 2 +-
> arch/sh64/kernel/sys_sh64.c | 2 +-
> arch/sparc/kernel/sys_sparc.c | 4 ++--
> arch/sparc/kernel/sys_sunos.c | 10 +++++++++-
> arch/sparc64/kernel/sys_sparc.c | 4 ++--
> arch/sparc64/kernel/sys_sunos32.c | 10 +++++++++-
> arch/sparc64/solaris/misc.c | 6 +++---
> arch/um/drivers/mconsole_kern.c | 6 +++---
> arch/um/kernel/syscall_kern.c | 12 ++++++------
> arch/um/sys-x86_64/syscalls.c | 2 +-
> arch/x86_64/ia32/sys_ia32.c | 10 +++++++++-
> arch/x86_64/kernel/sys_x86_64.c | 2 +-
> arch/xtensa/kernel/syscalls.c | 2 +-
> drivers/char/random.c | 4 ++--
> fs/cifs/connect.c | 28 ++++++++++++++++++++++++++++

```
>  fs/exec.c                       |    2 +-
>  fs/lockd/clntproc.c             |    4 ++--
>  fs/lockd/mon.c                  |    2 +-
>  fs/lockd/svclock.c              |    2 +-
>  fs/lockd/xdr.c                  |    2 +-
>  fs/nfs/nfsroot.c                |    2 +-
>  include/linux/lockd/lockd.h     |    2 +-
>  kernel/sys.c                    |   14 +++++++-------
>  32 files changed, 121 insertions(+), 121 deletions(-)
>
> 92a8cf13a78415ed5ec9068698b5039ddcc00210
> diff --git a/arch/alpha/kernel/osf_sys.c b/arch/alpha/kernel/osf_sys.c
> index 31afe3d..b793b96 100644
> --- a/arch/alpha/kernel/osf_sys.c
> +++ b/arch/alpha/kernel/osf_sys.c
> @@ -402,15 +402,15 @@ osf_utsname(char __user *name)
>
>   down_read(&uts_sem);
>   error = -EFAULT;
> - if (copy_to_user(name + 0, system_utsname.sysname, 32))
> + if (copy_to_user(name + 0, utsname()->sysname, 32))
>    goto out;
> - if (copy_to_user(name + 32, system_utsname.nodename, 32))
> + if (copy_to_user(name + 32, utsname()->nodename, 32))
>    goto out;
> - if (copy_to_user(name + 64, system_utsname.release, 32))
> + if (copy_to_user(name + 64, utsname()->release, 32))
>    goto out;
> - if (copy_to_user(name + 96, system_utsname.version, 32))
> + if (copy_to_user(name + 96, utsname()->version, 32))
>    goto out;
> - if (copy_to_user(name + 128, system_utsname.machine, 32))
> + if (copy_to_user(name + 128, utsname()->machine, 32))
>    goto out;
>
>   error = 0;
> @@ -449,8 +449,8 @@ osf_getdomainname(char __user *name, int
>
>   down_read(&uts_sem);
>   for (i = 0; i < len; ++i) {
> -  __put_user(system_utsname.domainname[i], name + i);
> -  if (system_utsname.domainname[i] == '\0')
> +  __put_user(utsname()->domainname[i], name + i);
> +  if (utsname()->domainname[i] == '\0')
>     break;
>   }
>   up_read(&uts_sem);
> @@ -608,11 +608,11 @@ asmlinkage long
```

> osf_sysinfo(int command, char __user *buf, long count)
> {
>   static char * sysinfo_table[] = {
> - system_utsname.sysname,
> - system_utsname.nodename,
> - system_utsname.release,
> - system_utsname.version,
> - system_utsname.machine,
> + utsname()->sysname,
> + utsname()->nodename,
> + utsname()->release,
> + utsname()->version,
> + utsname()->machine,
>   "alpha", /* instruction set architecture */
>   "dummy", /* hardware serial number */
>   "dummy", /* hardware manufacturer */
> diff --git a/arch/i386/kernel/sys_i386.c b/arch/i386/kernel/sys_i386.c
> index 8fdb1fb..4af731d 100644
> --- a/arch/i386/kernel/sys_i386.c
> +++ b/arch/i386/kernel/sys_i386.c
> @@ -210,7 +210,7 @@ asmlinkage int sys_uname(struct old_utsn
>   if (!name)
>   return -EFAULT;
>   down_read(&uts_sem);
> - err=copy_to_user(name, &system_utsname, sizeof (*name));
> + err=copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   return err?-EFAULT:0;
> }
> @@ -226,15 +226,15 @@ asmlinkage int sys_olduname(struct oldol
>
>   down_read(&uts_sem);
>
> - error = __copy_to_user(&name->sysname,&system_utsname.sysname,__OLD_UTS_LEN);
> + error = __copy_to_user(&name->sysname,&utsname()->sysname,__OLD_UTS_LEN);
>   error |= __put_user(0,name->sysname+__OLD_UTS_LEN);
> - error |=
__copy_to_user(&name->nodename,&system_utsname.nodename,__OLD_UTS_LEN);
> + error |= __copy_to_user(&name->nodename,&utsname()->nodename,__OLD_UTS_LEN);
>   error |= __put_user(0,name->nodename+__OLD_UTS_LEN);
> - error |= __copy_to_user(&name->release,&system_utsname.release,__OLD_UTS_LEN);
> + error |= __copy_to_user(&name->release,&utsname()->release,__OLD_UTS_LEN);
>   error |= __put_user(0,name->release+__OLD_UTS_LEN);
> - error |= __copy_to_user(&name->version,&system_utsname.version,__OLD_UTS_LEN);
> + error |= __copy_to_user(&name->version,&utsname()->version,__OLD_UTS_LEN);
>   error |= __put_user(0,name->version+__OLD_UTS_LEN);
> - error |= __copy_to_user(&name->machine,&system_utsname.machine,__OLD_UTS_LEN);
> + error |= __copy_to_user(&name->machine,&utsname()->machine,__OLD_UTS_LEN);

>   error |= __put_user(0,name->machine+__OLD_UTS_LEN);
>
>   up_read(&uts_sem);
> diff --git a/arch/ia64/sn/kernel/sn2/sn_hwperf.c b/arch/ia64/sn/kernel/sn2/sn_hwperf.c
> index d917afa..a0632a9 100644
> --- a/arch/ia64/sn/kernel/sn2/sn_hwperf.c
> +++ b/arch/ia64/sn/kernel/sn2/sn_hwperf.c
> @@ -420,7 +420,7 @@ static int sn_topology_show(struct seq_f
>   "coherency_domain %d, "
>   "region_size %d\n",
>
> -   partid, system_utsname.nodename,
> +   partid, utsname()->nodename,
>   shubtype ? "shub2" : "shub1",
>   (u64)nasid_mask << nasid_shift, nasid_msb, nasid_shift,
>   system_size, sharing_size, coher, region_size);
> diff --git a/arch/m32r/kernel/sys_m32r.c b/arch/m32r/kernel/sys_m32r.c
> index 670cb49..11412c0 100644
> --- a/arch/m32r/kernel/sys_m32r.c
> +++ b/arch/m32r/kernel/sys_m32r.c
> @@ -206,7 +206,7 @@ asmlinkage int sys_uname(struct old_utsn
>   if (!name)
>   return -EFAULT;
>   down_read(&uts_sem);
> - err=copy_to_user(name, &system_utsname, sizeof (*name));
> + err=copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   return err?-EFAULT:0;
> }
> diff --git a/arch/mips/kernel/linux32.c b/arch/mips/kernel/linux32.c
> index 3f40c37..b9b702f 100644
> --- a/arch/mips/kernel/linux32.c
> +++ b/arch/mips/kernel/linux32.c
> @@ -1100,7 +1100,7 @@ asmlinkage long sys32_newuname(struct ne
>   int ret = 0;
>
>   down_read(&uts_sem);
> - if (copy_to_user(name,&system_utsname,sizeof *name))
> + if (copy_to_user(name,utsname(),sizeof *name))
>   ret = -EFAULT;
>   up_read(&uts_sem);
>
> diff --git a/arch/mips/kernel/syscall.c b/arch/mips/kernel/syscall.c
> index 2aeaa2f..8b13d57 100644
> --- a/arch/mips/kernel/syscall.c
> +++ b/arch/mips/kernel/syscall.c
> @@ -232,7 +232,7 @@ out:
>   */

> asmlinkage int sys_uname(struct old_utsname __user * name)
> {
> - if (name && !copy_to_user(name, &system_utsname, sizeof (*name)))
> + if (name && !copy_to_user(name, utsname(), sizeof (*name)))
>   return 0;
>   return -EFAULT;
> }
> @@ -249,15 +249,15 @@ asmlinkage int sys_olduname(struct oldol
>   if (!access_ok(VERIFY_WRITE,name,sizeof(struct oldold_utsname)))
>   return -EFAULT;
>
> - error = __copy_to_user(&name->sysname,&system_utsname.sysname,__OLD_UTS_LEN);
> + error = __copy_to_user(&name->sysname,&utsname()->sysname,__OLD_UTS_LEN);
>   error -= __put_user(0,name->sysname+__OLD_UTS_LEN);
> - error -=
__copy_to_user(&name->nodename,&system_utsname.nodename,__OLD_UTS_LEN);
> + error -= __copy_to_user(&name->nodename,&utsname()->nodename,__OLD_UTS_LEN);
>   error -= __put_user(0,name->nodename+__OLD_UTS_LEN);
> - error -= __copy_to_user(&name->release,&system_utsname.release,__OLD_UTS_LEN);
> + error -= __copy_to_user(&name->release,&utsname()->release,__OLD_UTS_LEN);
>   error -= __put_user(0,name->release+__OLD_UTS_LEN);
> - error -= __copy_to_user(&name->version,&system_utsname.version,__OLD_UTS_LEN);
> + error -= __copy_to_user(&name->version,&utsname()->version,__OLD_UTS_LEN);
>   error -= __put_user(0,name->version+__OLD_UTS_LEN);
> - error -= __copy_to_user(&name->machine,&system_utsname.machine,__OLD_UTS_LEN);
> + error -= __copy_to_user(&name->machine,&utsname()->machine,__OLD_UTS_LEN);
>   error = __put_user(0,name->machine+__OLD_UTS_LEN);
>   error = error ? -EFAULT : 0;
>
> @@ -293,10 +293,10 @@ asmlinkage int _sys_sysmips(int cmd, lon
>   return -EFAULT;
>
>   down_write(&uts_sem);
> - strncpy(system_utsname.nodename, nodename, len);
> + strncpy(utsname()->nodename, nodename, len);
>   nodename[__NEW_UTS_LEN] = '\0';
> - strlcpy(system_utsname.nodename, nodename,
> -   sizeof(system_utsname.nodename));
> + strlcpy(utsname()->nodename, nodename,
> +   sizeof(utsname()->nodename));
>   up_write(&uts_sem);
>   return 0;
>   }
> diff --git a/arch/mips/kernel/sysirix.c b/arch/mips/kernel/sysirix.c
> index 5407b78..1b4e7e7 100644
> --- a/arch/mips/kernel/sysirix.c
> +++ b/arch/mips/kernel/sysirix.c
> @@ -884,7 +884,7 @@ asmlinkage int irix_getdomainname(char _

```
>   down_read(&uts_sem);
>   if (len > __NEW_UTS_LEN)
>    len = __NEW_UTS_LEN;
> - err = copy_to_user(name, system_utsname.domainname, len) ? -EFAULT : 0;
> + err = copy_to_user(name, utsname()->domainname, len) ? -EFAULT : 0;
>   up_read(&uts_sem);
>
>   return err;
> @@ -1127,11 +1127,11 @@ struct iuname {
>  asmlinkage int irix_uname(struct iuname __user *buf)
>  {
>   down_read(&uts_sem);
> - if (copy_from_user(system_utsname.sysname, buf->sysname, 65)
> -     || copy_from_user(system_utsname.nodename, buf->nodename, 65)
> -     || copy_from_user(system_utsname.release, buf->release, 65)
> -     || copy_from_user(system_utsname.version, buf->version, 65)
> -     || copy_from_user(system_utsname.machine, buf->machine, 65)) {
> + if (copy_from_user(utsname()->sysname, buf->sysname, 65)
> +     || copy_from_user(utsname()->nodename, buf->nodename, 65)
> +     || copy_from_user(utsname()->release, buf->release, 65)
> +     || copy_from_user(utsname()->version, buf->version, 65)
> +     || copy_from_user(utsname()->machine, buf->machine, 65)) {
>    return -EFAULT;
>   }
>   up_read(&uts_sem);
> diff --git a/arch/parisc/hpux/sys_hpux.c b/arch/parisc/hpux/sys_hpux.c
> index 05273cc..9fc2c08 100644
> --- a/arch/parisc/hpux/sys_hpux.c
> +++ b/arch/parisc/hpux/sys_hpux.c
> @@ -266,15 +266,15 @@ static int hpux_uname(struct hpux_utsnam
>
>   down_read(&uts_sem);
>
> - error = __copy_to_user(&name->sysname,&system_utsname.sysname,HPUX_UTSLEN-1);
> + error = __copy_to_user(&name->sysname,&utsname()->sysname,HPUX_UTSLEN-1);
>   error |= __put_user(0,name->sysname+HPUX_UTSLEN-1);
> - error |=
__copy_to_user(&name->nodename,&system_utsname.nodename,HPUX_UTSLEN-1);
> + error |= __copy_to_user(&name->nodename,&utsname()->nodename,HPUX_UTSLEN-1);
>   error |= __put_user(0,name->nodename+HPUX_UTSLEN-1);
> - error |= __copy_to_user(&name->release,&system_utsname.release,HPUX_UTSLEN-1);
> + error |= __copy_to_user(&name->release,&utsname()->release,HPUX_UTSLEN-1);
>   error |= __put_user(0,name->release+HPUX_UTSLEN-1);
> - error |= __copy_to_user(&name->version,&system_utsname.version,HPUX_UTSLEN-1);
> + error |= __copy_to_user(&name->version,&utsname()->version,HPUX_UTSLEN-1);
>   error |= __put_user(0,name->version+HPUX_UTSLEN-1);
> - error |= __copy_to_user(&name->machine,&system_utsname.machine,HPUX_UTSLEN-1);
> + error |= __copy_to_user(&name->machine,&utsname()->machine,HPUX_UTSLEN-1);
```

```
>   error |= __put_user(0,name->machine+HPUX_UTSLEN-1);
>
>   up_read(&uts_sem);
> @@ -373,8 +373,8 @@ int hpux_utssys(char *ubuf, int n, int t
>    /*  TODO:  print a warning about using this?  */
>    down_write(&uts_sem);
>    error = -EFAULT;
> -  if (!copy_from_user(system_utsname.sysname, ubuf, len)) {
> -   system_utsname.sysname[len] = 0;
> +  if (!copy_from_user(utsname()->sysname, ubuf, len)) {
> +   utsname()->sysname[len] = 0;
>     error = 0;
>    }
>    up_write(&uts_sem);
> @@ -400,8 +400,8 @@ int hpux_utssys(char *ubuf, int n, int t
>    /*  TODO:  print a warning about this?  */
>    down_write(&uts_sem);
>    error = -EFAULT;
> -  if (!copy_from_user(system_utsname.release, ubuf, len)) {
> -   system_utsname.release[len] = 0;
> +  if (!copy_from_user(utsname()->release, ubuf, len)) {
> +   utsname()->release[len] = 0;
>     error = 0;
>    }
>    up_write(&uts_sem);
> @@ -422,13 +422,13 @@ int hpux_getdomainname(char *name, int l
>
>    down_read(&uts_sem);
>
> - nlen = strlen(system_utsname.domainname) + 1;
> + nlen = strlen(utsname()->domainname) + 1;
>
>   if (nlen < len)
>    len = nlen;
>   if(len > __NEW_UTS_LEN)
>    goto done;
> - if(copy_to_user(name, system_utsname.domainname, len))
> + if(copy_to_user(name, utsname()->domainname, len))
>    goto done;
>   err = 0;
>  done:
> diff --git a/arch/powerpc/kernel/syscalls.c b/arch/powerpc/kernel/syscalls.c
> index 9b69d99..d358866 100644
> --- a/arch/powerpc/kernel/syscalls.c
> +++ b/arch/powerpc/kernel/syscalls.c
> @@ -260,7 +260,7 @@ long ppc_newuname(struct new_utsname __u
>   int err = 0;
>
```

```
>   down_read(&uts_sem);
> - if (copy_to_user(name, &system_utsname, sizeof(*name)))
> + if (copy_to_user(name, utsname(), sizeof(*name)))
>     err = -EFAULT;
>   up_read(&uts_sem);
>   if (!err)
> @@ -273,7 +273,7 @@ int sys_uname(struct old_utsname __user
>   int err = 0;
>
>   down_read(&uts_sem);
> - if (copy_to_user(name, &system_utsname, sizeof(*name)))
> + if (copy_to_user(name, utsname(), sizeof(*name)))
>     err = -EFAULT;
>   up_read(&uts_sem);
>   if (!err)
> @@ -289,19 +289,19 @@ int sys_olduname(struct oldold_utsname _
>     return -EFAULT;
>
>   down_read(&uts_sem);
> - error = __copy_to_user(&name->sysname, &system_utsname.sysname,
> + error = __copy_to_user(&name->sysname, &utsname()->sysname,
>             __OLD_UTS_LEN);
>   error |= __put_user(0, name->sysname + __OLD_UTS_LEN);
> - error |= __copy_to_user(&name->nodename, &system_utsname.nodename,
> + error |= __copy_to_user(&name->nodename, &utsname()->nodename,
>       __OLD_UTS_LEN);
>   error |= __put_user(0, name->nodename + __OLD_UTS_LEN);
> - error |= __copy_to_user(&name->release, &system_utsname.release,
> + error |= __copy_to_user(&name->release, &utsname()->release,
>       __OLD_UTS_LEN);
>   error |= __put_user(0, name->release + __OLD_UTS_LEN);
> - error |= __copy_to_user(&name->version, &system_utsname.version,
> + error |= __copy_to_user(&name->version, &utsname()->version,
>       __OLD_UTS_LEN);
>   error |= __put_user(0, name->version + __OLD_UTS_LEN);
> - error |= __copy_to_user(&name->machine, &system_utsname.machine,
> + error |= __copy_to_user(&name->machine, &utsname()->machine,
>       __OLD_UTS_LEN);
>   error |= override_machine(name->machine);
>   up_read(&uts_sem);
> diff --git a/arch/sh/kernel/sys_sh.c b/arch/sh/kernel/sys_sh.c
> index 917b2f3..e4966b2 100644
> --- a/arch/sh/kernel/sys_sh.c
> +++ b/arch/sh/kernel/sys_sh.c
> @@ -267,7 +267,7 @@ asmlinkage int sys_uname(struct old_utsn
>   if (!name)
>     return -EFAULT;
>   down_read(&uts_sem);
```

> - err=copy_to_user(name, &system_utsname, sizeof (*name));
> + err=copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   return err?-EFAULT:0;
> }
> diff --git a/arch/sh64/kernel/sys_sh64.c b/arch/sh64/kernel/sys_sh64.c
> index 58ff7d5..a8dc88c 100644
> --- a/arch/sh64/kernel/sys_sh64.c
> +++ b/arch/sh64/kernel/sys_sh64.c
> @@ -279,7 +279,7 @@ asmlinkage int sys_uname(struct old_utsn
>   if (!name)
>   return -EFAULT;
>   down_read(&uts_sem);
> - err=copy_to_user(name, &system_utsname, sizeof (*name));
> + err=copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   return err?-EFAULT:0;
> }
> diff --git a/arch/sparc/kernel/sys_sparc.c b/arch/sparc/kernel/sys_sparc.c
> index 0cdfc9d..c8ad73c 100644
> --- a/arch/sparc/kernel/sys_sparc.c
> +++ b/arch/sparc/kernel/sys_sparc.c
> @@ -470,13 +470,13 @@ asmlinkage int sys_getdomainname(char __
>
>   down_read(&uts_sem);
>
> - nlen = strlen(system_utsname.domainname) + 1;
> + nlen = strlen(utsname()->domainname) + 1;
>
>   if (nlen < len)
>   len = nlen;
>   if (len > __NEW_UTS_LEN)
>   goto done;
> - if (copy_to_user(name, system_utsname.domainname, len))
> + if (copy_to_user(name, utsname()->domainname, len))
>   goto done;
>   err = 0;
> done:
> diff --git a/arch/sparc/kernel/sys_sunos.c b/arch/sparc/kernel/sys_sunos.c
> index 288de27..9f9206f 100644
> --- a/arch/sparc/kernel/sys_sunos.c
> +++ b/arch/sparc/kernel/sys_sunos.c
> @@ -483,13 +483,13 @@ asmlinkage int sunos_uname(struct sunos_
> {
>   int ret;
>   down_read(&uts_sem);
> - ret = copy_to_user(&name->sname[0], &system_utsname.sysname[0], sizeof(name->sname) -
1);

```
> + ret = copy_to_user(&name->sname[0], &utsname()->sysname[0], sizeof(name->sname) - 1);
>   if (!ret) {
> -  ret |= __copy_to_user(&name->nname[0], &system_utsname.nodename[0],
sizeof(name->nname) - 1);
> +  ret |= __copy_to_user(&name->nname[0], &utsname()->nodename[0], sizeof(name->nname)
- 1);
>    ret |= __put_user('\0', &name->nname[8]);
> -  ret |= __copy_to_user(&name->rel[0], &system_utsname.release[0], sizeof(name->rel) - 1);
> -  ret |= __copy_to_user(&name->ver[0], &system_utsname.version[0], sizeof(name->ver) - 1);
> -  ret |= __copy_to_user(&name->mach[0], &system_utsname.machine[0], sizeof(name->mach)
- 1);
> +  ret |= __copy_to_user(&name->rel[0], &utsname()->release[0], sizeof(name->rel) - 1);
> +  ret |= __copy_to_user(&name->ver[0], &utsname()->version[0], sizeof(name->ver) - 1);
> +  ret |= __copy_to_user(&name->mach[0], &utsname()->machine[0], sizeof(name->mach) - 1);
>   }
>   up_read(&uts_sem);
>   return ret ? -EFAULT : 0;
> diff --git a/arch/sparc64/kernel/sys_sparc.c b/arch/sparc64/kernel/sys_sparc.c
> index 7a86913..0453bd2 100644
> --- a/arch/sparc64/kernel/sys_sparc.c
> +++ b/arch/sparc64/kernel/sys_sparc.c
> @@ -707,13 +707,13 @@ asmlinkage long sys_getdomainname(char _
>
>    down_read(&uts_sem);
>
> - nlen = strlen(system_utsname.domainname) + 1;
> + nlen = strlen(utsname()->domainname) + 1;
>
>          if (nlen < len)
>                  len = nlen;
>   if (len > __NEW_UTS_LEN)
>    goto done;
> - if (copy_to_user(name, system_utsname.domainname, len))
> + if (copy_to_user(name, utsname()->domainname, len))
>    goto done;
>   err = 0;
>  done:
> diff --git a/arch/sparc64/kernel/sys_sunos32.c b/arch/sparc64/kernel/sys_sunos32.c
> index ae5b32f..ba98c47 100644
> --- a/arch/sparc64/kernel/sys_sunos32.c
> +++ b/arch/sparc64/kernel/sys_sunos32.c
> @@ -439,16 +439,16 @@ asmlinkage int sunos_uname(struct sunos_
>   int ret;
>
>   down_read(&uts_sem);
> - ret = copy_to_user(&name->sname[0], &system_utsname.sysname[0],
> + ret = copy_to_user(&name->sname[0], &utsname()->sysname[0],
>        sizeof(name->sname) - 1);
```

```
> - ret |= copy_to_user(&name->nname[0], &system_utsname.nodename[0],
> + ret |= copy_to_user(&name->nname[0], &utsname()->nodename[0],
>         sizeof(name->nname) - 1);
>   ret |= put_user('\0', &name->nname[8]);
> - ret |= copy_to_user(&name->rel[0], &system_utsname.release[0],
> + ret |= copy_to_user(&name->rel[0], &utsname()->release[0],
>         sizeof(name->rel) - 1);
> - ret |= copy_to_user(&name->ver[0], &system_utsname.version[0],
> + ret |= copy_to_user(&name->ver[0], &utsname()->version[0],
>         sizeof(name->ver) - 1);
> - ret |= copy_to_user(&name->mach[0], &system_utsname.machine[0],
> + ret |= copy_to_user(&name->mach[0], &utsname()->machine[0],
>         sizeof(name->mach) - 1);
>   up_read(&uts_sem);
>   return (ret ? -EFAULT : 0);
> diff --git a/arch/sparc64/solaris/misc.c b/arch/sparc64/solaris/misc.c
> index 5284996..5d0162a 100644
> --- a/arch/sparc64/solaris/misc.c
> +++ b/arch/sparc64/solaris/misc.c
> @@ -239,7 +239,7 @@ asmlinkage int solaris_utssys(u32 buf, u
>    /* Let's cheat */
>    err  = set_utsfield(v->sysname, "SunOS", 1, 0);
>    down_read(&uts_sem);
> -  err |= set_utsfield(v->nodename, system_utsname.nodename,
> +  err |= set_utsfield(v->nodename, utsname()->nodename,
>          1, 1);
>    up_read(&uts_sem);
>    err |= set_utsfield(v->release, "2.6", 0, 0);
> @@ -263,7 +263,7 @@ asmlinkage int solaris_utsname(u32 buf)
>   /* Why should we not lie a bit? */
>   down_read(&uts_sem);
>   err  = set_utsfield(v->sysname, "SunOS", 0, 0);
> - err |= set_utsfield(v->nodename, system_utsname.nodename, 1, 1);
> + err |= set_utsfield(v->nodename, utsname()->nodename, 1, 1);
>   err |= set_utsfield(v->release, "5.6", 0, 0);
>   err |= set_utsfield(v->version, "Generic", 0, 0);
>   err |= set_utsfield(v->machine, machine(), 0, 0);
> @@ -295,7 +295,7 @@ asmlinkage int solaris_sysinfo(int cmd,
>   case SI_HOSTNAME:
>    r = buffer + 256;
>    down_read(&uts_sem);
> -  for (p = system_utsname.nodename, q = buffer;
> +  for (p = utsname()->nodename, q = buffer;
>         q < r && *p && *p != '.'; *q++ = *p++);
>    up_read(&uts_sem);
>    *q = 0;
> diff --git a/arch/um/drivers/mconsole_kern.c b/arch/um/drivers/mconsole_kern.c
> index 28e3760..5f87323 100644
```

```
> --- a/arch/um/drivers/mconsole_kern.c
> +++ b/arch/um/drivers/mconsole_kern.c
> @@ -106,9 +106,9 @@ void mconsole_version(struct mc_request
>  {
>    char version[256];
>
> - sprintf(version, "%s %s %s %s %s", system_utsname.sysname,
> -  system_utsname.nodename, system_utsname.release,
> -  system_utsname.version, system_utsname.machine);
> + sprintf(version, "%s %s %s %s %s", utsname()->sysname,
> +  utsname()->nodename, utsname()->release,
> +  utsname()->version, utsname()->machine);
>    mconsole_reply(req, version, 0, 0);
>  }
>
> diff --git a/arch/um/kernel/syscall_kern.c b/arch/um/kernel/syscall_kern.c
> index 37d3978..d90e9ed 100644
> --- a/arch/um/kernel/syscall_kern.c
> +++ b/arch/um/kernel/syscall_kern.c
> @@ -110,7 +110,7 @@ long sys_uname(struct old_utsname __user
>    if (!name)
>     return -EFAULT;
>    down_read(&uts_sem);
> - err=copy_to_user(name, &system_utsname, sizeof (*name));
> + err=copy_to_user(name, utsname(), sizeof (*name));
>    up_read(&uts_sem);
>    return err?-EFAULT:0;
>  }
> @@ -126,19 +126,19 @@ long sys_olduname(struct oldold_utsname
>
>     down_read(&uts_sem);
>
> - error = __copy_to_user(&name->sysname,&system_utsname.sysname,
> + error = __copy_to_user(&name->sysname,&utsname()->sysname,
>             __OLD_UTS_LEN);
>    error |= __put_user(0,name->sysname+__OLD_UTS_LEN);
> - error |= __copy_to_user(&name->nodename,&system_utsname.nodename,
> + error |= __copy_to_user(&name->nodename,&utsname()->nodename,
>       __OLD_UTS_LEN);
>    error |= __put_user(0,name->nodename+__OLD_UTS_LEN);
> - error |= __copy_to_user(&name->release,&system_utsname.release,
> + error |= __copy_to_user(&name->release,&utsname()->release,
>       __OLD_UTS_LEN);
>    error |= __put_user(0,name->release+__OLD_UTS_LEN);
> - error |= __copy_to_user(&name->version,&system_utsname.version,
> + error |= __copy_to_user(&name->version,&utsname()->version,
>       __OLD_UTS_LEN);
>    error |= __put_user(0,name->version+__OLD_UTS_LEN);
```

```
> - error |= __copy_to_user(&name->machine,&system_utsname.machine,
> + error |= __copy_to_user(&name->machine,&utsname()->machine,
>       __OLD_UTS_LEN);
>   error |= __put_user(0,name->machine+__OLD_UTS_LEN);
>
> diff --git a/arch/um/sys-x86_64/syscalls.c b/arch/um/sys-x86_64/syscalls.c
> index 6acee5c..3ad014e 100644
> --- a/arch/um/sys-x86_64/syscalls.c
> +++ b/arch/um/sys-x86_64/syscalls.c
> @@ -21,7 +21,7 @@ asmlinkage long sys_uname64(struct new_u
>  {
>   int err;
>   down_read(&uts_sem);
> - err = copy_to_user(name, &system_utsname, sizeof (*name));
> + err = copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   if (personality(current->personality) == PER_LINUX32)
>    err |= copy_to_user(&name->machine, "i686", 5);
> diff --git a/arch/x86_64/ia32/sys_ia32.c b/arch/x86_64/ia32/sys_ia32.c
> index f182b20..6e0a19d 100644
> --- a/arch/x86_64/ia32/sys_ia32.c
> +++ b/arch/x86_64/ia32/sys_ia32.c
> @@ -801,13 +801,13 @@ asmlinkage long sys32_olduname(struct ol
>
>     down_read(&uts_sem);
>
> - error = __copy_to_user(&name->sysname,&system_utsname.sysname,__OLD_UTS_LEN);
> + error = __copy_to_user(&name->sysname,&utsname()->sysname,__OLD_UTS_LEN);
>    __put_user(0,name->sysname+__OLD_UTS_LEN);
> -  __copy_to_user(&name->nodename,&system_utsname.nodename,__OLD_UTS_LEN);
> +  __copy_to_user(&name->nodename,&utsname()->nodename,__OLD_UTS_LEN);
>    __put_user(0,name->nodename+__OLD_UTS_LEN);
> -  __copy_to_user(&name->release,&system_utsname.release,__OLD_UTS_LEN);
> +  __copy_to_user(&name->release,&utsname()->release,__OLD_UTS_LEN);
>    __put_user(0,name->release+__OLD_UTS_LEN);
> -  __copy_to_user(&name->version,&system_utsname.version,__OLD_UTS_LEN);
> +  __copy_to_user(&name->version,&utsname()->version,__OLD_UTS_LEN);
>    __put_user(0,name->version+__OLD_UTS_LEN);
>    {
>     char *arch = "x86_64";
> @@ -830,7 +830,7 @@ long sys32_uname(struct old_utsname __us
>   if (!name)
>    return -EFAULT;
>   down_read(&uts_sem);
> - err=copy_to_user(name, &system_utsname, sizeof (*name));
> + err=copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   if (personality(current->personality) == PER_LINUX32)
```

```
>    err |= copy_to_user(&name->machine, "i686", 5);
> diff --git a/arch/x86_64/kernel/sys_x86_64.c b/arch/x86_64/kernel/sys_x86_64.c
> index 6449ea8..76bf7c2 100644
> --- a/arch/x86_64/kernel/sys_x86_64.c
> +++ b/arch/x86_64/kernel/sys_x86_64.c
> @@ -148,7 +148,7 @@ asmlinkage long sys_uname(struct new_uts
>  {
>   int err;
>   down_read(&uts_sem);
> - err = copy_to_user(name, &system_utsname, sizeof (*name));
> + err = copy_to_user(name, utsname(), sizeof (*name));
>   up_read(&uts_sem);
>   if (personality(current->personality) == PER_LINUX32)
>    err |= copy_to_user(&name->machine, "i686", 5);
> diff --git a/arch/xtensa/kernel/syscalls.c b/arch/xtensa/kernel/syscalls.c
> index f20c649..30060c1 100644
> --- a/arch/xtensa/kernel/syscalls.c
> +++ b/arch/xtensa/kernel/syscalls.c
> @@ -129,7 +129,7 @@ out:
>
>  int sys_uname(struct old_utsname * name)
>  {
> - if (name && !copy_to_user(name, &system_utsname, sizeof (*name)))
> + if (name && !copy_to_user(name, utsname(), sizeof (*name)))
>    return 0;
>   return -EFAULT;
>  }
> diff --git a/drivers/char/random.c b/drivers/char/random.c
> index 86be04b..ec4c11d 100644
> --- a/drivers/char/random.c
> +++ b/drivers/char/random.c
> @@ -888,8 +888,8 @@ static void init_std_data(struct entropy
>
>   do_gettimeofday(&tv);
>   add_entropy_words(r, (__u32 *)&tv, sizeof(tv)/4);
> - add_entropy_words(r, (__u32 *)&system_utsname,
> -     sizeof(system_utsname)/4);
> + add_entropy_words(r, (__u32 *)utsname(),
> +     sizeof(*(utsname()))/4);
>  }
>
>  static int __init rand_initialize(void)
> diff --git a/fs/cifs/connect.c b/fs/cifs/connect.c
> index 0b86d5c..852ff41 100644
> --- a/fs/cifs/connect.c
> +++ b/fs/cifs/connect.c
> @@ -765,12 +765,12 @@ cifs_parse_mount_options(char *options,
>   separator[1] = 0;
```

```
>
>   memset(vol->source_rfc1001_name,0x20,15);
> - for(i=0;i < strnlen(system_utsname.nodename,15);i++) {
> + for(i=0;i < strnlen(utsname()->nodename,15);i++) {
>    /* does not have to be a perfect mapping since the field is
>    informational, only used for servers that do not support
>    port 445 and it can be overridden at mount time */
>    vol->source_rfc1001_name[i] =
> -   toupper(system_utsname.nodename[i]);
> +   toupper(utsname()->nodename[i]);
>   }
>   vol->source_rfc1001_name[15] = 0;
>   /* null target name indicates to use *SMBSERVR default called name
> @@ -2077,7 +2077,7 @@ CIFSSessSetup(unsigned int xid, struct c
>        32, nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bytes_returned =
> -      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release,
> +      cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release,
>        32, nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bcc_ptr += 2;
> @@ -2104,8 +2104,8 @@ CIFSSessSetup(unsigned int xid, struct c
>    }
>    strcpy(bcc_ptr, "Linux version ");
>    bcc_ptr += strlen("Linux version ");
> -  strcpy(bcc_ptr, system_utsname.release);
> -  bcc_ptr += strlen(system_utsname.release) + 1;
> +  strcpy(bcc_ptr, utsname()->release);
> +  bcc_ptr += strlen(utsname()->release) + 1;
>    strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
>    bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
>   }
> @@ -2346,7 +2346,7 @@ CIFSSpnegoSessSetup(unsigned int xid, st
>        32, nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bytes_returned =
> -      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release, 32,
> +      cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release, 32,
>        nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bcc_ptr += 2;
> @@ -2371,8 +2371,8 @@ CIFSSpnegoSessSetup(unsigned int xid, st
>    }
>    strcpy(bcc_ptr, "Linux version ");
>    bcc_ptr += strlen("Linux version ");
> -  strcpy(bcc_ptr, system_utsname.release);
> -  bcc_ptr += strlen(system_utsname.release) + 1;
```

```
> +  strcpy(bcc_ptr, utsname()->release);
> +  bcc_ptr += strlen(utsname()->release) + 1;
>    strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
>    bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
>   }
> @@ -2622,7 +2622,7 @@ CIFSNTLMSSPNegotiateSessSetup(unsigned i
>        32, nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bytes_returned =
> -      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release, 32,
> +       cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release, 32,
>        nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bcc_ptr += 2; /* null terminate Linux version */
> @@ -2639,8 +2639,8 @@ CIFSNTLMSSPNegotiateSessSetup(unsigned i
>   } else {  /* ASCII */
>    strcpy(bcc_ptr, "Linux version ");
>    bcc_ptr += strlen("Linux version ");
> -  strcpy(bcc_ptr, system_utsname.release);
> -  bcc_ptr += strlen(system_utsname.release) + 1;
> +  strcpy(bcc_ptr, utsname()->release);
> +  bcc_ptr += strlen(utsname()->release) + 1;
>    strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
>    bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
>    bcc_ptr++; /* empty domain field */
> @@ -3001,7 +3001,7 @@ CIFSNTLMSSPAuthSessSetup(unsigned int xi
>        32, nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bytes_returned =
> -      cifs_strtoUCS((__le16 *) bcc_ptr, system_utsname.release, 32,
> +       cifs_strtoUCS((__le16 *) bcc_ptr, utsname()->release, 32,
>        nls_codepage);
>    bcc_ptr += 2 * bytes_returned;
>    bcc_ptr += 2; /* null term version string */
> @@ -3053,8 +3053,8 @@ CIFSNTLMSSPAuthSessSetup(unsigned int xi
>
>    strcpy(bcc_ptr, "Linux version ");
>    bcc_ptr += strlen("Linux version ");
> -  strcpy(bcc_ptr, system_utsname.release);
> -  bcc_ptr += strlen(system_utsname.release) + 1;
> +  strcpy(bcc_ptr, utsname()->release);
> +  bcc_ptr += strlen(utsname()->release) + 1;
>    strcpy(bcc_ptr, CIFS_NETWORK_OPSYS);
>    bcc_ptr += strlen(CIFS_NETWORK_OPSYS) + 1;
>    bcc_ptr++; /* null domain */
> diff --git a/fs/exec.c b/fs/exec.c
> index 0291a68..d881479 100644
> --- a/fs/exec.c
```

```
> +++ b/fs/exec.c
> @@ -1347,7 +1347,7 @@ static void format_corename(char *corena
>     case 'h':
>      down_read(&uts_sem);
>      rc = snprintf(out_ptr, out_end - out_ptr,
> -           "%s", system_utsname.nodename);
> +           "%s", utsname()->nodename);
>      up_read(&uts_sem);
>      if (rc > out_end - out_ptr)
>       goto out;
> diff --git a/fs/lockd/clntproc.c b/fs/lockd/clntproc.c
> index f96e381..915e596 100644
> --- a/fs/lockd/clntproc.c
> +++ b/fs/lockd/clntproc.c
> @@ -130,11 +130,11 @@ static void nlmclnt_setlockargs(struct n
>   nlmclnt_next_cookie(&argp->cookie);
>   argp->state   = nsm_local_state;
>   memcpy(&lock->fh, NFS_FH(fl->fl_file->f_dentry->d_inode), sizeof(struct nfs_fh));
> - lock->caller  = system_utsname.nodename;
> + lock->caller  = utsname()->nodename;
>   lock->oh.data = req->a_owner;
>   lock->oh.len  = snprintf(req->a_owner, sizeof(req->a_owner), "%u@%s",
>      (unsigned int)fl->fl_u.nfs_fl.owner->pid,
> -    system_utsname.nodename);
> +    utsname()->nodename);
>   lock->svid = fl->fl_u.nfs_fl.owner->pid;
>   lock->fl.fl_start = fl->fl_start;
>   lock->fl.fl_end = fl->fl_end;
> diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
> index 3fc683f..547aaa3 100644
> --- a/fs/lockd/mon.c
> +++ b/fs/lockd/mon.c
> @@ -152,7 +152,7 @@ xdr_encode_common(struct rpc_rqst *rqstp
>    */
>   sprintf(buffer, "%u.%u.%u.%u", NIPQUAD(argp->addr));
>   if (!(p = xdr_encode_string(p, buffer))
> -  || !(p = xdr_encode_string(p, system_utsname.nodename)))
> +  || !(p = xdr_encode_string(p, utsname()->nodename)))
>    return ERR_PTR(-EIO);
>   *p++ = htonl(argp->prog);
>   *p++ = htonl(argp->vers);
> diff --git a/fs/lockd/svclock.c b/fs/lockd/svclock.c
> index d2b66ba..61b4791 100644
> --- a/fs/lockd/svclock.c
> +++ b/fs/lockd/svclock.c
> @@ -326,7 +326,7 @@ static int nlmsvc_setgrantargs(struct nl
>  {
>   locks_copy_lock(&call->a_args.lock.fl, &lock->fl);
```

>   memcpy(&call->a_args.lock.fh, &lock->fh, sizeof(call->a_args.lock.fh));
> - call->a_args.lock.caller = system_utsname.nodename;
> + call->a_args.lock.caller = utsname()->nodename;
>   call->a_args.lock.oh.len = lock->oh.len;
>
>   /* set default data area */
> diff --git a/fs/lockd/xdr.c b/fs/lockd/xdr.c
> index f22a376..4eec051 100644
> --- a/fs/lockd/xdr.c
> +++ b/fs/lockd/xdr.c
> @@ -516,7 +516,7 @@ nlmclt_decode_res(struct rpc_rqst *req,
>   */
> #define NLM_void_sz 0
> #define NLM_cookie_sz 1+XDR_QUADLEN(NLM_MAXCOOKIELEN)
> -#define NLM_caller_sz 1+XDR_QUADLEN(sizeof(system_utsname.nodename))
> +#define NLM_caller_sz 1+XDR_QUADLEN(sizeof(utsname()->nodename))
> #define NLM_netobj_sz 1+XDR_QUADLEN(XDR_MAX_NETOBJ)
> /* #define NLM_owner_sz 1+XDR_QUADLEN(NLM_MAXOWNER) */
> #define NLM_fhandle_sz 1+XDR_QUADLEN(NFS2_FHSIZE)
> diff --git a/fs/nfs/nfsroot.c b/fs/nfs/nfsroot.c
> index c0a754e..1d656a6 100644
> --- a/fs/nfs/nfsroot.c
> +++ b/fs/nfs/nfsroot.c
> @@ -312,7 +312,7 @@ static int __init root_nfs_name(char *na
>   /* Override them by options set on kernel command-line */
>   root_nfs_parse(name, buf);
>
> - cp = system_utsname.nodename;
> + cp = utsname()->nodename;
>   if (strlen(buf) + strlen(cp) > NFS_MAXPATHLEN) {
>   printk(KERN_ERR "Root-NFS: Pathname for remote directory too long.\n");
>   return -1;
> diff --git a/include/linux/lockd/lockd.h b/include/linux/lockd/lockd.h
> index 995f89d..ac15b87 100644
> --- a/include/linux/lockd/lockd.h
> +++ b/include/linux/lockd/lockd.h
> @@ -80,7 +80,7 @@ struct nlm_wait;
> /*
>   * Memory chunk for NLM client RPC request.
>   */
> -#define NLMCLNT_OHSIZE (sizeof(system_utsname.nodename)+10)
> +#define NLMCLNT_OHSIZE (sizeof(utsname()->nodename)+10)
> struct nlm_rqst {
>   unsigned int a_flags; /* initial RPC task flags */
>   struct nlm_host * a_host; /* host handle */
> diff --git a/kernel/sys.c b/kernel/sys.c
> index 0b6ec0e..bcaa48e 100644
> --- a/kernel/sys.c

```
> +++ b/kernel/sys.c
> @@ -1671,7 +1671,7 @@ asmlinkage long sys_newuname(struct new_
>   int errno = 0;
>
>   down_read(&uts_sem);
> - if (copy_to_user(name,&system_utsname,sizeof *name))
> + if (copy_to_user(name,utsname(),sizeof *name))
>    errno = -EFAULT;
>   up_read(&uts_sem);
>   return errno;
> @@ -1689,8 +1689,8 @@ asmlinkage long sys_sethostname(char __u
>   down_write(&uts_sem);
>   errno = -EFAULT;
>   if (!copy_from_user(tmp, name, len)) {
> -  memcpy(system_utsname.nodename, tmp, len);
> -  system_utsname.nodename[len] = 0;
> +  memcpy(utsname()->nodename, tmp, len);
> +  utsname()->nodename[len] = 0;
>    errno = 0;
>   }
>   up_write(&uts_sem);
> @@ -1706,11 +1706,11 @@ asmlinkage long sys_gethostname(char __u
>   if (len < 0)
>    return -EINVAL;
>   down_read(&uts_sem);
> - i = 1 + strlen(system_utsname.nodename);
> + i = 1 + strlen(utsname()->nodename);
>   if (i > len)
>    i = len;
>   errno = 0;
> - if (copy_to_user(name, system_utsname.nodename, i))
> + if (copy_to_user(name, utsname()->nodename, i))
>    errno = -EFAULT;
>   up_read(&uts_sem);
>   return errno;
> @@ -1735,8 +1735,8 @@ asmlinkage long sys_setdomainname(char _
>   down_write(&uts_sem);
>   errno = -EFAULT;
>   if (!copy_from_user(tmp, name, len)) {
> -  memcpy(system_utsname.domainname, tmp, len);
> -  system_utsname.domainname[len] = 0;
> +  memcpy(utsname()->domainname, tmp, len);
> +  utsname()->domainname[len] = 0;
>    errno = 0;
>   }
>   up_write(&uts_sem);
```

Subject: Re: [RFC][PATCH 0/5] uts namespaces: Introduction
Posted by dev on Tue, 11 Apr 2006 12:27:45 GMT
View Forum Message <> Reply to Message

Serge,

> This patchset is based on Kirill Korotaev's Mar 24 submission, taking
> comments (in particular from James Morris and Eric Biederman) into
> account.
thanks a lot for doing this!

> Some performance results are attached. I was mainly curious whether
> it would be worth putting the task_struct->uts_ns pointer inside
> a #ifdef CONFIG_UTS_NS. The result show that leaving it in when
> CONFIG_UTS_NS=n has negligable performance impact, so that is the
> approach this patch takes.
Serge, your testing approach looks really strange for me.
First of all, you selected the worst namespace to check performance
overhead on.
1) uts_ns is rarely used and never used on hot paths,
2) also all these test suites below doesn't test the code paths you
modified.

So I wonder what was the goal of these tests, especially dbench?!

Thanks,
Kirill

>
> -serge
>
> Performance testing was done on a 2-cpu hyperthreaded
> x86 box with 16G ram. The following tests were run:
> dbench (20 times, four clients, on reiser fs non-isolated partition)
> tbench (20 times, 5 connections)
> kernbench (20 times)
> reaim (20 times ranging from 1 to 15 users)
>
> They were run on 2.6.17-rc1:
> pristine
> patched, but with !CONFIG_UTS_NS ("disabled")
> patched with CONFIG_UTS_NS=y ("enabled")
>
> All results are presented as means +/- 95% confidence interval.
>
> Dbench results:
> pristine: 387.080727 +/- 9.344585
> patched disabled: 389.524364 +/- 9.574921
> patched enabled: 370.155600 +/- 30.127808

>
> Tbench results:
> pristine: 388.940100 +/- 18.095104
> patched disabled: 389.173700 +/- 23.658035
> patched enabled: 394.333200 +/- 25.813393
>
> Kernbench results:
> pristine: 70.317500 +/- 0.210833
> patched, disabled: 70.860000 +/- 0.179292
> patched, enabled: 70.346500 +/- 0.184784
>
> Reaim results:
> pristine:
>   Nclients   Mean   95% CI
>   1   106080.000000 11327.896029
>   3   236057.142000 18205.544810
>   5   247867.136000 23536.800062
>   7   265370.000000 21284.335743
>   9   262969.936000 18225.497529
>   11   278256.000000 6230.342816
>   13   284288.016000 8924.589388
>   15   286987.170000 7881.034658
>
> patched, disabled:
>   Nclients   Mean   95% CI
>   1   105400.000000 8739.978241
>   3   229500.000000 0.000000
>   5   252325.176667 16685.663423
>   7   265125.000000 6747.777319
>   9   271258.645000 11715.635212
>   11   280662.608333 7775.229351
>   13   277719.706667 8173.390359
>   15   278515.421667 10963.211450
>
> patched, enabled:
>   Nclients   Mean   95% CI
>   1   102000.000000 0.000000
>   3   224400.000000 14159.870036
>   5   242963.288000 40529.490781
>   7   255150.000000 8745.802081
>   9   270154.284000 8918.863136
>   11   283134.260000 12239.361252
>   13   288497.540000 11336.550964
>   15   280022.728000 8804.882369
>
>

Subject: Re: [RFC][PATCH 2/5] uts namespaces: Switch to using uts namespaces
Posted by Sam Vilain on Tue, 11 Apr 2006 21:04:06 GMT
View Forum Message <> Reply to Message

Kirill Korotaev wrote:

>Serge,
>
>BTW, have you noticed that NFS is using utsname for internal processes
>and in general case this makes NFS ns to be coupled with uts ns?
>
>

Either that, or each NFS vfsmount has a uts_ns pointer.

Sam.