Subject: Re: [RFC] Virtualization steps
Posted by dev on Tue, 28 Mar 2006 09:02:08 GMT
View Forum Message <> Reply to Message

> Oh, after you come to an agreement and start posting patches, can you
> also outline why we want this in the kernel (what it does that low
> level virtualization doesn't, etc, etc), and how and why you've agreed
> to implement it. Basically, some background and a summary of your
> discussions for those who can't follow everything. Or is that a faq
> item?
Nick, will be glad to shed some light on it.

First of all, what it does which low level virtualization can't:
- it allows to run 100 containers on 1GB RAM
  (it is called containers, VE - Virtual Environments,
   VPS - Virtual Private Servers).
- it has no much overhead (<1-2%), which is unavoidable with hardware
  virtualization. For example, Xen has >20% overhead on disk I/O.
- it allows to create/deploy VE in less than a minute, VE start/stop
  takes ~1-2 seconds.
- it allows to dynamically change all resource limits/configurations.
  In OpenVZ it is even possible to add/remove virtual CPUs to/from VE.
  It is possible to increase/descrease memory limits on the fly etc.
- it has much more efficient memory usage with single template file
  in a cache if COW-like filesystem is used for VE templates.
- it allows you to access VE files from host easily if needed.
  This helps to make management much more flexible, e.g. you can
  upgrade/repair/fix all you VEs from host, i.e. easy mass management.

OS kernel virtualization
~~~~~~~~~~~~~~~~~~~~~~~~~
OS virtualization is a kernel solution, which replaces the usage
of many global variables with context-dependant counterparts. This
allows to have isolated private resources in different contexts.

So VE means essentially context and a set of it's variables/settings,
which include but not limited to, own process tree, files, IPC
resources, IP routing, network devices and such.

Full virtualization solution consists of:
- virtualization of resources, i.e. private contexts
- resource controls, for limiting contexts
- management tools

Such kind of virtualization solution is implemented in OpenVZ
(http://openvz.org) and Linux-Vserver (http://linux-vserver.org) projects.
~~~~~~~~~~~~~~~~~~~~~~~~~

Summary of previous discussions on LKML
~~~~~~
- we agreed upon doing virtualization of each kernel subsystem
  separately, not as a single virtual environment.
- we almost agreed upon calling virtualization of subsystems
  "namespaces".
- we were discussing whether we should have global namespace context,
  like 'current' or bypass context as an argument to all functions
  which require it.
- we didn't agreed on whether we need a config option and ability to
  compile kernel w/o virtual namespaces.

Thansk,
Kirill
~~~~~~