Peskovniki

Racunalniski pojem peskovnik oznacuje mehanizem, ki zagnan program strogo loci od ostalih programov v sistemu, tako da program v peskovniku ne vpliva na gostitelja ali ostale programe. V praksi so peskovniki pogosto uporabljeni za izvajanje netestirane kode, kode neznanega izvora, programov uporabnikov in spletnih strani, ki niso vredne nasega zaupanja. To lahko pomeni, da program ne more dostopati do datotek gostitelja in drugih programov, ima omejen cas izvajanja, omejen dostop do vhodno-izhodnih naprav in podobno.

Izvedbe peskovnikov

Obstaja vec pristopov, ki se med razlikujejo v nivoju zascite glavnega sistema in dodatnih stroskih za izvedbo preskovnika. Ce ne upostevamo varnostnih lukenj, stroski izvedbe narascajo linearno z nivojem zascite prvotnega sistema. Izvedba peskovnika, ki na originalnem sistemu pusti najmanj sledi, je tako najdrazja. Locimo naslednje izvedbe peskovnikov:

Fizicna kopija sistema

Najdrazji nacin izvedbe peskovnika je replikacija sistema. Ta pristop pogosto uporabljajo raziskovalci, ki ugotavljajo vplive zlonamerne programske opreme na tocno doloceno strojno opremo.

Virtualni stroji

Bistvo tega pristopa je izvajanje programa znotraj virtualnega stroja. Virtualizacija nam omogoca omejevanje sistemskih virov, izolacijo programov in enkapsulacijo sistema, stranski ucinek tega pa je prenosljivost virtualiziranega sistema med razlicnimi racunalniki. Kot primere virtualnih strojev lahko nastejemo Virtualbox, KVM, VMware resitve, JVM (Java Virtual Machine) in ostale.

Virtualizacija aplikacij

Virtualizaciji operacijskega sistema soroden pristop je virtualizacija aplikacij, kjer program zazenemo s pomocjo nekega vmesnega ogrodja, preko tega pa potem program komunicira z OS. Enkapsuliranemu programu glede na zmoznosti ogrodja omejimo dovoljene operacije. Tak mehanizem uporablja vecina resitev za peskovnike na platformi Windows.

Kontejnerizacija

Program tece na istem jedru, a je s sistemskimi pravili izoliran od ostalih programov. Princip se od virtualizacije aplikacij razlikuje v tem, da je ogrodje za omejevanje in nadzor aplikacije vgrajeno v jedro operacijskega sistema, kar odstrani en nivo abstrakcije. Prednosti tega pristopa so hitrost izvajanja (dodatnih stroskov izvajanja prakticno ni) in enostavna uporaba, slabost pa je, da je virtualizirano okolje mocno odvisno od operacijskega sistema, kar pomeni, da istega kontejnerja ne moremo zagnati na GNU/Linux in Windows operacijskem sistemu, vcasih pa so razlike celo med razlicnimi verzijami jedra Linux. Orodje za kontejnerizacijo Docker se temu v operacijskih sistemih Windows in OS X izogne z virtualizacijo Linux sistema, kar pa negativno vpliva na zmogljivost. Primeri tega mehanizma so FreeBSD jail, lxc, lxd, libcontainer, lmctfy (Let Me Contain That For You) in drugi.

Omejevanje s sistemskimi mehanizmi

V nadaljevanju so opisane funkcionalnosti jedra Linux, ki skupaj omogocajo kontejnerizacijo.

chroot

Chroot je funkcionalnost *NIX operacijskih sistemov, ki procesu in njegovim otrokom spremeni korenski imenik. Posledicno proces ne more dostopati do datotek izven tega korena. Sistemski klic lahko uporabimo za postavitev virtualizirane kopije sistema. Izvajanje zapletenih programov pod spremenjenim korenskim imenikom je nerodno, ker morajo biti dostopne vse knjiznice, ki jih program uporablja. To vodi v podvajanje datotek, zato se chroot samostojno za varnostne namene ne uporablja. Je pa prirocno orodje za postavitev okolja za testiranje, prevajanje kode in popravljanje napak pri namestitvi operacijskega sistema.

seccomp

Secure computing mode procesu onemogoci uporabo vseh sistemskih klicev, razen *exit*, *sigreturn* in *read* ter *write* nad ze odprtimi deskriptorji. Precej bolj uporabna razsiritev je seccomp-bpf, kjer z *Berkley Packet Filter* pravili natancno dolocimo dovoljene operacije. Problem te razsiritve naj bi bila zapletenost - *BPF* pravila namrec definiramo s programom. To pomeni, da moramo za nadzor nasega programa napisati poseben, locen program.

cgroups

Control groups so del Linux jedra. Skrbijo za nadzor, omejevanje, prioritizacijo in merjenje uporabe sistemskih virov skupine procesov.

namespaces

Imenski prostori izolirajo sistemske vire za skupine procesov. Procesi v istem imenskem prostoru lahko vidijo samo sistemske vire sosednjih procesov, viri procesov iz drugih imenskih prostorov pa niso dostopni. Ob zagonu je proces v imenskem prostoru svojega prednika, lahko pa ustvari novega, ali pa se premakne v drugega.

Jedro Linux ponuja 6 imenskih prostorov:

- IPC (medprocesna komunikacija)
- Network (dostop do omreznega stacka)
- Mount (dostop do datotecnega sistema)
- PID (ID procesov)
- User (ID uporabnikov in skupin)
- UTS (hostname)

Linux Security Modules

To je ogrodje, katerega implementacije omogocajo *MAC*, oziroma *Mandatory Access Control*. Uporabniki in njihovi procesi lahko dostopajo in izvajajo akcije samo na tistih objektih, kjer jim je dostop izrecno dovoljen. Objekt lahko v tem kontekstu predstavlja datoteko, TCP vrata, vhodno/izhodno napravo, del pomnilnika in podobno. Znana primera implementacij sta SELinux in AppArmor.

FreeBSD jail

V operacijskem sistemu FreeBSD obstaja celovita resitev za sandboxing ze od leta 2000. Programe omeji z enakimi principi kot kontejnerizacija.

OpenBSD tame

Precej sveza zadeva je sistemski klic tame za OpenBSD, ki programerju omogoca omejevanje dovoljenih akcij svojega programa. Temelji na dejstvu, da program ob zagonu uporabi vecje stevilo sistemskih klicev, v glavni fazi izvajanja pa se stevilo razlicnih sistemskih klicev precej zmanjsa. Po zagonu si program sam omeji dovoljene poteze, v primeru krsitve pravil pa operacijski sistem programu poslje signal SIGKILL.

Viri

- Nanut, M. Varno izvajanje programov v operacijskem sistemu GNU/Linux http://eprints.fri.uni-lj.si/2895/
- OpenBSD tame http://marc.info/?l=openbsd-tech&m=143725996614627&w=2
- namespaces man page http://man7.org/linux/man-pages/man7/namespaces.7.html
- cgroups https://www.kernel.org/doc/Documentation/cgroups/cgroups.txt
- An introduction to virtualization http://www.kernelthread.com/publications/virtualization/
- A Tase of Computer Security - sandboxing http://www.kernelthread.com/publications/security/sandboxing
- Application Virtualization https://en.wikipedia.org/wiki/Application_virtualization
- LinuxContainers.org https://linuxcontainers.org/