

Virtualizáció szabad szoftverekkel

Mátó Péter <mato.peter@andrews.hu>

Az előzmények

- 1960: IBM CP-40
- PC
 - Unix: chroot (4.2BSD: 1982), BSD: jail (FreeBSD: 2006)
 - VMWare (VMW WS: 1999, VMW GSX: 2001)
 - Windows Virtual PC, Solaris Zones
 - UML (User Mode Linux), QEmu, XEN, VirtualBox, KVM, OpenVZ

Mi indította el a hódítást?

- Egyre erősebb x86 gépek nagyon olcsón
- Általában nem lehet egy szolgáltatásra kihasználni egy gép erejét (max. 30% kihasználtság)
- A gépek sokat fogyasztanak, nagyon drága a hűtés
- Mostanában fontos lett, hogy legyen kicsi az áramszámla
- Gépeket vásárolni és összerakni macerás és unalmas

Mire használható?

- Tesztkörnyezetek kialakítása
 - fejlesztők, diákok
- Szerverek virtualizációja
 - kevesebb gép, kényelmesebb adminisztráció
- Speciális helyzetek kezelése
 - aszimmetrikus HA-k kialakítása
 - ha a tech támogatja, akkor spéci HA megoldások

Mire használható még?

- Nincs többé újratelepítés vascsere miatt
- Egyszerűbb a mentés, archiválás, katasztrófa helyreállítás
- Vas migráció elvégezhető online (live migration módszer használatával)
- A gépek erőforrásai nagyon rugalmasan változtathatók

Pénzügyi megtakarítások

- Csökken az áramszámla
- Csökken a hűtés miatti áramszámla
- Csökken a méret, kevesebb rack-et kell bérelni
- Növekszik az adminisztrátor hatékonysága, kevesebb emberrel is el lehet látni ugyanazt a feladatot
- Rugalmasabb, hibatűrőbb rendszer, kevesebb kiesés, kevesebb veszteség

A lehetséges problémák

- Ha a géphez hozzáférnek, akkor minden rajta futó virtuális szolgáltatáshoz is hozzáférnek
- Ha a host valamilyen hiba folytán leáll, akkor az egész rendszer halott, minden komponensével

Mire van szükségünk hozzá?

A technológiától és az igényeinktől függően:

- Egy erős gép, sok mag, sok RAM
- 64bites rendszer a memória használat miatt
- Virtualizációs támogatás a prociba
 - AMD-v, Intel VT-x
- Közös diszk alrendszer (NAS vagy SAN)

Néhány alapfogalom

- Host, Guest vagy VM
- hypervisor alapú virtualizáció
- Teljes virtualizáció – paravirtualizáció (részleges, a guest módosítása szükséges)
- OS szintű – alkalmazás szintű v.
- Desktop virtualizáció

A rendszer felépítése – a diszkek

- az adatbiztonság növelésére RAID
- a rugalmasság növelésére LVM
- A közös használathoz közös elérés kell:
 - FC hálózat, iSCSI, ATA over ethernet, DRBD
 - Spéci fájlrendszer, ami képes kezelni a párhuzamos hozzáférést (pl. GFS – Global Filesystem)
- Mindezek együtt: NAS, SAN (pl.:FreeNAS)

A rendszer felépítése – processzor

- Mindenképp törekedni kell rá, hogy egy virtuális host-ban legyen minél több proci mag
- Nem szerencsés annyi magot a VM-nek adni, amennyi van (host OS-nek is kell, párhuzamosság, várakozás a procik felszabadulására)
- Ma már olcsón kapható sokmagos rendszer (pl. Core i7 alapon felépített rendszer)

A rendszer felépítése – a memória

- Legyen benne nagyon sok
- Általában ez a szűk keresztmetszet
- Ha lehet, figyeljünk a DDR2 és DDR3 csatornák optimális kihasználására
- Nem szerver vasba jelenleg 12G, ritkán 24G tehető (de az utóbbit én csak úgy hiszem el, ha a kereskedő összerakja a vasat, mielőtt kifizetem)

A rendszer felépítése – a hálózat

- Legalább egy GB csatlakozó
- Ha iSCSI vagy AOE használt, akkor erre érdemes egy dedikált GB alhálót kiépíteni
- A hálózathoz a virtuális hálózatot úgy kell kialakítani, mint egy szerverterem hálózatát
- Vagy maga a host, vagy egy VM legyen a tűzfal a virtuális gépek felé (Fokozott biztonság szükséges!)

Paravirtualizáció

- Kissé kényelmetlenné teszi a rendszer használatát, mivel módosítani kell a guest oprendszerét
- Ennek azonban megvan az előnye is
 - virtualizációra kiélezett guest rendszerek (pl. JEOS)
 - speciális hardver meghajtók (pl. Linux VIRTIO, VirtualBox guest additions...), speciális lehetőségek
 - a sebesség jobb lehet, mint a teljes virtualizációnál

Zárt vagy szabad?

- Kétségtelen, hogy a zárt virtualizációs megoldások előrébb járnak
- A legfontosabb funkciókat azonban minden rendszer megfelelően, stabilan támogatja
- A szabadszoftver megoldások nem fogják a felhasználókat csapdába ejteni
- A legrosszabb, ami történhet, hogy egy zárttá váló kiadás esetén a közösség fork-olja a fejlesztést

A Qemu

- 2005-2008 Fabrice Bellard
- processzor emulátor
- x86-on virtualizál
- KQemu csak később lett GPL
- processzorok:
x86, x86_64, ARM, Sparc, PowerPC, MIPS...

A KVM rendszer

- Kernel-based Virtual Machine
- 2006. dec. 18. A KVM bekerül a 2.6.20-as kernelbe
- 2008. szept. 5. A Redhat megvásárolta a Qumranet

A KVM jellemzői I.

- stabil, kicsi és egyszerű
- szerveren remekül használható
- a Qemu-t userspace részét használta fel
- része az Ubunut Jaunty Jackalope rendszernek
- módosítás nélküli guest kernellel működik
- hardver támogatás szükséges
 - Intel: Intel VT támogatás (`grep vmx /proc/cpuinfo`)
 - AMD: AMD-V támogatás (`grep svm /proc/cpuinfo`)

A KVM jellemzői II.

- erőforrások finomhangolása nem lehetséges
- 32 és 64 bites host támogatás
- 32 és 64 bites guest támogatás
- SMP host támogatás
- SMP guest támogatás (max. 16 CPU)
- guest swapping
- live migration

Támogatott guest rendszerek

- Csak néhány példa:
- Linux 2.6, 32/64bit
- *BSD, 32/64bit (Net 32)
- MS Windows Server 2008, 32/64bit
- MS Windows XP, 32/64bit
- MS Windows 7 és Vista, 32/64bit

A virtuális vas jellemzői

- proci - speciális PII-nek látszó
- memória - szabadon beállítható méret
- diszkek
 - IDE / SCSI / virtio
- CDROM
- hálózati eszközök
 - ne2k_pci, pcnet, rtl8139, virtio ...

A virtio jellemzői

- 2.6.25-ös kernelbe és a 60-as KVM-be került be
- A virtio diszk saját mérések szerint 10x gyorsabb, mint az SCSI emuláció
- A virtio net más mérései szerint 2-4x gyorsabb
- A guest-en a block eszköznél speciális beállításokra van szükség

A virtuális gépek erőforrásigénye

- Mini php host (2 párhuzamos user):
d: 600M s: 256M m: 64M p: 1
- Közepes Drupal (50 párhuzamos user):
d: 2G s: 1G m: 400M p: 1
- Nagyobb DBs (minden VM ezt használja):
d: 50G s: 2G m: 2G p: 2
- Nagyobb Drupal, közös DBs (500 párhuzamos user):
d: 10G s: 2G m: 2G p: 2

Erőforrás igény felmérése

- Ha már van gép, akkor azon mérni kell az processzor, I/O, memória, hálózat és diszk terhelését
- Ha nincs, akkor becslés alapján, túlméretezve létrehozni a virtuális gépet
- utána a free, a top, az iotop, df és iptraf, vagy inkább Munin figyelésével vissza a szükséges méretre
- a diszk átméretezésére szerencsésebb egy új diszk vagy LVM használható

Virtuális gépek kényelmesen

- libvirt és Virt-Manager
- Virtuális gépek létrehozása klónozással, pár beállítást megcsinál, csak néhány dolgot kell átállítani
 - hálózati cím (de ezt a MAC-ből generálhatjuk az első boot-kor)
 - gép neve
 - szükséges plusz csomagok, az alrendszer beállításai
- cron-apt, és egyéb időzített ellenőrzések

Biztonsági kérdések

- Ha a KVM kernel része hibás, akkor bukott a rendszer
- A host-ra minimális hozzáférést, csak admin csatornára, és csak szabályzott forrásból
- A host csomagszűrőjét felhasználva minimalizálni kell az áthallást a virtuális gépek között
- Rendszeres frissítés mindenhol
- A VM hálózaton naplózzon, minimum a host-ra

Erőforrások szabályzása

- a memória szabályzása viszonylag egyszerű
- a processzeket nice segítségével lehet priorizálni
- az I/O-t az ionice segítségével priorizálhatjuk
- a hálózat felhasználását a Netfilter limit moduljával szabályozhatjuk egyszerűen
- hálózatonál lehetőség van osztály bázisú sáv szélesség szabályzásra (pl. CBQ, HTB)

Köszönöm a figyelmet.

Kérdések?

Magyar Szabad Szoftver Tárház

Erdei Csaba <erdei.csaba@fsf.hu>

Mátó Péter <mato.peter@fsf.hu>