OpenVZ vs. VServer

Bevezető

Az OpenVZ és a VServer egy-egy operációs rendszer-szintű virtualizációs környezet. Bizonyára mindannyian ismerjük a Posix típusú operációs rendszerek chroot() nevű rendszerszolgáltatást, amely egy processzt bezár egy alkönyvtárba. Ez már egyfajta virtualizációnak is tekinthető. Azonban itt a meglévő fájlleíróin keresztül továbbra is tud az operációs root-ján kívül futó folyamatokkal kommunikálni, vagy megfelelő adminisztrátori jogosultságok (capability) birtokában olyan dolgokat végrehajtani, amelyeket nem szeretnénk egy „bezárt" virtuális környezet számára engedélyezni.
Erre a finomabb korlátozásra ad lehetőséget pl. a FreeBSD jail szolgáltatása, amely nem csak bezárja egy alkönyvtárba a folyamatokat, de a hálózat konfigurációjához való hozzáférést is elzárja előlük, sőt a jail-be zárt a folyamatokhoz tartozó környezet(-ek) egy(-egy) külön hálózati interfészt kap. Hasonló szolgáltatáskészlettel rendelkezik még az Open- ill. NetBSD sysjail szolgáltatása, a Solaris Container/Zone szolgáltatása. Bár nem a virtualizáció a célja az RSBAC biztonsági megoldásnak, de a hálózati „elkerítés" funkció avval is megvalósítható.
A hasonló operációs rendszer-szintű virtualizációs megoldásokról jó összehasonlítást, és kiindulási pontot tartalmaz a Wikipedia idevágó szócikke: http://en.wikipedia.org/wiki/Operating_system-level_virtualization
Eme írásban az OpenVZ és VServer operációs rendszer-szintű Linux-specifikus virtualizációs megoldások lesznek összehasonlítva.

Processztér

Az operációs rendszer-szintű virtualizáció egyik fontos szolgáltatása, hogy az egyes virtuális környezetekben futó folyamatokat is elkülöníti egymástól. A VServer ezt úgy valósítja meg, hogy minden egyes környezethez egy kontextusszámot rendel. A „host" (saját terminológiájával „root") környezet a 0-s context id-t kapja, a többi VServerünknek pedig a 2 és afölötti számokat adhatjuk context id gyanánt. Az 1-es context id speciális jelentéssel bír: Az itt futó folyamatok látják az összes kontextus minden folyamatát, de kill szignált nem tud küldeni nekik. A folyamatok így a nagy globális processztérből kapnak sorszámot, de kiegészülnek egy további id-vel, amely a saját illetve más folyamatokkal kapcsolatos láthatóságára vonatkozóan tartalmaz információt. A VServer további érdekessége, hogy minden egyes kontextusban az init processz id-je 1-es.
A Linux speciális sajátossága, hogy a folyamatokról szóló információk egy jó részét a procfs-en keresztül kapjuk, de találunk itt más, menet közben elérhető, információkat, mint pl. az usbfs. A vserver arra is lehetőséget ad, hogy a proc alatt látható speciális bejegyzések közül elrejtsünk / megjelenítsünk[1] néhányat igényeink szerint.
Az OpenVZ terminológiája szerint a virtuális környezeteket VE-knek (Virtuális Entitásoknak) hívjuk. Az OpenVZ a filozófiája szerint igyekszik úgy megvalósítani a VE-ket, mintha azok valódi virtuális számítógépek lennének. Így a VE id-je és a pid együtt határoznak meg egy folyamatot. Magyarán lehet egyszerre 212-es pid-ű folyamat két különböző VE-ben, azoknak attól még semmi közük nem lesz egymáshoz.

[1] Akkor kell megjeleníteni, ha a „minden tilos, kivéve amit engedünk", ill. akkor kell elrejteni, ha a „mindent szabad kivéve amit tiltunk" megközelítést használjuk.

Korlátozások

Egy-egy virtuális környezetben kétféle dolgot szokás korlátozni: Az elérhető erőforrások mennyiségét (pl.:cpu, io), valamint a jogosultságokat.
Mindkét virtuális környezet lehetőséget ad arra, hogy a környezetben futó folyamatok által felhasználható memória és processzoridő mennyiségét korlátozzuk. Mindkét megoldás ad lehetőséget arra, hogy a virtuális környezetek felhasználóihoz tartozóan tudjunk lemezkvótát kiszabni. A virtuális környezetek I/O-jára vonatkozó korlátozások mindkét esetben csak a CFQ lemezütemező segítségével vehetők igénybe. A VServer esetén ez az ütemezési lehetőség folyamatonkénti, az OpenVZ esetén pedig VE-nkénti.
A speciális eszközök elérhetőségével kapcsolatos szabályozások a VServerben úgy vannak megoldva, hogy az mknod, illetve a hozzá tartozó capability nem érhető el a guest környezetekben futó folyamatok számára, még az OpenVZ-ben egy-egy eszközről tudjuk meghatározni, hogy lássa-e a guest, de maga az mknod-olás lehetősége nincs tiltva.
Az OpenVZ-ben a capability-k terén is érvényesül, az alapvetően mindent szabad, legfeljebb nem tudja mire használni a jogosultságát megközelítés, még a VServerben már erőforrás-igénylésekor is „futhatunk” tiltásba. Szerencsére a VServer dokumentációjában külön fejezetet szenteltek Problematic Programs néven a témának, amelyeknél tipikusan vagy egy capability-t kell a szoftvert futtató környezet számára engedélyeznünk, vagy a programot kell módosított paraméterekkel lefordítani (pl. Bind9).

Hálózati elérés és korlátozások

A szigorítások mindkét virtualizáció esetében megtiltják, hogy egy guest rendszerben a hálózat konfigurációjára vonatkozó módosítást tegyünk. Azonban a két rendszer az alapfilozófiáját itt is tartja: A VServernél a host-on az indulás előtt kell a konfigurációval kapcsolatban mindent meghatározzunk. Még az OpenVZ esetén belül további módosításokra van lehetőségünk.
A VServernél a hálózat elérése történhet úgy, hogy:

- egy meglévő fizikai interfészünkhöz adunk egy új IPcímet,
- létrehozunk egy dummy interfészt privát címekkel, majd NAT segítségével tesszük elérhetővé a virtuális szervert
- a virtuális szerverek külön hálózati interfészre kerülnek (ilyenkor az első az interfészhez kapcsolódó virtuális szerver indulásakor, magát az interfészt - legyen az külön fizikai, vagy vlan interfész - is fel kell élesszük, valamint az utolsó ehhez kapcsolódó virtuális gép leállásakor azt le kell kapcsoljuk)

A virtuális szerveren belül routing, vagy interfész konfiguráció módosítására nincs lehetőségünk. IPv6 támogatást pedig csak a 2.3 fölötti verziók nyújtanak, valamint a 2.2-es vserver sorozathoz külön patch alkalmazásával nyerhetünk.
Az OpenVZ megközelítése a hálózat esetén: A külvilággal kapcsolatos interfészek konfigurációját kívül, a host-on előre fixen meg kell adni, de az avval kapcsolatos további konfiguráció már a guest-ben történhet. Az OpenVZ két szinten kinál kapcsolódási lehetőséget:
Layer 3 összeköttetést az ún. venet interfész(ek)en keresztül
Layer 2 összeköttetést az ún. veth interfész(ek)en keresztül
A külvilág elérése venet esetén történhet routeolással vagy natolással is történhet. A veth esetén pedig vagy a fizikai interfészünkkel bridgeljük össze, vagy a bridge interészünk egy külön alhálózatra kerül, ami publikus címek esetén a host-on keresztüli routeolással, privát címek használatakor pedig natolással történhet. Első esetben a host-on további szűréseket tudunk tenni netfilter/iptables segítségével, második esetben ha nagyon szigorúak akarunk lenni, és pl. vigyázni, hogy a fizikai interfésszel összebridgelt VE ne adjon egy IPaliast az interfészéhez, vagy legalábbis az ne lásson ki, ebtables-t is be kell vessünk.

A megközelítés másik nagy előnye, hogy az IPv6 gond nélkül megy az OpenVZ használata esetén. További lehetőség, hogy az OpenVZ esetén, a guest-eknek saját iptables szabálykészletük lehet, valamint minden VE esetén külön meghatározhatjuk, hogy mely iptables modulokat használhatják a szabályaikhoz.
A VServernél nincs ilyen, cserébe viszont a mindent a host-ról felügyelünk megközelítésnek, a host iptables szabályaiban akár uid/gid match-et is használhatunk, amely a vcontextbeli uid/gid értékén fog múlni.

Mount, NFS

Előfordulhat, hogy egy már futó virtuális szerverhez szeretnénk további mount-pontot adni, akár egy helyi eszköz felcsatolásával, akár egy NFS export elérésével. OpenVZ esetén ilyennel nincs probléma, hiszen a mount maga nincs tiltva a vserveren belül, azonban az eszközt láthatóvá kell tenni a VE számára. Ha egy a host rendszerbe is felcsatolt kötetről van szó, akkor pedig a /vz/root/VEID/... alá kell bind-mount segítségével elérhetővé tegyük. NFS mountjának VE-beli elérése két dolgon múlik: az nfs engedélyezve van-e a VE-ben, és hogy az elérendő NFS szerverrel a VE tud-e hálózati kapcsolatot létesíteni.
VServer esetén a guestekben nem engedélyezett a mount. Ezt a megfelelő capability adásával kiküszöbölhetjük, de akkor az egész kontextusban lehetőség nyílik mountra. Egyszeri alkalomhoz használjuk inkább a vnamespace parancsot, amelynek átadhatjuk, hogy milyen contextus id-vel futtasson le egy parancsot, de a chroot() kihagyásával. Így egy eszközt, vagy egy NFS exportot bemountolhatunk közvetlenül a virtuális kontextusba, annélkül hogy az NFS-szerverrel tudnának hálózati kapcsolatot létesíteni az ott futó folyamatok.

Deployment

A guestek telepítésére valamint telepítés utáni mangementjére a VServer több különböző módszert is kínál: használhatjuk debian alapú rendszerekhez a debootstrap-et, fai-t, de yum-os és apt/rpm alapú rendszereket is tud kezelni. További tipikus használat, hogy egy minta-vservert klónozunk, vagy használunk mintának.
A debian oldaláról a debootstrap-es telepítésre további könnyítést jelent, hogy a vserver-debiantools csomag tartalmaz egy newvserver parancsot, amelynek csak a legalapvetőbb paramétereket kell megadjuk (ipcím/netmask, node-név és domain név, célkönyvtár, debian disztrib neve, contextus id-je) és pillanatok alatt fel is telepíti a virtuális környezetet.
Az OpenVZ megközelítése a template-k használatát helyezi előtérbe:
A http://wiki.openvz.org/Download/template/precreated oldalon számos előre elkészített templatet találhatunk. Ha találunk, számunkra megfelelő template-t, akkor azt letöltjük a /vz/template/cache könyvtárunkba, majd egy pillanat alatt példányosíthatjuk azt egy VE-á.
Ha mégis saját magunk szeretnénk a telepítésen végigmenni, akár a debootstrap-pel, akkor az alábbi címen találunk leírást, hogyan készítsük el a saját template-ünket, vagy VE-nket:
http://wiki.openvz.org/Debian_template_creation
http://wiki.openvz.org/Deploying_Debian_VEs_without_Templates

Snapshot/Checkpoint

Az OpenVZ lehetővé teszi, hogy a futó VE-ről egy pillanatképet készítsünk, vagy hogy azt lementsük/ felfüggesszük működését. Ennek a funkcionalitásnak köszönhetően az is lehetővé válik, hogy az ún. live migráció végrehajtható legyen egy VE-n, vagyis menet közben áttenni egy futó

VE-t egyik host rendszerről egy másikra (pl. TMK miatt).

Out of the box support

A vserver támogatottsága a debianban már az etch-nél is elég erős, és jól használható volt. Az openvz-hez még azonban csak patch-csomag formájában volt kernel-támogatás az etch kiadásakor. Aki etch-et szeretett volna használni, de új gépe volt, és etch-n-half-ra esett a választása, mind a két virtualizációs megoldáshoz „kézzel” kellett előállítsa a kernelét. A lenny kiadására mindkét virtualizációs technológia rögtön a telepítés után igénybe vehető. Természetesen egy host-on csak az egyik.
Ubuntu esetén Operációs rendszer-szintű virtualizációs megoldások támogatása csak az LTS verziókban érhető el. A Dapper esetén egy nemhivatalos tároló volt elérhető a vserver-hez, a Hardy-ba pedig teljes körűen intergrálták az OpenVZ támogatásához szükséges szoftvereket.
Hogy a Debian illetve Ubuntu esetén a jövő mit hoz, arról biztosat nem lehet tudni.
Egyéb linux disztribúciókon az OpenVZ támogatottsága már alábbhagy. Out of the box támogatás nem elérhető bennük OpenVZ-hez, saját magunknak kell megoldani a működését. Out of the box támogatás a VServer esetén sincs a nem-debian alapú rendszerekhez, de elég részletesen dokumentált ArchLinux, Centos, Fedora, Gentoo és Slackware útmutatókkal, valamint az out of the box élményt nyújtó tárolókkal rendelkezik.

Zárszó

Az egyes projektek weboldalai az alábbi címen érhetők el:
http://linux-vserver.org/
http://openvz.org/
A témában érdemes lehet még korábbi networkshop előadásokat is megnézni, például a 2007-es NetWorkShopon Csillag Tamással közösen tartott előadásunkat.
A Proxmox egy szintén figyelemre méltó projekt, ahol az OpenVZ lehetőségeit egy könnyen használható User Interfésszel ötvözik, a könnyű használatbavételt pedig BMI[2]-vel könnyítik meg.
Az előadás anyaga, jelen írás, és néhány további szemléltető videó a http://www.bibl.u-szeged.hu/~pasztor/nws2k9/ címen érhető el:

- **openvz-temp-deploy.avi** egy OpenVZ VE telepítése template-ből[3]
- **openvz-net.avi** az előző videóban feltelepített VE számára hálózat elérhetővé tétele venet interfészen keresztül
- **vserver-prepare.avi** a virtuális szerver telepítésének előkészítését mutatja be[4]
- **vserver-install.avi** a telepítés végső fázisa

[2]Bare Metal Installer
[3]Feltételezve, hogy a vzctl csomag, és az openvz-patchelt kernel fel van telepítve, és az openvz kernel fut
[4]Feltételezve, hogy a vserver-debiantools csomag, és a vserver-patchelt kernel fel van telepítve, és a vserver kernel fut