OpenVZ: Operating System-level Virtualization

Απόλλων Οικονομόπουλος
apollon@noc.grnet.gr

Τεχνικές παρουσιάσεις HELLUG - Software Libre Society Πανεπιστημίου Πειραιά

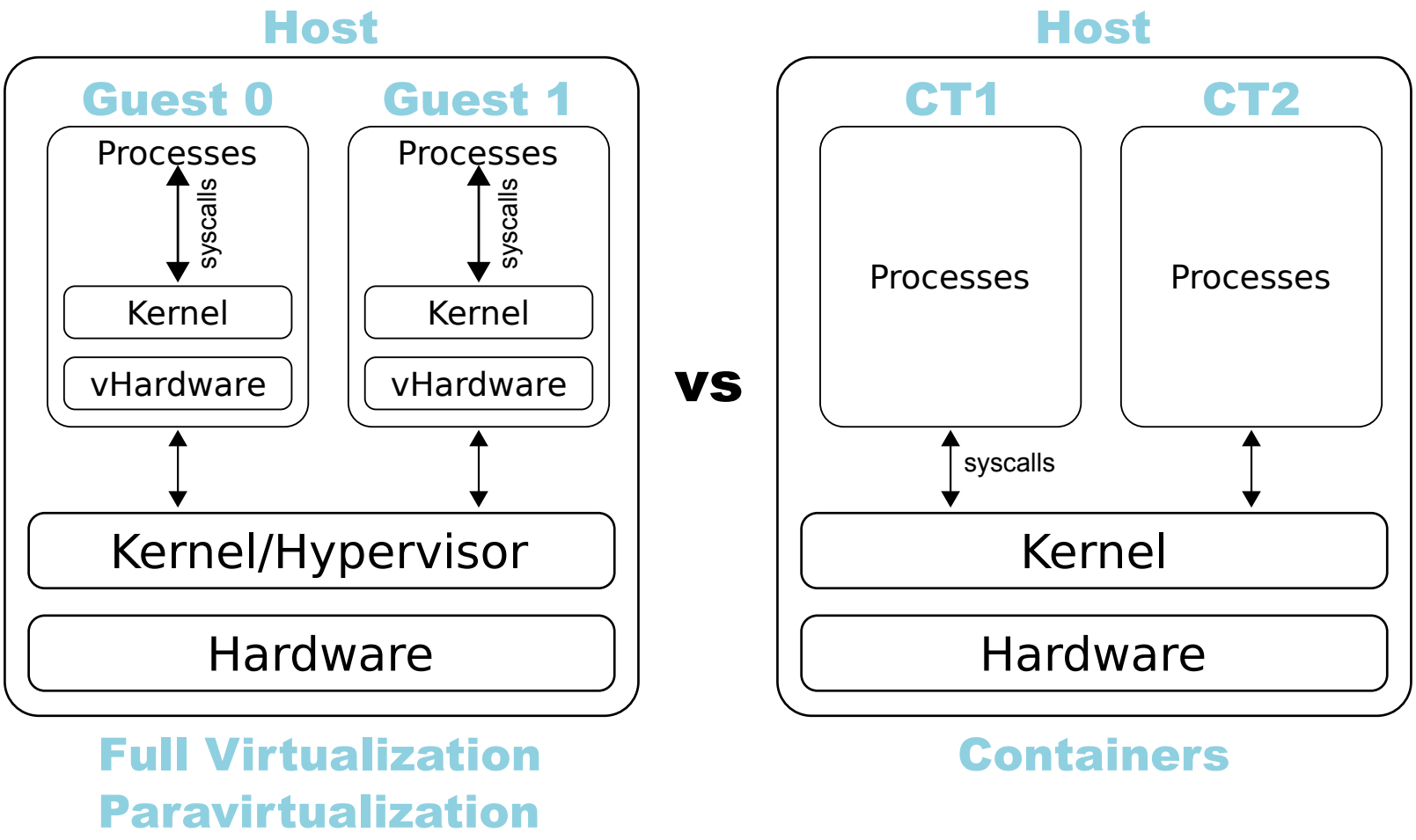
Περιεχόμενα

Virtualization

Η κατάτμηση ενός συστήματος σε περισσότερα απομονωμένα «εικονικά» συστήματα που μοιράζονται τους ίδιους πόρους.

- Full Virtualization: abstraction στο επίπεδο του HW
- Paravirtualization: Full Virtualization + εναλλακτικά data-paths με μικρότερο overhead
- Containers: abstraction στο επίπεδο του kernel

Containers vs Full Virtualization



chroot(2): ο παππούς των containers

- UNIX V7, 1979 και Bill Joy, 1982, πριν το 4.2BSD
- Αλλάζει το root directory ενός process και όλων των παιδιών του
- Αρχική χρήση: δοκιμές του V7 και του 4.2BSD χωρίς εγκατάσταση
- Χρησιμοποιήθηκε κατά κόρον ως μηχανισμός ασφαλείας, αν και παρακάμπτεται

Namespaces

- Βασική ιδέα: γενίκευση του chroot(2) και σε άλλα υποσυστήματα πέρα από το filesystem
- PID namespaces
- UID namespaces
- Network namespaces
- IPC namespaces
- UTSNAME namespaces

Namespaces: kernel implementation

include/linux/sched.h:

```
#include <linux/nsproxy.h>

struct task_struct {
   ...
   struct nsproxy *nsproxy;
   ...
```

include/linux/nsproxy.h:

```
struct nsproxy {
        atomic_t count;
        struct uts_namespace *uts_ns;
        struct ipc_namespace *ipc_ns;
        struct mnt_namespace *mnt_ns;
        struct pid_namespace *pid_ns;
        struct net           *net_ns;
};
```

OpenVZ

- Ελεύθερη έκδοση του Virtuozzo
- Patch στον επίσημο πυρήνα Linux

```
$ git diff --shortstat v2.6.32.12..v2.6.32-avdeyev
 597 files changed, 59043 insertions(+), 1879 deletions(-)

$ git diff --shortstat v2.6.31..v2.6.32
 10308 files changed, 1091675 insertions(+), 529314 deletions(-)
```

- Namespaces
- Resource accounting & control
- Userspace management tools
- Υποστήριξη: RHEL (OpenVZ kernels), Debian Lenny & Squeeze (Debian kernels)

OpenVZ: αρχιτεκτονική

- Ο host χωρίζεται σε containers (Virtual Environments - VEs)
- VE0: το privileged VE, στο οποίο ξεκινά η init του συστήματος κατά το boot. Η διαχείριση του συστήματος γίνεται μέσα από το VE0.
- Κάθε VE:
  - Έχει ένα VEID > 100 (32-bit)
  - Βλέπει ένα κομμάτι του filesystem, με 2-level quota
  - Έχει ελεγχόμενη πρόσβαση στους πόρους του συστήματος (user_beancounters)
  - Έχει δικό του hostname, δικούς του χρήστες και TCP/IP stack, iptables, ip6tables, NFS context
  - Έχει capabilities που περιορίζουν την αλληλεπίδραση με το υπόλοιπο σύστημα (π.χ. net_raw, sys_chroot, sys_admin, sys_boot, mknod, . . .)

OpenVZ: booting

1. Τα userspace εργαλεία, μέσω syscalls, δημιουργούν το κατάλληλο περιβάλλον (chroot, resource limits, κλπ)
2. Κάνουν `fork(2)` και `execve(2)` την νέα `init(8)`
3. Η `init(8)` βρίσκεται να τρέχει κανονικά με PID 1, κάτω από έναν kernel
4. Η διαδικασία προχωρά σαν την εκκίνηση ενός φυσικού μηχανήματος

OpenVZ: processes

- Τροποποιημένο `task_struct`:

  `include/linux/sched.h`:

```
struct task_struct {
...
#ifdef CONFIG_BEANCOUNTERS
        struct task_beancounter task_bc;
#endif
#ifdef CONFIG_VE
        struct ve_task_info ve_task_info;
#endif
...
}
```

- `ve_task_info`: VZ-specific πληροφορίες
- `task_beancounter`: resource accounting

OpenVZ: processes (2)

- Κάθε process ανήκει σε ακριβώς ένα container
- Όλα τα processes «φαίνονται» από το VE0
- Κάθε process έχει 2 PIDs: ένα global και ένα container-local
- Μεταξύ τους τα processes ενός container βλέπουν μόνο τα container-local PIDs

User beancounters: resource accounting

- `/proc/user_beancounters` ή `/proc/bc/<VEID>/resources`
- Resource accounting and limiting:

`kmemsize`	`lockedpages`	`privvmpages`
`shmpages`	`numproc`	`physpages`
`vmguarpages`	`oomguarpages`	`numtcpsock`
`numflock`	`numpty`	`numsiginfo`
`tcpsndbuf`	`tcprcvbuf`	`othersockbuf`
`dgramrcvbuf`	`numothersock`	`dcachesize`
`numfile`	`numiptent`	`diskspace`

- Tunable μέσω του vzctl, ακόμα και όταν το container «τρέχει». Οι αλλαγές ισχύουν άμεσα.

OpenVZ: υπόλοιπα resources

- Disk quota (υλοποιημένο στο `simfs`)
- CPU Fair scheduler: 2-level scheduler (VE + processes)
- I/O priorities: per-VE I/O priority (CFQ)

OpenVZ: Checkpointing

- Επιτρέπει το «πάγωμα» και την αποθήκευση του full process state ενός container
- Αποθηκεύονται όλα τα kernel objects (διεργασίες, δικτυακές συνδέσεις, IPC κλπ)
- Επιτρέπει migration των containers σε άλλο host
- Επιτρέπει λήψη «snapshots» του container
- Έλεγχος μέσω `ioctl(2)` στο `/proc/cpt` και το `/proc/rst`

OpenVZ: Userspace tools

- vzctl: Βασικός έλεγχος, δημιουργία, ρύθμιση, εκκίνηση, τερματισμός, ...
- vzmigrate: Live migration, με rsync των δεδομένων
- vzlist: Κατάλογος των VEs που είναι ορισμένα στο σύστημα
- vzpid: Αντίστοιχιση VE0 PID $\rightarrow$ VE PID
- vzcalc: Εμφάνιση στατιστικών για τους πόρους ενός VE
- vzsplit: Διαίρεση του physical host σε n VEs

Container templates

- Το 1ο container δημιουργείται χειροκίνητα (π.χ. debootstrap)
- Template: tarball με το standard tree
- Δημιουργία containers από template $\rightarrow$ ταχύτητα, ομοιομορφία
- Συνοδεύονται από scripts που αναλαμβάνουν την περαιτέρω παραμετροποίηση (hostname, IP address, ...)
- Θέλει προσοχή:
  - inittab: ~~tty's, gettys~~
  - ~~udev~~
  - Δημιουργία SSH host keypair στο πρώτο boot
  - fstab: μόνο `/proc`, `/dev/pts`

Παραδείγματα χρήσης 1: ακαδημαϊκό εργαστήριο

- Containers:
  - Mail server
  - File server
  - 2 Web servers
  - 1-15 containers για εργαστήρια φοιτητών
  - LDAP server
  - Φιλοξενούμενα projects foss.ntua
- Μερικά εκ μετατροπής (P2V)
- Πολλά με μικρή διάρκεια ζωής (π.χ. εργαστήρια/demos)
- Έχει χρησιμοποιηθεί live migration προκειμένου να γίνουν εργασίες στο hardware

Παραδείγματα χρήσης 2: production/staging server

- OpenVZ kernel ως domU σε Xen host (pv_ops)
- Containers:
  - Mail server
  - Web server
  - FTP server
  - DNS+LDAP server
  - Test/staging containers για υπηρεσίες
- Ομαλή λειτουργία κάτω από Xen
- Κατά καιρούς έχει φιλοξενήσει μέχρι και 20 containers ταυτόχρονα

LXC: LinuX Containers

- Containers στο mainline kernel
- Πολλά από τα χαρακτηριστικά του OpenVZ merged {PID, UTSNAME, IPC, UID}-namespaces
- Ακόμα λείπουν features
  - Resource control βασισμένο στα cgroups, όχι όσο feature-complete όσο το OpenVZ
- Υπό ανάπτυξη, λειτουργικό σε 2.6.29+
- Μεγάλη συνεισφορά από το OpenVZ και την IBM

Containers: συμπεράσματα

Συνιστώμενες χρήσεις:

- Linux-only servers
- Πιο ασφαλή chroots (π.χ. private shells)
- Development/build testbeds

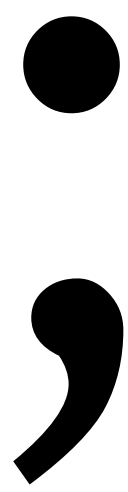
Ακατάλληλο για:

- Desktop virtualization
- Περιβάλλοντα με πολλαπλά Λ/Σ

Πηγές

- OpenVZ: http://wiki.openvz.org
- OpenVZ kernel git tree:
  git://git.openvz.org/pub/linux-2.6.32-openvz
- LXC: http://lxc.sf.net
- Debian: http://www.debian.org
- Proxmox VE
  (KVM & OpenVZ cluster management για Debian):
  http://www.proxmox.com
- Linux VServer: http://linux-vserver.org

Ευχαριστώ



Ερωτήσεις;