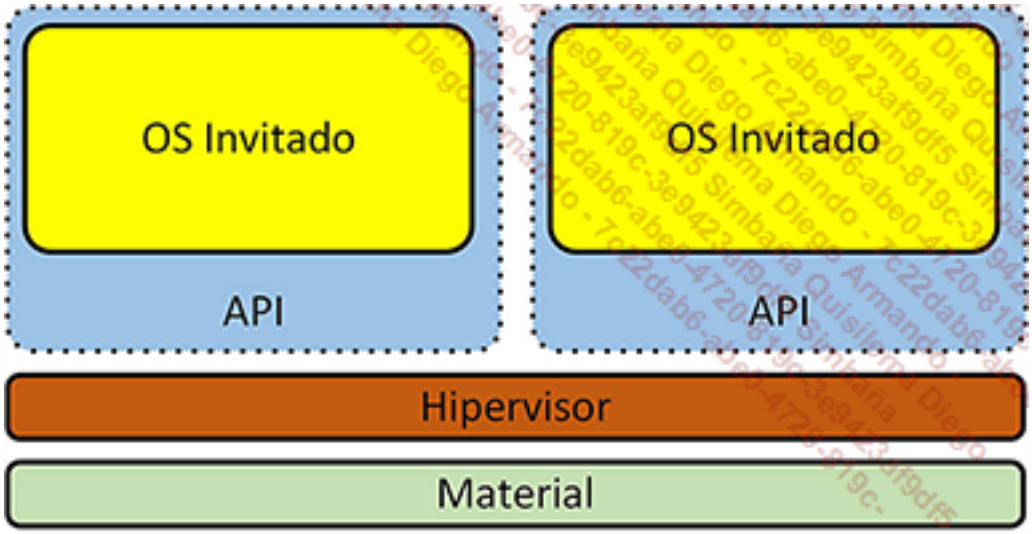
# Paravirtualización

## 1. Principio

Cuando nos encontramos en presencia de un hipervisor de tipo 1 o 2, el sistema anfitrión puede proponer una interfaz aplicativa optimizada y similar a la del material real. Es lo que se llama paravirtualización. La interfaz aplicativa API permite que el sistema invitado tenga un acceso casi directo a uno o a varios componentes materiales.




*Paravirtualización: el sistema accede al anfitrión gracias a las API*

Esta posibilidad la puede ofrecer el núcleo, la arquitectura material (por ejemplo el microprocesador), o incluso los dos.

En el caso del acceso al material realizado por los sistemas invitados los pilotos que se instalarán establecerán una comunicación directa con la API del sistema anfitrión.

En Linux los productos XEN y KVM utilizan técnicas de paravirtualización si están disponibles.

## 2. Virtio

**Virtio**, *Virtual Input Output*, se trata de la interfaz de programación (API) del núcleo Linux dedicada a los pilotos de los periféricos de las máquinas virtuales (o más bien de los sistemas invitados).

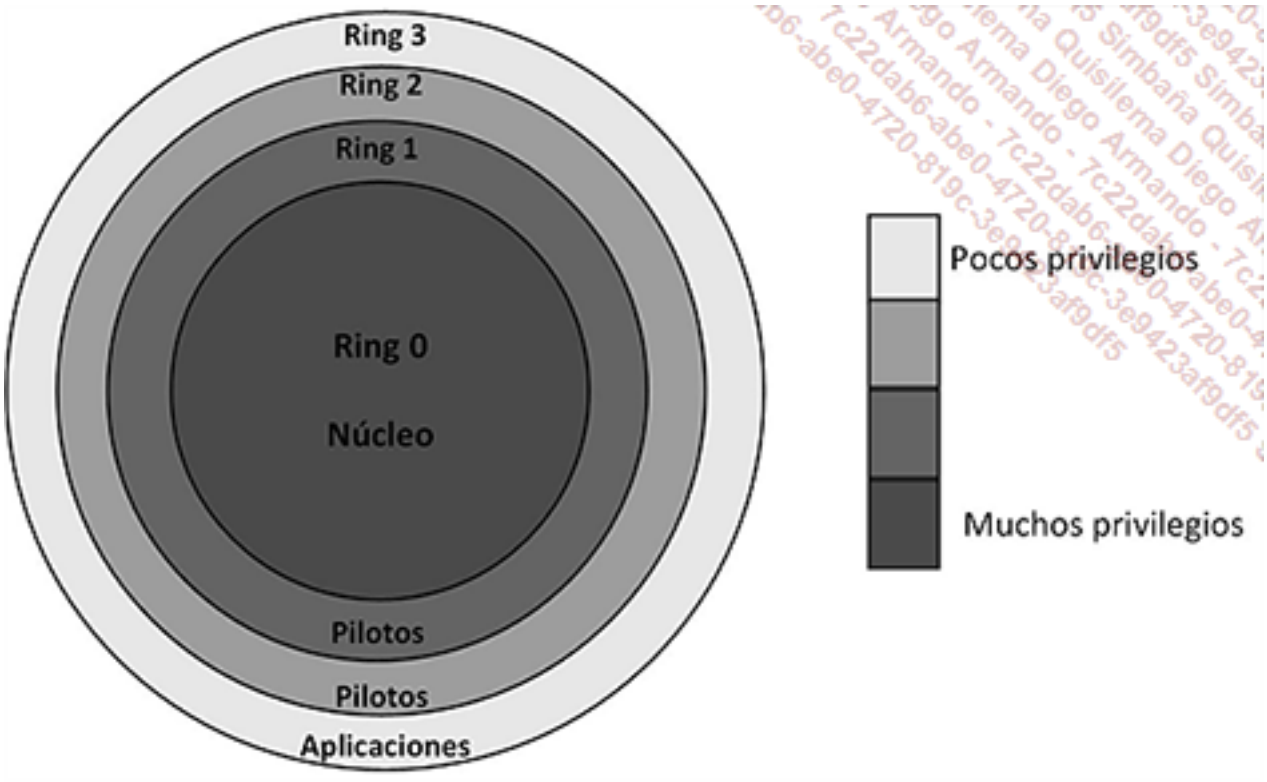
Una comunicación de tipo FIFO se establece entre el hipervisor (el núcleo del sistema anfitrión) y el núcleo del sistema invitado. Los pilotos se basarán en esta interfaz.

Virtio es la API por defecto de KVM, y Linux propone los pilotos de red y discos asociados. Hay otros pilotos que también están disponibles para Windows. VMWare y Virtualbox disponen de un soporte VirtIO.

# 3. Asistencia material

### a. Anillos de protección

Con el objetivo de mejorar el rendimiento de los supervisores, los fabricantes de microprocesadores añaden modos de funcionamiento y juegos de instrucciones suplementarios. Los procesadores de tipo x86 disponen de niveles de ejecución, o anillos de protección llamados **rings**. Estos anillos definen los privilegios de ejecución de los programas. Cuanto más bajo sea el número de anillo en el que está instalado un programa, más control ejercerá en el sistema.




*Los anillos definen los privilegios*

En la arquitectura x86 32 bits, existen cuatro anillos 0, 1, 2 y 3. En el material 64 bits, ya que se constata que los niveles 1 y 2 no se utilizan, solamente existen dos: 0 y 3. el sistema operativo funciona en el ring 0 y dispone del más alto nivel de control mientras que las aplicaciones operan en el ring 3, el más elevado. Estas no pueden modificar lo que se ejecuta en los rings inferiores al suyo. De esta manera, una aplicación no puede parar el sistema operativo, mientras que este sí puede hacerlo.

- Ring 0: el sistema operativo
- Ring 3: las aplicaciones

### b. Anillos y virtualización

En un entorno virtualizado ya no solo hay un sistema operativo sino varios. El primero es el hipervisor, necesita un nivel de privilegios muy alto ya que controla los recursos físicos y los sistemas invitados. En las máquinas virtuales de este hipervisor se instalan otros sistemas operativos. Como el hipervisor se encuentra en el ring 0, hay que ponerlos en el ring 3; ahora bien, los sistemas operativos se conciben para ser ejecutados en el ring 0.

Estos comprueban regularmente que se encuentran en este ring, ya que hay muchas instrucciones que solo se ejecutan desde el ring 0 o hacia este. Además, funcionar en un ring más bajo que las otras aplicaciones les garantiza poder controlarlas. Para evitar este problema hay dos métodos. Uno de ellos es la paravirtualización (utilizada por Xen) que consiste en modificar los sistemas operativos invitados para permitir que se ejecuten en otro ring que no sea el 0. Una translación binaria confunde el sistema operativo con respecto al lugar que ocupa realmente en el sistema interceptando algunas de las operaciones. Sin embargo esto provoca que el hipervisor tenga que trabajar más y, por lo tanto consumir más recursos.

## 4. AMD-V e Intel-VT

Intel y AMD han querido suprimir este problema de vigilancia, intercepción y traslación binaria del código, provocado por el cambio de ring. Para ello han modificado la arquitectura y los juegos de instrucciones de sus procesadores:

- AMD con Pacífica, ahora llamado AMD-V
- Intel con Vanderpool, ahora llamado INTEL-VT

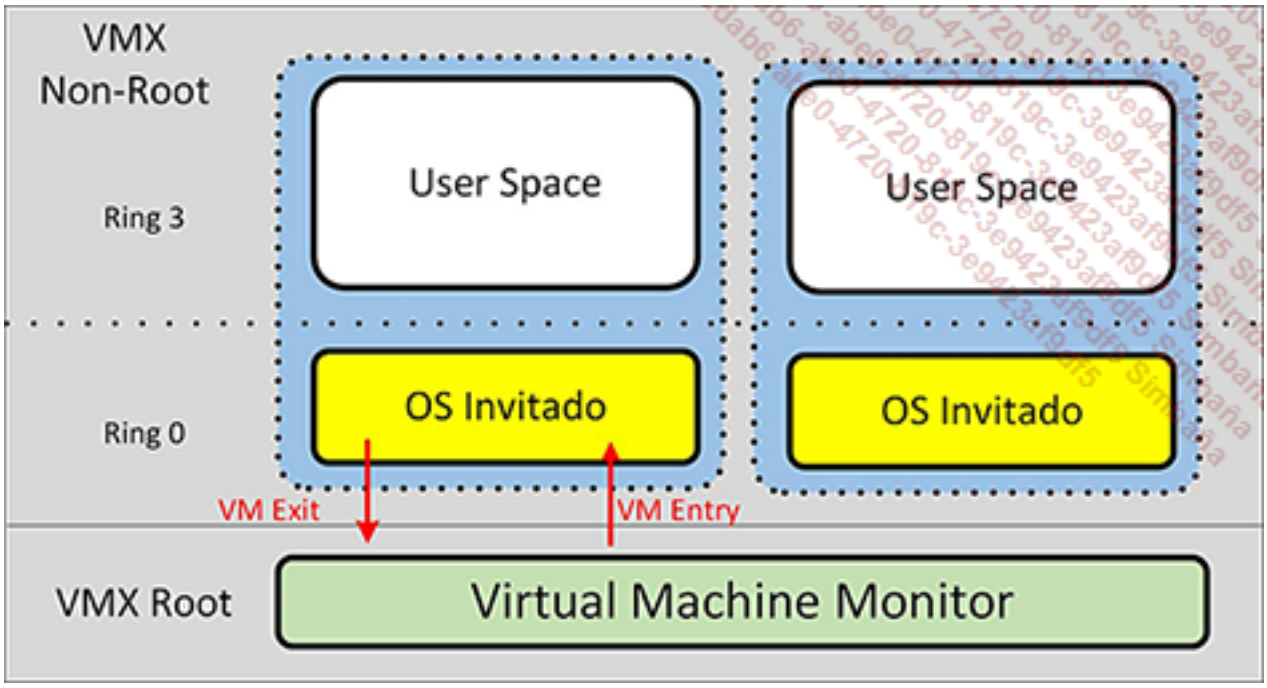
Las dos tecnologías son equivalentes aunque incompatibles. La descripción siguiente solo se basa en Intel-VT. La idea general es simple. Nuevas instrucciones permiten cambiar el procesador hacia un nuevo modo de ejecución dedicado a los sistemas invitados y viceversa. De esto resultan dos modos de ejecución distintos del procesador:

- VMX root: modo usado por el VMM, el hipervisor (sistema anfitrión)
- VMX non-root: modo usado por los sistemas invitados.

Cada uno de estos dos módulos dispone de los anillos de 0 a 3 (32 bits) o de 0 y 3 (64 bits). Un sistema invitado que no tenga opción de esos dos modos de ejecución se ejecutará en el ring 0 en el modo ejecución VMX non-root. No se necesitará que el sistema invitado sea modificado y, por lo tanto, la translación binaria no será necesaria.

El hipervisor utiliza el modo de ejecución VMX root, accediendo de esta manera a un nivel de control y de privilegios más importante. El hipervisor tiene que ser modificado para poder usar la extensión Vanderpool y administrar los nuevos modos de ejecución VMX.

El modo de ejecución VMX root presenta nuevas instrucciones dedicadas al hipervisor para facilitar la compartición y el control de recursos de la máquina anfitriona y mejorar su rendimiento. Los cambios, respaldos y restauraciones de contextos entre los sistemas invitados son gestionados en el procesador por instrucciones explícitas (VM entry, VM exit...). Los contextos están almacenados en estructuras llamadas VMCS (*Virtual Machine Control Structure*) suministradas por el procesador y guardadas en una zona reservada de la memoria.




*Intel-VT: Cada modo VMX dispone de los anillos 0 y 3*

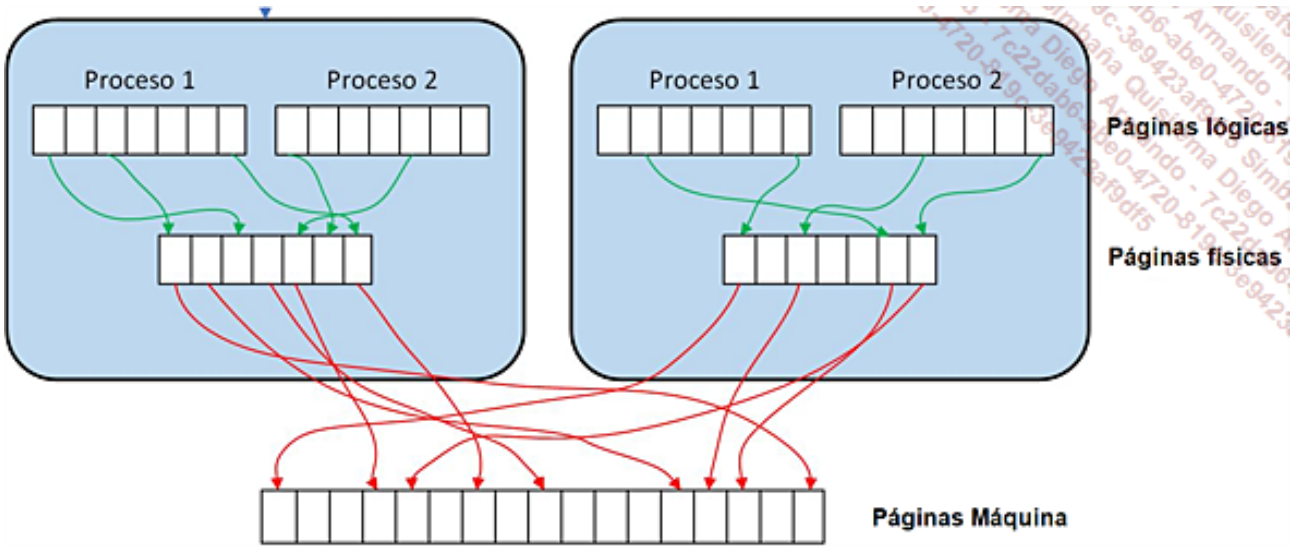
## 5. Virtualización de la memoria

Instalado por sí solo, un sistema operativo se ocupa íntegramente de la gestión de la memoria. Asigna zonas de memoria según el principio de gestión lineal, realizado en una memoria virtual que incluye a la vez la memoria viva (tarjetas de memoria) y los archivos o particiones de intercambio (swap). La gestión de la correspondencia entre la administración de zonas de memoria virtuales y de zonas de memoria física la hace un componente específico llamado MMU (*Memory Management Unit*). La MMU actualiza una tabla de correspondencia de páginas de memoria llamada PT Page Table o sobre todo TLB (*Translation Look-aside-Buffer*).

La MMU también se usa en 64 bits, aunque solo sea para la paginación. Solo la segmentación se elimina por completo en beneficio del modelo lineal.

Bajo la presencia de sistemas invitados, el hipervisor (VMM) atribuye a cada uno cierta cantidad de memoria definida en el arranque y reservada en el seno de la memoria virtual del anfitrión. Esto es un problema, porque el sistema invitado debe acceder a esta zona de memoria como si se tratara de un sistema anfitrión: el direccionamiento tiene que empezar en 0 y no en la dirección dada por la MMU. El VMM tiene que gestionar una tabla de páginas « caché » para hacer coincidir las direcciones de las páginas de memoria entre el anfitrión y el invitado. Esta tabla se llama SPT, *Shadow Page Table*. Los accesos de memoria ralentizan el sistema invitado porque se administran localmente.

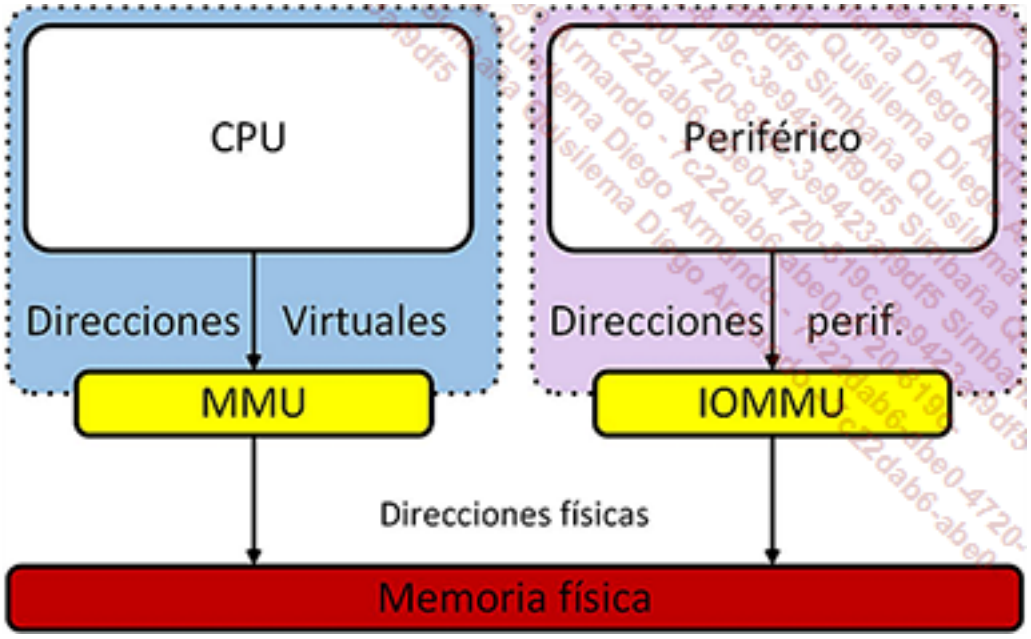
Los procesadores Intel y AMD proporcionan las tecnologías EPT (*Extended Page Tables*) (para Intel) y RVI (*Rapid Virtualization Indexing*) (para AMD), básicamente idénticas. Estas tecnologías permiten implementar materialmente, a través de instrucciones, las SPT. Estas son administradas directamente por la MMU. Hay una tabla EPT para cada sistema invitado, que puede acceder a su memoria sin tener que pasar por el hipervisor, lo que provoca una pequeña pérdida de velocidad.




*EPT administra materialmente los accesos de memoria de una VM*

## 6. Virtualización de los periféricos

Si el acceso al procesador es ahora transparente gracias a las tecnologías Intel-VT y AMD-V, el acceso a los periféricos se hace esencialmente por emulación dentro del hipervisor: cada programa deberá crear sus propios periféricos virtuales. Los drivers adaptados a estos últimos deben ser desarrollados y el rendimiento se ve, en general, degradado (acceso a los discos, tarjetas gráficas, redes). Los hipervisores deben también reproducir la gestión de la memoria de los periféricos. Sin embargo, esto no es suficiente: si un programa de la máquina virtual intenta acceder directamente al material, correrá el riesgo, simplemente, de no funcionar. Los dos constructores proponen una nueva tecnología llamada IOMMU (*Input Output Memory Management Unit*).




*Con IOMMU los periféricos virtuales obtiene acceso directo a los periféricos físicos*

Estas tecnologías permiten crear un acceso virtual a los periféricos físicos de entrada y salida. Si la máquina virtual accede a un periférico como una tarjeta gráfica, la máquina anfitriona perderá el acceso, lo que presupone que haya una tarjeta gráfica dedicada a la máquina virtual. Se pueden compartir la mayoría de los periféricos, desde los puertos USB hasta los conectores PCI-Express (y, por lo tanto, las tarjetas instaladas), teniendo siempre en cuenta que el acceso es exclusivo.

Tenga precaución, porque no todos los procesadores y chipsets soportan todas las tecnologías. Deberá tener cuidado en comprobar en las especificaciones del fabricante, tanto del procesador como de la tarjeta madre, que estas tecnologías están soportadas. Además, algunos periféricos solicitan drivers específicos para poder ser utilizados con

VT-d o AMD-Vi.

Tenga en cuenta también la existencia de VT-c, para las redes, VMDQ (archivos de periféricos), o GVT-d/g/s para las tarjetas gráficas Intel.

En la práctica, la virtualización de los periféricos tiene, esencialmente, dos usos: ofrecer capacidades gráficas correctas a una máquina virtual (dedicándole una tarjeta gráfica) o - lo que es más corriente - permitir que una máquina en un servidor pueda disponer de su propia tarjeta de red. Por otro lado, existen tarjetas madre dedicadas para ese tipo de instalación (y que disponen de distintas tarjetas de red).

# 7. Seguridad

Ya que se trata de un servidor como otro cualquiera, la máquina virtual debe responder a las mismas exigencias de seguridad que se expusieron en el capítulo correspondiente en este libro. Sin embargo, hay que recordar que la seguridad del sistema invitado está directamente ligada a la seguridad del anfitrión (hipervisor), y de la red asociada.

Los riesgos a menudo se asocian con problemas de configuración o a productos que presentan fallos, que ya no están soportados, son antiguos y, a veces, están mal parcheados. Los problemas también pueden estar relacionados con el firmware, pero de manera general, se trata de problemas aplicativos, que pueden ser solucionados a ese nivel.

Desde enero de 2018, vemos como aparece una amenaza mucho más importante, cuyas correcciones parecen más complejas: los agujeros de seguridad de tipo hardware, especialmente, relacionados con los procesadores Intel y AMD. Los más conocidos son Spectre y Meltdown, Foreshadow o ZombieLoad. Estos explotan tecnologías internas de los microprocesadores, como la predicción de rama, ejecución en desorden o la memoria intermediaria, estos fallos permiten, en ciertas condiciones, ganar un acceso a la memoria del sistema, pasando por encima de la seguridad inducida por el aislamiento de los procesos o la segmentación de la memoria.

Es grave en el caso de un servidor clásico, pero es aún peor en un hipervisor, ya que una máquina virtual podría tener acceso a datos de otra máquina.

Existen parches; la mayoría de ellos suprimen la funcionalidad que provoca estos fallos o

la vuelven a implementar de manera aplicativa. El parche definitivo, si sale algún día, implicará un cambio de procesador o de chipset. En noviembre de 2019, Intel corrigió 77 fallos de seguridad en sus procesadores. Intel parece más impactado que AMD.

El problema más grande que generan estos parches es que ralentiza, a menudo de manera significativa, el sistema: los fallos casi siempre se presentan en los mecanismos de aceleración material.

¿Qué pasará el día que se encuentre un fallo en las extensiones AMD-V o Intel VT?

# 8. Consideraciones prácticas

Cuando crea una máquina virtual Linux, el resultado no es muy diferente al de un servidor físico, pero hay algunas adaptaciones que pueden ser necesarias antes de arrancar la máquina, y también dentro de la misma.

La primera será adaptar la configuración de su sistema operativo: memoria, CPU, discos, tipo de red, etc. Pero sobre todo, se mirará la aceleración gráfica y la mejora de la velocidad de la máquina virtual. No dude en activar las opciones de paravirtualización y de aceleración material, especialmente activando virtio para el almacenamiento, las redes y la memoria.

- Los periféricos de tipo bloque se verán como /dev/vd[a-z].
- Los periféricos de red pueden guardar el nombre eth[0-9], veth[0-9], net[0-9] o presentar un nombre de tipo ensXXX, dependiendo de la distribución.

Para saber si usa los drivers virtio, use **lsmod**:

```
$ lsmod | grep virtio
virtio_pci
virtio_net
...
```

La segunda adaptación será comprobar, especialmente si ha clonado un servidor o si usa una imagen o un modelo que ya existía, que los parámetros del servidor se hayan

reinicializado como la generación de nuevos UUID para los volúmenes de datos o las direcciones MAC para las interfaces de red virtuales. Piense en suprimir el archivo /etc/maquina-id y el archivo /var/lib /dbus/maquina-id.

En el cloud, estas tareas las gestiona cloud-init.