Subject: Re: Problem starting VPS
Posted by dev on Tue, 28 Feb 2006 14:12:49 GMT
View Forum Message <> Reply to Message

Can you try this debug patch please?

```
--- ./kernel/capability.c.capdbg 2006-02-28 16:09:25.000000000 +0300
+++ ./kernel/capability.c 2006-02-28 17:04:29.000000000 +0300
@@ -192,8 +192,12 @@ asmlinkage long sys_capset(cap_user_head
      if (get_user(pid, &header->pid))
       return -EFAULT;

-     if (pid && pid != virt_pid(current) && !capable(CAP_SETPCAP))
+     if (pid && pid != virt_pid(current) && !capable(CAP_SETPCAP)) {
+      printk("capset: (%s): pid = %d, vpid = %d, caps=%08lx\n",
+        current->comm, pid, virt_pid(current),
+        current->cap_effective);
            return -EPERM;
+     }

      if (copy_from_user(&effective, &data->effective, sizeof(effective)) ||
    copy_from_user(&inheritable, &data->inheritable, sizeof(inheritable)) ||
@@ -235,5 +239,9 @@ out:
      read_unlock(&tasklist_lock);
      spin_unlock(&task_capability_lock);

+     if (ret == -EPERM) {
+      printk("sys_capset: %s:%d, pid=%d\n",
+        current->comm, current->pid, pid);
+     }
      return ret;
 }
--- ./security/commoncap.c.capdbg 2006-02-28 16:09:25.000000000 +0300
+++ ./security/commoncap.c 2006-02-28 16:59:39.000000000 +0300
@@ -78,27 +78,49 @@ int cap_capget (struct task_struct *targ
 int cap_capset_check (struct task_struct *target, kernel_cap_t *effective,
         kernel_cap_t *inheritable, kernel_cap_t *permitted)
 {
+ int ret;
+
   /* Derived from kernel/capability.c:sys_capset. */
   /* verify restrictions on target's new Inheritable set */
   if (!cap_issubset (*inheritable,
        cap_combine (target->cap_inheritable,
      current->cap_permitted))) {
-  return -EPERM;
+  ret = 1;
```

```
+  goto oops;
  }

  /* verify restrictions on target's new Permitted set */
  if (!cap_issubset (*permitted,
       cap_combine (target->cap_permitted,
      current->cap_permitted))) {
-  return -EPERM;
+  ret = 2;
+  goto oops;
  }

  /* verify the _new_Effective_ is a subset of the _new_Permitted_ */
  if (!cap_issubset (*effective, *permitted)) {
-  return -EPERM;
+  ret = 3;
+  goto oops;
  }

  return 0;
+
+oops:
+ printk("cap_capset_check: %d (%s:%d), curcaps=%08lx:%08lx:%08lx, target (%s:%d)
caps=%08lx:%08lx:%08lx, set %08lx:%08lx:%08lx\n",
+    ret,
+    current->comm, current->pid,
+    current->cap_effective,
+    current->cap_permitted,
+    current->cap_inheritable,
+    target->comm, target->pid,
+    target->cap_effective,
+    target->cap_permitted,
+    target->cap_inheritable,
+    *effective,
+    *permitted,
+    *inheritable
+        );
+ return -EPERM;
 }

 void cap_capset_set (struct task_struct *target, kernel_cap_t *effective,
--- ./security/dummy.c.capdbg 2006-01-03 06:21:10.000000000 +0300
+++ ./security/dummy.c 2006-02-28 17:00:37.000000000 +0300
@@ -56,6 +56,7 @@ static int dummy_capset_check (struct ta
           kernel_cap_t * inheritable,
           kernel_cap_t * permitted)
 {
+ printk("dummy_capset_check: %s:%d\n", current->comm, current->pid);
```

```
  return -EPERM;
 }

--- ./security/selinux/hooks.c.capdbg 2006-02-28 16:09:25.000000000 +0300
+++ ./security/selinux/hooks.c 2006-02-28 17:02:41.000000000 +0300
@@ -1314,9 +1314,15 @@ static int selinux_capset_check(struct t

   error = secondary_ops->capset_check(target, effective, inheritable, permitted);
   if (error)
-   return error;
+   goto err;
+
+ error = task_has_perm(current, target, PROCESS__SETCAP);
+ if (error)
+   goto err;

- return task_has_perm(current, target, PROCESS__SETCAP);
+ return 0;
+err:
+ printk("selinux_capset_check: %s:%d err=%d\n", current->comm, current->pid, error);
 }

 static void selinux_capset_set(struct task_struct *target, kernel_cap_t *effective,
```

Also I noted, that you have following config options, different from our config and potentially conflicting with virtualization:
CONFIG_SECURITY=y
CONFIG_AUDIT=y
please turn it off.

Maybe you can check the whole kernel with OVZ config?