Sandboxing

Índex

Sandboxing és una tècnica de seguretat que consisteix a executar un programa dins d'un entorn aïllat i restringit, de manera que només pugui accedir als recursos del sistema (fitxers, xarxa, altres processos, dispositius) que se li permetin explícitament.

La idea bàsica

Imagina que un programa normal té accés a “tota la casa”: pot llegir i escriure qualsevol fitxer al qual el seu usuari tingui permisos, obrir connexions de xarxa, veure altres processos, etc. El sandboxing el tanca en una “habitació” amb només els mobles que necessita per fer la seva feina, i cap més.

Si aquell programa és compromès (per un bug, una vulnerabilitat, o codi maliciós), l'atacant només hereta els permisos limitats de la “habitació”, no els de tota la casa.

Per què serveix

- **Limitar el dany d'una vulnerabilitat**: si una app té un forat de seguretat, l'aïllament evita que un atacant escali a la resta del sistema.
- **Reduir la superfície d'atac**: menys accés = menys coses que un atacant pugui abusar.
- **Executar codi no confiable amb seguretat**: provar programes desconeguts, obrir fitxers sospitosos, o executar codi de tercers sense arriscar tot el sistema.

Mecanismes típics per aconseguir-ho

- **Namespaces del kernel Linux**: aïllen la vista que té un procés dels processos, xarxa, punts de muntatge o usuaris del sistema --- el procés “creu” que està sol al sistema.
- **Control de capacitats (capabilities)**: en lloc de donar accés total de root, es concedeixen només privilegis molt concrets (per exemple, “pot obrir un port per sota del 1024” però no “pot fer qualsevol cosa que root pugui fer”).
- **Filtratge de syscalls (seccomp)**: es restringeix quines crides al sistema operatiu pot fer un procés, bloquejant les que no necessita.
- **Sistemes de fitxers de només lectura o privats**: es munten parts del sistema de fitxers com a inaccessibles o només-lectura des del punt de vista del procés confinat, encara que la resta del sistema hi tingui accés normal.
- **Mandatory Access Control (MAC)**: sistemes com AppArmor o SELinux defineixen perfils que especifiquen exactament quins fitxers, capacitats i accions pot fer cada programa, independentment dels permisos Unix tradicionals.

Un exemple simple

Un navegador web modern és un cas clàssic de sandboxing: cada pestanya s'executa en un procés aïllat que no pot llegir directament el disc dur, la memòria d'altres pestanyes, ni accedir a la càmera o el micròfon sense demanar permís explícit --- encara que el codi JavaScript d'una pàgina maliciosa intenti fer-ho.

El compromís (trade-off)

Com més restrictiu és el sandbox, més segur és el sistema, però també és més probable que el programa “trenqui” si intenta fer alguna cosa legítima que el sandbox no havia previst. Per això, sovint cal ajustar els perfils de confinament quan una aplicació necessita accedir a un recurs que inicialment se li havia denegat.

Ubuntu 26.04 endureix per defecte la unitat systemd d'Apache

La unitat systemd d'Apache2 ara activa `MemoryDenyWriteExecute=yes` per defecte com a mesura d'enduriment de seguretat, cosa que prevé mapes de memòria simultàniament escrivibles i executables.

El paquet apache2 d'Ubuntu 26.04 ha canviat la unitat systemd per defecte respecte a 24.04, afegint-hi directives de confinament que abans no hi eren.

El patró és més ampli, no és només Apache

Aquest canvi forma part d'una tendència general de 26.04: hi ha un enduriment de sandboxing més agressiu a nivell de systemd en diversos serveis (per exemple `polkit.service` amb `ProtectSystem`, `ProtectHome`, `PrivateTmp`, `NoNewPrivileges`), i SSSD i OpenLDAP han estat despullats de privilegis root.

A més, Ubuntu 26.04 és l'última versió que admet compatibilitat amb scripts SysV init dins de systemd, i un cop es traiu aquesta capa, cada servei ha de fer servir un fitxer d'unitat systemd, cosa que permet que qualsevol servei sigui sandboxat amb `ProtectSystem`, `PrivateTmp`, `NoNewPrivileges`, etc. --- és a dir, Ubuntu està aprofitant aquesta LTS per aplicar aquest tipus de confinament de forma molt més sistemàtica als paquets del dipòsit.

Per què 24.04 no ho patia

A 24.04, els mainters del paquet `apache2` (i molts altres) encara no havien afegit aquestes directives d'enduriment a la unitat per defecte --- el confinament amb `ProtectSystem`/`MemoryDenyWriteExecute` no formava part del paquet base. Amb 26.04, Canonical ha fet un esforç deliberat d'enduriment de seguretat per defecte across-the-board (el mateix any que reforcen `polkit`, SSSD, OpenLDAP i eliminen SysV init), i Apache n'és una víctima col·lateral pel que fa a compatibilitat amb aplicacions PHP.

Versions d'aquest document

- HTML - sandboxing.html
- PDF - sandboxing.pdf
- ODT - sandboxing.odt
- MD - sandboxing.md