JOSY

Migrations Xen⟶KVM
Linux-Vserver⟶LXC

www.univ-nantes.fr

UNIVERSITÉ DE NANTES

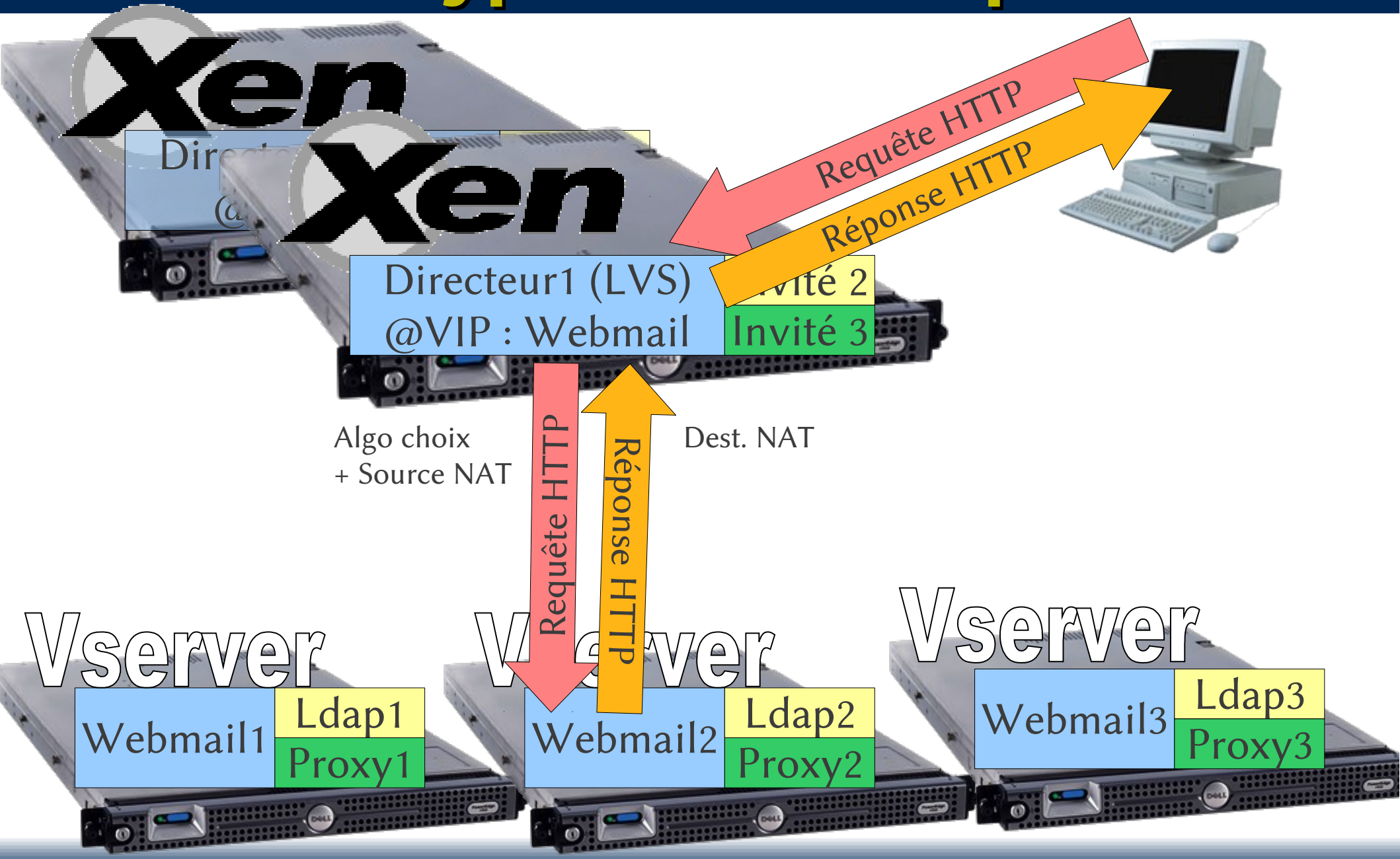
Jeudi 9 Juin 2011

Jean-Phippe Menil, Yann Dupont
Service IRTS, DSI Université de Nantes

Historique à Nantes

- 2002 : Début virtualisation avec Linux-Vserver
- 2003 : Virtualisation « massive »
- 2005 : Xen est utilisé pour la virtualisation lourde
- 2007 : Xen est progressivement remplacé par KVM
- 2010 : Linux-vserver est progressivement remplacé par LXC.

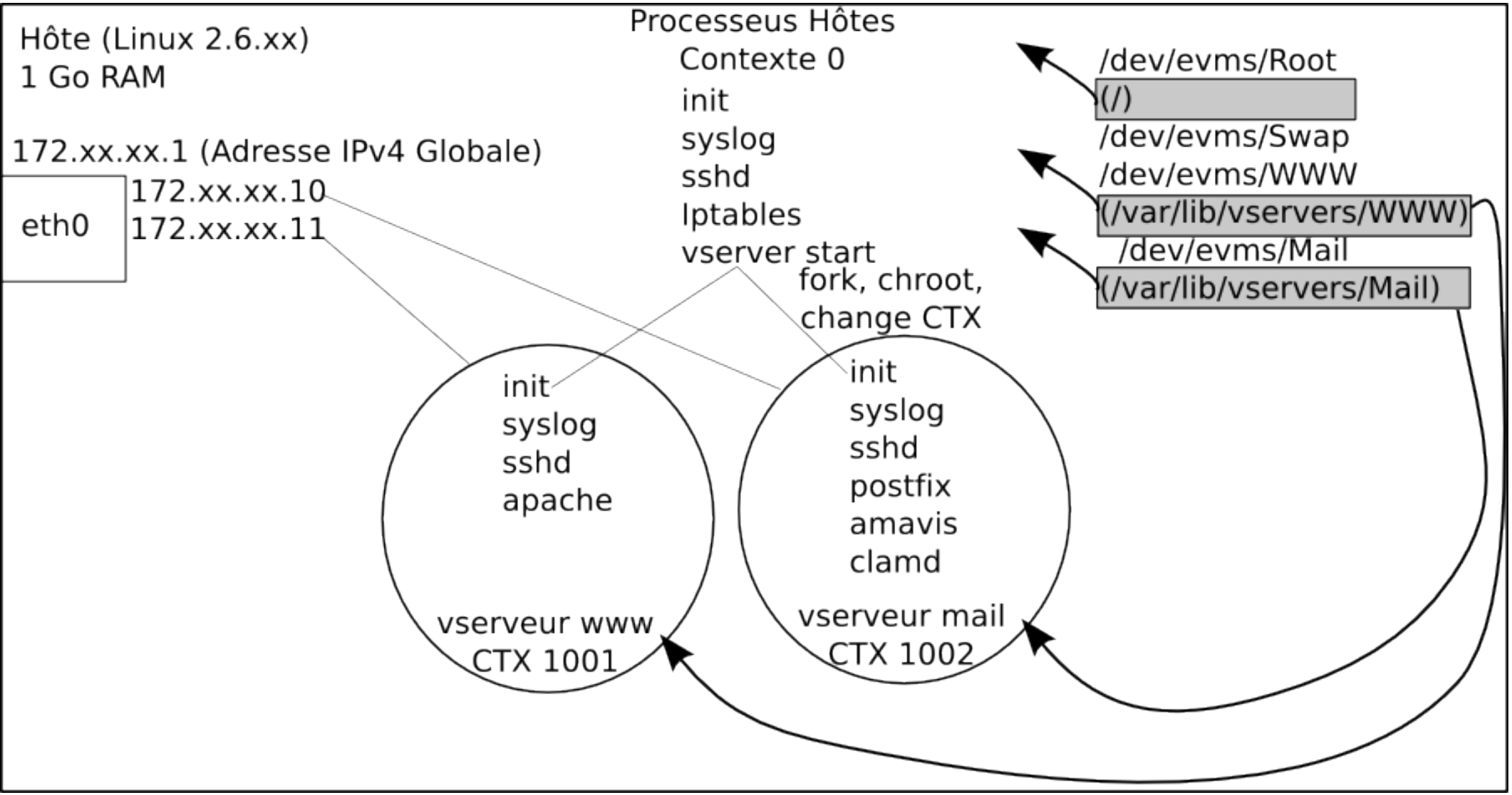
Utilisation type : Haute disponibilité



Linux-Vserver

- Virtualisation Légère, gérée directement par le noyau Linux
  (Ajout d'un contexte à chaque processus)
- Isolation des processus ayant un contexte différent, avec mutualisation des ressources
- Unique instance de noyau, et donc limitée à des invités Linux
- Les systèmes de fichiers sont un sous répertoire du maître
- (Quasi) pas de coût associé : imbattable en performance
- Virtualisation incomplète : ex : couche réseau
- Pas dans le kernel standard :
  nécessite patches et outils, (simples à appliquer)
  Support Debian jusqu'à Squeeze

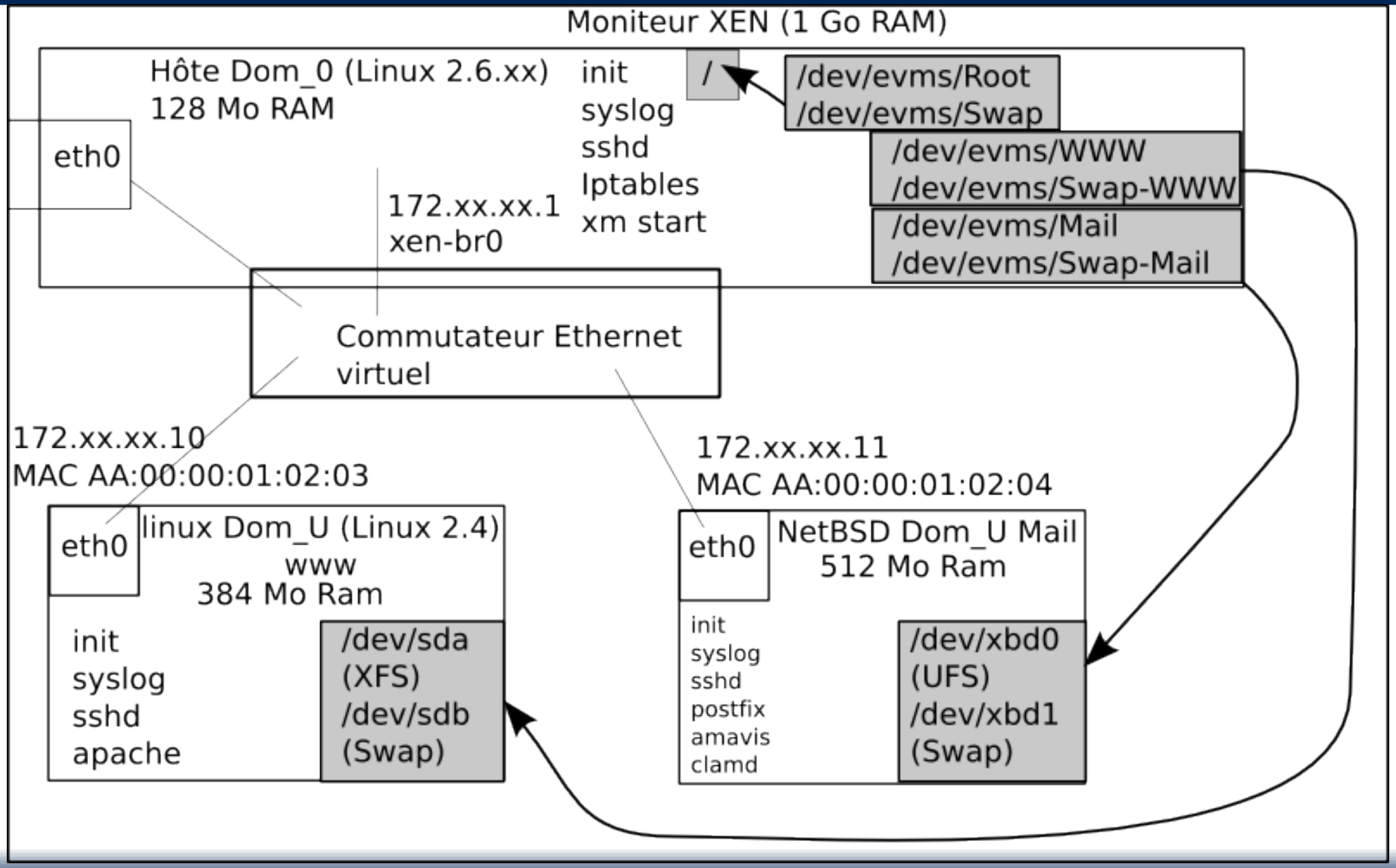
Linux-Vserver



XEN

- Virtualisation complète du système, gérée par un hyperviseur
- Isolation complète des ressources, sans mutualisation
- Si le CPU ne supporte pas les instructions « magiques », limitation à des invités Linux ou *BSD (2005!!!)
- Si le CPU est récent, ouverture à d'autres O.S.
- Pas dans le kernel standard (< 3.0) : nécessite patches et outils (kernel dom0, kernel domU, pas très simple en 2005...)
  Support Debian
- Coût associé à la virtualisation faible sur les calculs,
  plus important sur les entrées sorties
  en restant raisonnable (para-virtualisation)

XEN


Moniteur XEN (1 Go RAM)
Hôte Dom_0 (Linux 2.6.xx)
128 Mo RAM
init
syslog
sshd
Iptables
xm start
/
/dev/evms/Root
/dev/evms/Swap
/dev/evms/WWW
/dev/evms/Swap-WWW
/dev/evms/Mail
/dev/evms/Swap-Mail
eth0
172.xx.xx.1
xen-br0
Commutateur Ethernet
virtuel
172.xx.xx.10
MAC AA:00:00:01:02:03
172.xx.xx.11
MAC AA:00:00:01:02:04
eth0
linux Dom_U (Linux 2.4)
www
384 Mo Ram
init
syslog
sshd
apache
/dev/sda
(XFS)
/dev/sdb
(Swap)
eth0
NetBSD Dom_U Mail
512 Mo Ram
init
syslog
sshd
postfix
amavis
clamd
/dev/xbd0
(UFS)
/dev/xbd1
(Swap)

XEN → KVM

- Le savoir faire de Xen dans l'émulation des 17 instructions non natives a perdu de l'importance avec l'arrivée de la virtualisation native
- Gestion virtualisation « native » (Xen 3 HVM)
  plus compliquée que KVM à l'époque, 'usineàgazesque' :
  exemple avec un invité Windows
- Promesses de simplicité et meilleures performances avec KVM (virtio...)
- Noyau STANDARD, outils simples, connus et supportés dans la distribution
- Pertinence du dom-0 ? Linux fait un bon hyperviseur !
- Doutes sur la pérennité : Rachat par CITRIX, version opensource parent pauvre ?

KVM

- Nécessite les extensions de virtualisation des processeurs
- Virtualisation complète, supporte tous les systèmes d'exploitation
- Les fonctionnalités sont dans les noyaux Linux standards
- Très petit : Linux EST l'hyperviseur
  Partie utilisateur basée sur qemu
  1 ligne de commande suffit pour lancer une machine virtuelle
- Pilotes 'virtio' natifs pour les instances Linux, existent pour Windows
- Évolue régulièrement, rapidement (trop?)
- GPL, supporté par RedHat et autres sociétés

Conteneurs et LXC

- Depuis Linux 2.6.29, gestion générique des conteneurs dans le noyau Linux
- LXC est « juste » un ensemble d'outils autour de cette fonctionnalité du kernel
- Linux-Vserver *pourrait* utiliser les conteneurs dans un futur *éventuel* – Les fonctionnalités ne sont pas 100% identiques
- Travail supporté par IBM (entre autres)

Linux-Vserver vs LXC (1)

	Linux-Vserver	*LXC*
Intégré Noyau	NON. Nécessite un patch	OUI.
Suivi des versions du noyau	NON. Les patches ne s'appliquent qu'à des versions spécifiques du noyau. Des portages 'NON officiels' existent parfois.	OUI, du fait de l'intégration
Virtualisation complète	NON.	NON.
Virtualisation Réseau	NON. Pas de carte réseau, alias sur interface maître avec masquage des autres adresses. Problème avec localhost. Pas de filtrage possible entre les invités.	Quasi complète. Plusieurs façon d'implanter les interfaces virtuelles (comme KVM).
IPv6	OUI MAIS avec des limitations selon les versions.	OUI, car virtualisation de la couche réseau.
Routage avancé, multi vlan, multi homing	Compliqué. Tables de routages spécifiques à chaque invité et règles de routages (ip rule) à produire. Les paquets ne sortent pas toujours par la bonne interface...	Simple. Chaque invité est indépendant. (Simple connexions de bridges et interfaces virtuelles). Routages, IPVS, FW possibles.
Gestion outils tiers	NON.	OUI MAIS: Libvirt (supporte conteneurs, pas les outils LXC).
Support distributions	NON sauf Debian (mais squeeze dernière supportée)	OUI sur les distributions récentes.

Linux-Vserver vs LXC (2)

	Linux-Vserver	*LXC*
Sécurisation de /proc	OUI.	**NON !** Un utilisateur peut facilement éteindre une machine maître !!
Virtualisation de l'utilisation mémoire, de l'uptime	OUI.	NON. Les données du serveur maître sont retournées.
Maturité des outils	BONNE.	Faible. Les scripts par défaut ne permettent pas un arrêt correct des invités. (contournable). Parfois les interfaces réseau ne sont pas libérées à l'arrêt d'une machine.
Support kernel log	Désactivé, ne pose pas de soucis. Syslog ne gère que la partie utilisateur.	**OUI, mais incomplet !** Dans le cas d'iptables, les logs se retrouvent mutualisés entre maîtres et invités. Cause des soucis de duplications avec syslog.
Qualité de la documentation	Moyenne. Confuse, pas toujours à jour, et/ou contradictoire.	Pire ! **Très peu de documentation. (Jeunesse de la solution?)**
Pénalité à l'exécution	Quasi nulle.	Quasi nulle, mais passage dans bridge Linux et Tun/TAP

LXC et cgroups

- Repose sur cgroups (Control Groups) pour limiter les permissions et les ressources des invités
- cgroups est une infrastructure générique du noyau (ne sert pas qu'à LXC)
- But : Grouper des processus ensemble ET leur appliquer des règles :
  - Restrictions sur ressources
    - Peut restreindre la mémoire, les CPU assignés, les périphériques accédés, activer les débugs...
  - Accounting

- Le pseudo système de fichiers cgroup doit être monté
  (exemple de /etc/fstab) :
  cgroup /cgroup cgroup defaults 0 0
  - Il est manipulable facilement (ex : echo 1,2,3,4> /cgroup/01-dns1-lmb/cpuset.cpus)
- Chaque invité LXC démarré se voit assigner un groupe spécifique (portant le nom de l'invité)

Création d'invités LXC

- Démarrer via les scripts fournis par les outils lxc
  - Sur Debian, repose sur debootstrap
  - (Une fois finalisée et configurée, la racine de l'invité pourra être sauvegardée en tar.gz pour faire un patron rapide à déployer... Comme avec linux-vserver)
- Créer le fichier de configuration de l'invité

Configuration de LXC(1)

Fichier /var/lib/lxc/01-dns1-lmb/config

```
lxc.utsname = dns1-lmb


lxc.tty = 4
lxc.pts = 1024
lxc.rootfs = /var/lib/lxc/01-dns1-lmb/rootfs


## Restriction des droits de l'invité
lxc.cap.drop = audit_control audit_write fsetid ipc_lock ipc_owner lease linux_immutable
mac_admin mac_override mac_admin mknod setfcap setpcap sys_admin sys_boot
sys_module sys_nice sys_pacct sys_ptrace sys_rawio sys_resource sys_time sys_tty_config
```

Racine de l'invité (Sous-répertoire).
On l'a mise sur un même volume
que la configuration.

Par défaut , tout est
autorisé.

Mknod : création de devices
Doit être restreint !!!

Configuration de LXC(2)

- Configuration de l'interface cgroups
  - Lxc.cgroup correspond, sur le maître, au sous répertoire de /cgroup correspondant à l'invité, ici, /cgroup/01-dns1-lmb/

```
lxc.cgroup.devices.deny = a
# /dev/null et zero
lxc.cgroup.devices.allow = c 1:3 rwm
lxc.cgroup.devices.allow = c 1:5 rwm
# consoles
lxc.cgroup.devices.allow = c 5:1 rwm
lxc.cgroup.devices.allow = c 5:0 rwm
lxc.cgroup.devices.allow = c 4:0 rwm
lxc.cgroup.devices.allow = c 4:1 rwm
# /dev/{,u}random
lxc.cgroup.devices.allow = c 1:9 rwm
lxc.cgroup.devices.allow = c 1:8 rwm
lxc.cgroup.devices.allow = c 136:* rwm
lxc.cgroup.devices.allow = c 5:2 rwm
# rtc
lxc.cgroup.devices.allow = c 254:0 rwm
```

RAPPEL : on peut configurer cgroups pour limiter la mémoire d'un invité, son utilisation CPU, prioriser les I/O, par exemple.

Configuration de LXC(3)

- points de montages

```
lxc.mount.entry=proc /var/lib/lxc/01-dns1-lmb/rootfs/proc proc nodev,noexec,nosuid 0 0
lxc.mount.entry=devpts /var/lib/lxc/01-dns1-lmb/rootfs/dev/pts devpts defaults 0 0
lxc.mount.entry=sysfs /var/lib/lxc/01-dns1-lmb/rootfs/sys sysfs defaults  0 0
lxc.mount.entry=tmpfs /var/lib/lxc/01-dns1-lmb/rootfs/dev/shm tmpfs defaults 0 0
```

- Alternativement, on peut déclarer un fichier fstab spécifique

Configuration de LXC(4)

```
#### le reseau
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = V2003
lxc.network.name = eth0
lxc.network.mtu = 1500
lxc.network.hwaddr = BA:BE:BA:BE:01:01
lxc.network.veth.pair = dns1-lmb-clus
```

Le type d'interface réseau choisie peut être phys, vlan... pas de NAT
ICI Veth, interface TUN/TAP

Le nom du bridge sur lequel va être connecté l'interface créée
Il doit exister préalablement

Si non spécifiée, allocation dynamique de l'adresse MAC

Nom symbolique de l'interface

Ici, on laisse l'invité positionner lui même ses adresses IP
(on pourrait la déclarer ici, mais pas spécifier de route)

Configuration préalable du réseau

- Exemple /etc/network/interfaces du maître :

```
auto V2003
iface V2003 inet manual
        bridge_ports eth2.2003
        bridge_stp off
        bridge_maxwait 0
```

Tag 802.1Q

- Résultat après lancement des invités :

brctl show

```
bridge name  bridge id       STP enabled   interfaces
U06          8000.0024e86f3dfb  no     eth0.110
V2002        8000.0024e86f3e03  no
                                dns1-lmb-serv
                                eth2.2002
                                ldap1-lmb-serv

V2003        8000.0024e86f3e03  no
                                dns1-lmb-clus
                                eth2.2003
                                ldap1-lmb-clus
```

Script init à modifier.

- Chaque distribution a actuellement ses scripts d'initialisation
- Par défaut, sur debian, /etc/init.d/lxc stop arrête brutalement tous les processus internes au conteneur (!)
  - Problème au redémarrage des bases de données
  - Pas de corruption du système de fichiers, car celui ci est toujours monté sur le maître (différence avec KVM)
- Utilisation de scripts modifiés pour arrêter proprement les invités (basé sur inotify pour vérifier les processus encore en activité)

Conclusion

- LXC est très prometteur, mais encore un peu jeune
  - En particulier en ce qui concerne ses utilitaires
  - L'administrateur système doit se documenter et s'approprier la technologie
- LXC ne peut actuellement pas héberger des services où l'utilisateur a accès au shell, la sécurité est insuffisante
- La migration de Linux-Vserver vers LXC ne se fera qu'au fur et à mesure des progrès de LXC
  (juin 2011 : 31 invités LXC contre 284 vservers, 74 KVM)
- LXC est actuellement utilisé exclusivement pour les services à haute disponibilité ou dans des cas particuliers

Conclusion

La virtualisation par conteneur est appropriée dans la grande majorité des cas :
pas besoin de KVM pour héberger des invités Linux.

Merci de votre attention
Questions ?

Jeudi 9 Juin 2011
Jean-Phippe Menil, Yann Dupont
Service IRTS, DSI Université de Nantes