mkosi , systemd-nspawn i machinectl

La comanda *mkosi* (https://github.com/systemd/mkosi) -de "Make Operating System Image"- és una eina per generar imatges (generalment en forma de fitxers) que contenen al seu interior una estructura de particions que incorporen un sistema operatiu mínim, llest per ser arrencat ja sigui adjuntant aquesta imatge com a disc d'un contenidor o bé, si així s'especifica en la configuració, també d'una màquina virtual.

Concretament, *mkosi* s'encarrega de particionar l'interior de la imatge seguint l'estàndard GPT (això vol dir que dins de la imatge sempre es generarà com a mínim dues particions per defecte: la ESP i la partició arrel), de formatejar aquestes particions internes (el sistema arrel podrà estar, a escollir, en Ext4, Btrfs o també Squashfs, que és un forat denomés lectura), i de descarregar-hi allà (des d'un repositoris preestablerts) tots els paquets bàsics necessaris per tenir allà un minisistema operatiu funcional, també a escollir (les possibilitats són Debian, Ubuntu, Fedora, Suse, Arch o CentOS). Depenent de si s'escull generar una imatge arrencable o no (és a dir, preparada per fer-se servir com a màquina virtual autònoma o bé només com a simple contenidor), s'afegirà un gestor d'arranc i un kernel+initrd (ubicats a la partició ESP, muntada sota la carpeta /efi), o no.

La creació pròpiament dita de les imatges dels diferents sistemes operatius *mkosi* delega en eines especialitzades que tenen una llarga història dins la comunitat. Així per exemple, per crear imatges Debian/Ubuntu fa servir l'eina *debootstrap (o cdebootstrap o grml-debootstrap o vmdebootstrap*, segons les circumstàncies); per crear imatges Fedora/CentOS fa servir l'eina *dnf* amb el paràmetre *–installroot;* per crear imatges Arch fa servir l'eina *pacstrap* i per crear imatges Suse fa servir *zypper*. Això vol dir, no obstant, que depèn d'on estigui instal.lat *mkosi*, aquest serà capaç de generar imatges de més o menys sistemes diferents: per exemple, a sistemes Debian/Ubuntu no existeix el paquet "dnf", amb la qual cosa serà impossible generar imatges Fedora/CentOS; en canvi, a sistemes Fedora/CentOS sí que existeix el paquet "debootstrap", i, per tant, en aquest cas sí que es pot generar imatges Debian/Ubuntu.

> NOTA: Cadascuna d'aquestes eines (debootstrap, etc) ja ve configurada per accedir als repositoris oficials de la distribució corresponent i descarregar d'allà els paquets necessaris per tal d'omplir de contingut les imatges. No obstant, si es desitja, *mkosi* permet escollir explícitament els repositoris a usar gràcies al seu paràmetre *-m* per especificar el servidor-mirror a usar i/o *--repositories* per indicar un repositori concret

A continuació es presenta una taula amb els paràmetres més importants de *mkosi*. Tal com es pot veure allà, una comanda per exemple com...:

sudo mkosi -d debian -r bullseye -t gpt_ext4 -b -o imatge.raw -p apt,iproute2,nano --password 1234

...crearà un fitxer-imatge anomenat "imatge.raw" contenint un sistema Debian bàsic arrencable amb un usuari "root" amb contrasenya 1234 i amb un grapat de paquets instal.lats addicionals.

> NOTA: Altres opcions més concretes que no veurem són: *mkosi build, mkosi clean, mkosi shell, mkosi boot* o *mkosi qemu*

Si no volguésim indicar tants paràmetres, tal com es pot veure també a la taula, es pot crear un arxiu anomenat "***mkosi.conf***" a la mateixa carpeta on executarem *mkosi* que contingui una sèrie de seccions i directives que estableixen els valors per defecte per tots els paràmetres que desitgem i així no caldrà especificar-los a mà (en el cas que s'especifiquessin, però, tindrien preferència) ; un exemple seria:

```
[Distribution]
Distribution=debian
Release=bullseye
[Output]
Format=gpt_ext4
Bootable=yes
Output=imatge.raw
[Partitions]
RootSize=2G
[Packages]
Packages=apt iproute2 nano
[Validation]
Password=1234
```

NOTA: En qualsevol cas, si un paràmetre no s'especifica enlloc (ni a l'arxiu "mkosi.conf" ni directament, es pot esbrinar quin és el seu valor per defecte executant la comanda: *mkosi summary*

Paràmetre	**Directiva equivalent a l'arxiu *mkiso.conf***	**Significat**
-d ...	*Distribution=...* (sota la secció [Distribution])	Indica la distribució a instal.lar. Valors possibles: "fedora","ubuntu","debian", "opensuse","arch","centos"
-r ...	*Release=...* (sota la secció [Distribution])	Indica la versió de la distribució. Valors possibles: 35, "bullseye","jammy", etc. Si no s'indica, s'escull el valor més modern (per exemple, a Debian serà "unstable")
-t ...	*Format=...* (sota la secció [Output])	Indica el format de la imatge a crear. Valors possibles: *gpt_ext4* : la imatge serà un fitxer dins del qual hi haurà un sistema GPT amb una 1ª partició ESP (muntada a /efi) -si la imatge es marca com arrencable- i una 2ª partició (arrel del sistema) de tipus Ext4. Equivalent a *raw_gpt* *gpt_xfs* : la imatge serà un fitxer dins del qual hi haurà un sistema GPT amb una 1ª partició ESP (muntada a /efi) -si la imatge es marca com arrencable- i una 2ª partició (arrel del sistema) de tipus XFS. *gpt_btrfs* : la imatge serà un fitxer dins del qual hi haurà un sistema GPT amb una 1ª partició ESP (muntada a /efi) -si la imatge es marca com arrencable- i una 2ª partició (arrel del sistema) de tipus Btrfs. Equivalent a *raw_btrfs* *gpt_squashfs* : la imatge serà un fitxer dins del qual hi haurà un sistema GPT amb una 1ª partició ESP (muntada a /efi) -si la imatge es marca com arrencable- i una 2ª partició (arrel del sistema) de tipus Squashfs (és a dir, de només lectura). Equivalent a *raw_squashfs* *directory* : la imatge serà una carpeta dins de la qual només hi haurà el contingut del sistema arrel (per tant, no es pot combinar amb *-b*) *subvolume* : la imatge serà un subvolum Btrfs dins del qual només hi haurà el contingut del sistema arrel (per tant, no es pot combinar amb *-b*) *tar* : la imatge serà un arxiu tar dins del qual només hi haurà el contingut del sistema arrel (per tant, no es pot combinar amb *-b*)
-b	*Bootable=yes* (sota la secció [Output])	Fa que la imatge pugui ser arrencable com a VM autònoma. És a dir: afegeix una partició ESP amb un bootloader i, a la partició arrel, una carpeta /boot amb un kernel+initrd. Només funciona pels formats "gpt_xxx". NOTA: La parella "kernel+initrd" és generada per Dracut en ser invocat per *mkosi* durant els darrers moments de la seva execució.
-o ...	*Output=...* (sota la secció [Output])	Indica la ruta i nom de la imatge a crear (per defecte: "image.raw") NOTA: Si s'executa mkosi vàries vegades, no sobreescriurà la imatge generada si ja existia d'abans, a no ser que s'indiqui el paràmetre *-f*
--root-size nº *--esp-size nº*	*RootSize=nº* *EspSize=nº* (sota la secció [Output])	Estableix el tamany del sist. fitxers arrel (p. defecte 3G) Estableix el tamany de la partició ESP (p. defecte 256M) Només funcionen pels formats "gpt_xxx"
-p ...	*Packages=...* (sota la secció [Packages])	Indica un paquet addicional que s'instal.larà a la imatge. Si es vol afegir més d"un cal indicar-los separats per comes. A l'arxiu de configuració basta amb una sola línia Packages= indicant els diferents paquets un rera l'altre separats per espai en blanc. Paquets útils: "nano", "apt/dnf", "iproute2/iproute" o "iputils-ping/iputils"
--password ...	*Password=...* (sota la secció [Validation])	Assigna la contrasenya a l'usuari root. Interessant també és el paràmetre *--password-is-hashed* , el qual si val "true" indica que el valor de *--password* no és la contrasenya tal qual sinó el hash SHA512 directe a escriure a l'arxiu "/etc/shadow" Només funciona pels formats "gpt_xxx".
--read-only		Fa el sistema arrel de només lectura (gpt_squashfs ja ho és) Només funciona pels formats "gpt_xxx"
--hostname ...		Estableix el nom de la màquina dins la imatge
--kernel-commandline ...		Estableix els paràmetres del kernel responsable d'arrencar la imatge (si aquesta és arrencable, és clar)

A la mateixa carpeta on executem mkosi també poden haver opcionalment els següents elements:

*Subcarpeta "**mkosi.extra**" : Tots els arxius que hi hagi al seu interior es copiaran (amb propietari "root") dins de la partició arrel dins de la imatge, respectant l'arbre de subcarpetes (és a dir, l'arxiu "mkosi.extra/etc/unarxiu.conf" es copiarà sota la carpeta "/etc" del sistema arrel)

*Subcarpeta "**mkosi.cache**" : Usada com cache dels paquets descarregats, per no haver de tornar a descarregar-los si es repeteix l'execució de mkosi vàries vegades

*Script Bash executable "***mkosi.postinst***" : És invocat com a últim pas a la generació de la imatge i s'executa dins del context del sistema existent a la imatge. Permet poder alterar la imatge "a la carta"

NOTA: Si fos necessari que aquest script tingui accés a la xarxa, caldria afegir el paràmetre *--with-network* a la comanda *mkosi* (o bé afegir la línia *WithNetwork=yes* dins de la secció [Packages] de l'arxiu "mkosi.conf").

*Fitxer "***mkosi.rootpw***" : Conté la contrasenya que s'establirà a l'usuari "root" de la imatge (si a l'arxiu apareix un salt de línia al final es treu automàticament, així que no és problema). Aquest arxiu ha de ser propietat de l'usuari "root" de la màquina host i tenir permisos 600 o menys. Si aquest arxiu no apareix i tampoc s'usa el paràmetre *--password*, mkosi establirà la contrasenya per defecte que tingui l'usuari "root" al sistema a construir, la qual normalment sol estar bloquejada).

Un cop generada la imatge, tenim diferents possibilitats d'ús:

*Si no és arrencable (és a dir, si no hem indicat el paràmetre *-b* a *mkosi*), podem entrar-hi fent-la servir com a contenidor amb la comanda (inclosa dins el paquet "systemd-container"):

sudo systemd-nspawn -i imatge.raw

o bé, si volem iniciar Systemd dins del propi contenidor (això té certes avantatges com per exemple la posta en marxa del canal D-Bus dins del contenidor -del qual depèn el funcionament de certes aplicacions- o bé -que està relacionat- poder comunicar aquest amb la màquina amfitriona, entre altres) podem fer en canvi:

sudo systemd-nspawn -bi imatge.raw

NOTA: Si la imatge fos una carpeta o subvolum en comptes d'un fitxer "raw", caldria executar llavors la comanda *systemd-nspawn -D carpeta (o systemd-nspawn -bD carpeta)*
NOTA: L'ús d'imatges mkosi en contenidors LXC/LXD no està documentat

*Si sí és arrencable (és a dir, si sí hem indicat el paràmetre *-b* a *mkosi*), a més de poder fer servir la imatge com un contenidor igual que com si no ho fos, podem generar a més una màquina virtual Qemu amb una comanda similar a aquesta...:

qemu-system-x86_64 -m 512 -enable-kvm
-bios=/usr/share/OVMF/OVMF_CODE.fd
-drive if=virtio,format=raw,file=imatge.raw

...o, alternativament, fer servir Libvirt (i així fins i tot "cockpit-machines" podria treballar-hi amb ell), així:

virt-install -n nomMaquina -r 512 --disk path=imatge.raw --boot uefi --import

NOTA: És molt recomanable afegir a la comanda virt-install anterior el paràmetre *--graphics vnc,listen=0.0.0.0,port=5910* per tal de tenir permanentment activat un servidor VNC mentre la màquina estigui encesa i així poder veure la pantalla de la màquina virtual des d'un client VNC qualsevol

També es pot escriure aquesta imatge arrencable "imatge.raw" directament a un pendrive USB mitjançant *dd* i fer-lo servir com a sistema d'arranc.

Quadre resum del significat del paràmetre *-b* a *mkosi* i a *systemd-nspawn*:

****mkosi* sense paràmetre *-b***: La imatge només conté un sistema arrel "/". Per tant, la imatge només es pot fer servir com a contenidor.

****mkosi* amb paràmetre *-b***: La imatge conté el sistema arrel "/" i una carpeta "/boot" on hi ha un kernel+initrd a dins. Per tant, la imatge es pot fer servir tant com a contenidor com a disc dur d'una màquina virtual KVM

****systemd-nspawn* sense paràmetre *-b***: No es pot usar Systemd dins del contenidor

****systemd-nspawn* amb paràmetre *-b***: Sí es pot usar Systemd dins del contenidor

EXERCICIS *mkosi* i *systemd-nspawn* bàsic

1.-a) Entra en una màquina VirtualBox Fedora que tinguis a mà, converteix-te en administrador (per exemple, amb *sudo -i*). Instal.la els paquets "mkosi" i "systemd-container" (i, a més, necessitaràs els paquets "apt" i "zstd"). Seguidament, crea dins de la carpeta /root una subcarpeta anomenada "mkosi.cache". Finalment, mou-te dins d'aquesta carpeta /root i executa allà la comanda *mkosi* adient per tal de generar una imatge amb les següents característiques (atenció, trigarà una estona!):

-D'un sistema Fedora 37
-Anomenada "fedora.raw"
-De tipus "gpt_ext4"
-Arrencable
-Amb els paquets "dnf", "nano", "iputils", "iproute" i "systemd-resolved" extra instal.lats
-Amb un usuari "root" que tingui com a contrasenya "1234"
-Amb una mida per la partició arrel del sistema de 2G

b) Torna a executar la comanda *mkosi* des de la mateixa carpeta però ara per generar una imatge amb les següents característiques (atenció, trigarà una estona!):

-D'un sistema Ubuntu Jammy
-Anomenada "ubuntu.raw"
-També de tipus "gpt_ext4"
-No arrencable
-Amb els paquets "apt", "nano", "iputils-ping" i "iproute2" extra instal.lats
-Amb un usuari "root" que tingui com a contrasenya "1234"
-Amb una mida per la partició arrel del sistema de 2G

2.-a) Executa la comanda *systemd-nspawn* adient per tal d'arrencar la imatge "fedora.raw" com un contenidor sense Systemd al seu interior. Un cop a dins del contenidor, executa la comanda *uname -r* per confirmar quin és el kernel que està fent servir el contenidor. Surt del contenidor executant la comanda *exit*

aII) Es pot executar una comanda individual finita directament dins d'un contenidor sense haver "d'entrar" explícitament dins d'ell gràcies al paràmetre *-a*. Executa la comanda: *systemd-nspawn -i fedora.raw -a uname -r* i comprova que obtens el mateix resultat que a l'apartat anterior.

b) Llegeix aquest article: https://es.wikipedia.org/wiki/Loop_device i seguidament executa, dins del contenidor, la comanda *lsblk* per veure els dispositius de bloc muntats. Observa i raona en quin dispositiu està muntada la carpeta arrel del sistema. ¿Té coherència amb el que veus en executar *df -hT* ?

c) Què és el que veus en fer *ls /* dins del contenidor? I en fer *ls /home*? Per què?

d) Què és el que veus en fer *ps -e*? Per què?

e) Executa la comanda *systemd-detect-virt* dins del contenidor. ¿Per a què serveix?

f) A l'interior de la carpeta */efi/EFI/Linux* de dins del contenidor es troba un fitxer *.efi que representa un kernel+initrd de tipus "Stub". ¿Quan/Per a què creus que es podria fer servir aquest fitxer?. Surt del contenidor un altre cop

3.-a) Arrenca la mateixa imatge "fedora.raw" però ara amb el seu Systemd propi. ¿Quina diferència evident hi trobes només arrencar-la respecte l'arrencada sense Systemd?

b) Si ara tornes a fer tots els apartats de l'exercici anterior (a, b, c, d, e, f) ¿en quin d'aquests apartats veuràs alguna diferència respecte les respostes ja trobades anteriorment i en quins no? ¿Per què?

c) Torna a sortir del contenidor. Ara, però, ho hauràs de fer pulsant tres vegades seguides "CTRL+]" o bé "CTRLdret + 5" (si escrius *exit* l'unic que faràs és sortir de la sessió d'usuari)

4.-a) Executa de nou la comanda *systemd-nspawn* però ara per arrencar la imatge "ubuntu.raw" al seu interior (amb o sense Systemd propi, és irrellevant per l'exercici). Un cop a dins del contenidor, executa la comanda *uname -r* per confirmar quin és el kernel que s'està fent servir. ¿Veus alguna diferència amb el kernel vist a l'apartat a) de l'exercici 2? Per què?

b) Executa dins del contenidor la comanda *lsblk* ¿Quants dispositius "loop" veus ara? ¿On estan muntats cadascun? ¿Veus alguna diferència amb la sortida obtinguda a l'apartat b) de l'exercici 2? En aquest sentit, ¿quines conseqüències té que no existeixi cap punt de muntatge "/efi" (ni partició associada)? ¿Què hagués calgut fer per a què sí que existís?. Surt del contenidor.

5.-a) Crea dins de la carpeta "/etc" de la màquina amfitriona un arxiu de text anomenat "pepe.conf" amb un contingut qualsevol i dins de la carpeta "/opt" de la màquina amfitriona un altre arxiu de text anomenat "paca.conf" amb un altre contingut qualsevol. Executa seguidament la comanda *systemd-nspawn -i ubuntu.raw --bind /etc/pepe.conf:/etc/manolo.conf --bind /opt --bind-ro /etc/fstab*?

*¿Què veus dins de la carpeta "/opt" del contenidor? ¿Per què?
*¿Pots modificar el contingut del fitxer "/opt/paca.conf" en el contenidor?
*¿Quin és el contingut del fitxer "/etc/fstab" en el contenidor? ¿El pots modificar? ¿Per què?
*¿Quin és el contingut del fitxer "/etc/manolo.conf" en el contenidor? ¿El pots modificar? ¿Per què?
*¿Per què no existeix el fitxer "/etc/pepe.conf" en el contenidor *?*

NOTA: Si ho necessites, consulta l'apartat que parla del paràmetre *--bind* de la pàgina de manual de *systemd-nspawn* per respondre la darrera pregunta

b) ¿Què passa si executes la comanda *systemd-nspawn -i ubuntu.raw --read-only* i tot seguit, dins d'aquest contenidor, executes *touch a.txt*? ¿Té a veure amb algun valor que aparegui sota la columna OPTIONS mostrada en executar *findmnt /* ? ¿Té a veure amb el valor "1" que apareix sota la columna RO mostrada en executar *lsblk*?

bII) Executa la comanda *systemd-nspawn -i fedora.raw --volatile* ; seguidament, dins d'aquest contenidor executa *touch a.txt*, surt del contenidor i finalment torna a entrar amb *systemd-nspawn -i fedora.raw* ¿Existeix el fitxer *a.txt*? ¿Per què? ¿Té a veure amb algun valor que aparegui sota la columna FSTYPE mostrada en executar *findmnt /* ?

NOTA: Una opció similar a la pràctica a *--volatile* (encara que internament implementada de forma diferent) és *–ephemeral* (*-x*)