Proyecto Fin de Carrera

Ingeniería de Telecomunicación

Diseño de un entorno de trabajo para PYMES mediante virtualización sobre Proxmox VE

Autor: Carlo de Cástulo Navarro Álvarez

Tutor: Juan Antonio Ternero Muñiz

Departamento de Ingeniería Telemática

Escuela Superior de Ingenieros

Universidad de Sevilla

Sevilla, 2016

Escuela Técnica Superior de
INGENIERÍA DE SEVILLA

[PÁGINA DEJADA EN BLANCO A PROPÓSITO]

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Diseño de un entorno de trabajo para PYMES mediante virtualización sobre Proxmox VE

Autor:

Carlo de Cástulo Navarro Álvarez

Tutor:

Juan Antonio Ternero Muñiz

Profesor Titular

Dep. Ingeniería Telemática

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2016

[PÁGINA DEJADA EN BLANCO A PROPÓSITO]

Proyecto Fin de Carrera:

Diseño de un entorno de trabajo para PYMES mediante virtualización sobre Proxmox VE

Autor: Carlo de Cástulo Navarro Álvarez

Tutor: Juan Antonio Ternero Muñiz

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2016

El Secretario del Tribunal

Índice de contenido

Índice de tablas

Índice de figuras

Capítulo I: Introducción

1. Motivación del proyecto

Para entender los factores que llevan al nacimiento de la idea principal de este proyecto, quiero empezar hablando del contexto en el que surge. Informaticasa Soluciones S.L. (informática – telecomunicaciones – soluciones) es una pequeña empresa recién creada en 2015, con base en el municipio de Aracena, en la Sierra de Aracena y Picos de Aroche (Huelva). Esta empresa la formamos un equipo de jóvenes multidisciplinares con el objetivo de atender las necesidades en materia de informática y comunicaciones (y tecnológicas en general) en nuestro entorno de la sierra, poniendo el foco principalmente en profesionales y empresas. Nos lanzamos a esta aventura tras llevar unos siete años moviéndonos como trabajadores autónomos y teniendo ya una perspectiva de las características y necesidades de la zona.

La Sierra de Aracena es un entorno rural, formado por pequeños pueblos y aldeas cuya capital comarcal es Aracena, una localidad con unos 8000 habitantes. El tejido empresarial está formado principalmente por pequeñas empresas, las cuales en su mayoría no alcanzan los diez empleados. A lo largo de este tiempo trabajando por la zona hemos observado que la penetración digital en los negocios es todavía mínima, y en la mayoría de los casos problemática. La alfabetización digital tiene todavía mucho camino por recorrer y los profesionales se ven empujados a usar herramientas que no controlan y no aprovechan.

Hoy día es muy difícil encontrar una empresa que no dependa de un ordenador para su trabajo diario, por muy pequeña que sea. Un bar o restaurante que tiene un TPV para emitir

tickets o diseñar la carta, un taller mecánico con software de diagnóstico de vehículos, estudios de arquitectura o administraciones locales pequeñas. Bases de datos de clientes, programas de facturación, contabilidad, diseños, documentación y todo tipo de información crucial para el funcionamiento de estas empresas se almacena sobre infraestructuras muy frágiles, en equipos obsoletos, mal cuidados y sin medidas de protección alguna.

Los conceptos de redundancia, copias de seguridad, ley de protección de datos, monitorización, etc. parecen reservados para las grandes empresas y suelen aprenderse demasiado tarde. Y cuando sí se tienen en cuenta, en la mayoría de los casos se aplican mal y resultan ineficaces. Aparecen en escena cuando surge un problema y se pierde información valiosa o hay que detener la actividad comercial durante un tiempo mientras se soluciona la incidencia. Es muy común la sensación en estas empresas pequeñas de que la digitalización de la información sólo trae carga de trabajo extra y problemas. La inversión en la infraestructura tecnológica parece algo secundario y se intenta reducir al mínimo. No hay conciencia de que, cada día más, conforma los cimientos de la empresa y se desconoce todo lo que puede llegar a aportar.

Por poner un ejemplo práctico para entender mejor este apunte, cabe quizás hablar de una amenaza a la que nos hemos enfrentado con bastante frecuencia en los últimos tiempos en Informaticasa: Cryptolocker y sus variantes, o el ramsomware en general[1][2][3]. Este tipo de malware cifra la información en el disco duro y pide un rescate para su recuperación. Se propaga a discos externos y ubicaciones de red. Han sido demasiados los casos de empresas (y particulares) en que se ha perdido prácticamente todo lo digitalizado. Uno de ellos, especialmente grave, nos lo encontramos en una pequeña empresa con cuatro puestos en red. El malware afectó a los cuatro equipos y al disco USB donde almacenaban las copias de seguridad, por lo que no quedó nada a salvo. El impacto fue enorme y optaron incluso por pagar el rescate para recuperar lo perdido. Un buen sistema de copias de seguridad, apartadas y no accesibles, habría minimizado el daño causado por este incidente.

1 http://www.bleepingcomputer.com/virus-removal/threat/ransomware/
2 http://www.pandasecurity.com/spain/mediacenter/malware/cryptolocker/
3 https://es.wikipedia.org/wiki/CryptoLocker

De forma un tanto paradójica, y gracias principalmente a la gran aceptación del smartphone, términos como "la Nube", el "Cloud Computing", o servicios como Dropbox, Google Drive, iCloud, etc. sí son conocidos por una mayoría. Están presentes en las tareas cotidianas de los ciudadanos pero no se suelen aprovechar en el entorno profesional. Además, gracias a la reducción de los precios de la electrónica y al mundo del Software Libre, es posible elaborar soluciones de altas prestaciones y bajo coste.

En este documento se pretende diseñar un sistema de alta fiabilidad, flexible, escalable y de bajo coste que pueda servir como base para una infraestructura de telecomunicaciones de garantías a cualquier organización. A su vez, se estudiarán y experimentará con técnicas de virtualización. Esta tecnología es en gran medida la madre de la famosa "Nube", será la base de nuestro sistema y es un campo puntero en los avances informáticos de la actualidad.

2. Objetivos y alcance

El núcleo del sistema a desarrollar será un equipo (o varios, según los requisitos de la instalación) que proporcione múltiples servicios de forma segura y ágil a los usuarios.

Siempre existe y se ha contemplado la opción de recurrir a servicios de terceros como el alquiler de servidores privados virtuales externos, pero, al menos en esta zona, no es viable debido a la calidad de los accesos a internet. En la mayor parte de la Sierra de Aracena se dispone, como máximo, de acceso ADSL con 10mbps de descarga y 1mbps en el enlace ascendente. Y no son pocas las zonas donde se reduce a 6mbps/600kbps o acceso rural por satélite, con altas latencias y aún menor ancho de banda. El trabajo con ficheros de gran tamaño se hace prácticamente imposible: una copia de un fichero de 10MB llevaría casi 2 minutos en completarse con una velocidad de subida de 800kbps. Por otro lado, no solo la velocidad del acceso es el problema. La baja calidad de las instalaciones y la frecuencia con que se producen cortes, hacen que sea muy arriesgado externalizar los servicios y depender de ellos. Es preferible mantener el grueso del trabajo en red local y dejar "la nube" como herramienta secundaria para usos puntuales.

Teniendo en cuenta que deseamos que este sistema sea apto también para pequeñas empresas, con poca base tecnológica, debemos esforzarnos en que sea lo más transparente posible para el usuario. No debe suponer un cambio radical en su modo de trabajo ni implicar un

esfuerzo significativo de adaptación. Deberá ser suficientemente flexible como para permitir una inversión inicial mínima e ir ampliando de forma sencilla conforme cambien las necesidades.

Entrando ya en los aspectos técnicos que se persiguen:

- La base, la capa física de nuestro sistema, será hardware fiable y redundante. Esto proporcionará protección frente a posibles fallos en la electrónica. Según los requisitos del despliegue, la **redundancia** la podremos conseguir mediante servidores trabajando en **cluster** (más adelante se profundizará en este concepto) o discos duros en **RAID**.
- **Abstracción del hardware**. Este es quizás el aspecto central y más destacable de este proyecto. Los servicios que se ofrecen al usuario no estarán sujetos y limitados por los equipos instalados. Mediante **virtualización** eliminaremos la dependencia de la electrónica. Esto facilita la recuperación de desastres, las ampliaciones son menos costosas y más sencillas, el hardware obsoleto se puede ir sustituyendo sin afectar al software y los tiempos de parada de los sistemas para actualizaciones o reparaciones se reducen notablemente. Una desventaja que podríamos achacar a la virtualización es la sobrecarga de procesamiento que conlleva. Pero esto, hoy en día, con la potencia de cálculo que presenta incluso el ordenador personal más económico, es un factor que podemos despreciar.
- En los dos puntos anteriores nos hemos centrado en el hardware, en limitar los problemas que nos puede ocasionar y en evitar la dependencia con equipos específicos. La siguiente capa es la de las aplicaciones. ¿Para qué se usará el sistema? ¿Qué servicios requieren los usuarios? No importa. **Podremos desplegar tantos servidores virtuales** (siempre teniendo en cuenta las limitaciones del soporte físico elegido) **como necesitemos**. Esto nos permitirá modularizar la infraestructura manteniendo servicios separados y manejar configuraciones sencillas. No tener que mezclar en una misma máquina diferentes tareas aporta además una mejora considerable en la seguridad de nuestra red.

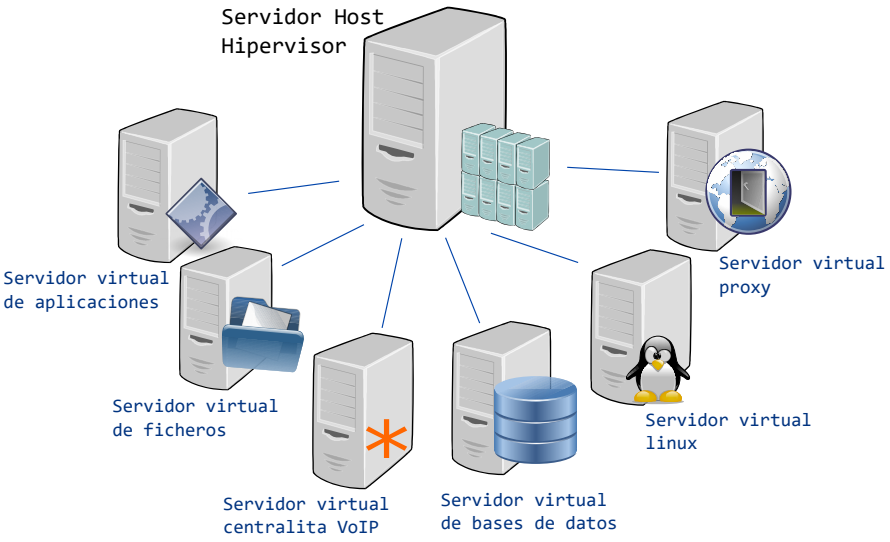


Ilustración 1: Servidores virtuales de ejemplo.

- Una vez tenemos las aplicaciones funcionando sobre nuestros servidores nos encontramos con otro punto muy importante: la seguridad de esta información. La redundancia hardware protege frente a un fallo de un componente físico, pero no impide la pérdida de datos por un borrado accidental o el ataque de algún programa malintencionado. **Las copias de seguridad** son una parte vital en una infraestructura que pretenda pasar por fiable. El hecho de trabajar con máquinas virtuales vuelve a ser clave en este aspecto: podremos realizar copias de los servidores completos sin necesitar ni siquiera pausar su funcionamiento. Al tratarse además de instantáneas completas el tiempo de restauración es muy reducido. No es necesario reinstalar y reconfigurar los servicios dañados. Otra característica muy destacable de este sistema de copias es su ocultación. Como se realizan en un nivel inferior al que trabaja el usuario no están fácilmente accesibles y se protegen de forma natural.

En este proyecto analizaremos las distintas técnicas y plataformas de virtualización disponibles, desarrollaremos un entorno de trabajo sobre Proxmox VE que nos permita alcanzar los puntos anteriores y se crearán algunos servidores virtuales de ejemplo.

3. Estructura de la memoria

Para abordar los objetivos presentados en la sección anterior este documento se organizará del siguiente modo:

- ***Capítulo 2: Estado del arte.*** En esta sección se realiza un estudio teórico sobre virtualización, analizando las técnicas y las plataformas existentes en el mercado.
- ***Capítulo 3: Proxmox Virtual Environment.*** Análisis en profundidad de la plataforma de virtualización Proxmox, que será la que se emplee como base para desarrollar nuestro sistema.
- ***Capítulo 4: Desarrollo del entorno de trabajo.*** En este capítulo plantearemos la configuración hardware: equipos, almacenamiento, sistema de copias de seguridad, etc. y desarrollaremos el servidor base sobre el que correrán las máquinas virtuales.
- ***Capítulo 5: Creación de servidores virtuales.*** Una vez se ha puesto en marcha el servidor ***hipervisor***, pasaremos a desplegar algunos servicios en máquinas virtuales usando distintas técnicas (KVM y contenedores) y así probar su funcionamiento.
- ***Capítulo 6: Tareas de administración.*** Una punto fuerte de este proyecto es tratar de planificar y facilitar el mantenimiento del entorno una vez se esté trabajando sobre él, intentando no dejar nada al azar. Veremos aquí cómo configurar el sistema de copias de seguridad, algunos procedimientos de actuación ante determinadas situaciones y varias opciones para monitorizar el entorno.
- ***Capítulo 7: Conclusiones y líneas de trabajo futuro.*** Con esta sección se finaliza la memoria. Se valoran los resultados obtenidos y se plantean múltiples opciones con las que se puede continuar trabajando en este proyecto.

Cabe mencionar que para el desarrollo de este documento se ha intentado utilizar únicamente herramientas de Software Libre y se ha tenido especial cuidado en respetar las licencias de uso del material referenciado o incluido. Las principales herramientas de trabajo han sido las siguientes:

- Debian GNU Linux como Sistema Operativo en los equipos de trabajo. https://www.debian.org/

- Libreoffice, en su versión 5, para la redacción de la memoria. https://es.libreoffice.org/

- Zotero como gestor de referencias bibliográficas. https://www.zotero.org/

- Inkscape / GIMP como herramientas de retoque y creación de gráficos. https://inkscape.org/es/ - http://www.gimp.org/

- La mayoría de gráficos que no se han diseñado expresamente y algunos iconos que se han utilizado se han extraído de https://openclipart.org/. En los otros casos se han verificado los permisos de uso y se han citado expresamente sus fuentes.

Capítulo II: Estado del arte

1. La Virtualización

"Half the work that is done in the world is to make things appear what they are not"
(E. R. Beadle)

Para el término virtualización podemos encontrar numerosas definiciones:

- Acción de abstraerse de los límites físicos de una tecnología(Wolf y Halter 2005).
- Creación a través de software de una versión virtual de algún recurso tecnológico, como puede ser una plataforma de hardware, un sistema operativo, un dispositivo de almacenamiento u otros recursos de red («Virtualización» 2015).
- Engaño. Algo que no existe físicamente como tal, sino que está creado mediante software para aparentar que lo hace (Santana 2013).
- Sistema o metodología para dividir los recursos de una computadora en múltiples entornos de ejecución aislados (Kovari y Dukan 2012).
- Tecnología diseñada para proveer una capa de abstracción entre el hardware de los equipos informáticos y el software que se ejecuta sobre ellos (Ali, Susandri y Rahmaddeni 2015).

En el mundo de las tecnologías de la información este término no es nuevo. La idea de virtualización se lleva utilizando y aplicando mucho tiempo en diferentes contextos. Solo tenemos que pensar, por ejemplo, en las VLAN (Virtual Local Area Network), la máquina virtual de Java o los sistemas de almacenamiento en RAID (Redundant Array of Independent Disks[4]). En estos tres casos se están aprovechando beneficios de la virtualización:

- Las redes locales virtuales nos permiten separar subredes de forma "casi física" (en capa 2) aunque su tráfico comparta el mismo cable o switch.
- La máquina virtual de Java permite ejecutar la misma aplicación en diferentes ordenadores, independientemente de su arquitectura o sistema operativo.
- RAID permite configurar varios discos duros en distintos modos (espejo, conjunto dividido, etc.) engañando al sistema operativo. A este se le presenta un volumen estándar por lo que no necesita ninguna característica especial y puede funcionar con normalidad.

Entonces, si el término no es nuevo y lleva mucho utilizándose, ¿qué hace que sea uno de los campos más punteros y que más interés está suscitando en los últimos años? Volviendo a la primera definición: *"abstraerse de los límites físicos de una tecnología"*. Se han desarrollado métodos para aplicar la virtualización en prácticamente todos los aspectos de las redes de comunicaciones. Hoy en día se virtualizan ordenadores, servidores, redes, routers, almacenamiento, aplicaciones, etc. En definitiva, se pueden crear completos centros de datos virtuales. Entre otras ventajas, las infraestructuras virtuales aportan:

- Reducción de costes y mejor aprovechamiento de recursos.
- Mejoras en estabilidad y alta disponibilidad, así como facilidades para políticas de seguridad y aislamiento de entornos.
- Simplificación de procesos operacionales (actualizaciones, reparaciones, migraciones...).

Estos aspectos hacen que la mayoría de las empresas del mundo *IT* estén profundamente implicadas en el desarrollo de las tecnologías de virtualización y sus aplicaciones.

4 Originalmente las siglas RAID provenían de *Redundant Array of Inexpensive Disks*. La motivación de la tecnología RAID era conseguir sistemas de almacenamiento fiables aprovechando varios discos duros económicos.

Veamos una línea temporal con un resumen de avances destacados en la historia de la virtualización. No se trata de una lista exhaustiva, tan solo se muestran aquellos que se consideran más importantes en el contexto de este trabajo:

- 1962. Memoria virtual (Universidad de Manchester).
- 1972. Máquina virtual (IBM).
- 1987. RAID (Universidad de California, Berkeley).
- 1999. Virtualización x86 (VMware).
- 2001. Virtualización de almacenamiento (DataCore), VMware ESX (VMware) .
- 2003. XEN (University of Cambridge).
- 2004. Microsoft Virtual Server (Microsoft).
- 2005. OpenVZ Project (SWsoft). Proxmox Virtual Environment (Proxmox Server Solutions GmbH).
- 2007. KVM (Open Source).
- 2008. Fibre Channel over Ethernet (Cisco, Emulex, QLogic), Hyper-V (Microsoft).
- 2009. Infraestructura vSphere (VMware), Distributed Virtual Switch (VMware).
- 2010. Open vSwitch (Open Source), OpenStack (Rackspace and NASA).
- 2011. Virtual Network Data Path (Cisco), Virtual eXtensible Local Area Network (VMware, Cisco, Red Hat, Citrix and others), OpenFlow (Open Networking Foundation).
- 2012. vCloud Suite (VMware), Cisco OpenStack Edition (Cisco), and Cisco Open Network Environment (Cisco).
- 2013. Docker es liberado como Open Source.
- 2014. LXC – Linux Containers (Open Source – Canonical).

Podemos apreciar la evolución desde soluciones específicas para problemas concretos a la construcción de completos entornos de servicios integrados virtualizados. Como decíamos antes, y lo observaremos también al repasar la variedad de plataformas existentes en las

siguientes secciones, la creación de infraestructuras virtuales totalmente abstraídas del hardware es un campo con mucho movimiento en el desarrollo tecnológico actual.

En la siguiente sección clasificaremos las distintas tecnologías de virtualización agrupándolas según el campo de aplicación y sus características. Nos centraremos en las relacionadas con la virtualización de máquinas o servidores, pues son el componente central del entorno que pretendemos desarrollar.

1.1. Tecnologías de virtualización

Para ordenar y analizar las distintas técnicas de virtualización existen múltiples formas de clasificación. Puesto que hay principalmente 3 tipos de virtualización, muchas clasificaciones empiezan organizándolas de ese modo: ***técnicas de agrupamiento, técnicas de abstracción y técnicas de particionamiento***.

Las técnicas de agrupamiento o *pooling,* son aquellas en que distintos elementos trabajan juntos para formar una única entidad lógica. El particionamiento consiste en subdivisiones lógicas de algún recurso hardware emulando cada una de ellas el comportamiento del recurso padre. La abstracción se caracteriza simplemente por proveer una capa de emulación, sin que tenga porqué existir ni agrupamiento ni particiones.

Sin embargo, parece más intuitivo, separar las tecnologías de virtualización primero según el área de aplicación, para luego caracterizarlas en función de qué tarea realizan y cómo lo hacen.

Los campos de aplicación ya se han podido entrever en los párrafos de la sección anterior. Podemos decir que existen las siguientes:

- Técnicas de virtualización en redes.
- Técnicas de virtualización en almacenamiento.
- Técnicas de virtualización de servidores.
- Técnicas de virtualización de aplicaciones. Esta categoría puede ser muy extensa. Son principalmente técnicas de abstracción, se suelen dar dentro del S.O. de los dispositivos con el objetivo de lograr la ejecución universal de un mismo software sin necesidad de adaptarlo específicamente. Quedan bastante fuera del alcance de este trabajo, por lo que no entraremos en su estudio.

- Tecnologías de integración para crear plataformas completas de virtualización. Esta no es una categoría como las anteriores de aplicación directa, pero es la base de los entornos actuales o el Cloud Computing y de este proyecto.

En el siguiente gráfico se muestra un resumen con algunas de las tecnologías existentes en las tres primeras áreas:

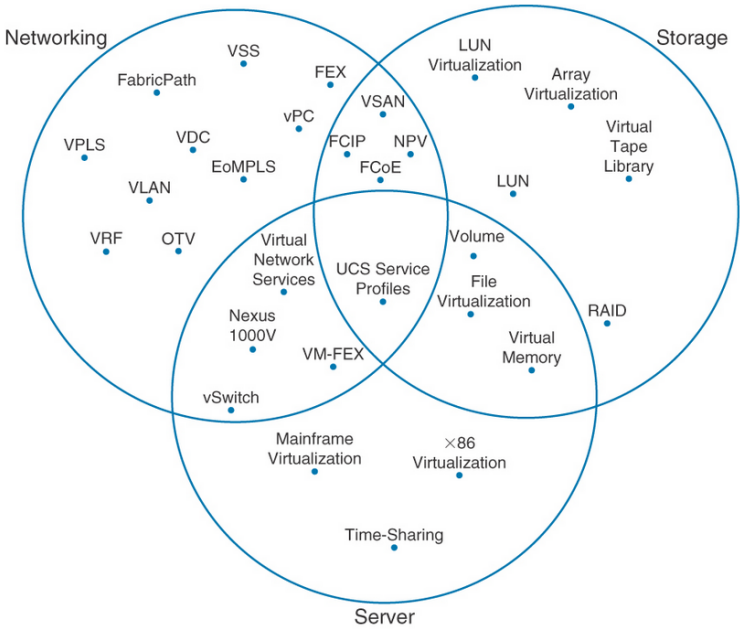


Ilustración 2: Tecnologías de virtualización según campo de aplicación (Santana 2013)

En el gráfico podemos observar que hay muchas tecnologías que pertenecen a más de una zona, es difícil establecer unos límites de forma exacta. Tienden a ser cada vez más complejas y abarcar múltiples características. Es un campo muy extenso y hay tal variedad de técnicas en las 3 áreas que resulta imposible pasar por todas ellas sin que el presente proyecto se alargue demasiado. De las tecnologías de redes y almacenamiento haremos un breve resumen nombrando los conceptos más relacionados con nuestro objetivo, sin entrar en profundidad. En el ámbito de la virtualización de servidores y las plataformas sí nos detendremos, pues conforman el núcleo de este trabajo.

1.2. Virtualización en tecnologías de red

El término virtualización de red, como ocurre con todos los que hemos visto hasta ahora en este terreno, está sobrecargado. Tiene numerosas connotaciones y depende fuertemente del

contexto en que se use. En (Wang et al. 2013) se hace un esfuerzo por dar una definición unificada:

La virtualización de red es cualquier forma de partición o combinación de recursos de red con una presentación a los usuarios de manera que, cada usuario perciba que su conjunto de recursos es una vista única y separada de la red. Los recursos pueden ser fundamentales (nodos o enlaces) o derivados (topologías), y pueden ser virtualizados recursivamente.

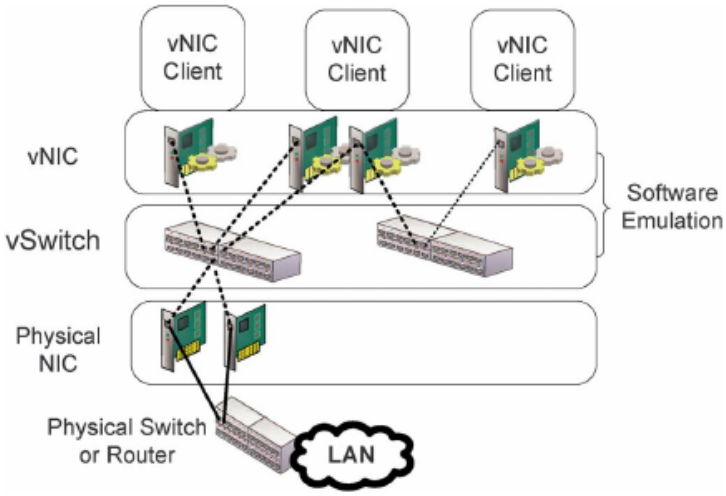


Ilustración 3: Switch e interfaz de red virtual

1.2.1. Virtualización de dispositivos (o nodos)

El gráfico anterior [fuente: (Wang et al. 2013)] ilustra una configuración genérica donde se emulan interfaces y conmutadores. Esta virtualización suele darse dentro de un sistema hipervisor y los clientes conectados a las vNIC son máquinas virtuales.

Una vNIC es un dispositivo simulado por software con su propia MAC y dirección IP, que puede ser conectado, también mediante un enlace virtual creado por software, a una NIC[5] física, a un switch virtual, o a cualquier otro dispositivo habilitado como extremo válido en la implementación realizada. Una importante tarea de las vNIC es la de compartir la misma interfaz física entre varios sistemas virtualizados.

Un vSwitch o conmutador virtual es un dispositivo software que simula un conmutador hardware. Se comporta de la misma manera y dependiendo de la implementación puede soportar prácticamente todas las características de un switch estándar (VLAN, STP…). Puede

5 Network Interface Card

conectarse a una NIC del hipervisor para dar conectividad a la red externa desde las máquinas virtuales, o no hacerlo, y crear un red interna para que exista comunicación entre ellas.

Para ilustrar mejor estos conceptos, y de paso ir conociendo unas técnicas que más adelante serán importantes en nuestro servidor, veamos algunos ejemplos de uso en GNU Linux:

Ejemplo de dispositivo	Ejemplo de comando generador	Descripción	Conexión con interfaz física
eth0:0	ifconfig eth0:0 192.168.2.44 up	Interfaz virtual creada para el propio host. Se suele usar para añadir más direcciones IP a una misma interfaz física.	Directamente conectada a eth0.
tun0	ip tuntap add name tun0 mode tun	Dispositivo virtual de capa 3 (con dirección IP). Por un lado queda conectado a la pila de red del S.O. anfitrión y por al otro al espacio de usuario (algún programa).	Dependerá del enrutamiento que se realice en el anfitrión.
tap0	ip tuntap add name tap0 mode tap	Similar a la interfaz tun, pero en este caso de capa 2. La comunicación entre el S.O. anfitrión y el espacio de usuario se realiza a nivel de enlace (MAC).	Se suele conectar a un puente virtual en el anfitrión. Dependerá de si este puente también contiene alguna interfaz física.
veth00	ip link add veth00 type veth peer name veth01	Pareja de interfaces virtuales ethernet. Se crean dos dispositivos conectados virtualmente mediante un cable cruzado de capa 2.	Depende de cómo se utilicen los extremos.
venet00	--	Interfaz virtual específica de OpenVZ que proporciona conectividad en capa 3 a un contenedor virtual. Más adelante se verá con más detalle.	Dependerá del enrutamiento que se realice en el anfitrión.

Tabla 1: Interfaces de red virtuales en GNU Linux

Las interfaces ***veth*** y ***venet*** son muy utilizadas en un tipo concreto de virtualización (contenedores de Linux). Se puede apreciar un claro paralelismo en su funcionamiento (capa2 – capa3) con la pareja tun/tap. La siguiente tabla detalla las diferencias entre ambas:

Feature	veth	venet
MAC address	Yes	No
Broadcasts inside CT	Yes	No
Traffic sniffing	Yes	No
Network security	Low	High
Can be used in bridges	Yes	No
IPv6 ready	Yes	Yes
Performance	Fast	Fastest

Tabla 2: Diferencias entre veth y venet[6]

La capacidad de crear un puente virtual entre varias interfaces está presente en el kernel de GNU Linux desde la versión 2.2 (año 1999). ***bridge-utils*** es el paquete de herramientas de espacio de usuario que permite gestionar estos puentes virtuales. Es una tecnología muy madura, estable y con características avanzadas.

Con el despunte de la virtualización en los últimos años ha surgido otro proyecto que pretende aportar flexibilidad y características al mundo de los switches virtuales en Linux: ***Open vSwitch***[7].

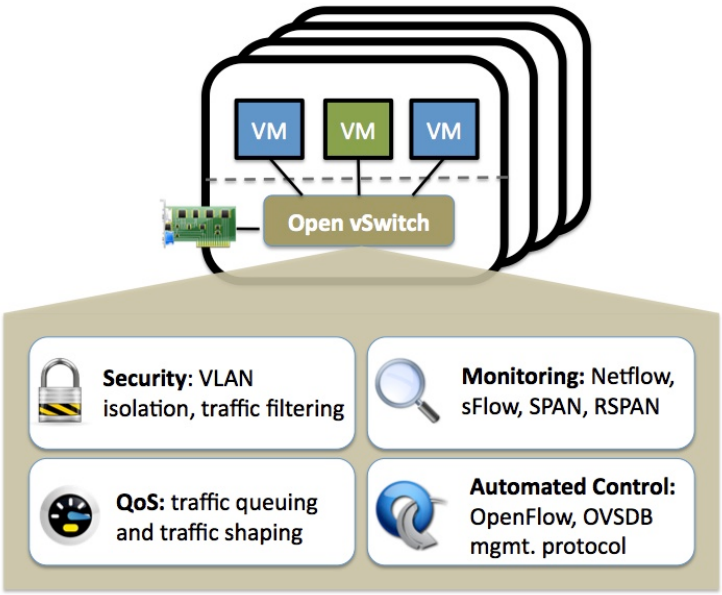


Ilustración 4: Proyecto Open vSwitch

Open vSwitch ya está soportado en diversas plataformas de virtualización como XenServer 6.0, Proxmox VE o VirtualBox y está integrado en sistemas como OpenStack, openQRM u OpenNebula.

6 Fuente: https://openvz.org/Differences_between_venet_and_veth
7 http://openvswitch.org/

En la siguiente ilustración [fuente («VMware Virtual Networking Concepts» 2007)] se muestra la topología planteada en VMWare Infraestructure 3 para la virtualización de la red. Sigue un esquema similar al que hemos ido viendo hasta ahora. Si se desea profundizar en esta implementación, la referencia citada documenta con bastante detalle el planteamiento de VMWare.

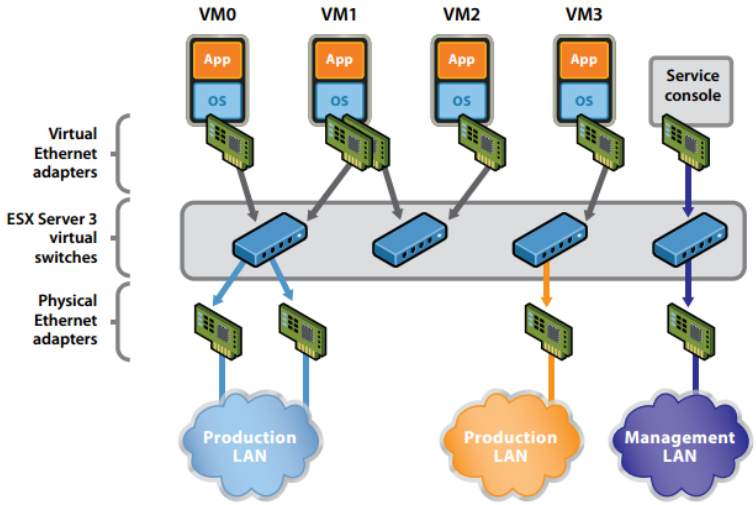


Ilustración 5: vNIC y vSwitch en VMWare ESX Server

Una vez hemos visto cómo las tarjetas de red y los switches son convertidos en entidades virtuales, es lógico esperar que el siguiente dispositivo a emular sea el router. En este tema podemos considerar 3 categorías de virtualización con funcionalidades muy diferenciadas:

1. ***Routers sobre sistemas operativos virtuales (o routers virtuales).***

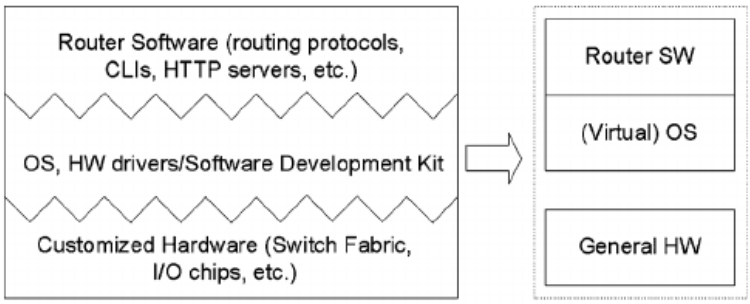


Ilustración 6: Evolución de router clásico a router virtual

Los routers clásicos eran equipos con el hardware y el software fuertemente acoplados. El sistema operativo se desarrollaba a medida para la plataforma, poniendo especial énfasis en el

rendimiento. Ejemplos como Mikrotik RouterOS[8], Untangle[9] o VyOS Linux[10] muestran un cambio de paradigma. Estas distribuciones hacen un esfuerzo por separar las capacidades de equipo enrutador de la plataforma física. Mediante un S.O. virtual pueden ser ejecutados en distintas plataformas hardware (incluso x86) o en máquinas virtuales.

Si integramos el concepto de router en el primer gráfico donde esquematizábamos las interfaces y los switches virtuales (y además damos nombres un poco más concretos), se empieza a vislumbrar claramente una completa infraestructura virtual, que como ya hemos comentado anteriormente, es la realidad hacia la que nos están conduciendo estas tecnologías.

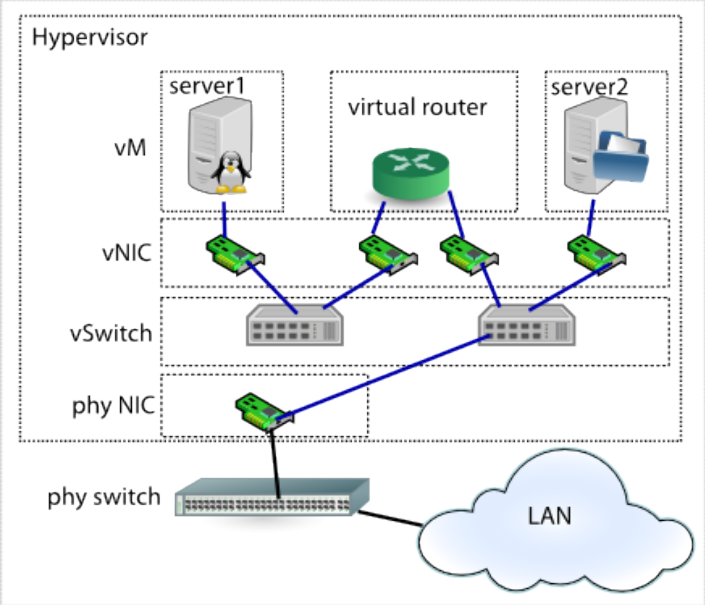


Ilustración 7: Interfaces, switches y routers virtuales.

2. ***Virtualización a nivel de control/ datos dentro del router.***

Un componente principal de un router es la tabla de encaminamiento. Con ella dirige los paquetes recibidos a los puertos de salida correspondientes. Si en un dispositivo existen varias tablas de encaminamiento y distintos procesos encargados de tomar las decisiones usando una tabla u otra en función de un contexto (por ejemplo, según la interfaz de entrada) estamos hablando de la tecnología VRF, Virtual Routing and Forwarding. Algunos equipos más avanzados permiten incluso una separación lógica de procesos para diferentes protocolos de enrutamiento y configuraciones. Fabricantes como Cisco, Alcatel-Lucent o Juniper dan soporte a esta

8 http://www.mikrotik.com/software
9 http://www.untangle.com/untangle-ng-firewall
10 http://vyos.net/wiki/Main_Page

tecnología en sus equipos desde hace tiempo[11]. En GNU Linux ha recibido el nombre de ***Network Namespaces***, se soporta desde el kernel 2.6.24 y las herramientas de gestión están integradas en el conocido paquete ***iproute2.***

3. ***Routers virtualizados en hardware particionado.***

Este tipo de virtualización se suele encontrar en dispositivos avanzados. Un hardware anfitrión especializado mantiene múltiples instancias de routers virtuales. Generalmente la partición se realiza por tarjetas de línea[12] y se despliega un router virtual por cada una de ellas. En Cisco y Juiniper Networks, por ejemplo, se conoce esta tecnología como HVRs(Hardware-Isolated Virtual Router)[13] y PSD[14] respectivamente.

1.2.2. Virtualización de enlaces

En el terreno de las conexiones lógicas nos volvemos a encontrar el problema de la amplitud del término virtualización. Esquivaremos las técnicas que trabajan a nivel físico del canal (multiplexación TDM, FDM..., circuitos, gestión de anchos de banda, etc.) pues quedan más lejos de nuestro contexto de virtualización por software y recordaremos las que no modifican el canal en sí misma (capa 2 y superiores del modelo OSI). Estas técnicas se implementan en los nodos de las comunicaciones y se basan principalmente en dos métodos: etiquetado de los paquetes y encapsulación o túneles punto a punto.

En este campo, la tecnología más madura y extendida en las redes ethernet (el primer estándar 802.11Q tiene fecha de 1998) ya se ha nombrado anteriormente. Se trata de las VLANs. Mediante una etiqueta añadida a la cabecera ethernet podemos hacer una separación lógica de subredes que comparten el mismo medio. Prácticamente todas las herramientas de virtualización de interfaces, switches o routers actuales soportan la norma 802.11Q y permiten asignación dinámica a la VLAN deseada.

11 http://www.cisco.com/c/en/us/td/docs/net_mgmt/active_network_abstraction/3-7/reference/guide/ANARefGuide37/vrf.html
12 Placa física con interfaces de red que es gestionada desde el controlador principal en el chasis donde se inserta.
13 Cisco Router Virtualization in Service Provider
14 Juniper Protected System Domains

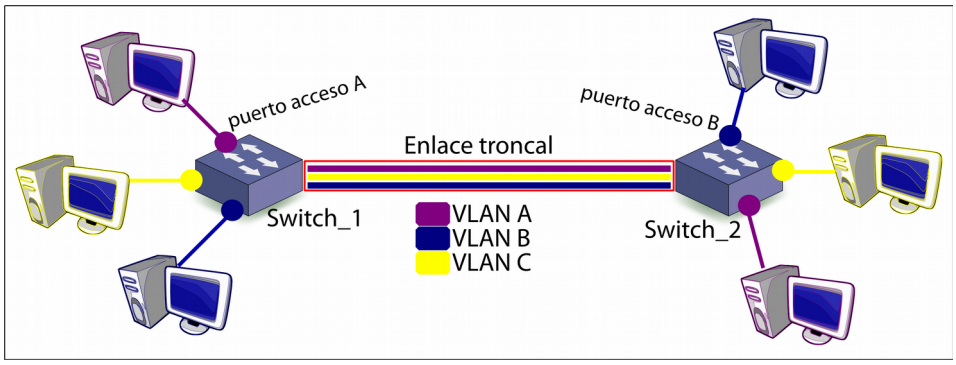


Ilustración 8: Ejemplo de 3 VLANs en dos switches conectados por un único cable

Un avance más reciente en este campo de los enlaces virtuales ha sido el protocolo MPLS o ***Multiprotocol Label Switching***. Se trata de un protocolo que opera entre la capa 2 y la capa 3 del modelo OSI y permite mantener circuitos virtuales independientemente del protocolo de red que transporte encima. Su funcionamiento se basa en una pila de etiquetas que se van añadiendo o extrayendo en los nodos de la comunicación a los paquetes transmitidos. Está definido en la RFC 3031[15]del IETF.

MPLS es un protocolo pensado para ser implementado entre enrutadores, *en el backbone de la red*. Pero visto que los routers también se virtualizan es lógico que el protocolo se extienda hasta los supuestos extremos finales. Diversas plataformas (por ejemplo en OpenStack Neutron[16], el conjunto de herramientas para la abstracción de la red de la suite OpenStack) ya están integrando su soporte.

El resto de técnicas que podemos incluir en esta categoría son las relacionadas con la creación de túneles. Encapsulan unos protocolos de red sobre otros protocolos de red añadiendo características extra, por ejemplo, autenticación y cifrado, y por este motivo se mencionan en este trabajo, permiten virtualizar infraestructuras de comunicaciones. Existe gran variedad de técnicas de este tipo, operan en distintas capas y según la técnica analizada, le encontramos más o menos características relacionadas directamente con la virtualización. Como ejemplos podemos mencionar protocolos como GRE [17], IPSEC [18], PPTP [19] , L2TP [20], o aplicaciones tipo OpenVPN [21]. Un uso muy extendido de estas herramientas se da en entornos empresariales

15 https://tools.ietf.org/html/rfc3031
16 https://wiki.openstack.org/wiki/Neutron/MPLSVPNaaS
17 https://tools.ietf.org/html/rfc1701 y https://tools.ietf.org/html/rfc1702
18 https://tools.ietf.org/html/rfc4301 y https://tools.ietf.org/html/rfc4309
19 https://tools.ietf.org/html/rfc2637
20 https://tools.ietf.org/html/rfc2661
21 https://openvpn.net/index.php/open-source/overview.html

para configurar redes privadas virtuales (VPN). Consiguen así conectar sedes remotas a través de internet de forma segura, compartir recursos y funcionar como si trabajaran en sus mismas instalaciones privadas de red.

1.3. Virtualización en almacenamiento

El primer dispositivo de almacenamiento tipo disco duro fue introducido por IBM en 1956 en el IBM 305 RAMAC, tenía un tamaño de aproximadamente 1.7m de alto, 1.5m de largo, 0.8m de fondo y capacidad para algo menos de 5MB. Desde entonces, el desarrollo de los soportes de almacenamiento ha tenido un desarrollo incluso superior al predicho por Moore para la electrónica: los bits por unidad de disco se doblan cada año. En 1996, el almacenamiento digital pasó a ser más económico que el papel para guardar información (Morris, Robert JT y Truskowski 2003).

Originalmente, el almacenamiento consistía simplemente en el disco duro u otro soporte físico y el primer objetivo de la investigación era aumentar en capacidad. Rápidamente, con la extensión de su uso, se hizo evidente que no era suficiente con mejorar en este aspecto. Se requerían otras características como fiabilidad, mayor rendimiento, facilidad de gestión y crecimiento, recuperación de desastres, etc. y para ello se han ido añadiendo capas de software y hardware que las aportan.

A continuación veremos algunas de estas técnicas importantes en la evolución de los sistemas de almacenamiento, en especial las que están más relacionadas y que usaremos en este trabajo. No analizaremos sistemas de ficheros como pueden ser ext3, ext4, ReiserFS, NTFS, y un largo etc. sino tecnologías que aportan características extra como las mencionadas en el párrafo anterior.

Las organizaremos en dos categorías: locales o distribuidas. Como almacenamiento local entendemos aquel cuyo soporte está físicamente confinado en el mismo equipo o servidor que lo gestiona y utiliza. Los almacenamientos distribuidos van un paso más allá en la virtualización de recursos: la ubicación real de los datos puede estar repartida en lugares geográficamente muy separados, pero desde el punto de vista del usuario esto es transparente. En ocasiones se consigue mediante hardware dedicado (como las redes SAN), en otras, y aquí se están produciendo los avances más recientes, mediante capas de software que coordinan almacenamientos locales (GlusterFS o CEPH).

1.3.1. Almacenamiento local

1.3.1.1. RAID

Al inicio de este capítulo ya hemos mencionado RAID. Es una tecnología que podemos calificar como antigua, pero que sigue siendo ampliamente usada, es estable y cumple perfectamente su cometido con un mínimo coste. Se publicó en 1988 (Patterson, Gibson y Katz 1987) con el objetivo de alcanzar propiedades similares a sistemas de alta gama combinando múltiples discos económicos. Según se configuran estos conjuntos de discos alcanzamos unas características diferentes tales como aumento del rendimiento o redundancia de los datos, a estas configuraciones se les llama niveles de RAID. No entra dentro del alcance de este proyecto estudiar todos estos niveles y sus características, pero veamos en una tabla los más comunes y sus rasgos principales.

NIVEL	Nº MINÍMO DE DISCOS	EFICIENCIA EN ESPACIO	RENDIMIENTO	TOLERANCIA A FALLOS	DESCRIPCIÓN
RAID 0	2	100%	Muy alto en lectura y escritura.	No puede fallar ningún disco	Los discos "se suman". Los datos se reparten entre ellos aumentando el espacio y el rendimiento.
RAID 1	2	50%	Similar al del dispositivo más lento del conjunto.	Pueden fallar todos menos 1	Espejo. Los datos se replican en los distintos discos. El espacio total que obtendremos se limita al del disco de menor tamaño.
RAID 5	3	66%	Alto. Más operaciones de escritura en los discos	Soporta el fallo de 1 disco	Reparte los datos y un cálculo de paridad sobre varios discos. Aumenta el rendimiento y permite a un disco fallar, pero conlleva más procesamiento.
RAID 10	4	50%	Muy alto, sin cálculos de paridad.	Puede fallar un disco de cada espejo	Suma de espejos. Se combinan dos RAID1 en un RAID0. Aumenta rendimiento y fiabilidad.

Tabla 3 Niveles de RAID más comunes

Para montar estos conjuntos y configurar un RAID existen tres métodos:

- **Controladora RAID hardware**: la distribución de los datos y la configuración de los discos es gestionada por un componente dedicado. Es el método más eficaz, pues no añade sobrecarga al sistema y es transparente al S.O. La principal desventaja es el coste y la dificultar para acceder a los datos en caso de fallo de la controladora.

- **RAID híbrido, falso-RAID, o RAID por firmware**: este tipo de RAID lo suelen incorporar placas base de gama media – baja. Es prácticamente un RAID por software pues el trabajo lo realizan entre la BIOS y el driver instalado en el S.O, con ayuda de algún chipset acelerador. En rendimiento es algo superior al RAID por software y en coste resulta mucho más económico que el hardware dedicado. Por otro lado tiene el inconveniente de la necesidad de compatibilidad de los drivers con el S.O., genera sobrecarga y al igual que las controladoras dedicadas, suelen configurar los discos de forma propia y si falla el hardware se complica la recuperación de información.

- **RAID por software**: en este caso se carga al procesador con el trabajo de gestionar el RAID. Se configura en el S.O. implementado mediante una capa de abstracción, una componente del sistema de ficheros o una aplicación. Linux **mdadm**, LVM, el administrador de discos dinámicos de Microsoft Windows, o en las últimas versiones de Windows Server, los espacios de almacenamiento[22] son ejemplos de RAID por software.

Este tipo de RAID tiene la principal desventaja de la penalización en rendimiento. Según los requerimientos de nuestras aplicaciones y la potencia de procesamiento del equipo puede que debamos tener en cuenta esta sobrecarga. Sin embargo, con los procesadores que hay en el mercado actual y si la exigencia no es muy alta, las ventajas resultan muy atractivas:

- Los datos son más accesibles desde el S.O., tenemos más control, flexibilidad y opciones de monitorización.
- El coste es mínimo. Una solución como **mdadm** (software libre) es gratuita y muy estable.
- En caso de fallo hardware del sistema la información se puede recuperar desde cualquier equipo con ese software. Evitamos dependencia de plataformas o hardware concreto.

1.3.1.2. Linux Volume Manager

LVM es un gestor de volúmenes lógicos para el kernel de GNU Linux. Aporta una capa de abstracción que nos permite diseñar un mapa de particiones independiente de la estructura física de los dispositivos subyacentes. Algunas de las características que nos aporta:

- Redimensión y migración de los volúmenes de almacenamiento en vivo.

22 https://blogs.msdn.microsoft.com/b8/2012/01/05/virtualizing-storage-for-scale-resiliency-and-efficiency/

- RAID 0: permite sumar bloques de distintos dispositivos para una partición lógica.
- Independencia del orden de los volúmenes físicos para la reestructuración del espacio.
- Instantáneas en vivo del sistema de ficheros. Permite las copias de seguridad de todo el sistema sin parar los servicios.
- Máxima flexibilidad: como dispositivos físicos acepta unidades de disco duro, dispositivos creados por software RAID, particiones…

Para su lograr estas cualidades LVM se estructura en un diseño por capas:

1. Nivel más bajo: volumen físico. una partición preparada sobre el dispositivo físico con formato LVM, que será integrada en un grupo en el siguiente nivel.
2. NIvel medio: grupos de volúmenes. Cada grupo será un dispositivo de almacenamiento virtual. Su tamaño dependerá de los volúmenes físicos del nivel inferior que le pertenezcan y sobre él se crean las particiones finales.
3. Nivel superior: volumen lógico. Particiones sobre las que se crean los sistemas de ficheros finales. Su tamaño puede ser cualquiera, siempre dentro de un mismo grupo de volúmenes y por tanto con tamaño máximo igual al del grupo al que pertenece.

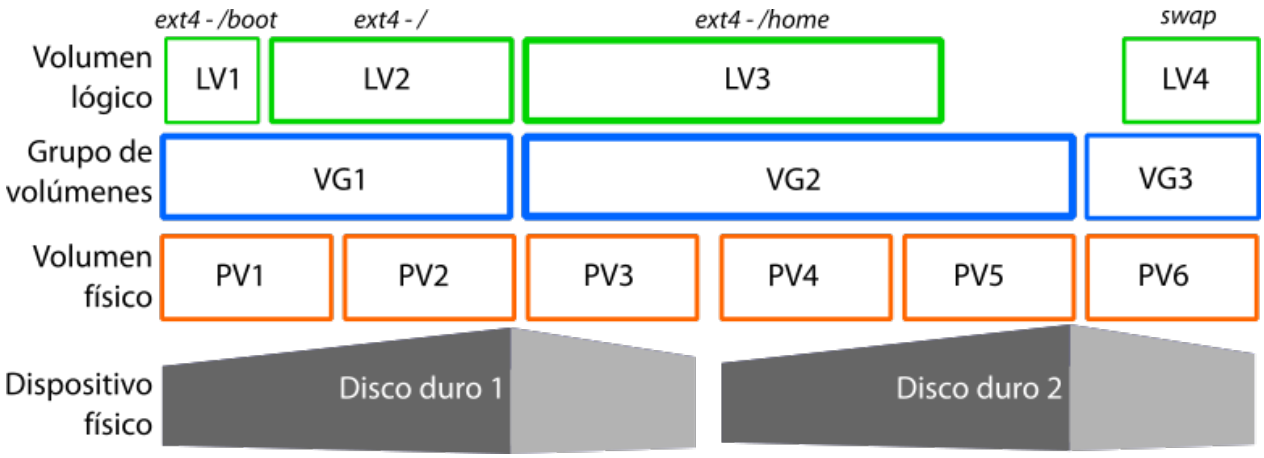


Ilustración 9: Logical Volume Manager

1.3.1.3. Otros: ZFS, JBOD, Veritas Volume Manager

JBOD son las siglas de *"Just a Bunch Of Disks"*, o traducido, sólo un grupo de discos. JBOD equivale a un RAID 0 más sencillo y con algunas diferencias. Crea un volumen virtual con la suma de los dispositivos que componen el conjunto, pero los datos se almacenan de forma independiente. No supone mejoría alguna en el rendimiento pero el fallo de un disco no provoca

la pérdida de los datos del resto de componentes. Además, en un RAID-0, el tamaño está limitado por la unidad más pequeña: si configuramos dos discos de 500GB y 750GB, el resultado será un dispositivo de 500GB+500GB = 1TB, por lo que desaprovechamos 250GB. En el caso de JBOD se aprovecha todo el espacio, y el resultado sería una unidad de 1250GB.

Veritas Volume Manager (VxVM) es un administrador de volúmenes lógicos similar a LVM, propietario, desarrollado por Veritas Software. Es compatible con Windows, AIX, Solaris, Linux o HP-UX. Para ampliar en sus especificaciones podemos consultar la guía de administrador[23] .

ZFS es un sistema de ficheros originalmente cerrado y propietario desarrollado por SUN Microsystems para su sistema operativo Solaris. En 2005 su código fue liberado en el proyecto OpenSolaris, y a partir de ahí OS X, BSD, GNU Linux y derivados disponen de su propia implementación. Aunque hemos dicho que no vamos a hablar de sistemas de ficheros, ZFS es un caso especial: integra administrador de volúmenes lógicos y características avanzadas para configurar RAID por software, realizar instantáneas y clones o control de integridad de los datos. Todo ello gestionado por el propio sistema de ficheros. Su funcionamiento es similar al de LVM: en capas. Los dispositivos físicos se agrupan en los llamados *storage pools*, para después organizarse en *datasets*. Estos pueden ser de tres tipos: sistema de ficheros estándar para ficheros, zvols (dispositivos de bloques) o instantáneas de otros datasets. La documentación oficial de ZFS está disponible en la web de Oracle[24].

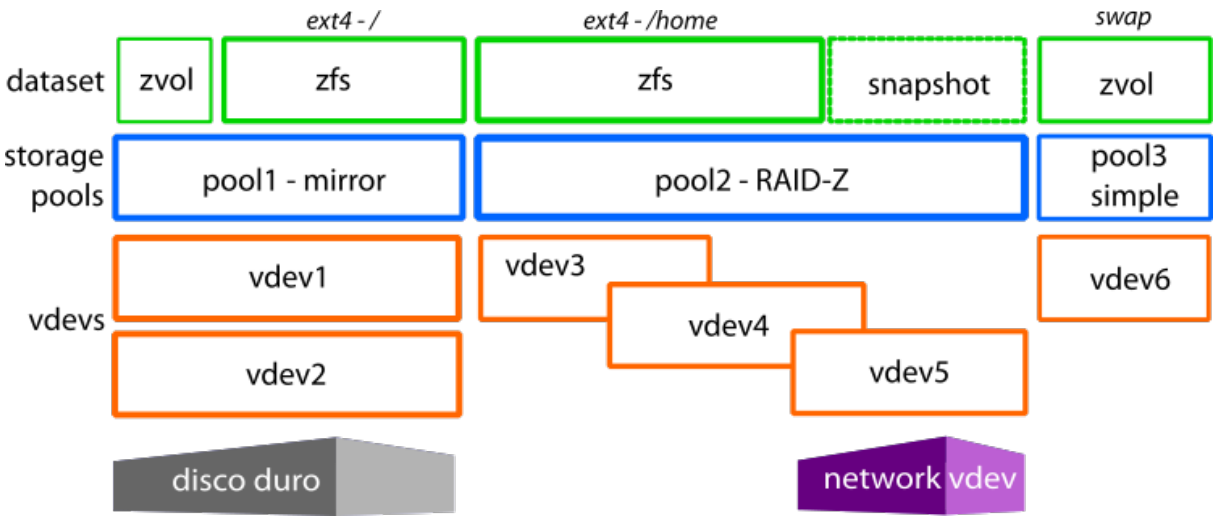


Ilustración 10: ZFS

23 Veritas™ Volume ManagerAdministrator's Guide
24 Oracle Solaris ZFS Administration Guide

1.3.2. Almacenamiento distribuido

El origen de los almacenamientos ubicuos lo podemos situar en los sistemas SAN y NAS. La aparición de las tecnologías LAN de bajo coste al final de los 1980s condujo rápidamente a la interconexión de los PCs y a infraestructuras de trabajo basadas en el modelo cliente-servidor. Con ello, la información abandonó el espacio del *Centro de Datos* y se dispersó por diferentes equipos y ubicaciones. Esto trajo consigo nuevos problemas asociados a la pérdida de control sobre ella. Surgió entonces la necesidad de volver a centralizar el almacenamiento, o al menos su control, aunque estuviera repartido físicamente en varios dispositivos.

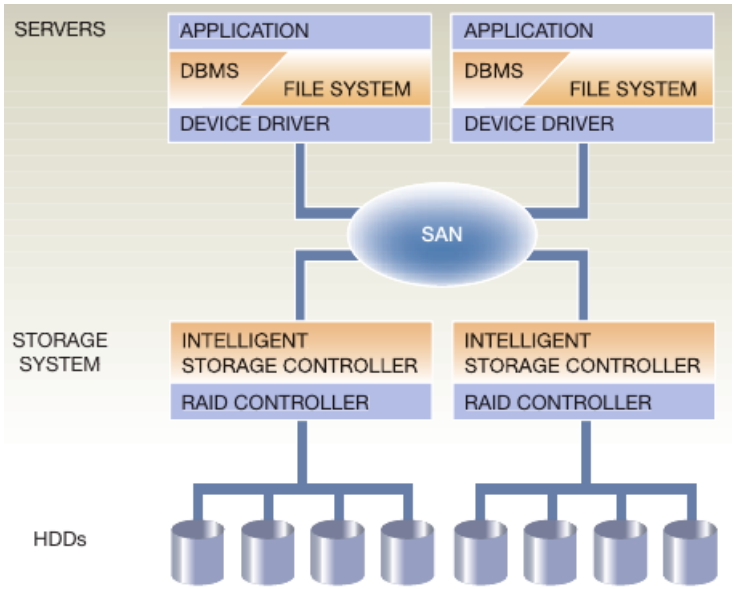


Ilustración 11: Esquema de una red SAN (Morris, R. J. T. y Truskowski 2003)

Las redes SAN son infraestructuras dedicadas completamente al almacenamiento. Constan de dispositivos de almacenamiento (discos duros, cabinas de discos, librerías de cintas) y su red propia de interconexión de alta velocidad, con equipos de interconexión, canales de datos y protocolos de transporte propios (SCSI, FCP, FCoE, iFCP, ISCSI). Generalmente se trata de conexiones de fibra y switches de gran rendimiento por lo que son muy costosas. Con la mejora en velocidad de las conexiones ethernet (redes 10-gigabit o gigabit) han surgido protocolos como iSCSI o FCoE (Fibre Channel Over Ethernet) que reducen los costes considerablemente aprovechando esta infraestructura y facilitan la extensión de su uso.

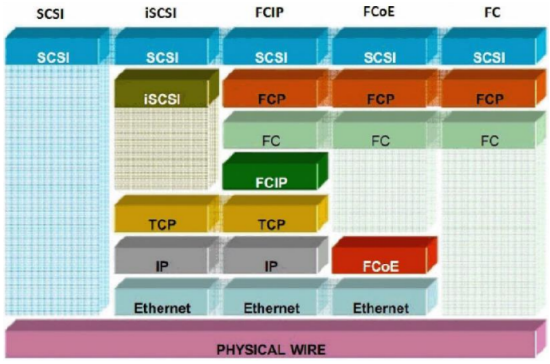


Ilustración 12: Protocolos SCSI – almacenamiento SAN

El almacenamiento de una red SAN es exportado como dispositivos de bloques, o lo que es lo mismo, los usuarios o clientes perciben una unidad de disco. Frente a esta tecnología de almacenamiento remoto está el NAS. Un NAS gestiona sus discos de forma local mediante un sistema de ficheros y una configuración determinada, y exporta el almacenamiento como ficheros sobre protocolos de más alto nivel. Los más extendidos son CIFS o SAMBA y NFS.

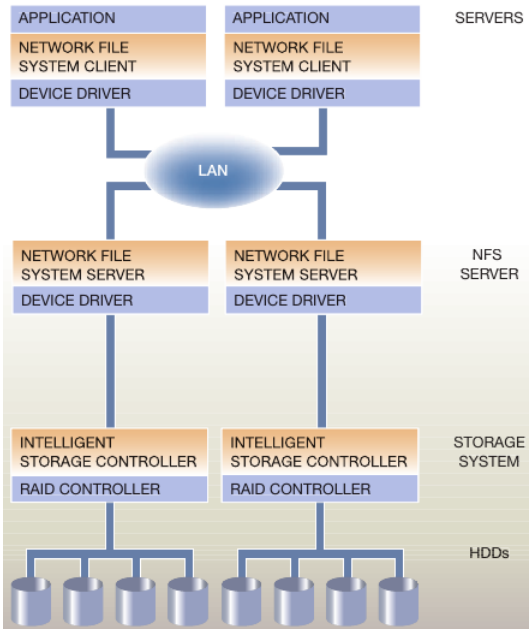


Ilustración 13: Esquema de almacenamiento NAS (Morris, R. J. T. y Truskowski 2003)

En el mundo del software libre existen dos proyectos dignos de mención como son FreeNAS[25] y OpenFILER[26]. Son sistemas operativos diseñados para crear soluciones de almacenamiento en red muy potentes sobre servidores o equipos corrientes, integrando características de SAN y NAS.

25 http://www.freenas.org/
26 http://www.openfiler.com/

Los dos tipos de almacenamientos que acabamos de ver son almacenamientos en red, y son distribuidos en la medida en que no están ubicados físicamente el el lugar donde se usan los datos. Actualmente el concepto de almacenamiento distribuido se usa más para hacer referencia a tecnologías que funcionan en una capa un tanto superior: los sistemas de ficheros distribuidos.

Un sistema de ficheros distribuido es el resultado de la coordinación de varias máquinas o nodos en un clúster. Cada una almacena y gestiona parte de información, pero desde el punto de vista de un cliente, el almacenamiento se percibe como un bloque compacto. Para entender mejor el concepto, veamos algunas de las tecnologías existentes.

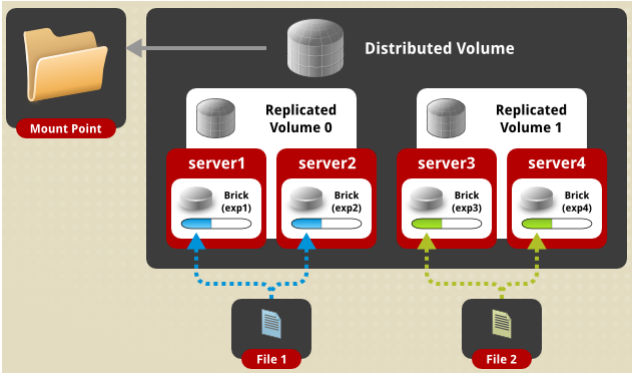


Ilustración 14: GlusterFS – Sistema de ficheros replicado y distribuido

- **GlusterFS**[27]

GlusterFS es un sistema de ficheros distribuido que funciona sobre redes TCP/IP, Infiniband o SDP y basado en el modelo cliente-servidor. Cada servidor ejecuta una instancia del demonio **glusterfsd** y exporta un sistema de archivos local (o parte) como un volumen. El equipo cliente utilizará varios de estos volúmenes mediante un ***traductor,*** pudiendo configurarlos en diferentes modos: distribuidos, replicados, fragmentados o combinaciones de estos.

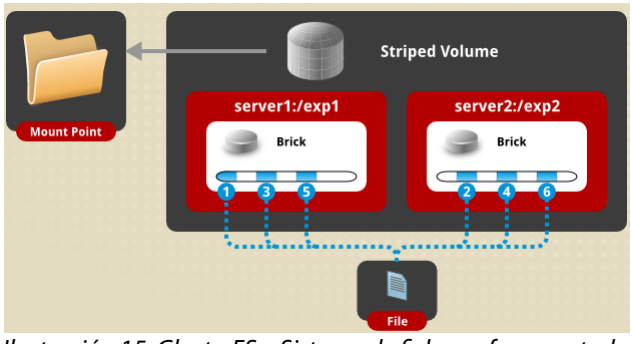


Ilustración 15: GlusterFS – Sistema de ficheros fragmentado

27 http://gluster.readthedocs.io/en/latest/

- **Ceph**[28]

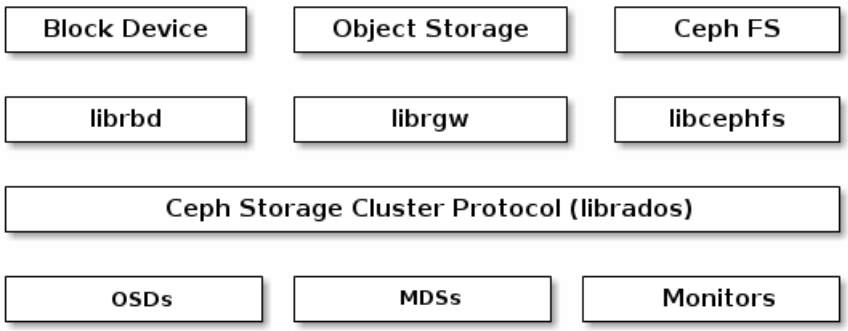


Ilustración 16: Ceph – Arquitectura de alto nivel

Otro sistema de almacenamiento distribuido muy extendido en la actualidad es Ceph. Tras su desarrollo están compañías como Canonical, Cisco o, principalmente, Red Hat (que también es responsable de Gluster[29]). La principal diferencia respecto a GlusterFS es que este ofrece tres modos de acceso para el cliente: dispositivo de bloques, de objetos o sistema de ficheros. Basa su estructura en clusters compuestos por Monitores (mantienen un mapa con el estado del cluster) y OSDs (Object Storage Devices, gestionan la lectura/escritura en su unidad de disco, almacenan la información como objetos e informan de su estado y el de otros OSD a los monitores).

Otros proyectos emergentes en la misma línea dignos de mención son **DRBD**[30] y **DRBD9** de Linbit o **Sheepdog**[31]. DRBD y DRBD9 están orientados a la alta disponibilidad y la seguridad de los datos. Sheepdog está pensado para ofrecer volúmenes de almacenamiento a máquinas virtuales QEMU, contenedores o cualquier cliente iSCSI.

28 http://docs.ceph.com/docs/master/start/intro/
29 https://www.redhat.com/es/technologies/storage
30 http://www.linbit.com/en/p/products/drbd
31 https://sheepdog.github.io/sheepdog/

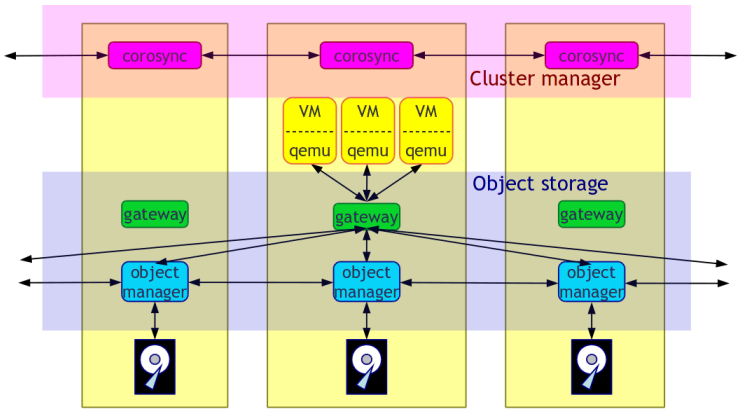


Ilustración 17: Sheepdog - Arquitectura

1.4. Virtualización de máquinas, servidores y aplicaciones

Entramos ya en el área principal de aplicación de la virtualización para el propósito de este proyecto. Hasta ahora hemos podido comprobar que es un término muy sobrecargado pues, desde los inicios de la informática, los avances tecnológicos han ido creando capas de abstracción sobre los medios físicos. Prácticamente todos los protocolos de comunicación o lenguajes de programación de alto nivel, sin definir muy estrictamente el término, se pueden considerar alguna clase de virtualización. Pero, si hay que elegir un tipo concreto como elemento central de los entornos actuales y detonante del sistema que queremos desarrollar es la ***virtualización de ordenadores dentro de ordenadores***.

Para estudiar este área de virtualización volvamos a una de las definiciones que vimos en el primer apartado de este capítulo:

Sistema o metodología para dividir los recursos de una computadora en múltiples entornos de ejecución aislados. (Kovari y Dukan 2012).

Según el nivel de emulación y la forma de llevar a cabo este reparto de recursos se pueden diferenciar las categorías de virtualización que analizaremos a continuación.

1.4.1. Virtualización completa

Simula completamente las características de la plataforma virtualizada. Se crea un entorno virtual que permite ejecutar cualquier Sistema Operativo (compatible con la arquitectura generada) sin modificaciones. Digamos que el Sistema Operativo huésped no es consciente de que está siendo virtualizado. Cada máquina virtual reproduce su BIOS, dispositivos, periféricos, gestión de memoria y realiza una traducción completa para el anfitrión. Debido a la gran capa de abstracción añadida es de los métodos menos eficientes y que más carga generan.

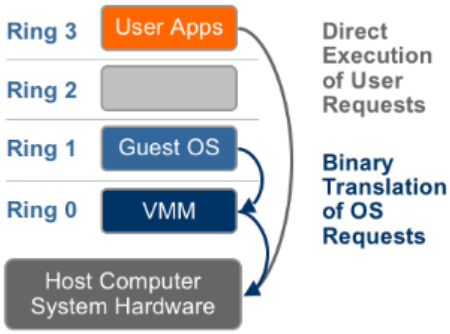


Ilustración 18: Virtualización completa.

Algunas soluciones existentes en el mercado que proporcionan virtualización completa:

Nombre	URL del proyecto
Oracle Virtualbox	https://www.virtualbox.org/
bochs	http://bochs.sourceforge.net/
Xen	http://www.xenproject.org/
KVM	http://www.linux-kvm.org/page/Main_Page
QEMU *Se apoya en Xen o KVM	http://wiki.qemu.org/Main_Page
VMWare Workstation	https://www.vmware.com/es/products/workstation
Microsoft Hiper-V	https://technet.microsoft.com/library/hh831531.aspx
Windows VirtualPC	https://www.microsoft.com/es-es/download/details.aspx?id=3702

Tabla 4: Plataformas con virtualización completa

1.4.2. Virtualización asistida por hardware

También conocida como *Virtualización acelerada* o *Virtualización nativa*. Es un tipo de virtualización completa donde se aprovechan características hardware del equipo anfitrión para hacerla más eficiente. En este caso es necesario que el Sistema Operativo invitado tenga soporte específico de estas para poder aprovecharlas. Los fabricantes de electrónica cada vez tienen más en cuenta la virtualización y van añadiendo características para facilitarla desde el más bajo nivel. Desde el año 2006 los dos grandes fabricantes AMD e INTEL incorporan en sus procesadores las mejoras VT-x y AMD-V respectivamente. Estas permiten capturar algunas instrucciones y traspasarlas directamente al hipervisor sin necesidad de traducción binaria o paravirtualización. Prácticamente todas las plataformas existentes en el mercado aprovechan estas mejoras para mejorar su rendimiento.

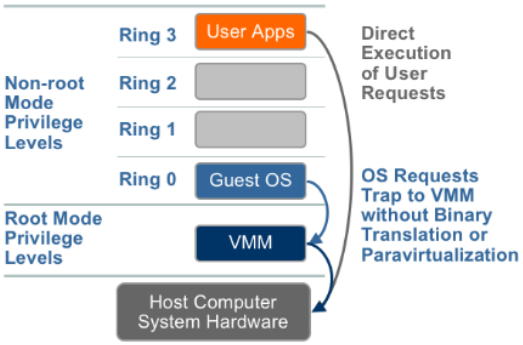


Ilustración 19: Virtualización asistida por hardware

1.4.3. Para-virtualización o virtualización asistida por S.O.

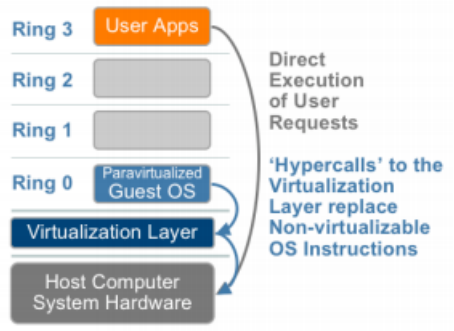


Ilustración 20: Para-virtualización

La principal diferencia entre la paravirtualización y la virtualización completa es que el S.O. huésped es modificado y adaptado (por ejemplo, mediante la instalación de drivers específicos). El monitor le proporciona una serie de llamadas que reemplazan operaciones que no se pueden simular, evitando la traducción binaria. Se mejora en rendimiento y eficiencia a costa de una menor compatibilidad y mayores costes en términos de mantenimiento y soporte, pues hay que modificar el sistema huésped.

Dos ejemplos de plataformas que permiten este tipo de emulación, aparte de la virtualización completa como vimos antes, son Xen y KVM.

1.4.4. Virtualización a nivel de Sistema Operativo o Contenedores

En este tipo de virtualización se generan múltiples entornos de ejecución aislados dentro del mismo S.O. anfitrión. Son instancias en espacio de usuario, esto quiere decir que se ejecutan fuera del núcleo del sistema, con permisos como cualquier proceso o librería típico para operaciones de entrada/salida o aplicaciones estándar. Tienen su propio espacio de memoria virtual y no pueden acceder a la de otros procesos. Al no realizarse simulación de plataforma

alguna, no se pueden ejecutar diferentes núcleos de S.O., todos los contenedores presentan el mismo del anfitrión, pero desde dentro, la percepción es que cada contenedor es un sistema real e independiente. A su vez, como no existe traducción y toda la gestión de recursos es realizada por el mismo kernel, esta tecnología ofrece mejor rendimiento y densidad que las vistas hasta ahora.

El origen de los contenedores lo podemos situar en 1979, cuando se incluye por primera vez la llamada al sistema ***chroot*** (change-root) en la versión 7 de UNIX. Esta operación permite cambiar la raíz aparente del sistema operativo para el proceso actual y sus hijos, creando un entorno aislado del que, teóricamente, estos procesos no pueden salir. A este entorno se le suele llamar jaula CHROOT. La evolución de esta idea, junto con el desarrollo de técnicas como los kernel namespaces o cgroups (Rosen [2013]), es lo que ha desembocado en las actuales opciones de virtualización por contenedores. Citamos a continuación algunas de las más populares.

Nombre	URL del proyecto
FreeBSD Jals	https://www.freebsd.org/doc/handbook/jails.html
Solaris Zones	http://www.oracle.com/technetwork/server-storage/solaris/containers-169727.html
OpenVZ Containers	https://openvz.org/Main_Page
LXC Linux containers	https://linuxcontainers.org/

Tabla 5: Soluciones de virtualización a nivel de sistema operativo

En http://pivotal.io/platform/infographic/moments-in-container-history se puede consultar un gráfico con un eje temporal que cubre la evolución histórica de los contendores.

Aunque no se trata realmente de tecnologías de virtualización a nivel de sistema operativo, hay que mencionar en este apartado los proyectos Docker[32] y LXD[33]. Ambos están orientados a proporcionar plataformas para gestionar múltiples contenedores y agilizar el despliegue de servicios y aplicaciones sobre ellos.

32 https://www.docker.com/what-docker
33 https://linuxcontainers.org/lxd/

2. Plataformas de virtualización. Computación distribuida

Por plataforma de virtualización entendemos una estructura compuesta por diferentes servicios que gestionan múltiples tecnologías de virtualización como las vistas hasta ahora. Integran subsistemas de almacenamiento, redes y máquinas virtuales en un mismo entorno, centralizando y simplificando su gestión. Son herramientas de **virtualización de centros de datos completos** y conforman la realidad existente bajo el concepto de la nube. Gracias a su desarrollo ha surgido el modelo de negocio **infraestructura como servicio (IaaS)**, en el que los proveedores mantienen el hardware y alquilan servidores, espacios de almacenamiento y redes (todo virtualizado) a sus clientes.

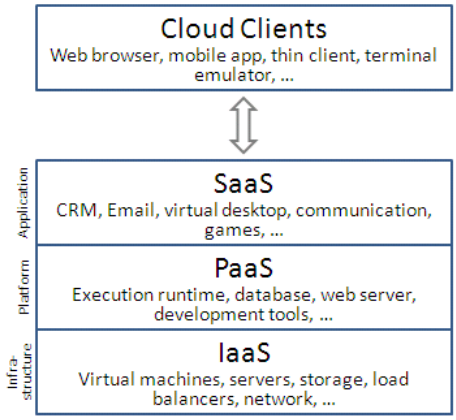


Ilustración 21: Capas en el modelo de Cloud Computing [fuente: Wikimedia]

Plataforma	Tipo de hipervisor - S.O. anfitrión	Virtualización, sistemas invitados soportados	Licencia
Citrix XenServer	Tipo 1	Hipervisor XEN. Virtualización completa, paravirtualización. Sistemas Windows , Linux, Unix y derivados.	GNU GPLv2+
	https://www.citrix.es/products/xenserver/		
VMWare vSphere	Tipo 1	Hipervisor ESXi. Virtualización completa, paravirtualización. Sistemas Windows , Linux, Unix y derivados.	Propietaria
	https://www.vmware.com/es/products/vsphere		
Microsoft Hyper-V Server	Tipo 2 – Microsoft Windows Server	Hipervisor Hyper-V. Virtualización completa, paravirtualización. Sistemas Windows , Linux, Unix y derivados.	Propietaria
	https://msdn.microsoft.com/es-es/library/hh831531(v=ws.11).aspx		
Virtuozzo	Tipo 2 - Virtuozzo Linux (RHEL-based)	Hipervisor KVM/QEMU y OpenVZ. Virtualización completa, paravitualización y contenedores. Sistemas Windows , Linux, Unix y derivados.	Propietaria
	https://virtuozzo.com/		
Proxmox VE	Tipo 2 – Debian GNU Linux	Hipervisor KVM/QEMU y LXC. Virtualización completa, paravitualización y contenedores. Sistemas Windows , Linux, Unix y derivados.	AGPL-V3
	http://www.proxmox.com/en/		

Tabla 6: Plataformas de virtualización más extendidas

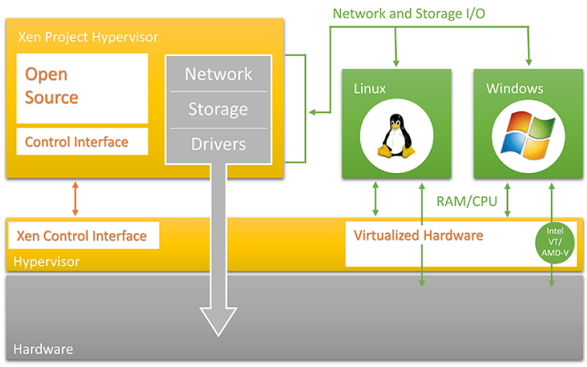


Ilustración 22: XenServer - Arquitectura

La tabla anterior contiene las plataformas de virtualización más extendidas. Cada una con sus características propias, pero todas integran, como ya hemos dicho, gestión de almacenamientos virtuales de distintos tipos, virtualización de redes, configuraciones en clúster de alta disponibilidad, migración de máquinas entre nodos en caliente, o capacidad de instantáneas sin detener el sistema invitado. Aunque permiten simular perfectamente topologías o centros de datos, estas plataformas anteriores están orientadas a virtualizar y gestionar máquinas o servidores.

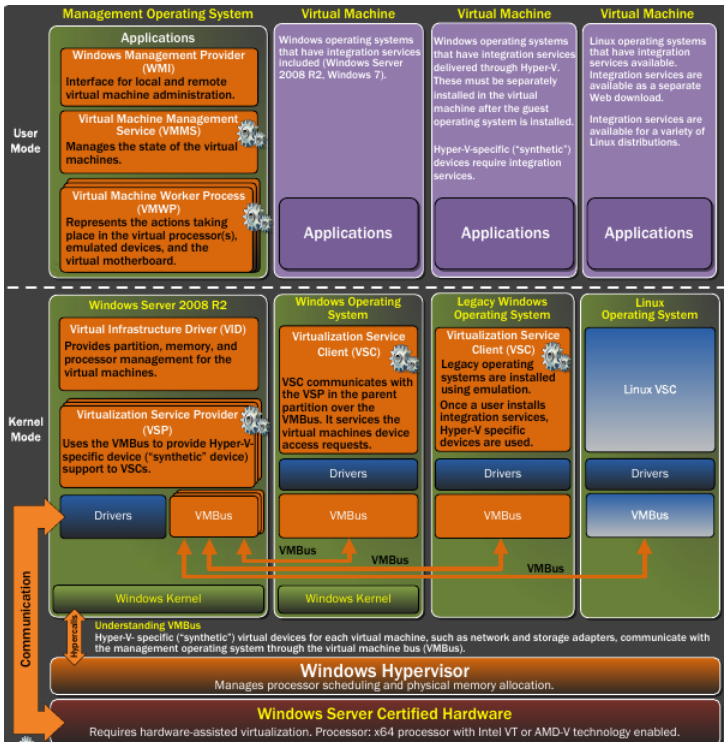


Ilustración 23: Microsoft Hyper-V Server 2008R2 - Arquitectura

El siguiente paso en las plataformas de virtualización lo han dado proyectos como OpenStack, OpenNode, OpenNebula y OpenQRM. Estos desarrollos están orientados a generar infraestructuras complejas, en crear los verdaderos entornos CLOUD. A diferencia de las vistas en la tabla anterior, el peso de estas plataformas ya no se pone en las máquinas virtuales, el objetivo es la propia infraestructura. Para entender algo mejor esta diferencia pensemos en estos dos casos: en OpenStack el subsistema de computación se integra con los proyectos XenServer, QEMU, LXC, Virtuozzo o Hyper-V entre otros[34] y OpenNebula es usado en muchas ocasiones para proveer una capa de abstracción, provisionamiento o alquiler sobre una infraestructura de VMware vCenter[35].

34 http://docs.openstack.org/developer/nova/support-matrix.html
35 http://opennebula.org/about/technology/

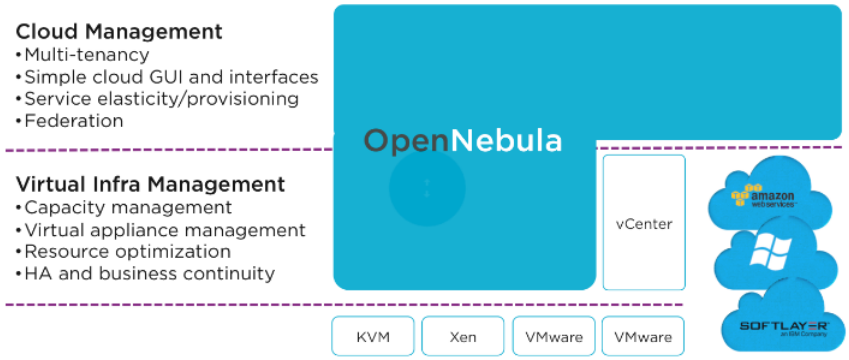


Ilustración 24: Proyecto OpenNebula

OpenStack es quizás el desarrollo con más empuje en este campo. Arrancó en 2010 y tiene detrás a más de 200 compañías, algunas como Canonical, AT&T, HP, IBM, Intel o Red Hat son miembros Platino de la Fundación Openstack. Su arquitectura nos puede ayudar a comprender el funcionamiento y el objetivo de este tipo de proyectos:

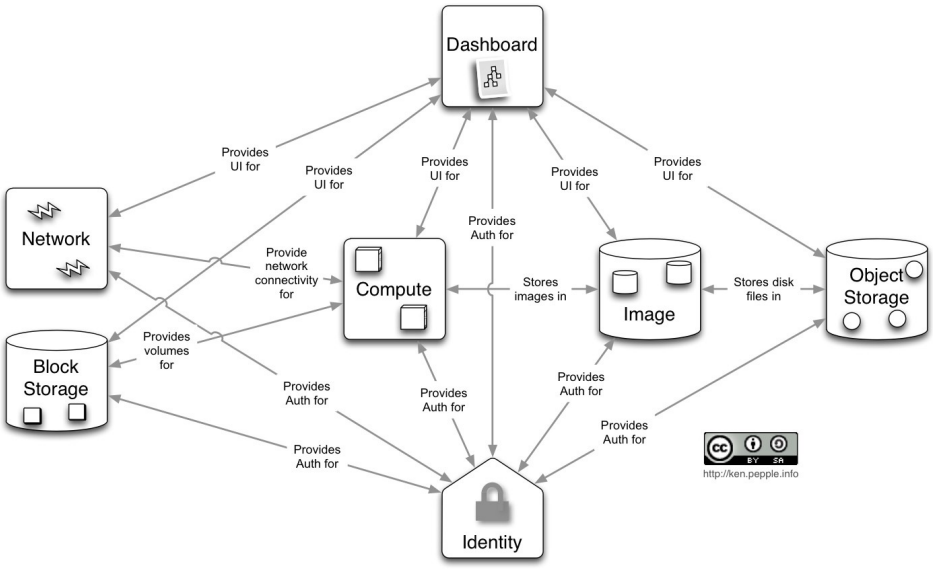


Ilustración 25: OpenStack – Arquitectura

Capítulo III: Proxmox Virtual Environment

1. Visión general e historia

Hoy en día, Proxmox VE es un entorno de virtualización maduro, estable, con buen soporte y preparado para un uso profesional en servidores de empresa. Integra la gestión de contenedores, máquinas virtuales, redes virtuales, almacenamiento y clúster de alta disponibilidad. Todo ello administrado desde línea de comandos o una interfaz web sencilla, y sin necesidad de un controlador dedicado externo, pues todo puede desplegarse a partir de un único nodo.

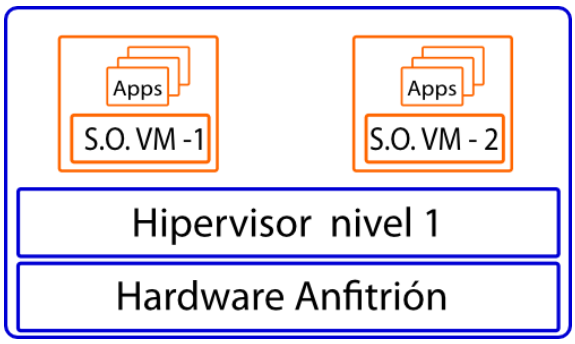


Ilustración 26: Hipervisor de virtualización de nivel 1

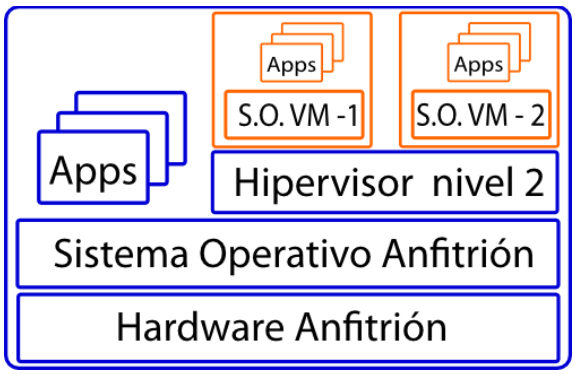


Ilustración 27: Hipervisor de virtualización de nivel 2

Como se muestra en las imágenes anteriores, un hipervisor de nivel 1 funciona directamente sobre el hardware y no aporta funciones más allá de gestionar el sistema de virtualización.

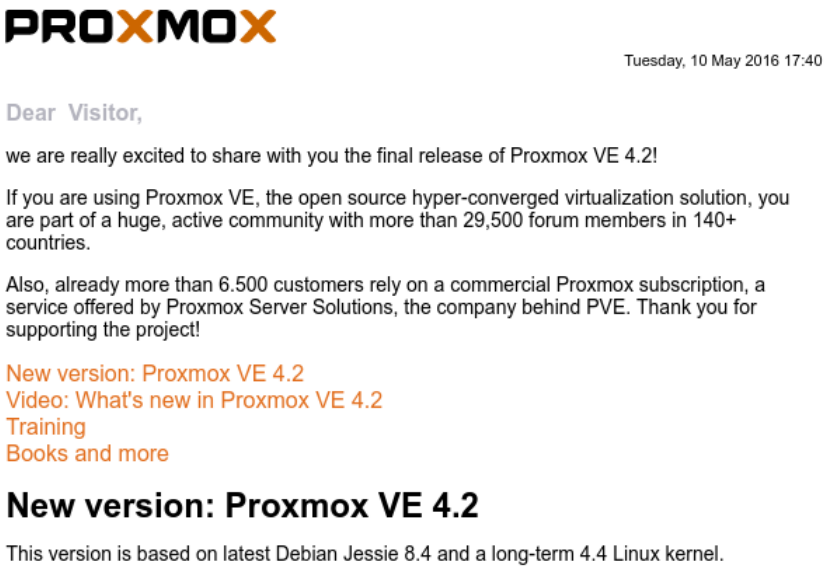
Proxmox VE es un hipervisor de nivel 2: la capa de virtualización se monta sobre un sistema operativo anfitrión, completamente funcional y autónomo, que puede realizar otras tareas además de gestionar la virtualización. Proxmox está desarrollado sobre el proyecto Debian. Una distribución GNU Linux ampliamente reconocida por su fiabilidad, comprometida con la seguridad y el software libre y con una fuerte comunidad de desarrolladores que la respaldan. [36]

La licencia bajo la que se publica Proxmox es la GNU Affero General Public License (V3) [37]. Se trata íntegramente de una GNU GPL con una cláusula que añade la obligación de distribuir el software si éste se ejecuta para ofrecer servicios a través de una red de ordenadores. El principal objetivo de esta licencia es, por tanto, asegurar la cooperación con la comunidad. Tenemos libertad para inspeccionar el código fuente, modificarlo y contribuir con el desarrollo.

La flexibilidad que aporta el ser un hipervisor de nivel 2, la licencia AGPL-V3 y la garantía de que esté basado en Debian GNU Linux son tres factores que tienen mucho peso en la elección de Proxmox como el entorno elegido para el desarrollo de este proyecto.

La primera versión estable del entorno se publicó en 2008, la v1.0. Fue un desarrollo pionero al integrar soporte de contenedores y máquinas virtuales. La última disponible en el momento de desarrollo de este documento, la v4.2, se ha publicado en abril de 2016. Como podemos leer en su anuncio, existe una comunidad activa de más de 29500 miembros del foro [38]

36 https://www.debian.org/intro/about#what
37 http://www.gnu.org/licenses/agpl-3.0.html
38 https://forum.proxmox.com/



PROXMOX

Tuesday, 10 May 2016 17:40

Dear Visitor,

we are really excited to share with you the final release of Proxmox VE 4.2!

If you are using Proxmox VE, the open source hyper-converged virtualization solution, you are part of a huge, active community with more than 29,500 forum members in 140+ countries.

Also, already more than 6.500 customers rely on a commercial Proxmox subscription, a service offered by Proxmox Server Solutions, the company behind PVE. Thank you for supporting the project!

New version: Proxmox VE 4.2
Video: What's new in Proxmox VE 4.2
Training
Books and more

New version: Proxmox VE 4.2

This version is based on latest Debian Jessie 8.4 and a long-term 4.4 Linux kernel.

Ilustración 28: Anuncio publicación Proxmox VE 4.2.

en más de 140 países, y se han vendido ya por encima de 6500 suscripciones comerciales. Se trata de un proyecto con 8 años de experiencia, con una evolución y mejoría constantes en el que podemos confiar como base sólida para nuestro desarrollo.

Los siguientes 10 puntos son los hitos más relevantes en la historia del proyecto según la propia compañía[39]:

- Febrero de 2005: Martin y Dietmar Maurer crean Proxmox Server Solutions GmbH en Viena.
- 29 de abril de 2005: se publica la versión 1.0 de Proxmox Mail Gateway.
- Mayo de 2005: se activa el foro para la comunidad online y se unen los primeros miembros.
- 2007: publicación principal de la versión 2.0 de Proxmox Mail Gateway 2.0 con virtualización por contenedores en Virtuozzo™ soporte de OpenVZ.
- 2008: publicación de Proxmox Virtual Environment (VE) en la versión0.9. En octubre aparece la primera estable, la 1.0.
- Marzo de 2012: se libera Proxmox VE 2.0 con opciones de alta disponibilidad sobre Red Hat Cluster y Corosync. Se despliegan los primeros servicios de soporte comercial.

39 http://www.proxmox.com/en/news/press-releases/10-years-of-proxmox

- Julio de 2012: Proxmox Mail Gateway versión 3.0 con integración de Commtouch/Cyrens anti-spam/antivirus.
- 2013: se libera Proxmox VE 3.0 con plantillas y clonación de VMs. Aparece también el programa de formación en Proxmox VE.
- 2014: Proxmox se une a la Linux Foundation. Las versiones 3.2 y 3.3 traen el soporte de SPICE y spiceterm, Ceph server, Open vSwitch, consola HTML5, Proxmox VE Firewall, autenticación en 2 pasos, plugin de almacenamiento ZFS.
- Enero de 2015: versión 4.0 de Proxmox Mail Gateway. En febrero Proxmox VE 3.4 soporta completamente ZFS. Proxmox cumple 10 años.

En el capítulo 2 hemos estudiado conceptos y repasado las tecnologías actuales de virtualización, analizando las usadas en Proxmox con algo más de detalle y viendo un resumen de sus características. En las siguientes secciones estudiaremos nuestro entorno de virtualización, cómo funciona y sus características para sacarle el máximo rendimiento.

2. Contenedores y máquinas

El núcleo de Proxmox está formado por tres tecnologías abiertas que ya hemos mencionado: **LXC, KVM y QEMU.**

Ilustración 29: Proxmox VE – crear contenedor (CT) o máquina(VM)

La combinación de KVM/QEMU es la que posibilita el trabajo con máquinas virtuales, aportando paravirtualización o virtualización completa. KVM (Kernel-based Virtual Machine) forma parte del kernel de Linux desde febrero de 2007. Se basa en tres módulos (kvm.ko, kvm-intel.ko y kvm-amd.ko) y su función es aportar virtualización completa aprovechando las extensiones hardware del equipo (Intel VT-X o AMD-V). La interfaz de comunicación con el espacio de usuario se integra en QEMU.

Ilustración 30: Logotipos KVM y LXC

QEMU es un emulador / virtualizador genérico y open source. Como virtualizador se apoya en XEN o KVM y alcanza un rendimiento casi de sistema nativo. Proporciona una interfaz que permite control al usuario y puede ser automatizada mediante scripts.

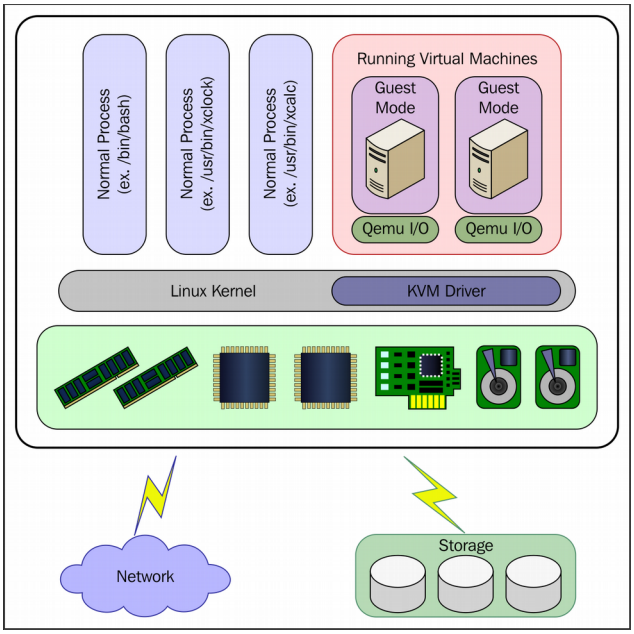


Ilustración 31: Integración KVM – QEMU

Para la gestión de contenedores Proxmox se apoya en LXC. Ya hemos explicado la virtualización a nivel de sistema operativo y cómo funciona, pero recordemos que su objetivo era proporcionar un entorno lo más parecido posible a una máquina virtual, pero evitando la sobrecarga de un nuevo kernel y la simulación del hardware. Por tanto, solo son válidos para simular invitados del mismo tipo que el anfitrión, con kernel GNU Linux. Para sistemas Windows, hay que recurrir a KVM/QEMU.

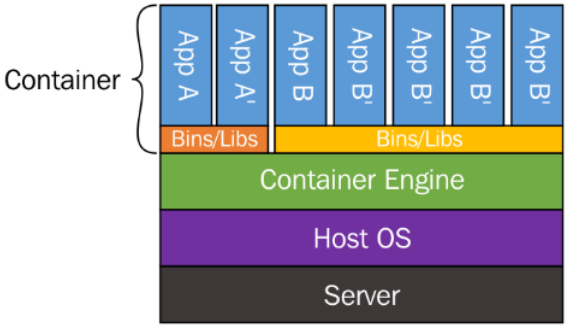


Ilustración 32: Contenedores LXC

Las características principales que distinguen a LXC de la virtualización completa son las siguientes:

- Consumo mucho menor de CPU y de RAM. La capa de virtualización es mucho más ligera que en la completa. Esto es porque funcionan sobre el mismo kernel, y aunque los contenedores están completamente aislados entre ellos, comparten las librerías dinámicas cargadas en RAM con el anfitrión.
- Un usuario puede modificar cualquier configuración en un contenedor o instalar software sin interferir en el resto o en el anfitrión.
- Los contenedores arrancan en segundos, mientras que una máquina virtual puede tardar minutos con los mismos recursos.
- El tráfico de red está perfectamente aislado entre ellos. Un contenedor no puede capturar datos de otro.
- Tienen acceso a características avanzadas de gestión de tráfico. Es posible ejecutar sistemas de firewall y enrutamiento dentro los contenedores.

La instalación y configuración de una máquina QEMU/KVM se realiza tal y como se haría sobre un equipo físico real, usando el instalador del sistema operativo y adaptándose al hardware emulado. Para aprovechar las características de paravirtualización se instalarán controladores personalizados.

La creación de un contenedor se realiza a partir de una plantilla. En Proxmox VE existen dos tipos de las mismas:

- Plantillas de sistema operativo. Contienen un conjunto mínimo de paquetes, librerías, scripts y utilidades básicas de alguna distribución GNU Linux. Están pensadas para

aprovechar los beneficios de un sistema linux completo y limpio sobre el mismo kernel del anfitrión. No incluyen generalmente software extra para permitir su personalización para el uso deseado.

- Plantillas de aplicación. Estas plantillas se crean con el objetivo de proveer un servicio concreto, se apoyan en el principio JeOS (Just enough Operating System). Incluyen el mínimo de componentes necesarios para que funcione una aplicación determinada, de la forma más segura y eficiente posible. Pueden proporcionar una funcionalidad más general, como por ejemplo, un sistema LAMP, o algo más específico como puede ser un Wordpress, un CMS. Todo ello con una mínima intervención del usuario para el despliegue y configuración.

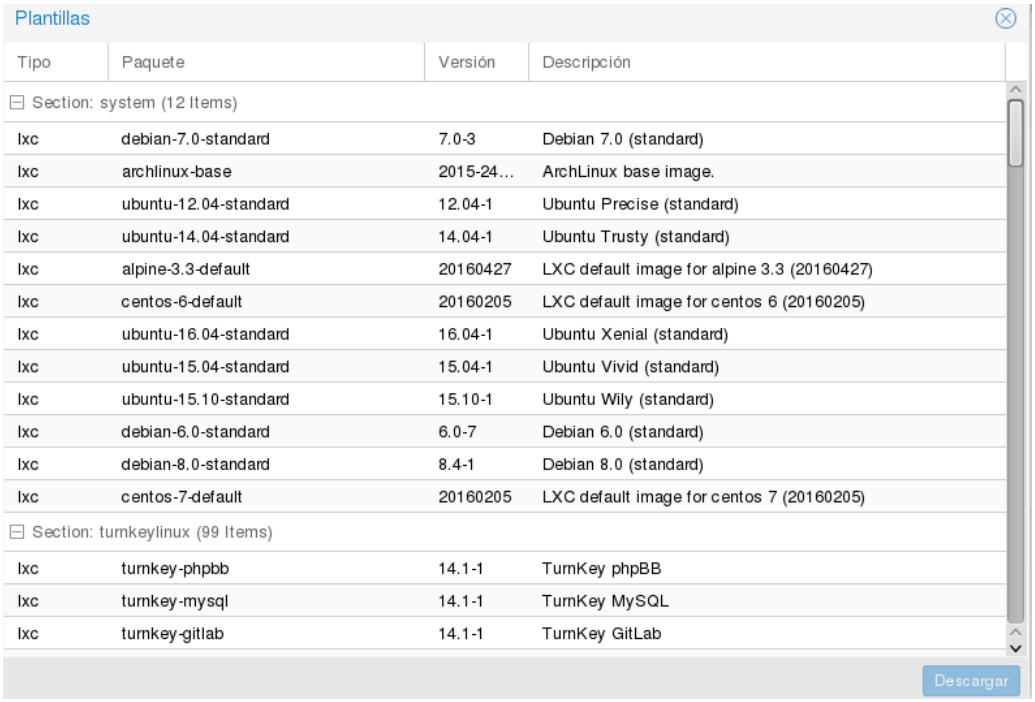


Ilustración 33: Plantillas de sistema y aplicaciones en Proxmox VE

Ambos tipos de plantilla se pueden descargar directamente desde la interfaz de Proxmox. En la imagen se muestran 12 disponibles de sistema y 99 de aaplicación. Estas se obtienen del proyecto Turnkey Linux[40], que ofrece más de 100 contenedores preconfigurados, con base Debian 8, utilizables en distintos entornos como Proxmox, VMWare, Docker o Xen.

Dispondremos entonces, tanto en los contenedores como en las máquinas, de entidades aisladas que pueden ser reiniciadas de forma independiente, tener su propio conjunto de

40 https://www.turnkeylinux.org/all

usuarios y grupos, y acceso root o administrador. Cada una con su configuración, dirección (o direcciones) IP, memoria, procesos, librerías, etc.

3. Modelo de almacenamiento

El sistema de almacenamiento en Proxmox VE destaca por su flexibilidad. Permite ubicaciones locales, compartidas en red o distribuidas y de distintos tipos. Se puede usar cualquier tecnología que esté disponible para Debian GNU Linux. Un dato importante a tener en cuenta para la elección del almacenamiento es que el tipo compartido permite la migración en vivo de máquinas virtuales entre nodos del clúster.

Las tecnologías de almacenamiento en red soportadas y gestionables directamente desde la interfaz web son las siguientes:

- Dispositivos iSCSI
- Grupos LVM (sobre dispositivos iSCSI)
- Comparticiones NFS
- Dispositivos de bloques Ceph
- Conexión directa con LUNs de iSCSI (unidades lógicas)
- GlusterFS

En almacenamiento local:

- LVM Group (sobre dispositivos locales)
- Directorios (montaje en cualquier sistema de ficheros local)
- ZFS

Hay que tener en cuenta que cada tipo de almacenamiento tiene un propósito diferente, y no todos almacenan cualquier información. Para mantener actualizada esta lista de compatibilidad podemos consultar la documentación en:

http://pve.proxmox.com/wiki/Storage_Model

En Proxmox, la información que se gestiona desde el hipervisor se estructura en cinco categorías:

- Imágenes: discos virtuales de máquinas KVM/QEMU.
- Contenedores: discos virtuales de los contenedores LXC.
- ISO: ficheros .iso para cargar en los lectores de cdrom de las máquinas virtuales.
- Plantillas: las fuentes para crear contenedores que hemos visto anteriormente, de tipo sistema o aplicación.
- Backups: copias de seguridad de contenedores y máquinas.

4. Modelo de red

La infraestructura virtual de red de Proxmox VE se basa en un modelo de puente, en capa 2. Todas las máquinas (y contenedores) pueden compartir el mismo puente como si cada una estuviera físicamente enchufada con un cable al mismo switch. El anfitrión es el encargado de conectar ese puente con el mundo real mediante sus interfaces de red, y de gestionar cómo se realiza este acceso.

Ilustración 34: Proxmox VE – Configuración de red

Para mayor flexibilidad, cada nodo de Proxmox puede mantener hasta 4094 puentes, admite configuraciones de VLAN (IEEE 802.11q), agregación de enlaces y toda la potencia de la pila de red de GNU Linux, con iptables y ebtables. Se pueden crear subredes enrutadas, con NAT o balanceos. Desde la versión 3.2[41], el proyecto Open vSwitch también está disponible en Proxmox.

41 http://pve.proxmox.com/wiki/Roadmap#Proxmox_VE_3.2

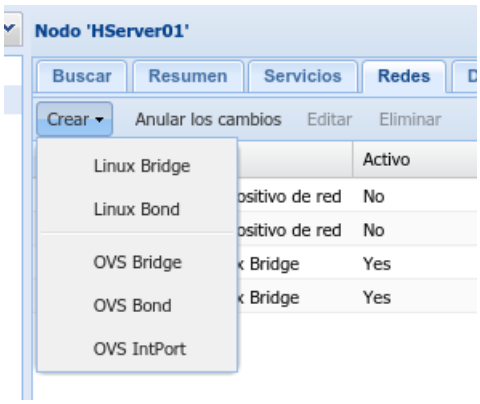


Ilustración 35: Proxmox VE – Linux bridges y Open vSwitch

5. Requisitos hardware de la plataforma

Un punto a destacar de esta plataforma de virtualización es la baja exigencia de infraestructura hardware que conlleva. A diferencia de otras plataformas, como VMware vSphere, Proxmox VE es un sistema *stand-alone:* no requiere de un equipo externo para la gestión del hipervisor, se puede realizar todo el trabajo sobre el mismo servidor. Para evaluación en entornos de prueba se recomienda un equipo que cumpla simplemente con las siguientes características:

- Procesador con arquitectura de 64bit y placa base con soporte de Intel VT o AMD-V para realizar virtualización completa sobre KVM.
- Mínimo de 1GB de RAM
- Disco duro y tarjeta de red.

Y como hardware para un sistema en producción nos recomiendan el siguiente:

- Servidor con dos o cuatro sockets y procesador de 4/6/8 núcleos.
- CPU de 64bit con Intel VT o AMD-V.
- 8GB de RAM, cuanta más mejor.
- Sistema para RAID por hardware y baterías de protección de caché.
- Unidades de disco rápidas, SAS o RAID10 recomendado
- Dos tarjetas de red Gigabit.

En las pruebas realizadas durante este trabajo hemos observado que estas recomendaciones están bastante por encima de lo necesario para obtener un sistema estable y con un rendimiento adecuado, aunque claro, esto siempre dependerá de nuestra exigencia y la carga a la que vayamos a someter al sistema. Una muestra de esto es el siguiente sistema:

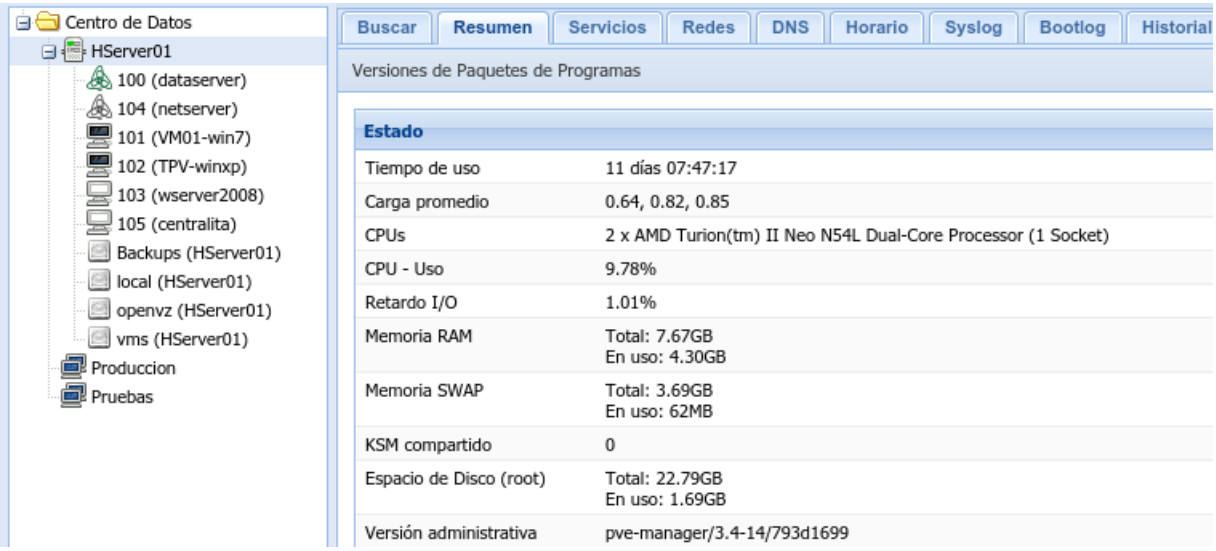


Ilustración 36: Proxmox VE – carga del sistema

Este equipo lleva un procesador AMD Turion II N54L de 2 núcleos, con reloj de 2.2GHZ (se trata de una CPU de bajo consumo, con disipación pasiva y rendimiento limtado). Tiene instalados 8GB de memoria RAM y dos discos duros configurados en RAID por software. Como podemos observar en la imagen, con un contenedor y dos máquinas virtuales ejecutándose (100, 103 y 104) la carga promedio del sistema durante los últimos 1, 5 y 15 minutos es de 0.64, 0.82 y 0.85 (para profundizar en el significado estos valores podemos consultar cualquier artículo sobre Unix Load Average[42]). Teniendo en cuenta que el sistema tiene dos núcleos, podemos entender que su utilización está por debajo del 50%.

6. Licencias y precio del soporte

El sistema de licencias de Proxmox VE ha sido otro factor de peso en su elección como plataforma para este proyecto. A diferencia de otros entornos, es muy sencillo y económico. No existe ningún coste de compra inicial y se basa en la calidad del soporte que queramos recibir. Incluso, si no queremos soporte y preferimos trabajar con los repositorios de la comunidad,

42 http://www.howtogeek.com/194642/understanding-the-load-average-on-linux-and-other-unix-like-systems/

usando el software en pruebas también lo podemos hacer sin ninguna limitación, pero esto no se recomienda para entornos de producción.

En la web del proyecto podemos consultar las opciones, que son las 4 siguientes:

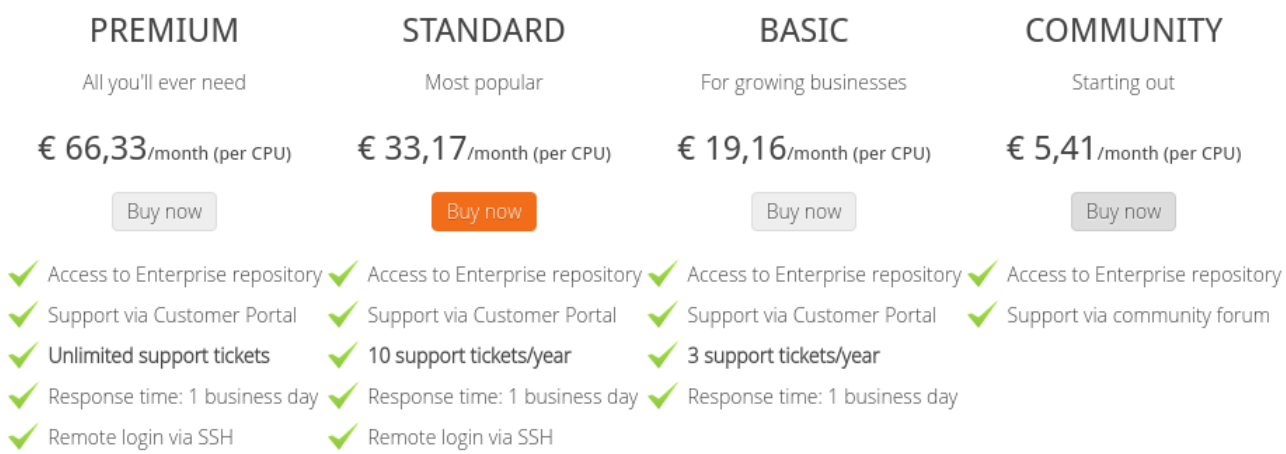


Ilustración 37: Proxmox VE – planes de subscripción

1. Community: Soporte de actualizaciones y acceso a repositorios estables. 5,41€ / mes por CPU.
2. Basic: Community + 3 tickets de soporte al año con respuesta en un día laborable. 19,16€ / mes por CPU.
3. Standard: actualizaciones, acceso a repositorios estables, 10 tickets de soporte al año, posibilidad de soporte remoto mediante SSH. 33,17€ / mes por CPU.
4. Premium: mismos servicios que Standard pero sin límite de tickets de soporte. 66,33€ / mes por CPU.

Capítulo IV: Desarrollo del entorno de trabajo

1. Diseño del sistema

1.1. Planteamiento general

Recordemos los puntos que nos marcamos como objetivo en el capítulo de introducción. Debemos construir un entorno que cumpla las siguientes características:

- Económico y escalable → Podremos empezar con un sistema mínimo, que se adapte a los requisitos y las posibilidades de una pequeña empresa, pero que pueda ampliarse de forma sencilla conforme lo haga esta.
- Base hardware sólida, redundante → Si es necesario queremos poder agrupar y duplicar recursos para asegurar nuestra infraestructura. La posibilidad de clústering o las configuraciones de discos en RAID serán básicas.
- Abstracción del hardware, flexibilidad → Virtualización. No queremos depender de ninguna plataforma hardware concreta.
- Copias de seguridad con garantías → Gestionadas desde el anfitrión de virtualización en equipos dedicados y seguros, aisladas del espacio de usuario. Fáciles de restaurar y mantener.

Una vez hemos estudiado las características de Proxmox VE, sabemos que estos objetivos los podemos alcanzar sin mucha dificultad usando sus herramientas. Partamos del siguiente esquema para desarrollar la idea.

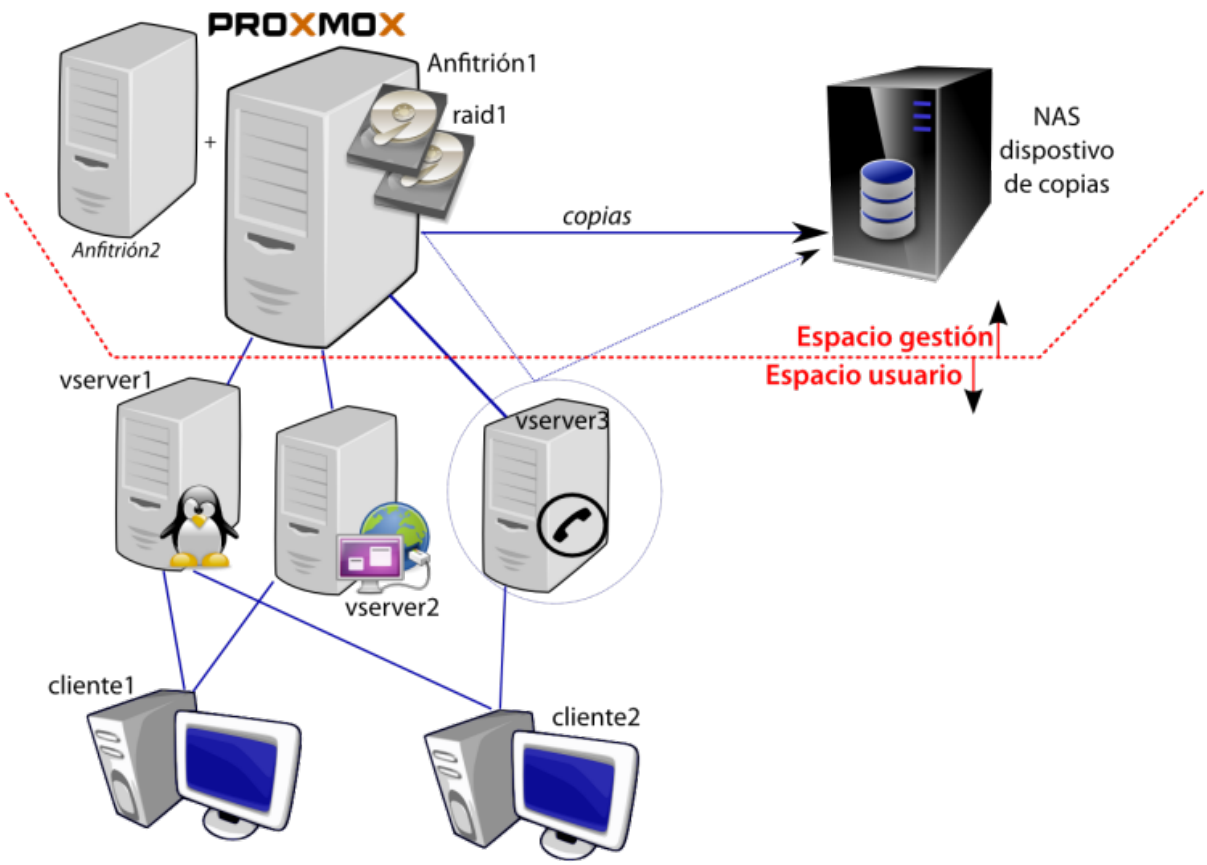


Ilustración 38: Diseño del entorno de trabajo Proxmox

En primer lugar planteamos una separación entre el espacio en que trabajará el usuario final y el espacio del administrador del sistema. Toda la infraestructura virtual será transparente al usuario, éste solamente interaccionará con los recursos necesarios o que el administrador decida. Añadimos así una capa de seguridad y prevenimos problemas por errores humanos. Del esquema podemos extraer los dispositivos que necesitamos para empezar:

- Un equipo servidor. Puesto que el entorno Proxmox VE no necesita un hipervisor independiente y puede funcionar en un único equipo partimos de un solo servidor. Sus especificaciones dependerán de los servicios que queramos desplegar: teniendo en cuenta los requisitos mínimos de Proxmox y las máquinas virtuales que levantaremos podemos dimensionar la RAM, CPU o espacio en disco a medida.

- La redundancia: para prevenir fallos hardware montaremos como mínimo dos discos duros locales en RAID modo espejo. Si la exigencia es mayor siempre podemos añadir una red SAN de alto rendimiento, o incluso un segundo servidor anfitrión con la misma configuración y así reducimos además el tiempo de recuperación en caso de que el fallo sea del propio equipo. Al diseño del almacenamiento dedicaremos una sección completa más adelante, pues es un pilar central del sistema.

- Para las copias de seguridad necesitamos un soporte independiente, preferiblemente que podamos instalar en una ubicación física alejada del hipervisor. Un NAS es una buena opción: económica, sencilla y cumple nuestros requisitos. Su potencia y su capacidad de almacenamiento (número de bahías) los elegiremos acorde a los cálculos realizados para el servidor. En caso de querer reducir aún más el coste del montaje, podemos usar para los backups un tercer disco duro instalado en el mismo servidor. Esos datos seguirán siendo transparentes al espacio de usuario y estarán relativamente a salvo, pero corremos el riesgo de que daños físicos (por ejemplo, un fuego) acaben con todo el sistema y no tengamos opción de recuperación.

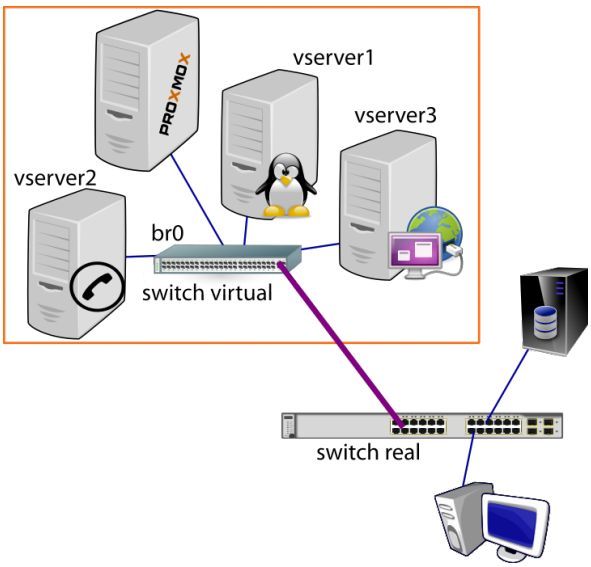


Ilustración 39: Modelo de red básico – entorno Proxmox

- Puesto que estamos planteando un entorno de trabajo en red, será necesaria una infraestructura mínima de comunicaciones. Lo ideal sería una red gigabit ethernet **cableada** (el rendimiento de las redes inalámbricas en la práctica deja mucho que desear, mejor evitarlas) entre puestos y servidor. Si deseamos una infraestructura de red más

avanzada (con diferentes subredes, vlans… etc) es suficiente con utilizar un switch gestionable para conectar el servidor. Como ya hemos visto, Proxmox soporta VLAN y es muy flexible en las configuraciones de red. Podríamos configurar un enlace troncal con varias VLAN y conectar cada servidor virtual a una subred independiente.

- Es importante también protegernos ante cortes en el suministro eléctrico para evitar daños o pérdidas de información. Un sistema de alimentación ininterrumpida (SAI) será imprescindible en la instalación. Debemos tener en cuenta que no sólo hay que proteger el servidor o el NAS, también la electrónica de red involucrada en la comunicación puesto – servidor para que no se produzca una desconexión.

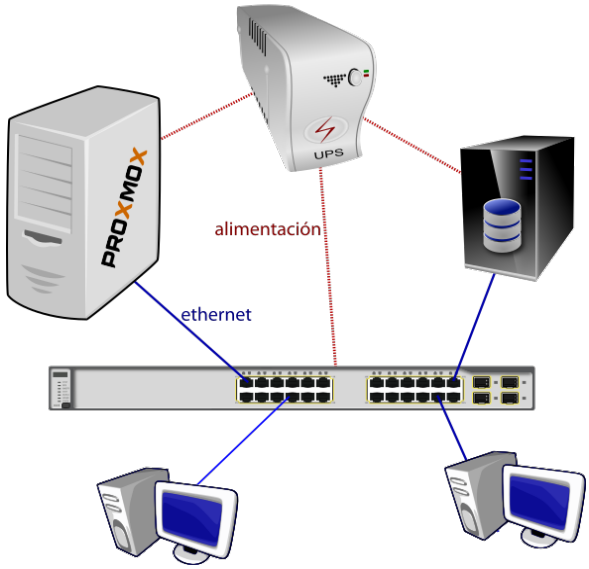


Ilustración 40: Conexionado entorno Proxmox

- El planteamiento de las máquinas virtuales a desplegar, el tamaño y el tipo de almacenamiento, la potencia y el número de los servidores… dependerá de los requisitos del cliente. Una vez los estudiemos diseñaremos la solución inicial adecuada, siempre teniendo en cuenta las posibilidades de expansión.

1.2. Análisis de requisitos

1.2.1. Entorno de trabajo típico en una PYME

Para este desarrollo nos basaremos en el escenario más común que nos hemos encontrado a diario en Informaticasa Soluciones. Concretando el entorno que ya se presentaba en el capítulo de introducción:

- PYME con 4-6 usuarios y puestos de trabajo. El usuario suele tener un nivel bajo de conocimientos informáticos. Los problemas derivados del mal uso y las infecciones por malware son muy habituales. La dependencia con el puesto de usuario final debemos reducirla al mínimo.
- No existe la figura del administrador de sistemas o responsable de TI en la propia empresa. El servicio de mantenimiento se provee desde fuera. Cuanto más opciones para el soporte remoto tenga la infraestructura más sencilla y ágil se hará esta tarea.
- Los trabajadores tienen necesidad de compartir información y trabajar sobre directorios comunes. A su vez, es importante limitar el alcance y los permisos de cada uno a los datos que realmente necesita.
- Software de facturación, contabilidad o similares, con bases de datos, que generalmente funcionan sobre un S.O Windows y que usan desde varios puestos.
- El volumen de información almacenada no es muy alto, principalmente trabajan con documentos de texto, hojas de cálculo, documentos PDF. La información esencial en los puestos rara vez se acerca a los 50GB.
- Ocasionalmente algún empleado puede necesitar trabajar desde una ubicación remota y acceder a todos los recursos de la LAN. Integrar una solución para el acceso remoto puede resultar muy práctico.

A partir de estas características podemos extraer que generalmente será necesario un servidor para compartir ficheros, y también, alguna máquina con sistema operativo de Microsoft. Siguiendo con la filosofía de este trabajo intentaremos aprovechar primero todo lo que nos ofrece el software libre. Recurriremos a servidores Linux sobre LXC (más eficientes) siempre que sea posible, y si no es así, desplegaremos Windows sobre KVM. Veamos a continuación los requisitos de estos S.O para dimensionar las máquinas virtuales.

1.2.2. Requisitos mínimos para Windows 7[43]

- CPU de 1GHz superior, 32-bit (x86) o 64-bit (x64)
- 1GB RAM (32-bit) o 2 GB RAM (64-bit)
- 16GB de espacio en disco para 32-bit o 20 GB para 64-bit

43 https://support.microsoft.com/en-us/help/10737/windows-7-system-requirements

- Tarjeta gráfica para DirectX9 - WDDM 1.0

1.2.3. Requisitos mínimos para Windows Server 2012 / 2012 R2 [44]

- CPU de 1.4 GHz 64-bit
- 512MB RAM *(para que la instalación funcione mejor más de 800MB)*
- 32GB de espacio en disco

1.2.4. Requisitos mínimos para Debian GNU Linux [45]

CPU: Pentium 4, 1GHz es el mínimo recomendado para un sistema de escritorio.

Tipo de instalación	RAM (mínimo)	RAM (recomendada)	Espacio en disco
Con entorno de escritorio	128 MB	512 MB	2 GB
Sin entorno de escritorio	256 MB	1 GB	10 GB

Tabla 7: Requisitos hardware mínimos Debian GNU Linux

1.3. Detalles de la solución adoptada

Recordando las características de Proxmox VE estudiadas en el capítulo 3 y el análisis de requisitos realizado, podemos plantear una estructura de máquinas virtuales que cumpla con nuestras necesidades. Una vez hecho esto, podremos elegir el hardware para el servidor anfitrión adecuado. La siguiente tabla contiene la propuesta sobre la que trabajaremos en este proyecto. Los recursos asignados a cada máquina cumplen sobradamente con el mínimo para cada sistema. La asignación de CPU, como ya se ha estudiado anteriormente, se realiza de forma dinámica a demanda.

44 https://technet.microsoft.com/en-us/library/dn303418(v=ws.11).aspx
45 https://www.debian.org/releases/stable/amd64/ch03s04.html.en

Nombre	Tecnología	S.O.	Recursos asignados	Descripción
dataserver	LXC	Debian GNU Linux	1 vCore 2GHz - HDD 600GB – 2GB RAM	Servidor de ficheros SAMBA
netserver	LXC	Debian GNU Linux	1vCore 2GHz - HDD 10GB – 512MB RAM	Servidor para tareas de gestión de RED, VPN, monitorización, dyndns...
winserver	KVM	Microsoft Windows 7 / Windows Server 2012 ...	2 vCore 2GHz - HDD 120GB – 3GB RAM	Servidor para aplicaciones Windows
pbx	*KVM*	*Linux (FreePBX)*	*1 vCore 2GHz - HDD 10GB – 1GB RAM*	*(opcional – futuro) Centralita de telefonía IP*

Tabla 8: Propuesta de máquinas virtuales para 4 – 6 puestos

Con estos datos ya podemos estimar las características para elegir un equipo anfitrión. Necesitaremos disponer de los siguientes recursos para los servidores virtuales:

- CPU: 4vCores de unos 2GHz. Aunque como ya hemos estudiado, la asignación se realiza de forma dinámica según demanda. No es necesaria una reserva de recursos que garantice el 100% del tiempo de proceso.
- Memoria RAM asignada: 5,5GB
- Espacio en disco: 730GB. Al diseño del almacenamiento dedicaremos una sección completa, pues con solo 2 discos duros y una configuración adecuada, apoyándonos en RAID y LVM lograremos una base fiable y flexible para el crecimiento futuro.

1.4. Selección del hardware.

El desarrollo práctico de este proyecto se ha llevado a cabo en las oficinas de Informaticasa Soluciones. El equipo se ha elegido entre la oferta disponible en los proveedores, teniendo en cuenta su coste y el estudio realizado en el apartado anterior. Se han valorado los mostrados en la siguiente tabla:

Equipo	Procesador	Memoria RAM	Almacenamiento	Otras especificaciones	Precio (IVA incluido)
HP Proliant Microserver G8 819185-421	Intel Celeron G1610T 2cores2.3 GHz	4GB DDR3 (2 ranuras, max 16GB)	No incluye HDD 4 bahías 3,5" SATA3 Max 16TB	2 x puertos RJ45 Gigabit 2xUSB3.0 5xUSB2.0	200€
HP Proliant ML310E G8 v2 470065-800	Intel Xeon E3-1220v3 4cores 3.1 GHz	8GB DDR3 (4 ranuras, max 32GB)	2 x HDD 1TB 4 bahías 3,5" SATA3 Max 16TB	2 x puertos RJ45 Gigabit 2xUSB3.0 5xUSB2.0	750€
HP Proliant ML150 G9 780849-425	Intel Xeon E5-2609 V3 6 cores, 1.9GHz	8GB DDR4 (16 ranuras, max 512GB)	No incluye HDD 10 bahías 3,5" SATA3 Max 60TB	2 x puertos RJ45 Gigabit 4xUSB3.0 2xUSB2.0	1000€
Dell Power Edge T20 20-3692	Intel Pentium G3220 2cores 3GHz	4GB DDR3 (4 ranuras – max 32GB)	1 x HDD 500GB 6 bahías 3,5" SATA3 Max 13TB	1 x puerto RJ45 Gigabit 4xUSB3.0 8xUSB2.0	330€

Tabla 9: Opciones de servidor para el desarrollo

La opción elegida ha sido el equipo HP Proliant ML310E G8 v2 (modelo 470065-800). Por precio y características es el que más se ajusta a los objetivos de este trabajo. Cumple los requisitos de CPU, RAM y viene configurado con dos discos duros de 1TB (necesitábamos unos 730GB para las VMs, y siempre dos unidades para realizar el RAID-1).

Ilustración 41: HP Proliant ML310e Gen8 v2

1.5. Diseño del sistema de almacenamiento local. Particionado.

Como ya hemos visto, la instalación *bare metal* de Proxmox VE no integra soporte para RAID por software, trabaja directamente sobre LVM. Para añadir esta funcionalidad y alcanzar nuestro objetivo de almacenamiento óptimo necesitamos instalar Debian GNU Linux, crear el mapa de particiones de forma manual y posteriormente integrar Proxmox VE. Empezaremos el

diseño por la capa inferior (la física, el disco duro) e iremos subiendo en nivel de abstracción hasta terminar con la partición lógica final con el sistema de ficheros.

1. Como dispositivo de almacenamiento se ha optado por 2 discos duros de 1TB. Los configuraremos en RAID-1 por software con mdadm, en un nivel superior. Si es posible, elegiremos discos duros diseñados para almacenamiento en red y uso 24/7, y montaremos dos de la misma capacidad y características pero distinto fabricante. Así reducimos la probabilidad de un fallo simultáneo (el tiempo medio entre fallos *MTBF* suele ser diferente).

2. El primer nivel de configuración consiste en las particiones "físicas" que se crean directamente sobre los discos duros. Nuestro esquema consta de 4 particiones, y deberá ser exactamente el mismo en los dos dispositivos:

Dispositivo	Tamaño	Tipo	Descripción
/dev/sda1 /dev/sdb1	1MB	BIOS BOOT	En discos grandes con tabla de particiones tipo GPT y BIOS (no en sistemas UEFI ni para tablas tipo MBR) es necesario dejar una partición de aproximadamente 1MB para el arranque de GRUB. El tipo es BIOS BOOT y si queremos instalar el gestor de arranque en los dos discos hay que dejarla en ambos.
/dev/sda2 /dev/sdb2	1GB	LINUX RAID	Particiones tipo LINUX RAID para configurar el dispositivo RAID-1 en el siguiente nivel. Este bloque está pensado para la partición de arranque /boot.
/dev/sda3 /dev/sdb3	19GB	LINUX RAID	Particiones tipo LINUX RAID para configurar el dispositivo RAID-1 en el siguiente nivel. Este bloque está pensado para la instalación del sistema operativo anfitrión.
/dev/sda4 /dev/sdb4	980GB (restante)	LINUX RAID	Particiones tipo LINUX RAID para configurar el dispositivo RAID-1 en el siguiente nivel. Este bloque está pensado para datos

Tabla 10: Nivel 2 del esquema de almacenamiento. Particiones físicas.

3. El siguiente paso consiste en utilizar *mdadm* para crear el RAID por software y obtener los dispositivos virtuales.

Dispositivo	Tamaño	Tipo	Descripción
/dev/sda1 /dev/sda2	N/ A		
/dev/md0	1GB	Dispositivo de bloques → ext4 /boot	Dispositivo de bloques virtual resultado del espejo /dev/sda2+ /dev/sdb2. Se usará para la partición tipo ext4 /boot. Esta partición no suele necesitar cambios ni aumento de tamaño con el tiempo, por lo que se deja fuera de LVM (pero sí en el RAID), para facilitar la compatibilidad y acelerar el arranque.
/dev/md1	19GB	Dispositivo de bloques → LVM PV	Dispositivo de bloques virtual resultado del espejo /dev/sda3+ /dev/sdb3. Se usará como volumen físico LVM para incluir en el grupo destinado al sistema anfitrión.
/dev/md2	980GB	Dispositivo de bloques → LVM PV	Dispositivo de bloques virtual resultado del espejo /dev/sda4+ /dev/sdb4. Se usará como volumen físico LVM para incluir en el grupo destinado a máquinas virtuales.

Tabla 11: Nivel 3 del esquema de almacenamiento. mdadm RAID.

4. Un nivel más arriba aparecen los LVM Volume Groups. Estos conforman unidades contenedoras de espacio. Serán tan grandes como los PV que contengan.

Dispositivo	Tamaño	Tipo	Descripción
VG SistemaHost	19GB	LVM VG	LVM Volume group. Contiene inicialmente a **/dev/md1**. Se usará para regular el espacio necesario para el sistema anfitrión.
VG VirtDatos	980GB	LVM VG	LVM Volume group. Contiene inicialmente a **/dev/md2**. Se usará para regular el espacio asignado a máquinas virtuales.

Tabla 12: Nivel 4 del esquema de almacenamiento. LVM Volume Groups.

5. En la capa superior configuramos los volúmenes lógicos, con el tamaño que queramos asignar de inicio, el sistema de ficheros y el punto de montaje.

Dispositivo	Tamaño	Tipo	Descripción
/dev/md0	1GB	ext4	**/boot** → Ficheros responsables del arranque del sistema
/dev/mapper/ SistemaHost-root	10GB	ext4	**/** → Raíz del sistema.
/dev/mapper/ SistemaHost-var	5GB	ext4	**/var** → Como su nombre indica, el subdirectorio var contiene datos variables, que se van modificando durante la ejecución del sistema operativo (por ejemplo, los LOGS). Por seguridad, para limitar su crecimiento y evitar una posible saturación de todo el sistema en caso de problemas, es recomendable confinarlo en su propia partición.
/dev/mapper/ SistemaHost-swap	4GB	swap	Área de intercambio. Caché del sistema.
/dev/mapper/ VirtDatos-*	--	--	Estas serán las unidades de disco de las máquinas virtuales desplegadas en un futuro. El espacio disponible inicialmente son los 980GB del grupo VirtDatos.

Tabla 13: Nivel 5 del esquema de almacenamiento. Particiones lógicas.

En el siguiente gráfico se ilustra de forma clara todo el diseño planteado. Se muestra también cómo aparecería un tercer disco duro instalado para realizar copias de seguridad locales en el propio anfitrión.

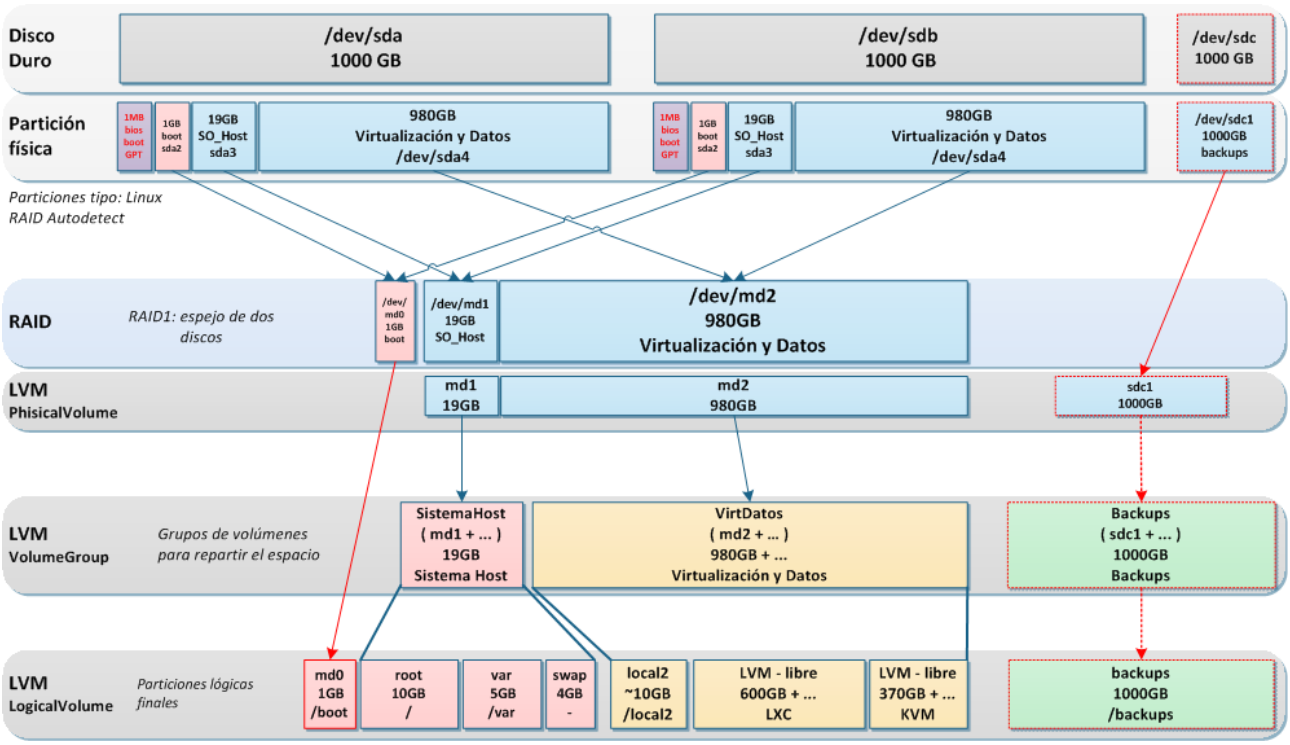


Ilustración 42: Diseño del sistema de almacenamiento local

2. Manos a la obra. Instalación de Debian GNU Linux.

2.1. Entorno de laboratorio.

Para el desarrollo de la práctica hemos contado con los siguientes recursos:

- Ordenador portátil con S.O. Debian GNU Linux como equipo de apoyo.
- Ordenador de sobremesa con S.O. Windows 7 como equipo de apoyo, conectado a un teclado, ratón y monitor mediante un KVM, al que conectaremos a continuación el equipo servidor a instalar.
- SAI para alimentar de forma segura el equipo de sobremesa y el servidor durante la instalación. El portátil ya cuenta con su propia batería.
- Subred 192.168.7.0/24 para el acceso a internet. Los tres equipos estarán conectados al mismo switch.

Desembalamos el equipo, recordemos que habíamos elegido el HP Proliant ML310E G8 v2 (modelo 470065-800). Ya viene con los dos discos duros de 1TB instalados, por lo que no necesitamos hacer ninguna configuración hardware. Conectamos alimentación, red y KVM para monitor, teclado y ratón, y empezamos.

Ilustración 43: HP Proliant ML310E

2.2. Obtención del Sistema Operativo

En primer lugar necesitamos el disco que utilizaremos para instalar el sistema operativo en el anfitrión. Descargaremos la ISO en versión netinstall (mínima, haremos una instalación actualizada por red) estable de Debian GNU Linux desde las fuentes oficiales.

https://www.debian.org/CD/netinst/#netinst-stable

La utilizada en nuestro caso es la siguiente:

http://cdimage.debian.org/debian-cd/8.3.0/amd64/iso-cd/debian-8.3.0-amd64-netinst.iso

Grabamos en un CDROM y la usaremos para arrancar en el servidor.

2.3. Configuración BIOS del equipo

Antes de comenzar a instalar Debian necesitamos ajustar algunos parámetros en la BIOS de nuestro servidor. Pulsando F9 entramos al menú de configuración.

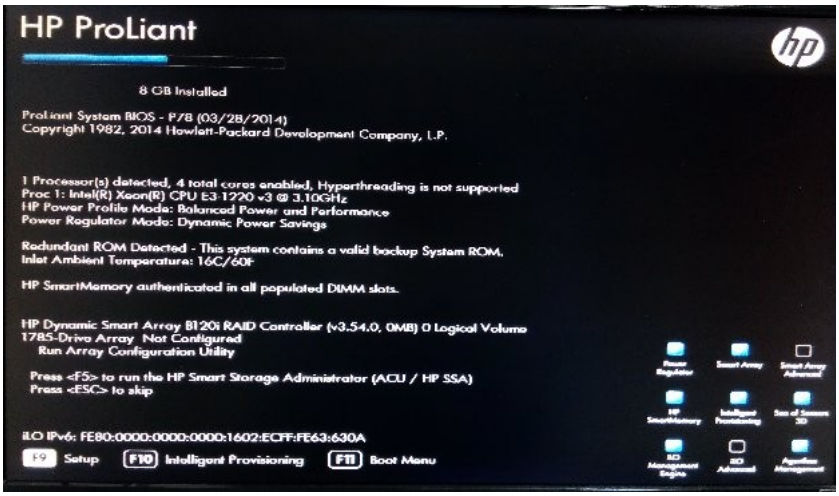


Ilustración 44: Arranque del servidor

1. Desactivar la controladora RAID integrada. Pondremos los discos en modo AHCI SATA para luego realizar el RAID con mdadm.

```
System Options → SATA Controller Options → Embedded SATA Configuration
→ Enable SATA AHCI Support
```

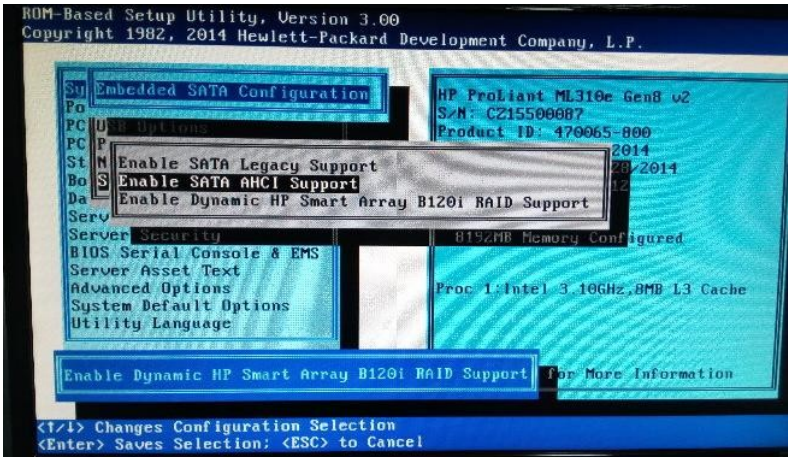


Ilustración 45: Configuración BIOS 1

2. Definir los discos duros como dispositivo de arranque principal. Para elegir otra fuente de forma puntual como el CDROM podemos hacerlo a mano pulsando F11.

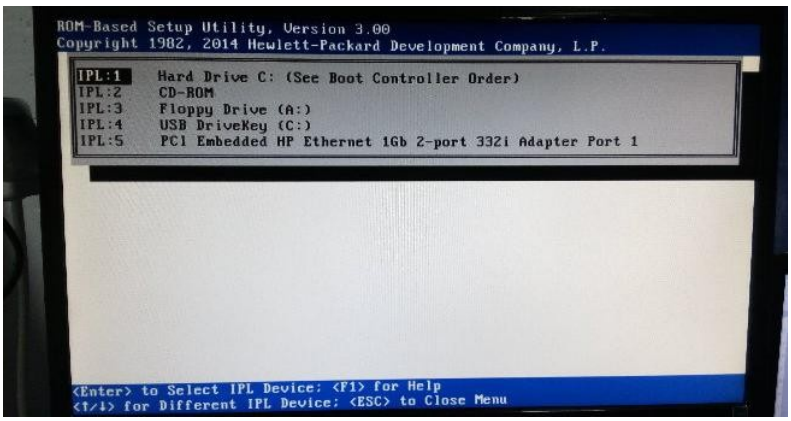


Ilustración 46: Configuración BIOS 2

3. Para evitar un molesto mensaje que aparece continuamente[46] en el registro del sistema (*dmesg*) desactivaremos la opción "Processor Power and Utilization Monitoring".

```
CTRL + A (hace aparecer la opción) → "Service Options." → "Processor
Power and Utilization Monitoring." → "Disable"
```

Fuente de la solución:

http://h20564.www2.hpe.com/hpsc/doc/public/display?docId=emr_na-c03265132

2.4. Instalación de Debian GNU Linux

Arrancamos desde el CDROM que hemos grabado anteriormente con la ISO de instalación. Para ello pulsamos F11 en la secuencia de arranque de la BIOS. En el menú inicial del instalador seleccionaremos la opción gráfica avanzada.

```
Advanced Options → Graphical expert install
```

Las siguientes opciones son bastante intuitivas. Configuramos el idioma:

```
Idioma: España – es_ES.UTF-8
```

Seleccionamos los componentes que queremos cargar en el instalador:

```
cfdisk-udeb, choose-mirror, parted-udeb, network-console
```

Los primeros nos aportan opciones avanzadas en el particionado y la selección del repositorio en red para instalación. En este caso hemos añadido también **network-console** para poder continuar la instalación desde un equipo remoto mediante ssh y así mejorar la calidad de las futuras capturas de pantalla, para una instalación corriente no sería necesario.

46 [Firmware Bug]: The BIOS Has Corrupted Hw-PMU Resources

A continuación la configuración de red se hace de forma automática por DHCP. Aparecerán nuestras dos interfaces Broadcom (eth0 y eth1) y obtendremos una IP de la subred 192.168.7.0/24 en la que hayamos conectado el cable. En el siguiente punto pasaremos ya a continuar la instalación desde el equipo remoto. La pantalla siguiente nos mostrará la información necesaria para establecer la conexión desde el cliente SSH.

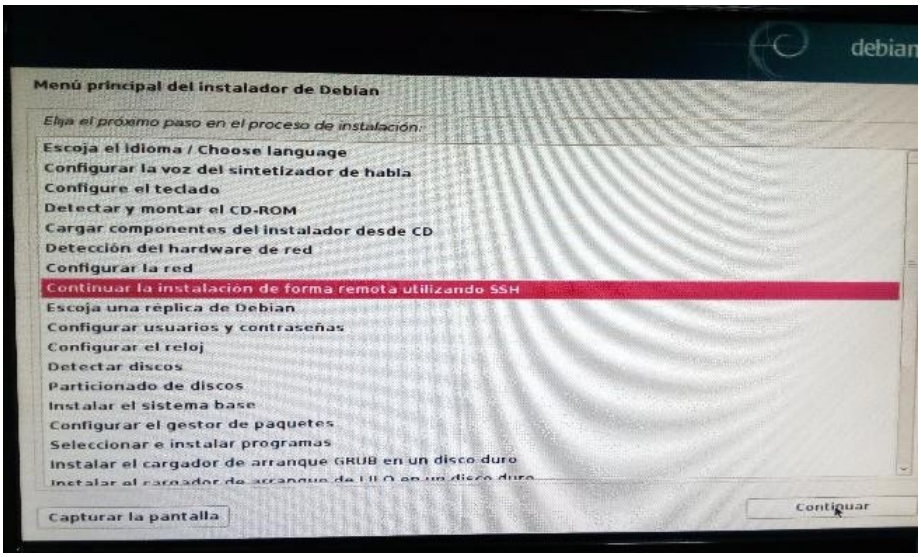


Ilustración 47: Cambio a instalación remota SSH

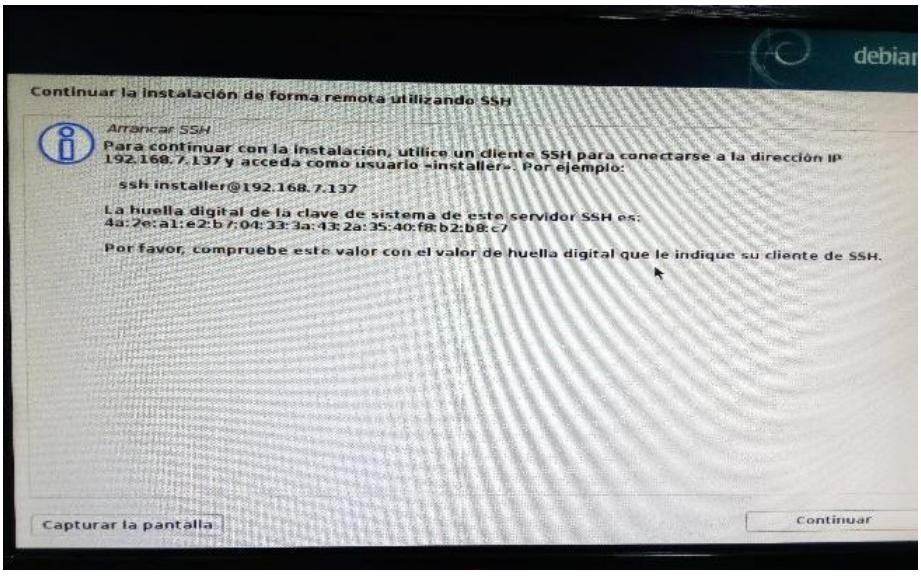


Ilustración 48: Cambio a instalación remota SSH - 2

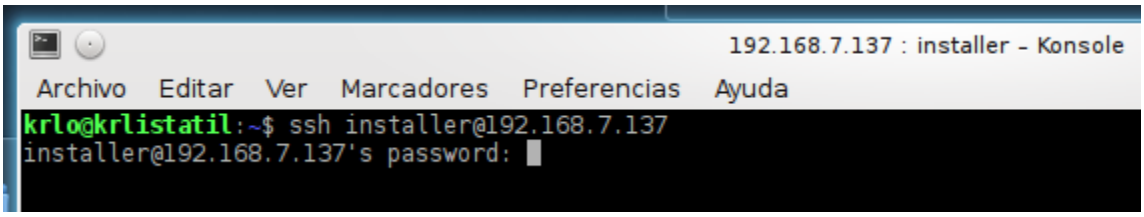


Ilustración 49: Conexión desde el cliente SSH

Las primeras opciones que se nos presentas son sencillas. No adjuntaremos capturas para no agrandar de forma innecesaria este documento. Comentaremos a continuación las más especiales y las que sacaremos del valor por defecto.

```
Iniciar el instalador (modo experto)
```

Por experiencias anteriores del autor de este documento con instalaciones de Debian, elegiremos un repositorio distinto al español, por ejemplo Francia, Suecia o Alemania. Suelen ser más rápidos y estables.

```
Escoja una réplica de Debian → Francia
```

El permitir el trabajo como root o forzar el uso de **sudo** es una discusión constante entre los administradores del mundo de Linux. En este trabajo confiaremos en el sentido común del administrador y nos decantaremos por habilitarlo y así eliminar el engorro de teclear sudo.

```
¿Permitir acceso como superusuario (root)? → sí
```

Establecemos una clave para root y crearemos un usuario estándar. Por ejemplo:

```
root – claveroot
usuario – claveusuario
```

Como servidor de hora podemos usar el del servidor de tiempo NTP instalado en el Real Instituto y Observatorio de la Armada, que es el oficial de España[47].

```
Hora con servidor NTP: sí →  hora.roa.es
```

2.4.1. Configuración de discos. Particionado.

El siguiente punto del asistente nos lleva a la configuración de discos y el particionado. En la sección "1.5 Diseño del sistema de almacenamiento local. Particionado." ya planteamos el esquema, ahora toca llevarlo a la práctica.

47 http://www.armada.mde.es/ArmadaPortal/page/Portal/ArmadaEspannola/ciencia_observatorio/prefLang_es/06_Hora--01_QueHoraEs

```
Detectar discos → Particionado → Manual
```

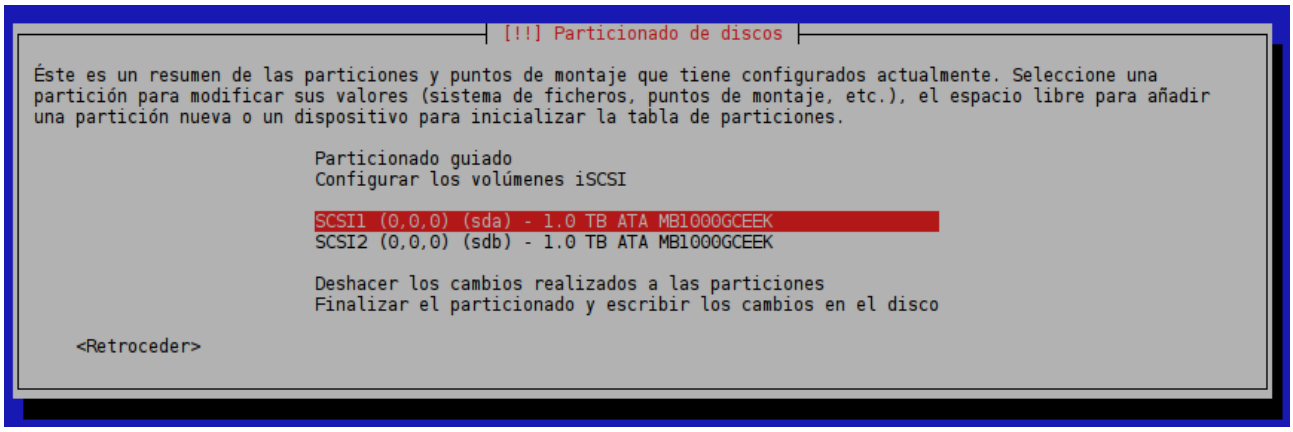


Ilustración 50: Inicio particionado de discos duros

Empezamos eligiendo el tipo de tabla de particiones. Aunque para un disco de 1TB podemos utilizar MBR (msdos) lo haremos ya con el formato más nuevo GPT, y así esta instalación es compatible con discos más grandes.

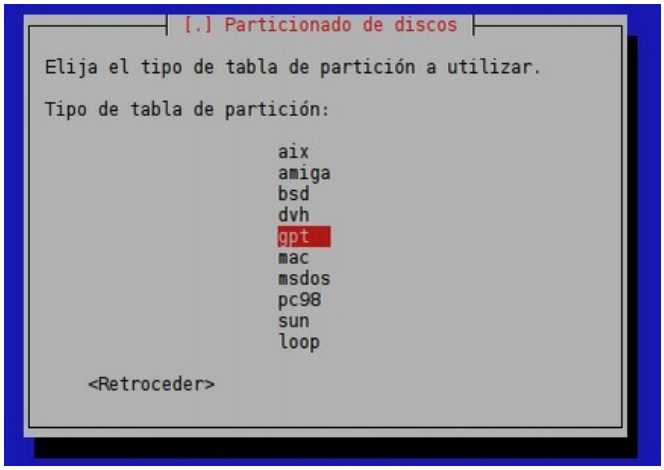


Ilustración 51: Formato tabla de particiones

En los siguientes pasos deberemos crear las 4 particiones correspondientes al nivel 1 de nuestro esquema (tipo biosgrub y linux raid), y hacerlo en los dos discos de la misma forma hasta alcanzar la configuración mostrada en la ilustración. Posteriormente, seleccionamos Configurar RAID por software y subimos a la preparación de la siguiente capa.

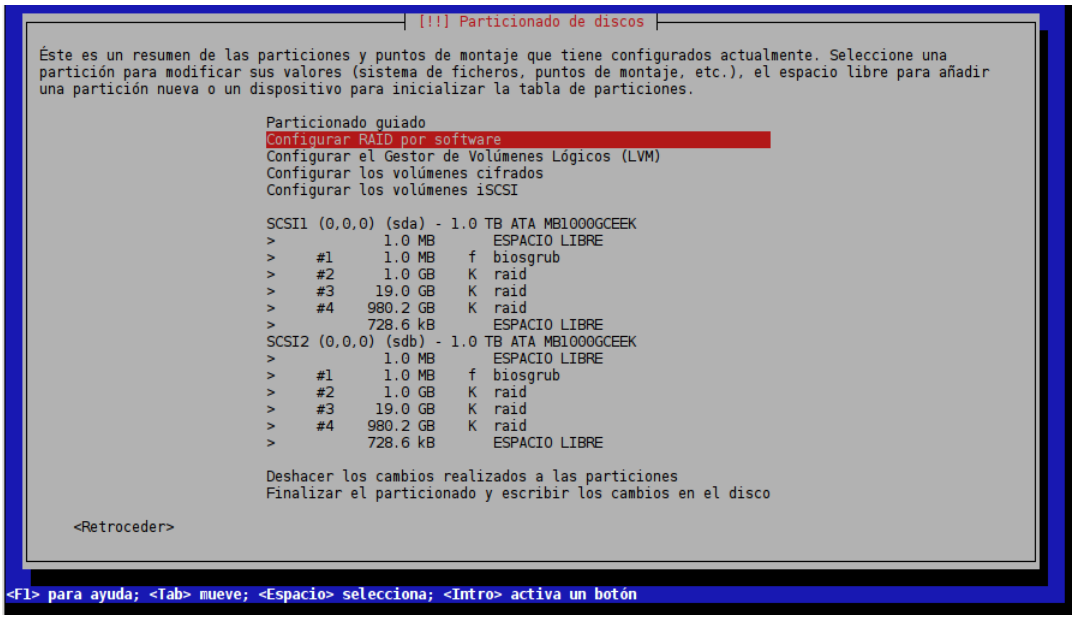


Ilustración 52: Particionado – nivel 1

Las capturas siguientes contienen los pasos a seguir para crear el dispositivo virtual md0. Para los espejos md1 y md2 repetiremos el mismo procedimiento, seleccionando sda3+sdb3 y sda4+sdb4 respectivamente.

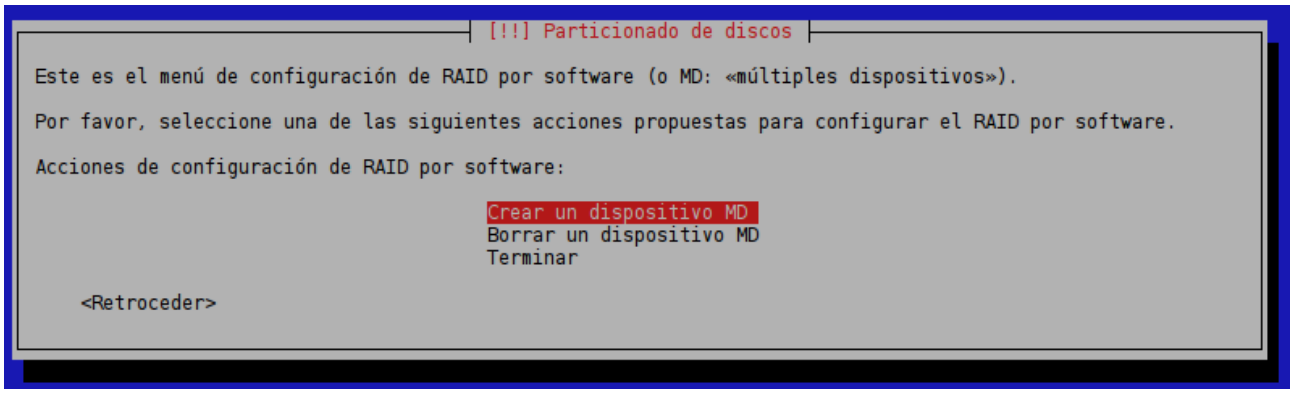


Ilustración 53: Particionado – creando md0 como RAID1

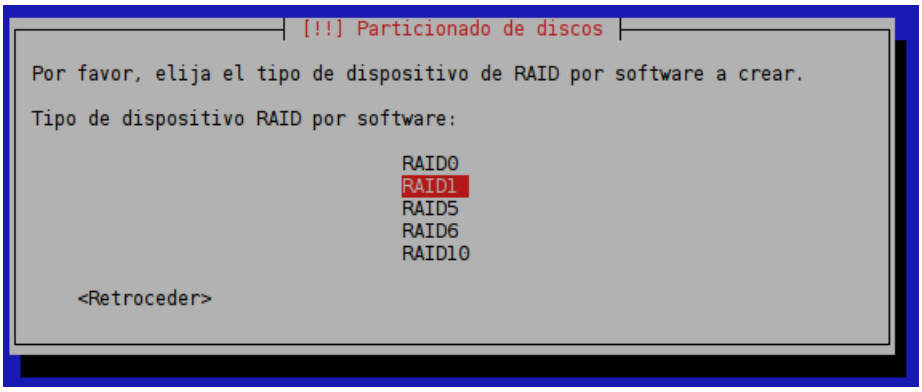


Ilustración 54: Particionado – creando md0 como RAID1

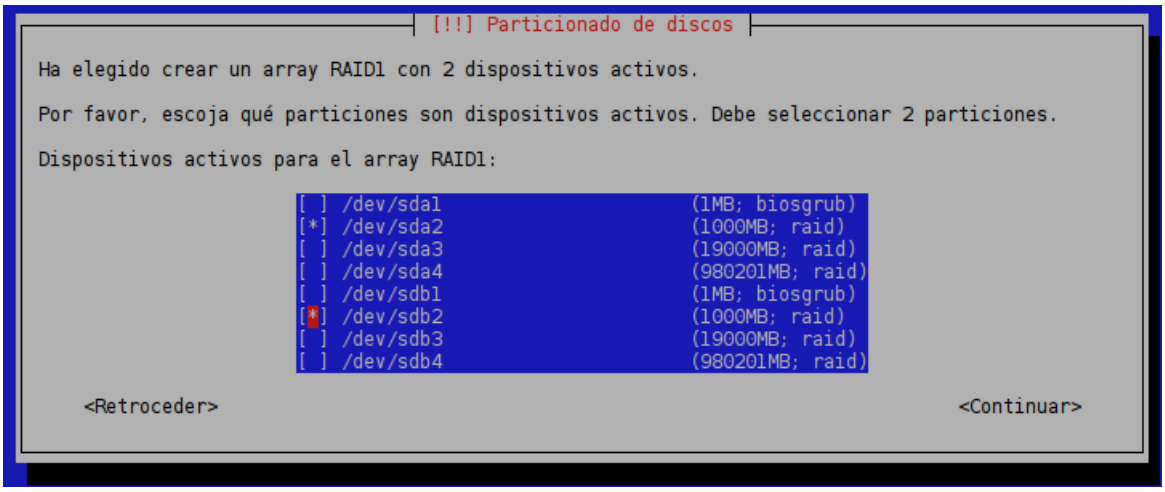


Ilustración 55: Particionado – creando md0 como RAID1

Una vez terminemos la configuración de RAID por software el mapa completo deberá mostrar los tres dispositivos md0,md1 y md2 disponibles para seguir trabajando.

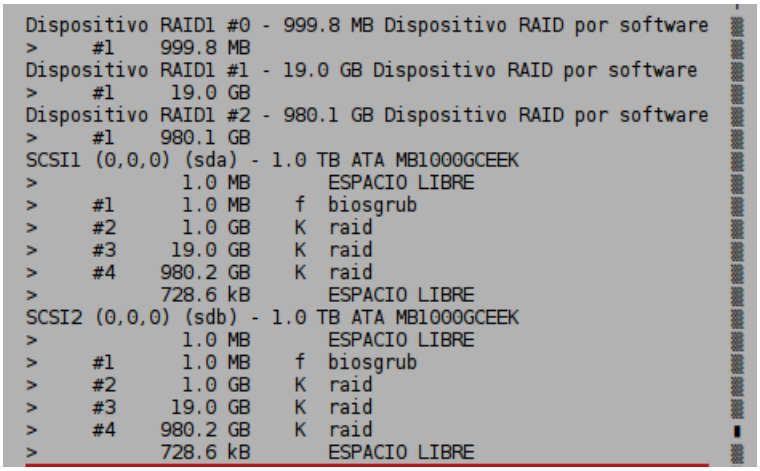


Ilustración 56: Particionado – nivel 2 – RAID completado

A continuación ascendemos un nivel más de nuestro diseño y pasamos a configurar LVM. En primer lugar hay que establecer el tipo de md1 y md2 a volumen físico para LVM. md0 contendrá el directorio /boot y no necesitamos LVM, por lo que formateamos directamente con ext4. Entrando en la opción "Configurar el Gestor de Volúmenes Lógicos (LVM)" empezamos la creación de los grupos de volúmenes y las particiones lógicas en LVM.

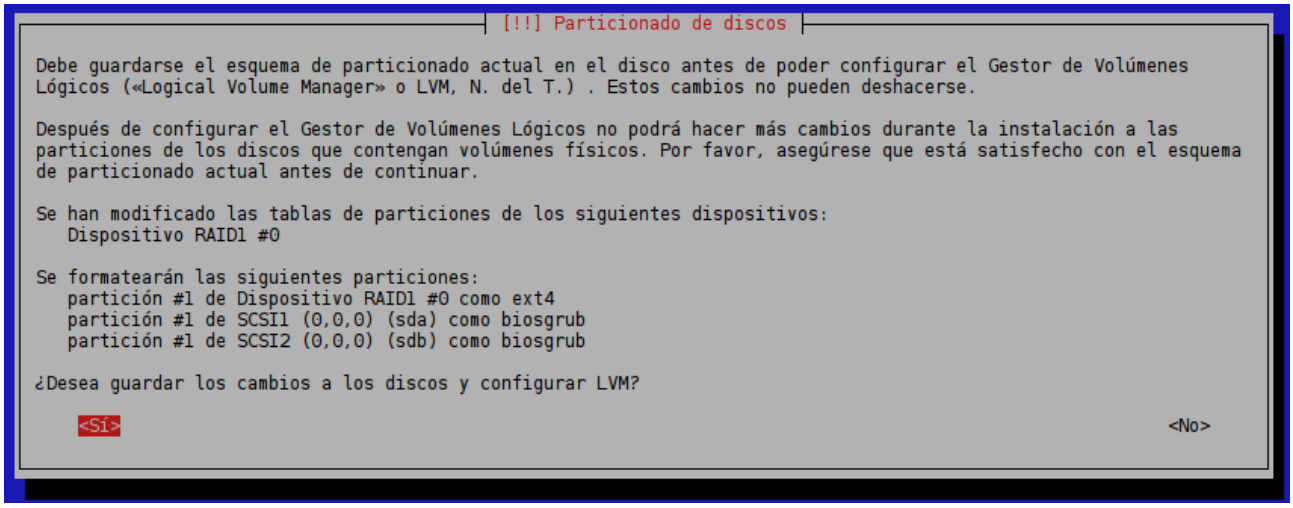


Ilustración 57: Particionado - inicio configuración LVM

Cuando terminemos de crear los grupos y las unidades lógicas planeadas con el asistente para LVM llegaremos al diseño final mostrado. Escribimos los cambios en el disco y continuamos con la instalación de Debian.

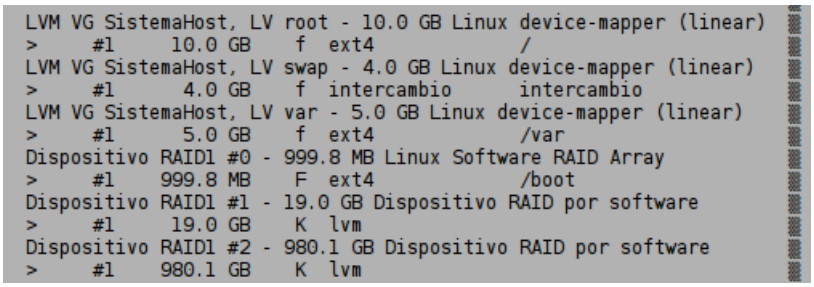


Ilustración 58: Particionado – esquema final

Podemos observar que el grupo LVM VirtDatos de 980GB queda sin usar inicialmente. Esto es porque en Proxmox 4 se ha migrado de OpenVZ a LXC como gestor de contenedores. LXC permite usar como almacenamiento directamente un volumen lógico LVM y no necesitamos crear una partición tipo ext4 donde ubicar el directorio raíz para los contenedores. En el caso de necesitarlo podríamos haber creado un volumen lógico de, por ejemplo, 500GB tipo ext4 y asignar el punto de montaje manual **/vz**.

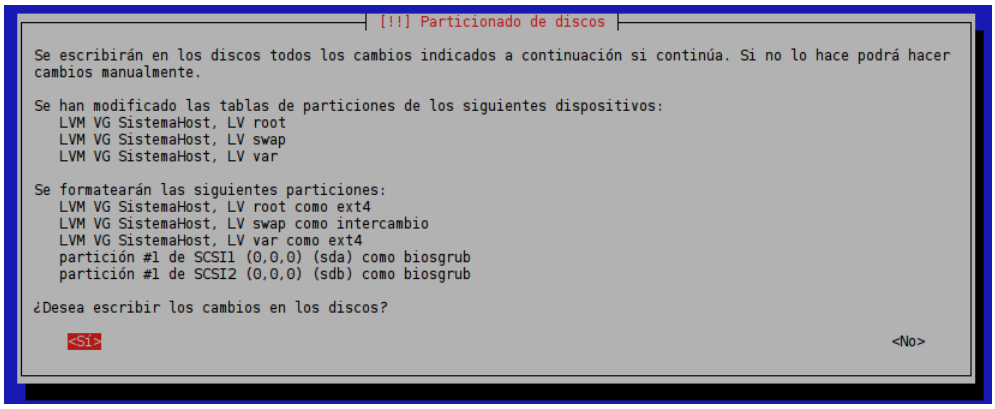


Ilustración 59: Particionado – guardar cambios

2.4.2. Instalación del sistema base

Una vez hemos configurado el esquema de particiones empezamos la instalación de los componentes de Debian GNU Linux. Las opciones a tener en cuenta son las siguientes:

Elección del núcleo: fijamos la versión linux-image-3.16.04 para evitar actualizaciones automáticas. Seleccionaremos un **initrd genérico** para facilitar un arranque exitoso si nos vemos obligados a cambiar el hardware del equipo en algún momento.

Como ya hemos dicho, nos apoyaremos en una réplica en red de Francia para descargar los paquetes actualizados. Activamos el software no libre para disponer de paquetes como java o el navegador chromium. Los necesitaremos para acceder a la interfaz web de Proxmox desde el propio servidor.

```
¿Usar software no libre? Sí
Actualizaciones de seguridad Sí
```

En la selección de los programas a instalar marcaremos únicamente:

```
SSH Server
Utilidades estándar del sistema
```

Por último el instalamos el cargador de arranque GRUB en el registro principal de arranque del primer disco duro. Más tarde lo haremos de forma manual también en el segundo.

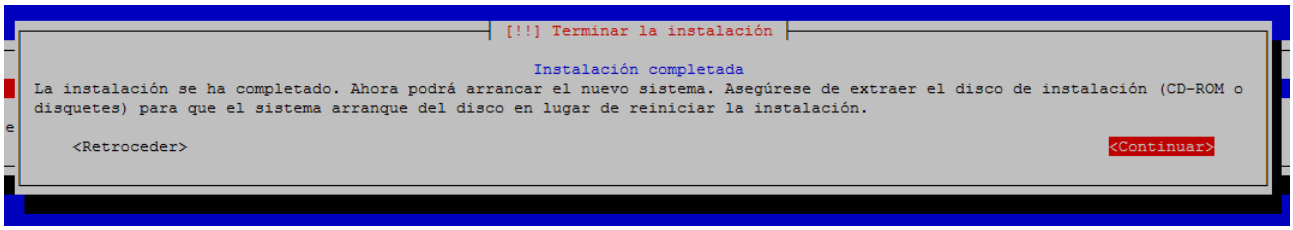


Ilustración 60: Final de la instalación de Debian GNU Linux

2.5. Instalación de Proxmox VE en Debian GNU Linux

Una vez que hemos completado la instalación de Debian y arrancado nuestro sistema nos toca configurarlo a nuestro gusto e instalar el entorno Proxmox VE. Para esta tarea, en lugar de ejecutar los comandos y hacer las configuraciones a mano, se ha optado por desarrollar una serie de scripts que nos guíen para hacerlo de forma semi-automática, y así acelerar el proceso si tenemos que preparar varios equipos. Estos scripts completos están disponibles en el Anexo A. A grandes rasgos, las operaciones que se realizan son las siguientes:

- Configuración de red: puente entre las interfaces, dirección IP estática, puerta de enlace, DNS.
- Personalización de nombre de host, mensajes de bienvenida (motd, issue.net) y otros.
- Configuración de repositorios para APT e instalación de software. Se añaden los repositorios de Proxmox VE y se instalan los paquetes necesarios.
- Copia de scripts auxiliares.
- Programación de tareas: informes de estado periódicos.
- Configuración de POSTFIX como agente de correo usando una cuenta de GMAIL para enviar informes.

A continuación veremos el proceso seguido para realizar la instalación utilizando ConfiguraProxmox.

2.5.1. Personalizar las opciones de ConfiguraProxmox

Al inicio del fichero se ha preparado una sección que permite personalizar la instalación para el lugar en que se vaya a implementar.

```
#############################################################################
## CONFIGURACIÓN
#############################################################################
# Nombre de empresa para identificar el lugar en informes y etc
NOMBRE_EMPRESA="EMPRESA"
# Versión de debian usada para la instalación y disponible en los repos de proxmox
# en dic de 2015 -> jessie
DEBIAN_VER=jessie
PROXMOX_VER=4.2.8-1
PROXMOX_KERNEL=4.2.8-1-pve
##########
# Nombre del servidor
NOMBRE_HOST="HServer01"
# Dominio, sufijo para nombrar el server: Hserver01.dominio
DOMINIO="dominio"
# Config de red
LAN_IFAZ=vmbr0
IP_ADDR=192.168.7.10
NETMASK=255.255.255.0
LAN_ADDR=192.168.7.0
LAN_CIDR=192.168.7.0/24
LAN_BCAST=192.168.7.255
GATEWAY=192.168.7.1
DNS_SERVER=192.168.7.1
##########
## Dirección a la que enviar un mail de prueba tras configurar POSTFIX
TEST_MAIL_RCV="carlona@informaticasa.es"
#############################################################################
```

La versión de Debian será la que elegimos cuando descargamos la ISO con el instalador. Para saber qué versión de Proxmox es la actual en el momento de la instalación recurrimos a:

http://pve.proxmox.com/wiki/Roadmap y http://pve.proxmox.com/wiki/Downloads

Tras las líneas: "*Reboot to activate the new Kernel, to check if you got all packages, run 'pveversion -v' and compare your output (all packages should have equal or higher version numbers): pve-server:~# pveversion -v"* Aparecen las versiones actuales de paquetes.

La configuración de los parámetros de red debemos hacerlos conforme a nuestro laboratorio de trabajo, y no al destino final donde se implementará el servidor, pues necesitamos mantener el acceso a internet durante la instalación. Una vez se haya finalizado toda la preparación ajustaremos las configuraciones al cliente.

2.5.2. Copia de los script de instalación al servidor

Una vez modificado el script debemos copiarlo al equipo para ejecutarlo. Para ello

Permitimos inicialmente el acceso SSH a root y, ya que tocamos la configuración del servicio SSH, cambiaremos el puerto de escucha al 1707 como medida de seguridad. Editamos **/etc/ssh/sshd_config**, las líneas *Port* y *PermitRootLogin*. Esta última la devolveremos a "no" cuando terminemos la instalación.

```
>$ nano /etc/ssh/sshd_config
```

```
# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 1707
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
…
# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes
…
```

Una vez editado reiniciamos el servicio SSH

```
>$ systemctl restart ssh.service
```

Y desde el puesto de trabajo copiamos el directorio con los ficheros de instalación al servidor

```
>$ scp -r -P1707 01_Host root@192.168.7.137:
```

```
The authenticity of host '[192.168.7.137]:1707 ([192.168.7.137]:1707)' can't
be established.
ECDSA key fingerprint is ed:a8:94:6e:c3:3a:20:a8:11:e8:fe:2a:b2:6a:05:59.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[192.168.7.137]:1707' (ECDSA) to the list of known
hosts.
root@192.168.7.137's password:
vzdumpHook.pl                      100% 1775     1.7KB/s   00:00
estadoHDDs                         100% 1965     1.9KB/s   00:00
getIPpub                           100%  233     0.2KB/s   00:00
backupWinVM                        100% 2115     2.1KB/s   00:00
informa                            100% 9125     8.9KB/s   00:00
ConfiguraProxmox                   100%   18KB  17.9KB/s   00:00
```

2.5.3. Instalación mediante ConfiguraProxmox (parte1)

La instalación del servidor se ha dividido en dos fases. Si ejecutamos el script ConfiguraProxmox sin argumentos se nos explica el porqué y cómo usarlo:

```
root@HServer01:~/01_Host# ./ConfiguraProxmox

 +++++ ConfiguraProxmox +++++
 + Uso: ConfiguraProxmox parte1 | parte2
 + parte1: primera fase de la configuración. Al final se debe reiniciar.
 + parte2: segunda fase. Se debe ejecutar una vez se complete la fase 1 y
 + se haya reiniciado el equipo con el kernel apropiado de proxmox.
 ++++++++++++++++++++++
```

Iniciemos la instalación. Recordemos que para todas estas tareas de configuración debemos trabajar como superusuario. En los fragmentos de código siguientes se han acortado

las listas de paquetes mostradas por APT y algunas líneas que no aportan información relevante para no extendernos de forma innecesaria.

```
usuario@HServer01:~$  >$ su -
root@HServer01:~# cd ~/01_Host
root@HServer01:~/01_Host# chmod +x ConfiguraProxmox
root@HServer01:~/01_Host# ./ConfiguraProxmox parte1
```

```
  ******************************************
  Autoconfiguración de HServer01 - PARTE 01.

  - Servidor de máquinas virtuales -
  Proxmox VE + Debian GNU Linux 64bits

  Los scripts personalizados deben estar en:
  ./Scripts
  ******************************************

Se van a realizar las primeras configuraciones básicas.
Pulsa ENTER para continuar
```

```
 ++++++++ Configuraciones iniciales ++++++++

+do_network: Configurando interfaces
Se instalarán los siguiente paquetes NUEVOS:
  bridge-utils
0 paquetes actualizados, 1 nuevos instalados, 0 para eliminar y 0 sin
actualizar.
Necesito descargar 32,8 kB de ficheros. Después de desempaquetar se usarán 146
kB.
Des: 1 http://ftp.fr.debian.org/debian/ jessie/main bridge-utils amd64 1.5-9
[32,8 kB]
Descargados 32,8 kB en 0s (108 kB/s)
Seleccionando el paquete bridge-utils previamente no seleccionado.
(Leyendo la base de datos ... 31446 ficheros o directorios instalados
actualmente.)
Preparando para desempaquetar .../bridge-utils_1.5-9_amd64.deb ...
Desempaquetando bridge-utils (1.5-9) ...
Procesando disparadores para man-db (2.7.0.2-5) ...
Configurando bridge-utils (1.5-9) ...

Waiting for vmbr0 to get ready (MAXWAIT is 2 seconds).
RTNETLINK answers: File exists
Failed to bring up vmbr0.
+do_network: hecho.

+do_nombres: configurando nombres de equipo...
+do_nombres: hecho.

+do_saludos: Configurando motd e issue.net...
+do_saludos: hecho.
+do_sourcesList: configurando APT...
+do_sourcesList: hecho.

+do_pcspkr: bloquea módulo pitido spkr...
+do_pcspkr: hecho.

+do_crontab: programando tareas...
+do_crontab: hecho.

Se van a instalar los scripts personalizados.
Pulsa ENTER para continuar

+ do_scripts: instala scripts personalizados...
 Se instalarán en: [ /usr/local/sbin ]

Copiando <backupWinVM>
Copiando <estadoHDDs>
Copiando <getIPpub>
Copiando <informa>
```

```
Copiando <vzdumpHook.pl>
+do_scripts: hecho.

Se van a instalar paquetes del sistema.
Pulsa ENTER para continuar

+do_instalasw: instalando paquetes...

+do_instalasw: PARTE 1
--2016-03-08 09:10:09--  http://download.proxmox.com/debian/key.asc
Resolviendo download.proxmox.com (download.proxmox.com)... 79.133.36.246
Conectando con download.proxmox.com (download.proxmox.com)
[79.133.36.246]:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 1703 (1,7K) [text/plain]
Grabando a: “STDOUT”

-                                   100%
[=======================================================>]   1,66K  --.-KB/s
en 0s

2016-03-08 09:10:09 (541 MB/s) - escritos a stdout [1703/1703]

OK
Ign http://download.proxmox.com jessie InRelease
… [líneas de descarga de índices]
Des: 19 http://ftp.de.debian.org jessie/non-free Translation-en [72,5 kB]
Descargados 12,3 MB en 11s (1.078 kB/s)

Estado actual: 11 actualizados [+11], 63 nuevos [+63].
Resolviendo las dependencias...
Se instalarán los siguiente paquetes NUEVOS:
  libnvpair1{a} libuutil1{a} libzfs2{a} libzpool2{a}
Se actualizarán los siguientes paquetes:
  dmeventd dmsetup grub-common grub-pc grub-pc-bin grub2-common iproute2
libdevmapper-event1.02.1 libdevmapper1.02.1
  liblvm2cmd2.02 lvm2
Se RECOMIENDAN los siguientes paquetes, pero NO se instalarán:
  libatm1
11 paquetes actualizados, 4 nuevos instalados, 0 para eliminar y 0 sin
actualizar.
Necesito descargar 6.039 kB de ficheros. Después de desempaquetar se liberarán
500 kB.
¿Quiere continuar? [Y/n/?]

…

Configurando grub2-common (2.02-pve4) ...
…
[líneas de configuración]
…
Procesando disparadores para initramfs-tools (0.120) ...

Estado actual: 0 actualizados [-11].
Se instalarán los siguiente paquetes NUEVOS:
  apparmor{a} … [ lista de paquetes ] … x11-common{a}
0 paquetes actualizados, 178 nuevos instalados, 0 para eliminar y 0 sin
actualizar.
Necesito descargar 116 MB de ficheros. Después de desempaquetar se usarán 152
MB.
No se satisfacen las dependencias de los siguientes paquetes:
 postfix : Entra en conflicto: mail-transport-agent que es un paquete virtual.
 exim4-daemon-light : Entra en conflicto: mail-transport-agent que es un
paquete virtual.
 exim4-config : Entra en conflicto: postfix pero se va a instalar 2.11.3-1.
Las acciones siguientes resolverán estas dependencias

     Eliminar los paquetes siguientes:
1)     exim4
2)     exim4-base
3)     exim4-config
4)     exim4-daemon-light

¿Acepta esta solución? [Y/n/q/?]Y
Se instalarán los siguiente paquetes NUEVOS:
  apparmor{a}[ … lista de paquetes … ] x11-common{a}
Se ELIMINARÁN los siguientes paquetes:
```

```
  exim4{a} exim4-base{a} exim4-config{a} exim4-daemon-light{a}
0 paquetes actualizados, 178 nuevos instalados, 4 para eliminar y 0 sin
actualizar.
Necesito descargar 116 MB de ficheros. Después de desempaquetar se usarán 148
MB.
¿Quiere continuar? [Y/n/?] Y

…

  ******************************************
  Ahora debes REINICIAR y arrancar con el kernel
  de proxmox:

  pve-kernel-4.2.8-1-pve

  Debes configurarlo por defecto en grub
  para no tener que cambiar a mano.

  Una vez se reinicie el equipo
  (comprueba que el kernel es correcto con uname -a)
  continúa con la configuración ejecutando este script
  para la parte 2:
  $>  ConfiguraProxmox parte2
  ******************************************
Pulsa ENTER para REINICIAR
```

2.5.4. Instalación mediante ConfiguraProxmox (parte2)

Tras el reinicio podemos acceder al servidor ya con la IP que le hayamos fijado en la configuración inicial:

```
>$ ssh usuario@192.168.7.10 -p 1707
```

Comprobamos que hemos cargado el kernel correcto:

```
root@HServer01:~# uname -a
Linux HServer01 4.2.8-1-pve #1 SMP Fri Feb 26 16:37:36 CET 2016 x86_64
GNU/Linux
```

Y seguimos con la segunda parte de la configuración:

```
root@HServer01:~# cd 01_Host
root@HServer01:~/01_Host# ./ConfiguraProxmox parte2
 ******************************************
  Autoconfiguración de HServer01 - PARTE 02.
  ******************************************

Se van a instalar paquetes del sistema.
Pulsa ENTER para continuar

+do_instalasw: instalando paquetes...

+do_instalasw: PARTE 2
Se instalarán los siguiente paquetes NUEVOS:
  htop ifstat libasn1-8-heimdal{a} libcurses-perl{a} libcurses-ui-perl{a}
libgssapi3-heimdal{a} libhcrypto4-heimdal{a}
  libheimbase1-heimdal{a} libheimntlm0-heimdal{a} libhx509-5-heimdal{a}
libkrb5-26-heimdal{a} libldb1{a} libntdb1{a}
  libperl5.20{a} libroken18-heimdal{a} libsensors4{a} libsmbclient{a} libsnmp-
base{a} libsnmp30{a} libtalloc2{a}
  libtdb1{a} libterm-readkey-perl{a} libtevent0{a} libwbclient0{a} libwind0-
heimdal{a} lshw lsscsi ntfs-3g
  python-crypto{a} python-ldb{a} python-ntdb{a} python-samba{a} python-
talloc{a} python-tdb{a} samba-common{a}
  samba-common-bin{a} samba-libs{a} smbclient sysv-rc-conf vim vim-runtime{a}
0 paquetes actualizados, 41 nuevos instalados, 0 para eliminar y 0 sin
actualizar.
```

```
Necesito descargar 19,3 MB de ficheros. Después de desempaquetar se usarán
77,1 MB.
¿Quiere continuar? [Y/n/?]
…
Se instalarán los siguiente paquetes NUEVOS:
  hddtemp libdate-manip-perl{a} libsys-cpu-perl{a} lm-sensors logwatch
smartmontools
0 paquetes actualizados, 6 nuevos instalados, 0 para eliminar y 0 sin
actualizar.
Necesito descargar 1.943 kB de ficheros. Después de desempaquetar se usarán
15,3 MB.
¿Quiere continuar? [Y/n/?]
Des: 1 http://ftp.de.debian.org/debian/ jessie/main libdate-manip-perl all
6.47-1 [923 kB]
Des: 2 http://ftp.de.debian.org/debian/ jessie/main libsys-cpu-perl amd64
0.61-1+b1 [10,3 kB]
…
Descargados 1.943 kB en 3s (512 kB/s)
Preconfigurando paquetes ...
…
Configurando logwatch (7.4.1-2) ...
Configurando smartmontools (6.3+svn4002-2+b2) ...
update-rc.d: warning: start and stop actions are no longer supported; falling
back to defaults
Configurando hddtemp (0.3-beta15-52) ...
Configurando lm-sensors (1:3.3.5-2) ...
Procesando disparadores para systemd (215-17+deb8u3) ...
Generating locales (this might take a while)…
...
Generation complete.

Current default time zone: 'Europe/Madrid'
Local time is now:      Tue Mar  8 09:55:21 CET 2016.
Universal Time is now:  Tue Mar  8 08:55:21 UTC 2016.

Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
+do_instalasw: hecho.

 ++++++++++++++++++++++++++++++++++++++++++++++++++
 + A continuación Activa/Desactiva los servicios que
 + quieres que arranquen automáticamente...
 ++++++++++++++++++++++++++++++++++++++++++++++++++
 + Pulsa ENTER para continuar


Se va a configurar POSTFIX.
Pulsa ENTER para continuar

+do_postfix:
 ++++++++++++++++++++++++++++++++++++++++++++++++++
 + Configuración del servidor de correo POSTFIX...
 + En el asistente siguiente deja todo por defecto excepto
 + el tipo de puesto. Debes escoger: SITIO DE INTERNET
 ++++++++++++++++++++++++++++++++++++++++++++++++++
 + Pulsa ENTER para continuar

setting synchronous mail queue updates: false
...
After modifying main.cf, be sure to run '/etc/init.d/postfix reload'.

Running newaliases
Procesando disparadores para libc-bin (2.19-18+deb8u3) ...
[ ok ] Reloading postfix configuration (via systemctl): postfix.service.
+do_instalasw: hecho.

 ++++++++++++++++++++++++++++++++++++++++++++++++++
 A continuación se muestran unas instrucciones con varios
 pasos recomendados para realizar a mano.
 ++++++++++++++++++++++++++++++++++++++++++++++++++
Pulsa ENTER para continuar
```

Si la configuración de POSTFIX ha tenido éxito recibiremos un correo electrónico en la dirección de prueba que hayamos configurado:

```
Test mail from postfix on HServer01.dominio
```

2.5.5. Algunas configuraciones manuales

Antes de finalizar la ejecución, ConfiguraProxmox nos muestra unas instrucciones con sugerencias de opciones que podemos configurar a mano.

- Instalar un entorno gráfico mínimo para permitir la administración de Proxmox (via web) desde el propio equipo si disponemos de monitor, teclado y ratón.

```
root@HServer01:~# aptitude install --without-recommends xorg xserver-xorg-video-mga fluxbox tightvncserver eterm feh
root@HServer01:~# aptitude install --without-recommends gsmartcontrol smartmontools iceweasel iceweasel-l10n-es-es openjdk-7-jre icedtea-7-plugin
```

Donde *xserver-xorg-video-mga* es el driver de la tarjeta gráfica y deberemos elegirlo en función del hardware del equipo. Mediante el comando ***lspci*** podemos verificar cual es nuestro dispositivo.

- Asegurarnos de que GRUB está correctamente configurado y por defecto selecciona el kernel de Proxmox. Desde Proxmox 4 esto se hace automáticamente y no es necesario modificar nada. Sí es importante el paso de instalar el gestor en ambos discos duros. Para versiones anteriores los cambios recomendados eran los siguientes:

```
root@HServer01:~# nano /etc/default/grub
```

Desactivar los SUBMENUS para que se vean todos los kernels:

```
GRUB_DISABLE_SUBMENU=y
```

Establecer el kernel de proxmox por defecto (normalmente es el número 2):

```
GRUB_DEFAULT=2
```

Actualizar la configuración con los cambios realizados e instalarlo en ambos HDDs para asegurar el arranque del sistema aunque falle uno de ellos:

```
root@HServer01:~# update-grub
root@HServer01:~# grub-install /dev/sda
root@HServer01:~# grub-install /dev/sdb
```

- Configuración de los sensores para monitorizar correctamente las temperaturas del equipo. Ejecutamos ***sensors-detect*** y permitimos que añada los módulos necesarios.

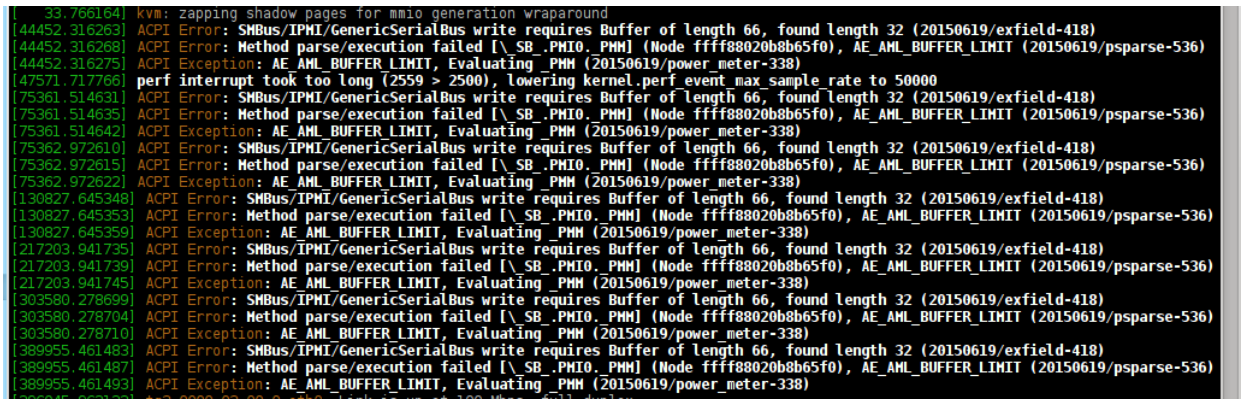


Ilustración 61: Error ACPI en la lectura de sensores

Después deberemos editar el fichero sensors3.conf para eliminar un molesto error que se repite en el registro del sistema:

Añadiendo las siguientes líneas, lo conseguimos resolver[48]:

```
root@HServer01:~# nano /etc/sensors3.conf
```

```
chip "power_meter-acpi-0"
    ignore power1
```

```
root@HServer01:~# root@HServer01:~# sensors
```

```
acpitz-virtual-0
Adapter: Virtual device
temp1:         +8.3°C  (crit = +31.3°C)

coretemp-isa-0000
Adapter: ISA adapter
Physical id 0:  +27.0°C  (high = +80.0°C, crit = +100.0°C)
Core 0:         +26.0°C  (high = +80.0°C, crit = +100.0°C)
Core 1:         +27.0°C  (high = +80.0°C, crit = +100.0°C)
Core 2:         +24.0°C  (high = +80.0°C, crit = +100.0°C)
Core 3:         +26.0°C  (high = +80.0°C, crit = +100.0°C)

power_meter-acpi-0
Adapter: ACPI interface
```

- Configurar las alertas por email de **mdadm.** De esta forma recibiremos un aviso si surge algún problema en el funcionamiento del RAID.

```
root@HServer01:~# nano /etc/mdadm/mdadm.conf
```

Y establecemos:

```
  # instruct the monitoring daemon where to send mail alerts
  MAILADDR monitor@informaticasa.es
```

- Configurar las alertas por email de ***smartmontools.*** De esta forma recibiremos avisos si surgen problemas en los discos duros gracias a la tecnología SMART.

Para activar el demonio lo hacemos en:

48 Fuente de la solución: http://www.nexusco.net/frequent-acpi-errors-starting-smbus-ipmi-write-requires-buffer-length-42/

```
root@HServer01:~# nano /etc/default/smartmontools
```

Y fijamos la dirección donde queramos recibir la alertas editando smartd.conf en la siguiente línea:

```
root@HServer01:~# nano /etc/smartd.conf
  DEVICESCAN -d removable -n standby -m monitor@informaticasa.es -M exec
/usr/share/smartmontools/smartd-runner
```

- Como estamos trabajando con la versión sin subscripción de Proxmox VE, cada vez que entramos en la interfaz web nos aparece un mensaje de aviso. Si nos resulta molesto y queremos desactivarlo se puede hacer del siguiente modo:

Editar el siguiente fichero alrededor de la línea 482:

```
root@HServer01:~# nano +482 /usr/share/pve-manager/ext4/pvemanagerlib.js
```

Buscamos:

```
  if (data.status !== 'Active') {
```

Y lo modificamos de esta forma:

```
  if (false) {
```

- Configuración del servidor SSH. Ya hemos cambiado el puerto de escucha, pero debemos desactivar el acceso al usuario root:

```
root@HServer01:~# nano /etc/ssh/sshd_config
   PermitRootLogin no
```

- Ajustar /etc/crontab para programar los informes o copias de seguridad con la periodicidad deseada.

Tras el mensaje para las configuraciones manuales, ConfiguraProxmox nos muestra el siguiente texto y finaliza:

```
*******************************************
FIN. La configuración inicial del servidor
se ha completado. Ahora puedes empezar a trabajar
con las máquinas virtuales accediendo a:
https://192.168.7.10:8006
mediante la interfaz web.
*******************************************
+ Pulsa ENTER para continuar
```

Hemos terminado de preparar nuestro anfitrión. El siguiente paso es empezar a trabajar en la interfaz web, creando los espacios de almacenamiento y nuestros servidores virtuales. Como nos dice el mensaje mostrado, accederemos mediante: https://192.168.7.10:8006.

2.6. Configuración de Proxmox VE.

La interfaz web del sistema que acabamos de instalar tiene el aspecto mostrado en la siguiente imagen. Como vemos, no aparece aún ninguna máquina virtual ni almacenamiento diferente del local (que corresponde a la partición root del sistema). El primer paso que deberemos dar es agregar los espacios de almacenamiento para los servidores virtuales que hemos planeado en la fase de particionado.

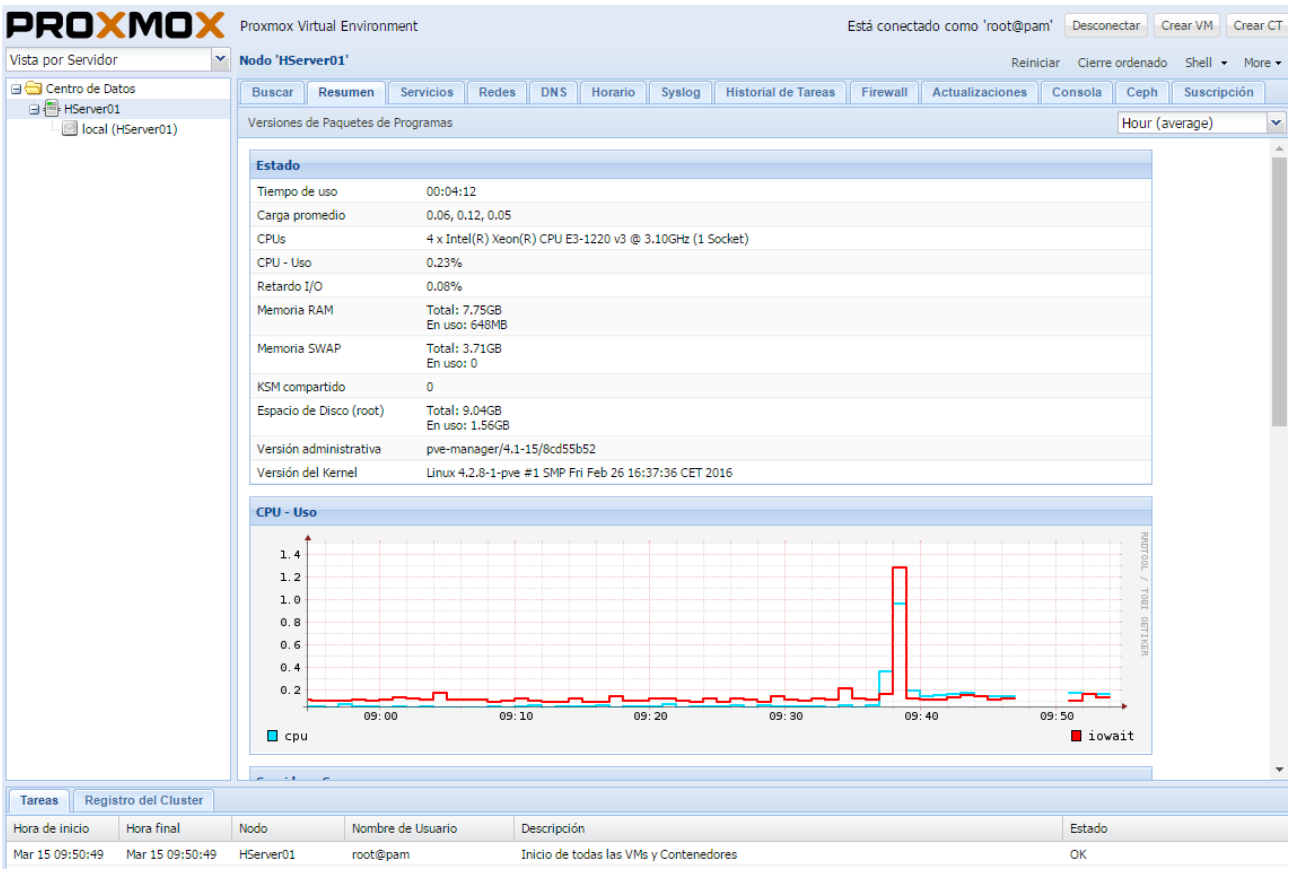


Ilustración 62: Interfaz web de Proxmox recién instalado.

2.6.1. Activando los espacios de almacenamiento.

Empezaremos creando 2 almacenamientos, serán los siguientes:

- Almacenamiento tipo directorio a partir de una partición lógica que crearemos sobre el grupo VirtDatos. Este almacenamiento lo usaremos para almacenar imágenes, ISOs, plantillas de contenedores y etc. Es necesario para no llenar los 10GB de la raíz. Veamos primero como crear el volumen LVM mediante la interfaz de comandos.

Recordemos la configuración de LVM existente. El comando vgdisplay nos muestra los grupos de volúmenes disponibles en nuestro sistema.

```
root@HServer01:~# vgdisplay
  --- Volume group ---
  VG Name               VirtDatos
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               912,76 GiB
  PE Size               4,00 MiB
  Total PE              233666
  Alloc PE / Size       0 / 0
  Free  PE / Size       233666 / 912,76 GiB
  VG UUID               VpAtNP-tp03-eydc-m5Pr-WV3D-q627-6jPCXc

  --- Volume group ---
  VG Name               SistemaHost
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  4
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                3
  Open LV               3
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               17,68 GiB
  PE Size               4,00 MiB
  Total PE              4525
  Alloc PE / Size       4525 / 17,68 GiB
  Free  PE / Size       0 / 0
  VG UUID               lhI0JA-JjSk-ISv7-PrWz-oUvH-Uz0v-O40wnj
```

lvdisplay es el equivalente pero para volúmenes lógicos. Aparecen los tres que creamos en la fase de instalación de Debian.

```
root@HServer01:~# lvdisplay
  --- Logical volume ---
  LV Path                /dev/SistemaHost/root
  LV Name                root
  VG Name                SistemaHost
  LV UUID                2mVYR0-V9Lm-Re1d-paR2-0N3E-sCSt-3mZGDL
  LV Write Access        read/write
  LV Creation host, time HServer01, 2016-03-01 14:41:03 +0100
  LV Status              available
  # open                 1
  LV Size                9,31 GiB
  Current LE             2384
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:0

  --- Logical volume ---
  LV Path                /dev/SistemaHost/var
  LV Name                var
  VG Name                SistemaHost
  LV UUID                cltkOl-2ERd-tE2w-dt9A-UNNh-buNd-upaw5G
  LV Write Access        read/write
  LV Creation host, time HServer01, 2016-03-01 14:41:35 +0100
  LV Status              available
```

```
  # open                 1
  LV Size                4,66 GiB
  Current LE             1192
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:2

  --- Logical volume ---
  LV Path                /dev/SistemaHost/swap
  LV Name                swap
  VG Name                SistemaHost
  LV UUID                NxoNw3-kLU3-gs5Y-Ybs6-kcJl-DItf-hQVt0E
  LV Write Access        read/write
  LV Creation host, time HServer01, 2016-03-01 14:41:45 +0100
  LV Status              available
  # open                 2
  LV Size                3,71 GiB
  Current LE             949
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:1
```

Queremos añadir un nuevo volumen lógico para nuestra partición de almacenamiento local. El comando que necesitamos y su sintaxis es la siguiente:

```
# lvcreate -L <tamaño> <grupo_de_volúmenes> -n <volumen_lógico>
```

En nuestro caso le asignaremos un tamaño de 10GB y lo llamaremos ***local***.

```
root@HServer01:~# lvcreate -L 10G VirtDatos -n local
  Logical volume "local" created.
```

```
root@HServer01:~# lvdisplay
...
  --- Logical volume ---
  LV Path                /dev/VirtDatos/local
  LV Name                local
  VG Name                VirtDatos
  LV UUID                2zMQof-a1Zl-kvyO-iU1j-RQM9-Pbew-UFaHjR
  LV Write Access        read/write
  LV Creation host, time HServer01, 2016-03-15 11:53:37 +0100
  LV Status              available
  # open                 0
  LV Size                10,00 GiB
  Current LE             2560
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:3
```

A continuación formateamos el sistema de ficheros y creamos el punto de montaje:

```
root@HServer01:~# mkfs.ext4 /dev/mapper/VirtDatos-local
mke2fs 1.42.12 (29-Aug-2014)
Se está creando El sistema de ficheros con 2621440 4k bloques y 655360 nodos-i

UUID del sistema de ficheros: 4af7b0c5-0fa7-45ec-85fb-2fa2a5995342
Respaldo del superbloque guardado en los bloques:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (32768 bloques): hecho
```

```
Escribiendo superbloques y la información contable del sistema de ficheros:
hecho
```

```
root@HServer01:~# mkdir /local
root@HServer01:~# mount /dev/mapper/VirtDatos-local /local
root@HServer01:~# mount
```

```
...
/dev/mapper/VirtDatos-local on /local type ext4 (rw,relatime,data=ordered)
…
```

Y por último hacemos el montaje permanente añadiendo la línea correspondiente en **/etc/fstab**

```
root@HServer01:~# nano /etc/fstab
```

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system>                 <mount point>   <type>  <options>
<dump>  <pass>
## RAIZ
/dev/mapper/SistemaHost-root    /               ext4    errors=remount-ro
0       1
## BOOT /boot was on /dev/md0 during installation
UUID=db6c70af-01bb-42a7-b19f-a755e0c6b500 /boot ext4    defaults
0       2
## VAR
/dev/mapper/SistemaHost-var /var            ext4    defaults
0       2
## LOCAL
/dev/mapper/VirtDatos-local /local      ext4    defaults
0       2
## SWAP
/dev/mapper/SistemaHost-swap none            swap    sw
0       0
## CDROM
/dev/sr0        /media/cdrom0   udf,iso9660 user,noauto     0       0
```

Una vez montada nuestra partición, ya la podemos activar en Proxmox. Añadimos seleccionando tipo directorio y elegimos qué contenido guardará: plantillas e imágenes ISO.

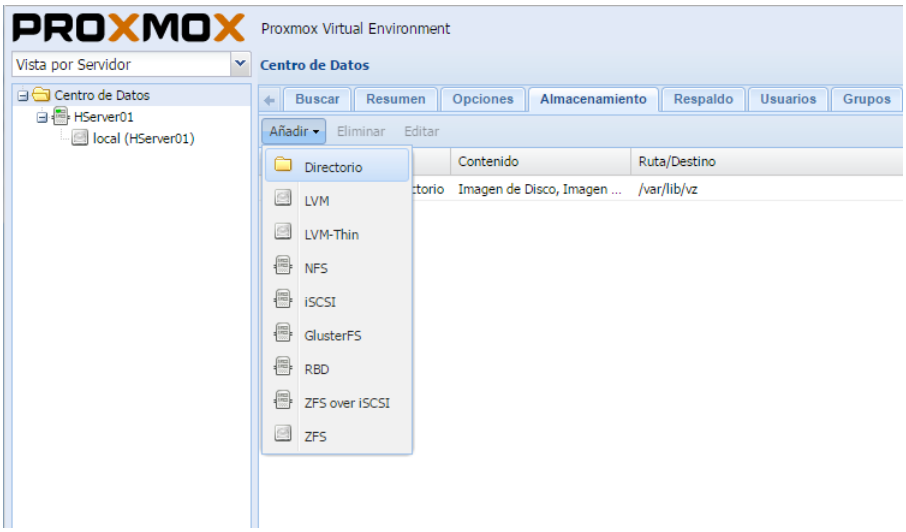


Ilustración 63: Añadiendo almacenamiento tipo directorio 1

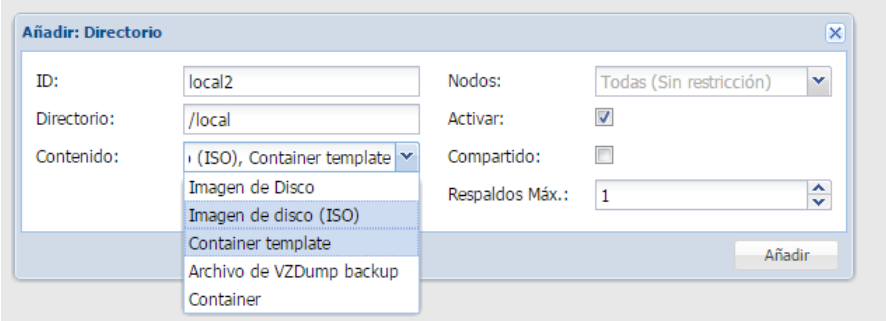


Ilustración 64: Añadiendo almacenamiento tipo directorio 2

- El segundo espacio de almacenamiento que debemos activar es para las máquinas virtuales. Recordemos que era tipo LVM a partir del grupo de volúmenes VirtDatos. Contendrá por tanto imágenes de disco (para KVM) y containers (LXC).

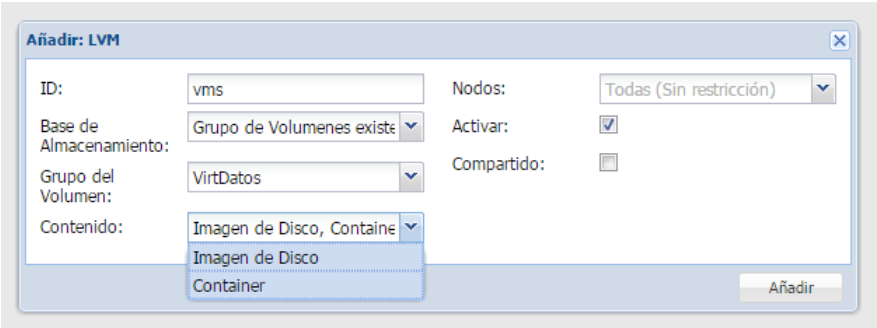


Ilustración 65: Añadiendo almacenamiento tipo LVM

Una vez hecho esto, la interfaz mostrará las tres opciones de espacio de almacenamiento y habremos terminado de configurar nuestro anfitrión Proxmox. El siguiente paso será levantar las máquinas virtuales para proveer los servicios deseados. Lo veremos en el siguiente capítulo.

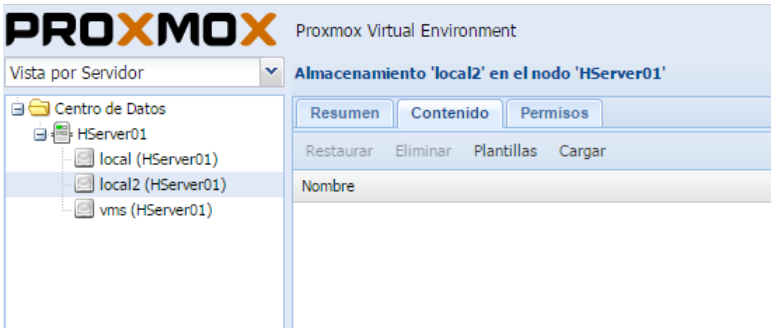


Ilustración 66: Interfaz web – almacenamientos creados

Capítulo V: Creación de servidores virtuales

1. dataserver: contenedor LXC - Debian

La primera máquina virtual que crearemos será la que hemos llamado ***dataserver***. Tipo contenedor LXC y con sistema operativo Debian GNU Linux. Le asignaremos un núcleo como procesador (esto es realmente un límite superior, no una reserva), un disco duro de 600GB y 2GB de RAM. Su propósito principal es ser un almacén seguro para los ficheros de los usuarios y facilitar el trabajo compartido en red. Se ha optado por utilizar LXC y Linux por el ahorro de recursos de LXC frente a KVM, la flexibilidad y potencia para configurar servicios SAMBA, FTP, NFS y la menor vulnerabilidad ante amenazas de malware o virus de Linux sobre los sistemas Windows.

1.1. Descarga de la plantilla

Aprovecharemos el repositorio de máquinas virtuales prefabricadas que ofrece Proxmox VE y nos ahorraremos así la instalación desde cero del sistema, que por otro lado ya hemos visto como hacerla para el anfitrión. En la web https://pve.proxmox.com/wiki/Debian_8.0_Standard podemos consultar los detalles de la plantilla que utilizaremos. Se trata simplemente de una preinstalación estándar de Debian Jessie. Para obtenerla lo haremos mediante la interfaz web de Proxmox, seleccionando el almacenamiento en que la vayamos a descargar (local2 en nuestro caso) y accediendo al listado de plantillas disponibles en la pestaña de contenido:

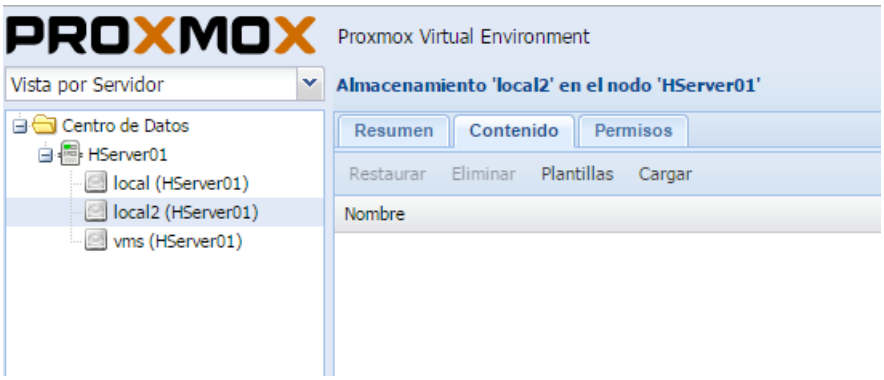


Ilustración 67: Descargando plantilla Debian 8 – almacenamiento

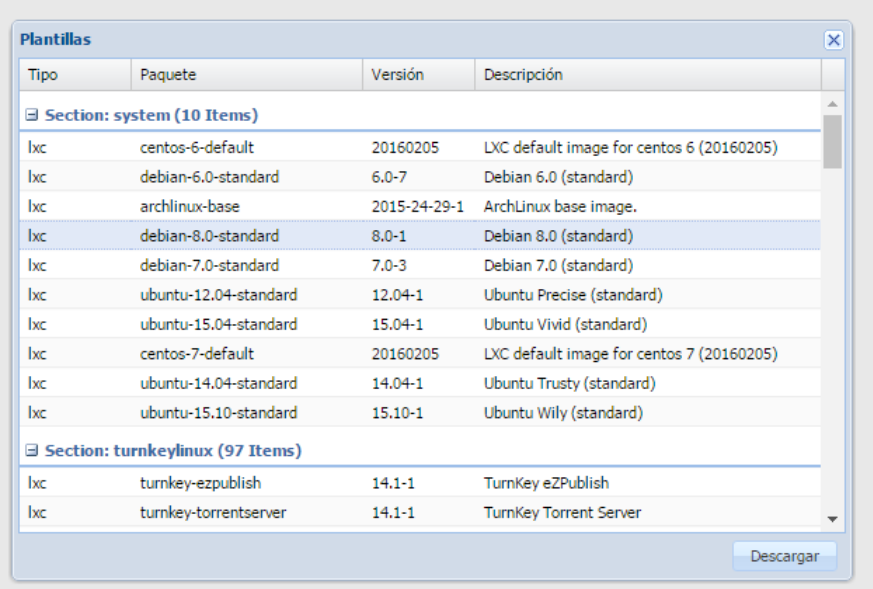


Ilustración 68: Descargando plantilla Debian 8 – listado plantillas

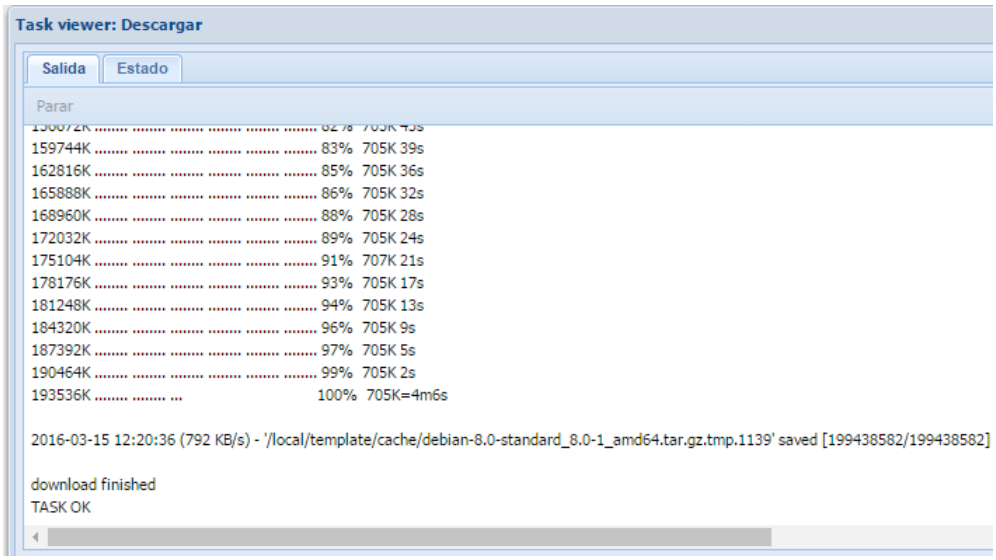


Ilustración 69: Descargando plantilla Debian 8 – progreso

Una vez se ha completado el proceso, en el contenido del almacenamiento local2 debemos disponer del fichero comprimido con la plantilla descargada. A partir de aquí la podremos utilizar para crear contenedores.

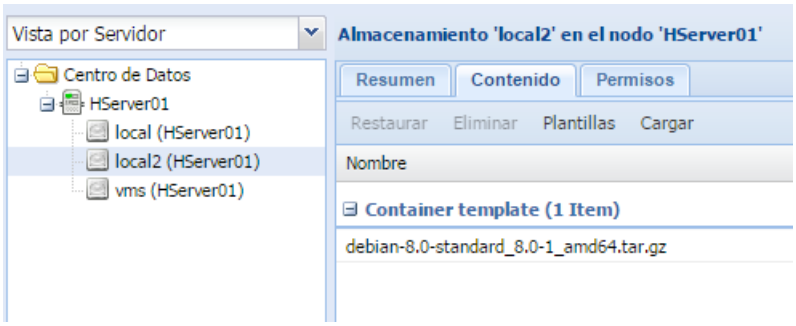


Ilustración 70: Descargando plantilla Debian 8 – finalizado

1.2. Preparación del contenedor

En la esquina superior derecha de la interfaz web podemos observar los botones para iniciar el asistente de creación de contenedores y máquinas virtuales.

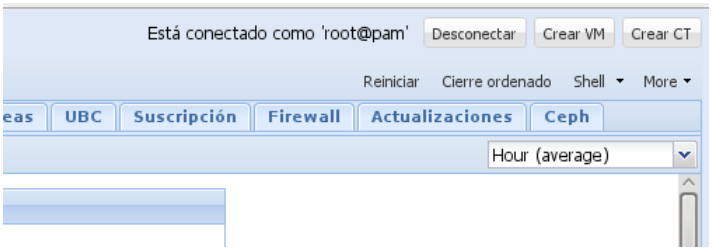


Ilustración 71 Interfaz Proxmox: Crear VM / Crear CT

Para crear nuestro contenedor deberemos definir los siguientes parámetros. Se han destacado los que debemos tener preparados y configurar a nuestro criterio:

- General
  - Nodo: En nuestro caso solo existe HServer01, no se trata de un cluster con varios nodos.
  - VM ID: identificador numérico para la máquina virtual.
  - **Nombre del host**: dataserver.
  - Conjunto de recursos: este parámetro se utiliza para agrupar y organizar las máquinas virtuales y los almacenamientos en grupos, como por ejemplo: *Producción* y *Pruebas*. Es útil en un entorno con más unidades creadas.
  - **Contraseña** / Confirmar contraseña: clave de root para el contenedor.

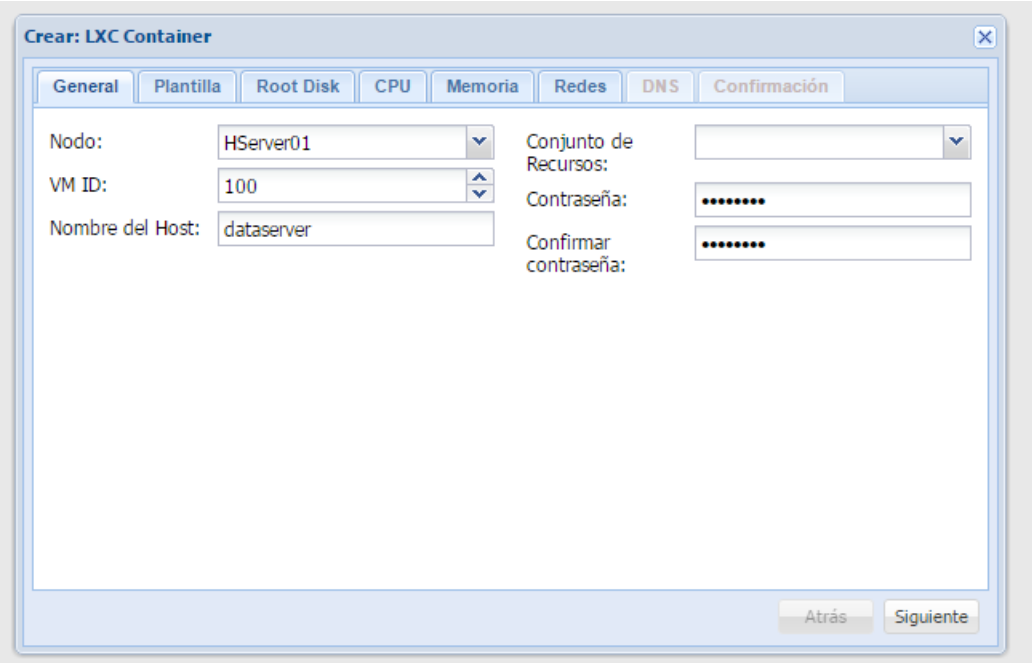


Ilustración 72: Creación de un contenedor LXC – general

- Plantilla
  - Almacenamiento: ubicación con plantillas descargadas, local2 en nuestro caso.
  - Plantilla: seleccionamos la deseada.

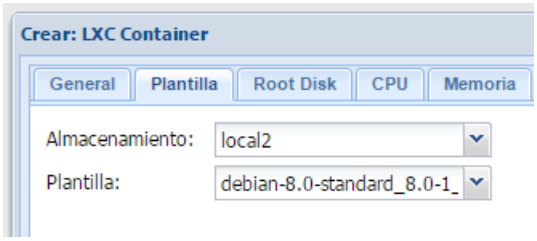


Ilustración 73: Creación de un contenedor LXC – plantilla

- Disco raíz
  - Almacenamiento: en qué conjunto de almacenamiento ubicaremos el disco virtual de la máquina. Habíamos preparado el grupo LVM **vms** para esta tarea.
  - Tamaño de disco: 600GB.

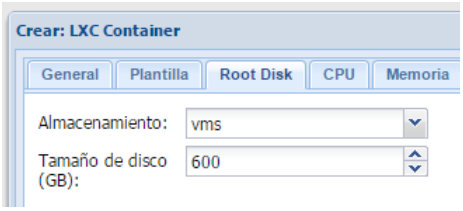


Ilustración 74: Creación de un contenedor LXC – disco

- CPU [49]
  - **CPU limit**: este parámetro configura el porcentaje máximo de proceso que permitimos usar al contenedor. Si nuestro servidor dispone de 4 núcleos, el máximo disponible será 4. Es importante comprender que no estamos confinando la ejecución en unos núcleos determinados, sino limitando el porcentaje de uso de todos los disponibles. Si, como en nuestro caso, fijamos el parámetro en 1 núcleo, lo que estamos haciendo realmente es limitar el uso al 25% de los 4 disponibles. De hecho, el contenedor ve el procesador al completo:

```
usuario@dataserver:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 60
model name      : Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
stepping        : 3
microcode       : 0x17
cpu MHz         : 3100.000
…
processor       : 1
vendor_id       : GenuineIntel
```

49 https://openvz.org/Resource_shortage

```
…
processor       : 2
vendor_id       : GenuineIntel
cpu family      : 6
model           : 60
model name      : Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
…
processor       : 3
vendor_id       : GenuineIntel
cpu family      : 6
model           : 60
model name      : Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
```

- Utilización de CPU (CPU units): asignación de prioridad para repartir el tiempo de proceso entre las máquinas virtuales. Esto se hace mediante pesos relativos. Veamos un ejemplo para entenderlo mejor. Pongamos que desplegamos tres máquinas. Si a la primera asignamos un peso de 1000 y a las otras 500, sabemos que (siempre dentro del límite máximo fijado en el parámetro anterior para cada una) la primera tiene doble prioridad para ejecutar sus tareas. Si están trabajando a pleno rendimiento, la de peso 1000 ocupará la CPU durante un 50% del tiempo, y las otras un 25% cada una. Para un reparto equitativo configuraremos el mismo valor a todas nuestras máquinas. Lo dejaremos en 1024 tal como viene por defecto.

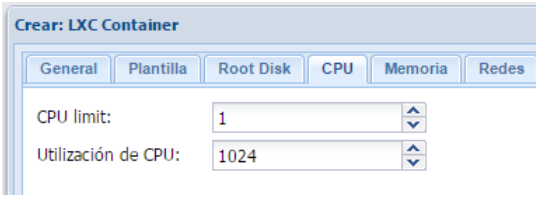


Ilustración 75: Creación de un contenedor LXC – CPU

- Memoria
  - RAM: tal como habíamos planificado, limitamos a un máximo de 2GB.
  - SWAP: apenas se usa en los equipos con suficiente RAM, pero asignaremos 512MB por si en algún instante fuera necesaria.

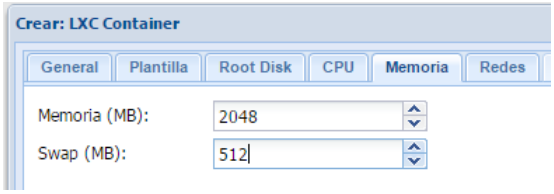


Ilustración 76: Creación de un contenedor LXC – memoria

- Redes

- ID: identificador interno para Proxmox VE.
- Nombre: nombre que recibirá la interfaz dentro del container. Se suelen usar eth0, eth1, eth2…

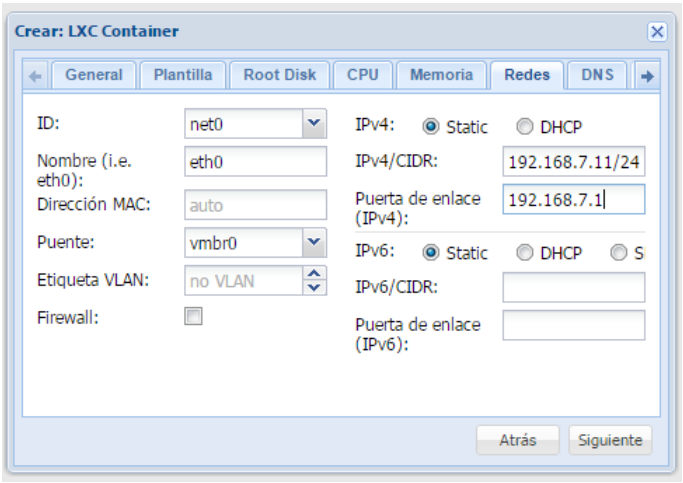


Ilustración 77: Creación de un contenedor LXC – redes

- Puente: nombre del puente virtual que creamos en el anfitrión para enganchar las máquinas virtuales. Recordemos que habíamos planteado un modelo de red conmutada, donde el servidor anfitrión y los virtuales estaban en la misma subred, conectados al mismo switch virtual. Configuraremos también el direccionamiento IP. Las VLAN, IPv6 o el firewall no los usaremos en nuestra instalación.

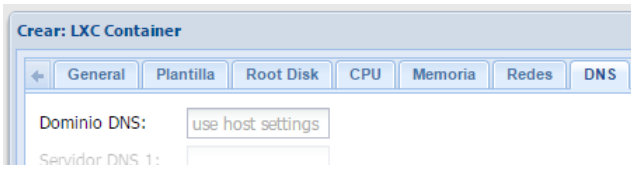


Ilustración 78: Creación de un contenedor LXC – DNS

- La última pestaña muestra un resumen con la configuración realizada. Si está todo correcto pulsamos en finalizado y se generará el contenedor.

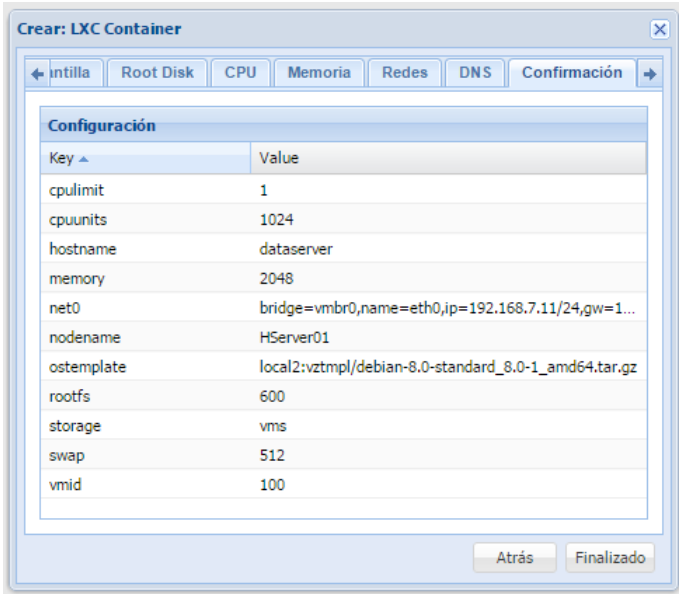


Ilustración 79: Creación de un contenedor LXC – resumen final

En la pestaña principal podemos ver nuestro nuevo contenedor. El icono aparece en gris porque actualmente está detenido. El almacenamiento vms deberá contener la imagen del disco. Si listamos los volúmenes lógicos mediante el comando lvdisplay en la terminal de comandos observaremos sus características detalladas.

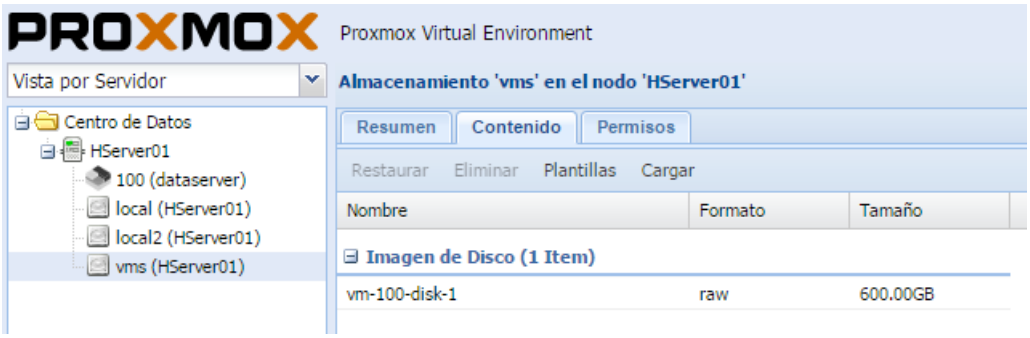


Ilustración 80: Creación de un contenedor LXC – dataserver terminado

```
root@HServer01:~# lvdisplay
```

```
  …
  --- Logical volume ---
  LV Path                /dev/VirtDatos/vm-100-disk-1
  LV Name                vm-100-disk-1
  VG Name                VirtDatos
  LV UUID                POFnw7-j5WT-e65v-ukHw-CMH9-3gt9-qh0T6p
  LV Write Access        read/write
  LV Creation host, time HServer01, 2016-03-15 12:29:55 +0100
  LV Status              available
  # open                 1
  LV Size                600,00 GiB
  Current LE             153600
  Segments               1
  Allocation             inherit
```

```
  Read ahead sectors      auto
  - currently set to      256
  Block device            252:3
…
```

Nuestra siguiente tarea será iniciar dataserver y configurarlo desde dentro para ofrecer los servicios deseados.

1.3. Configuración de dataserver

Para preparar el servidor de ficheros de nuestra empresa usaremos un script instalador similar al que desarrollamos para el anfitrión. De esta forma almacenamos el procedimiento, lo podemos ir mejorando con el tiempo y reutilizar tantas veces como necesitemos. El script completo (00_instalar.sh) está disponible en el Anexo B.1.

Al inicio de este script nos encontramos con la sección para personalizar los parámetros y adaptar dataserver a nuestra medida. Este será el primer paso del proceso. Además del script, como podemos leer en los comentarios de los parámetros, tenemos que personalizar los usuarios y claves que generaremos para los puestos remotos. La sección de configuración de red no se aplica desde las versiones 4.X de Proxmox, en estas se realiza desde el anfitrión.

```
################################################################################
## CONFIGURACIÓN
################################################################################
# Versión de debian usada para el container
# en junio de 2015 -> jessie
DEBIAN_VER=jessie
# Nombre de empresa para mostrar información
NOMBRE_EMPRESA="NombreEmpresa"
EMPRESA_CORTO="NomEmpre"
##### Usuarios y grupos unix
USU_PRI="usuarioPri"
USU_FULL_NAME="Nombre Usuario Pri"
USU_PRI_PASS="claveDelPri"
# Grupo común para los recursos compartidos de los usuarios del ayto
GRU_PRI="grupoPrincipal"
#
# Resto de usuarios que se crean para puestos remotos y comparticiones SAMBA
# estos usuarios no tienen acceso mediante SHELL.
# F_USUARIOS debe contener una lista con estos usuarios (uno por línea)
# donde se especifica su contraseña y grupo principal. El formato del fichero es por tanto:
# COL1  COL2    COL3
# user  clave   grupo
# pc01  pc011   grcps
#
USUARIOS="Usuario pc01 pc02 pc03 pc04 pc05 pc06 pc07 pc08 pc09 pc10"
F_USUARIOS="./usuarios.txt"

##########
# Grupo de trabajo SAMBA
SAMBA_WG="WORKGROUP"

# Config de red
## OJO:
# La configuración de la red en los LXC ya no es necesaria.
# Se realiza desde el HOST. Se mantiene esta sección por si hay que volver a OpenVZ.
#
###
IFAZ=eth0
IP_ADDR=192.168.7.11
```

```
NETMASK=255.255.255.0
LAN_ADDR=192.168.7.0
LAN_BCAST=192.168.7.255
GATEWAY=192.168.7.1
DNS_SERVER=192.168.7.1
##########

# Directorio origen con los ficheros de configuración para copiar (YA NO SE USA)
# CFDIR="./Conf_Files"
# Directorio origen con los scripts personalizados
SCDIR="./Scripts"
# Destino para instalar los scripts
SCRIPTS_INSTALL="/usr/local/sbin"
################################################################################
```

Una vez hemos modificado el fichero con nuestra configuración, debemos copiarlo al contenedor para ejecutarlo desde dentro. Repetiremos el proceso que seguimos para el host. Primero arrancamos el contenedor, nos conectamos mediante la consola Proxmox y habilitamos el acceso por SSH para el usuario root.

```
nano /etc/ssh/sshd_config
```

Cambiamos la línea:

```
PermitRootLogin without-password
```

a

```
PermitRootLogin yes
```

y reiniciamos el servidor SSH:

```
root@dataserver:~# service ssh restart
Restarting OpenBSD Secure Shell server: sshd.
```

Ahora podremos iniciar sesión desde nuestro puesto para empezar la instalación:

```
~$ ssh root@192.168.7.11
root@192.168.7.11's password:
Linux dataserver 4.2.8-1-pve #1 SMP Sat Mar 19 10:44:29 CET 2016 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Apr 26 06:10:06 2016
-bash: warning: setlocale: LC_ALL: cannot change locale (es_ES.UTF-8)
-bash: warning: setlocale: LC_ALL: cannot change locale (es_ES.UTF-8)
root@dataserver:~#
```

El siguiente paso será copiar los ficheros de la instalación:

```
scp -r 02_Dataserver root@192.168.7.11:
root@192.168.7.11's password:
bash: warning: setlocale: LC_ALL: cannot change locale (es_ES.UTF-8)
usuarios.txt                                                  100%  226
0.2KB/s   00:00
```

```
viruscan                                                    100% 1661
1.6KB/s   00:00
AddUser.sh                                                  100%  678
0.7KB/s   00:00
00_instalar.sh                                              100%   16KB
15.5KB/s   00:00
```

Y nos conectamos para empezar el proceso:

```
ssh root@192.168.7.11
root@dataserver:~# ls
```

```
02_Dataserver
```

```
root@dataserver:~# cd 02_Dataserver
root@dataserver:~/02_Dataserver# ls -la
```

```
total 36
drwxr-xr-x+ 3 root root  4096 Apr 26 06:30 .
drwx------+ 4 root root  4096 Apr 26 06:30 ..
-rw-------  1 root root 15900 Apr 26 06:30 00_instalar.sh
-rw-------  1 root root   678 Apr 26 06:30 AddUser.sh
drwxr-xr-x+ 2 root root  4096 Apr 26 06:30 Scripts
-rw-------  1 root root   226 Apr 26 06:30 usuarios.txt
```

```
root@dataserver:~/02_Dataserver# chmod +x 00_instalar.sh
root@dataserver:~/02_Dataserver#  ./00_instalar.sh
```

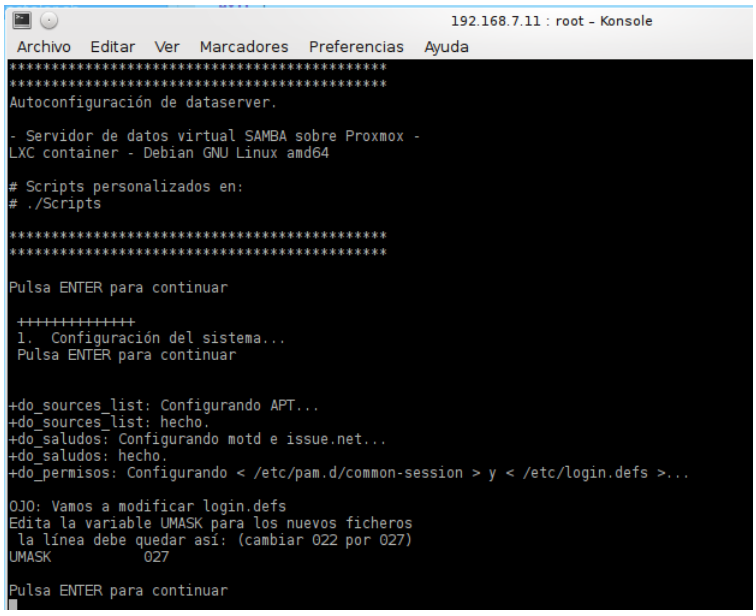


Ilustración 81: Configuración de dataserver. Inicio del script de instalación

El primer momento en que el script solicita nuestra intervención es para editar a mano el fichero ***login.defs*** y verificar que todo marcha correctamente. Ese cambio lo realizamos para que el sistema establezca correctamente los permisos en la creación de nuevos ficheros y resulte más sencillo la compartición de recursos con SAMBA. El proceso continúa hasta que nos pide

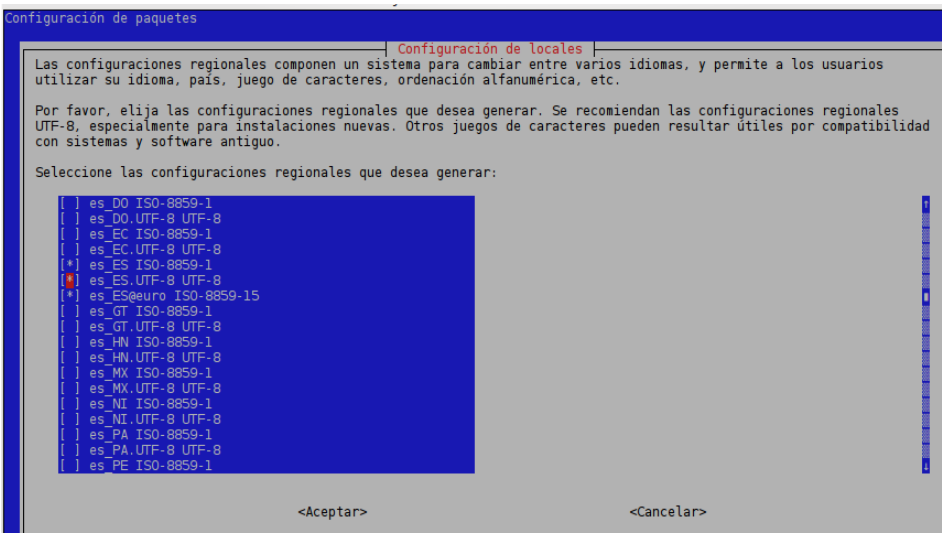


Ilustración 82: Configuración de dataserver - LOCALES

seleccionar los LOCALES (los juegos de caracteres utilizados en el sistema). Generamos los españoles y definimos UTF-8 como predeterminado.

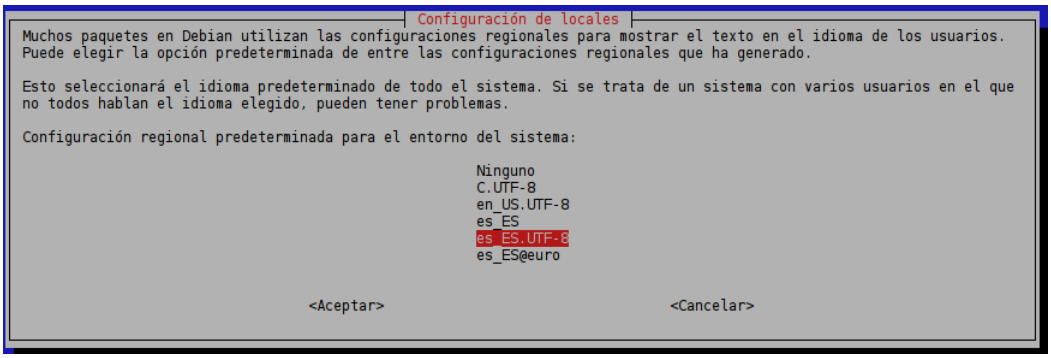


Ilustración 83: Configuración de dataserver - LOCALES II

A continuación se realizará la configuración de SAMBA y se crearán los usuarios y grupos tal como hayamos definido en el script.

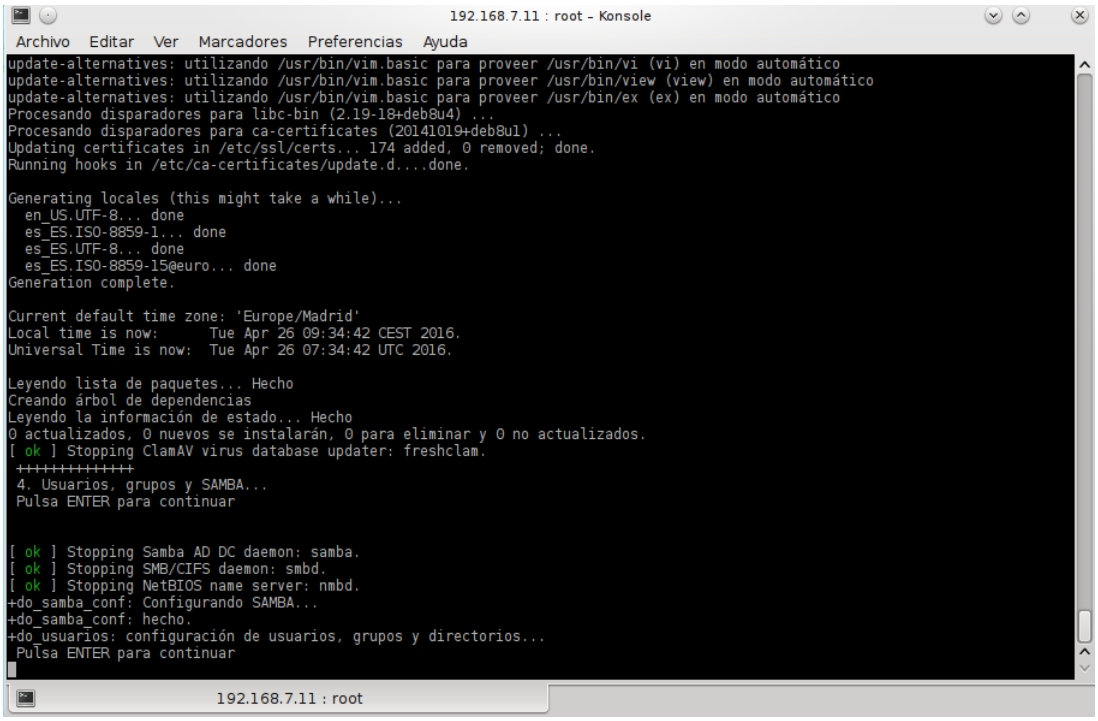


Ilustración 84: Configuración de dataserver – SAMBA, usuarios y grupos

```
++++++++++++++
4. Usuarios, grupos y SAMBA...
Pulsa ENTER para continuar


[ ok ] Stopping Samba AD DC daemon: samba.
[ ok ] Stopping SMB/CIFS daemon: smbd.
[ ok ] Stopping NetBIOS name server: nmbd.
+do_samba_conf: Configurando SAMBA...
+do_samba_conf: hecho.
+do_usuarios: configuración de usuarios, grupos y directorios...
Pulsa ENTER para continuar

Creando grupo de usuarios: < grupoPri >...
Añadiendo el grupo `grupoPri' (GID 1000) ...
Hecho.
...OK
A continuación se va a crear el usuario principal de dataserver...
Pulsa ENTER para continuar

Creando usuario principal: < usuPri > | Clave: < clavePri > | Grupo: <
grupoPri >...
Added user usuPri.
A continuación se van a crear los usuarios para los puestos remotos...
Pulsa ENTER para continuar

Creando usuario: < Usuario > | Clave: < usuClave1 > | Grupo: < grupoPri >...
Added user Usuario.
...OK
Creando usuario: < pc01 > | Clave: < pc011 > | Grupo: < grupoPri >...
Added user pc01.
...OK
Creando usuario: < pc02 > | Clave: < pc022 > | Grupo: < grupoPri >...
Added user pc02.
...OK
Creando usuario: < pc03 > | Clave: < pc033 > | Grupo: < grupoPri >...
Added user pc03.
...OK
Creando usuario: < pc04 > | Clave: < pc044 > | Grupo: < grupoPri >...
Added user pc04.
...OK
```

```
Creando usuario: < pc05 > | Clave: < pc055 > | Grupo: < grupoPri >...
Added user pc05.
...OK
Creando usuario: < pc06 > | Clave: < pc066 > | Grupo: < grupoPri >...
Added user pc06.
...OK
Creando usuario: < pc07 > | Clave: < pc077 > | Grupo: < grupoPri >...
Added user pc07.
...OK
Creando usuario: < pc08 > | Clave: < pc088 > | Grupo: < grupoPri >...
Added user pc08.
...OK
Creando usuario: < pc09 > | Clave: < pc099 > | Grupo: < grupoPri >...
Added user pc09.
...OK
Creando usuario: < pc10 > | Clave: < pc100 > | Grupo: < grupoPri >...
Added user pc10.
...OK
A continuación se van a crear los directorios comunes y a ajustar los
permisos...
Pulsa ENTER para continuar

Directorio personal del usuario < Usuario >:
 [ /home/Usuario ] ...
.
Directorio personal del usuario < pc01 >:
 [ /home/pc01 ] ...
.
Directorio personal del usuario < pc02 >:
 [ /home/pc02 ] ...
.
Directorio personal del usuario < pc03 >:
 [ /home/pc03 ] ...
.
Directorio personal del usuario < pc04 >:
 [ /home/pc04 ] ...
.
Directorio personal del usuario < pc05 >:
 [ /home/pc05 ] ...
.
Directorio personal del usuario < pc06 >:
 [ /home/pc06 ] ...
.
Directorio personal del usuario < pc07 >:
 [ /home/pc07 ] ...
.
Directorio personal del usuario < pc08 >:
 [ /home/pc08 ] ...
.
Directorio personal del usuario < pc09 >:
 [ /home/pc09 ] ...
.
Directorio personal del usuario < pc10 >:
 [ /home/pc10 ] ...
.
Directorio personal del usuario principal < usuPri >:
 [ /home/usuPri ] ...
.
Directorio compartido privado:
< /home/usuPri/Empresa >
.
Directorio compartido público:
< /home/usuPri/Publica >
.
Directorio de mantenimiento Informaticasa:
< /home/usuPri/Informaticasa >
.
+do_usuarios: hecho.
++++++++ ...hecho. ++++++++
```

Una vez termina el bucle de creación de usuarios y sus directorios personales hemos finalizado de preparar dataserver. El servidor que hemos preconfigurado ofrecerá

almacenamiento SAMBA para diferentes puestos remotos. La estructura de directorios que se propone de inicio es la siguiente:

- Un directorio privado para cada usuario definido, solo accesible por él.
- Un directorio común privado accesible por todos los usuarios creados, pero solamente por ellos.
- Un directorio público con permisos de lectura y escritura para todo el mundo.
- Un directorio público llamado Informaticasa que se destinará a almacenar ficheros relacionados con el mantenimiento informático.

Se ha planteado esta organización de carpetas porque generalmente se adapta muy bien a los requisitos básicos de almacenamiento de las empresas con que hemos trabajado, pero siempre es posible añadir más recursos, editar permisos y personalizar la estructura de forma muy sencilla modificando **/etc/samba/smb.conf** en dataserver.

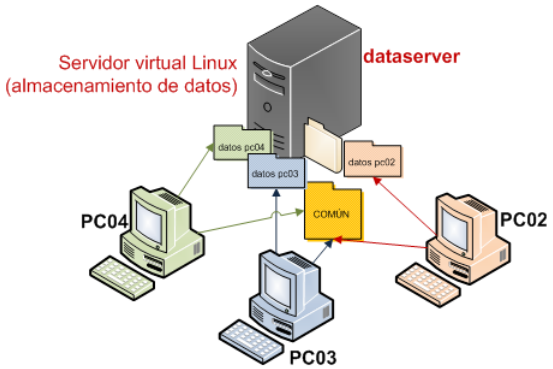


Ilustración 85: Estructura de comparticiones SAMBA en dataserver

Para facilitar el acceso desde los puestos a su directorio correspondiente usaremos un fichero .bat. Recordemos que generalmente nos encontramos con redes pequeñas, donde no es necesaria una infraestructura tipo dominio centralizado o directorio activo. Para simplificar el trabajo de configuración de usuarios y grupos el .bat mapeará los directorios deseados en unidades de red, realizando la tarea de autenticación. Estos .bat los podemos generar de manera automática en dataserver y dejarlos preparados para solo tener que copiarlos a cada puesto. El script makeUniRedScripts.sh (Anexo B.2) cumple este cometido.

```
root@dataserver:~/02_Dataserver# ./makeUniRedScripts.sh
unix2dos: convirtiendo archivo ./UniRedScripts/ConectaUniRedPC01.bat a formato
DOS ...
unix2dos: convirtiendo archivo ./UniRedScripts/ConectaUniRedPC02.bat a formato
DOS ...
```

```
...
...
unix2dos: convirtiendo archivo ./UniRedScripts/ConectaUniRedPC19.bat a formato
DOS …
```

```
root@dataserver:~/02_Dataserver# cat UniRedScripts/ConectaUniRedPC02.bat
NET USE z: \\192.168.7.11\pc02 /user:192.168.7.11\pc02 pc022
NET USE y: \\192.168.7.11\Compartida /user:192.168.7.11\pc02 pc022
```

Como utilidad extra para ayudar en la seguridad de la información se ha añadido a dataserver un script que facilita el uso de **clamav.** Clamav es un antivirus que permite analizar ficheros y directorios desde Linux en busca de malware para Windows. Programando una tarea y ejecutándolo periódicamente podemos mantener más limpia la información de los usuarios y evitar males mayores. En el Anexo B.3 está el código de este script, al que hemos llamado ***viruscan***.

2. netserver: contenedor LXC – Turnkey Linux OpenVPN

Para que los usuarios puedan trabajar desde cualquier lugar y acceder de forma segura a la subred desplegaremos un servidor OpenVPN[50]. Nos apoyaremos en OpenVPN por ser software libre, multiplataforma, seguro (el tráfico va cifrado por SSL y la autenticación se realiza por certificado), fácil de configurar y está perfectamente integrado en nuestro entorno Proxmox. Aprovecharemos la plantilla ofrecida por TurnkeyLinux:

http://www.turnkeylinux.org/openvpn

Esta plantilla consiste en un sistema Debian GNU/Linux 8 (Jessie) preconfigurado con OpenVPN y entre otras, las siguientes características añadidas:

- Asistentes para facilitar la configuración de OpenVPN según los tres perfiles o modos de funcionamiento más usados: servidor, puerta de enlace o cliente.
- Todos los perfiles soportan certificados SSL/TLS para autenticación e intercambio de claves.
- Los perfiles de servidor y puerta de enlace incluyen scripts para crear los clientes con todas sus claves y certificados de forma muy sencilla.

Los tres perfiles mencionados y su funcionamiento son los siguientes.

1. servidor: acepta conexiones desde clientes y opcionalmente habilita el acceso a las subredes privadas que queramos. Configura una ruta en el cliente indicándole que para alcanzar ciertas subredes debe usarlo como puerta de enlace.
2. gateway: acepta conexiones desde clientes y se encarga de enrutar todo el tráfico a internet de ellos. Digamos que el servidor en este caso se convierte en la puerta de enlace por defecto para los clientes de la VPN.
3. cliente: se conecta a otro servidor OpenVPN.

50 https://openvpn.net/index.php/open-source/339-why-ssl-vpn.html

En nuestro caso seleccionaremos el tipo servidor, pues queremos alcanzar la siguiente topología:

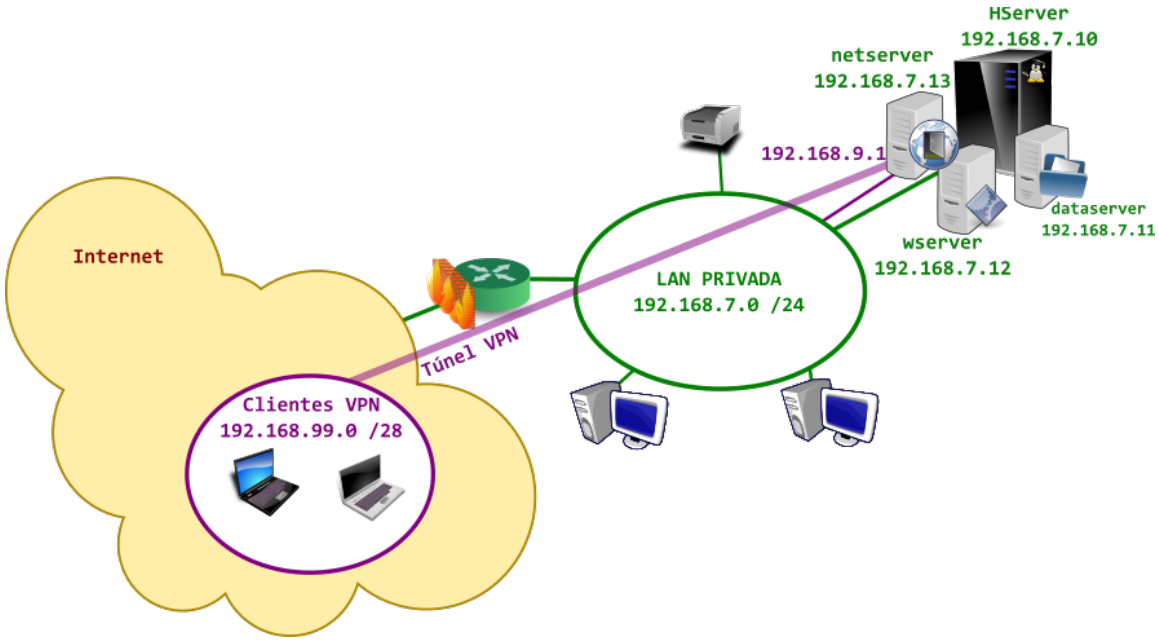


Ilustración 86: Topología para la VPN

2.1. Descarga de la plantilla

Repetimos el proceso realizado al descargar la imagen de Debian para dataserver, y bajaremos en local2 la plantilla de Turnkey – OpenVPN.

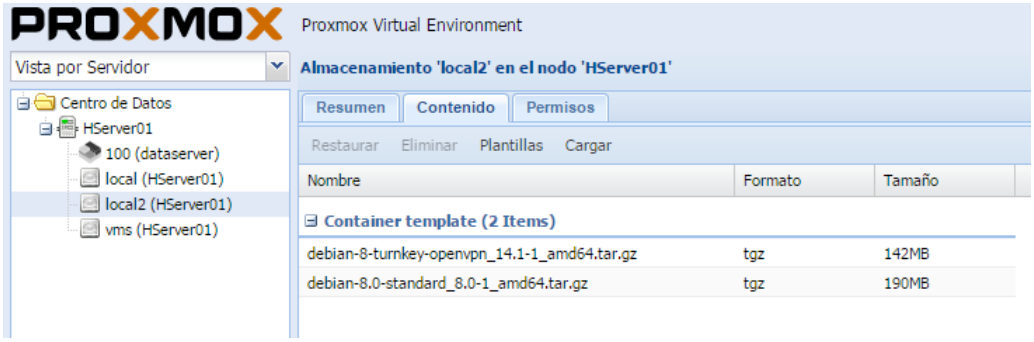


Ilustración 87: Plantilla turnkey openvpn descargada

2.2. Preparación del contenedor

Una vez hecho descargada la plantilla, repetimos el proceso para crear un contenedor que vimos con dataserver. Eligiendo esta vez la plantilla de turnkey-openvpn, asignándole como nombre del host ***netserver*** y los siguientes recursos:

- Almacenamiento: vms, tamaño: 4GB
- CPU: cpu limit=1, utlización CPU: 1024
- Memoria: RAM=512, SWAP=512
- Redes: PUENTE=vmbr0, IP=192.168.7.12/24, GW=192.168.7.1

En https://www.turnkeylinux.org/docs/hardware-requirements podemos verificar que con estas características cumplimos con creces el mínimo para un buen funcionamiento. De hecho, con solo 256MB de RAM están probados contenedores.

El resumen antes de confirmar la creación queda como sigue:

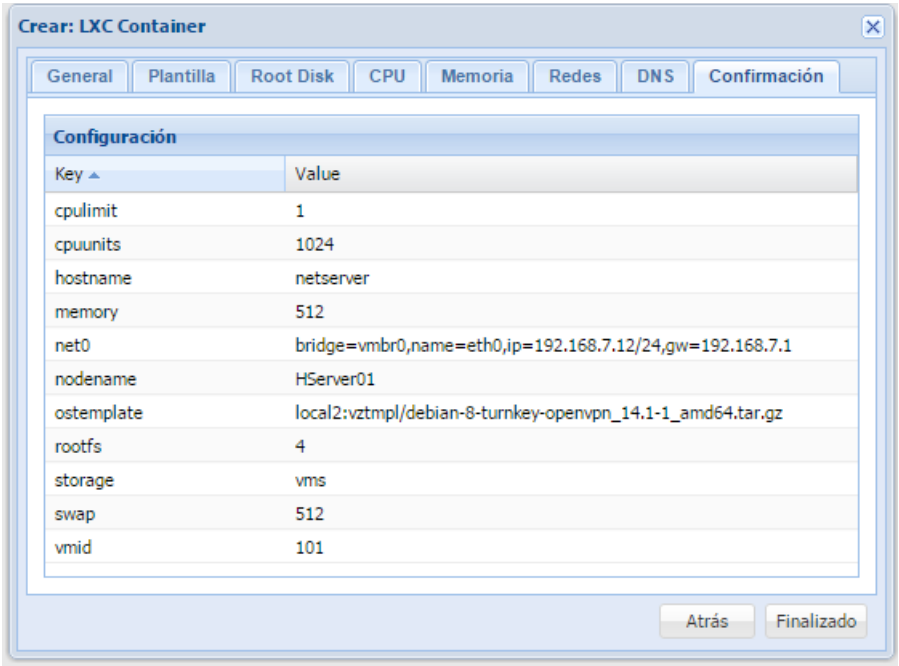


Ilustración 88: Resumen tras finalizar creación de netserver

Y una vez confirmado, contaremos con nuestro segundo contenedor (ID=101) en la lista de máquinas virtuales.

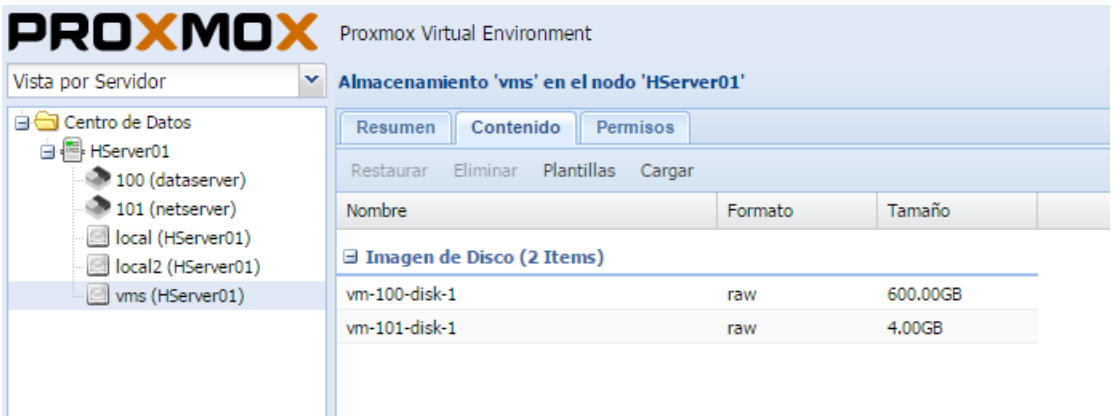


Ilustración 89: Creación de un contenedor LXC – netserver terminado

2.3. Configuración de netserver

En el primer arranque de nuestro servidor OpenVPN nos encontraremos con el asistente para seleccionar el perfil de trabajo y configurar las opciones. Deberemos tener preparados los siguientes datos:

- Una **dirección de email** para generar la clave de nuestro servidor OpenVPN. (por ejemplo netserver@midominio.es).
- FQDN (Fully Qualified Domain Name) de nuestro servidor. Esto es la **dirección que apuntará a la IP publica** de nuestro equipo, y que usarán los clientes para acceder desde internet. Si tenemos IP fija y dominio, algo como (vpn.midominio.es) será este. Si usamos IP dinámica podemos configurar aquí el dns dinámico: miservervpn.dyndns.org. En otro momento veremos también como usar netserver como actualizador de dyndns mediante ddclient.
- Dirección de la **subred** privada en que ubicaremos a los **clientes de la VPN** en formato CIDR. Usaremos en nuestro caso, por ejemplo, la subred 192.168.99.0/28.
- **Dirección de la subred privada** a la que vamos a habilitar el acceso en formato CIDR. En nuestro caso estamos trabajando con 192.168.7.0/24.

Veamos algunas capturas del proceso de configuración. Una vez completemos el asistente tendremos listo nuestro servidor, sólo necesitaremos crear nuestros clientes.

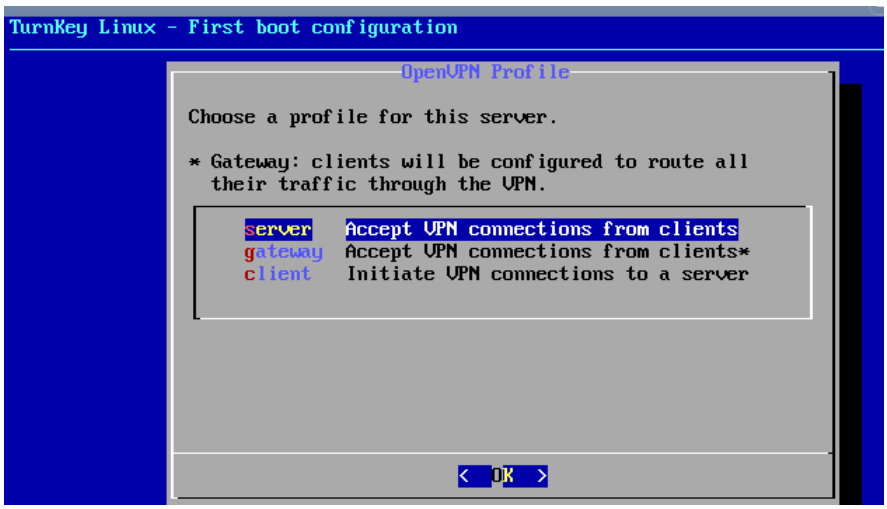


Ilustración 90: Asistente de inicio turnkeyvpn – selección de perfil

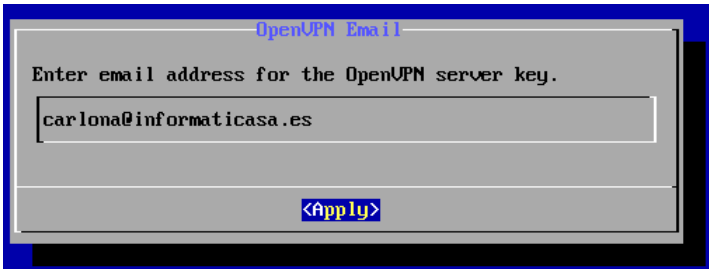


Ilustración 91: Asistente de inicio turnkeyvpn – correo

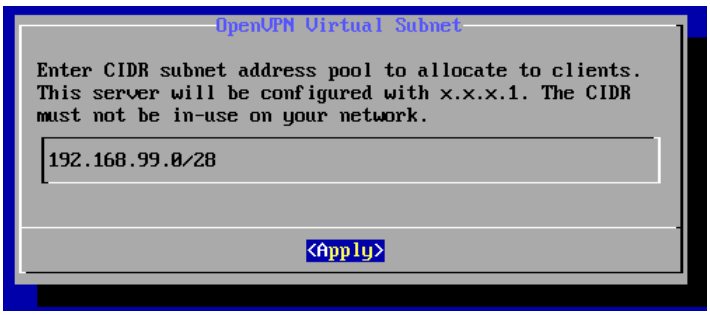


Ilustración 92: Asistente de inicio turnkeyvpn – subred clientes

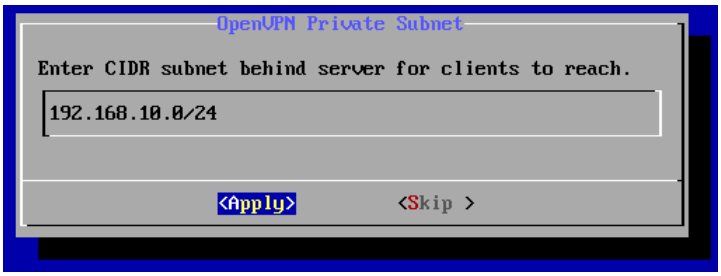


Ilustración 93: Asistente de inicio turnkeyvpn – subred a la que conceder acceso

2.3.1. Añadir clientes y configuración de los equipos remotos

El comando que nos proporciona TurnkeyLinux preparado para este propósito es el siguiente:

```
openvpn-addclient -h
```

```
Syntax: openvpn-addclient client-name client-email [private-subnet]
Generate keys and configuration for a new client

Arguments:
    client-name         Unique name for client
    client-email        Client email address
    private-subnet      CIDR subnet behind client (optional)
```

```
$ openvpn-addclient office office@example.com 172.16.1.0/24
$ openvpn-addclient fulanito correo@defulanito.es
```

```
INFO: generated /etc/openvpn/easy-rsa/keys/office.ovpn
```

Para configurar el cliente remoto solamente tendremos que trasladar el fichero .ovpn generado a su equipo y ejecutar el software de OpenVPN[51]. Este fichero contiene todos los datos de acceso y

2.4. Acceso a netserver desde internet

2.4.1. Servicio DynDNS

En el mismo netserver podemos instalar un cliente para servicios dyndns y así resolver el típico problema con direcciones IP dinámicas. Utilizaremos el software **ddclient.**

Para instalarlo:

```
apt-get install ddclient
```

Ejemplo de configuración con una cuenta en www.dyndns.com:

```
cat /etc/ddclient.conf
```

```
# Configuration file for ddclient generated by debconf
#
# /etc/ddclient.conf

protocol=dyndns2
use=web, web=checkip.dyndns.com, web-skip='IP Address'
server=members.dyndns.org
login=nuestroUsuario
password='claveDeNuestraCuenta'
miservervpn.dyndns.org
```

51 https://openvpn.net/index.php/open-source/downloads.html

2.4.2. Redirección de puertos en el router de acceso a internet

El escenario típico que nos encontraremos son subredes con una única IP dinámica para acceder a internet y NAT por puertos. Para que funcione el acceso a nuestro servidor desde el exterior deberemos redireccionar el puerto usado por OpenVPN. Por defecto trabaja con el 1194 y el tipo de tráfico es UDP.

Un ejemplo de las reglas necesarias si nuestro router es un Mikrotik y nuestro servidor OpenVPN tiene la IP interna 192.168.7.12 (nuestro entorno de laboratorio):

1. Regla en la tabla NAT (DSTNAT – realiza el reenvío):

```
add action=dst-nat chain=dstnat comment="OpenVPN udp" dst-port=1194 in-
interface=eth1-wan protocol=udp to-addresses=192.168.7.12 to-ports=1194
```

2. Regla en la tabla filter (FORWARD – permite el tráfico) :

```
add chain=forward comment="OpenVPN udp" connection-state=new dst-address=
192.168.7.12 dst-port=1194 in-interface=eth1-wan protocol=udp
```

2.5. Configuración requerida en el anfitrión (dispositivos TUN/TAP)

Para que el contenedor pueda ejecutar correctamente el servidor OpenVPN y crear los túneles necesarios debe poder activar dispositivos TUN/TAP. Para ello hay que concederle un permiso especial desde el anfitrión. Según sea un contenedor LXC u OpenVZ deberemos hacerlo de forma diferente.

2.5.1. En versiones anteriores a Proxmox 4 (OpenVZ)

1. En el anfitrión, editar */etc/vz/vz.conf* para añadir varios módulos extra a IPTABLES (estado, tabla NAT, etc.). Editaremos la linea con la variable *IPTABLES*= para que quede del siguiente modo:

```
root@HServer01:~# nano /etc/vz/vz.conf
## IPv4 iptables kernel modules to be enabled in CTs by default
IPTABLES="ipt_REJECT ipt_tos ipt_limit ipt_multiport iptable_filter
iptable_mangle ipt_TCPMSS ipt_tcpmss ipt_ttl ipt_length iptable_nat
ip_conntrack ipt_conntrack"
## IPv4 iptables kernel modules to be loaded by init.d/vz script
IPTABLES_MODULES="$IPTABLES"

## Enable IPv6
IPV6="no"

## IPv6 ip6tables kernel modules
IP6TABLES="ip6_tables ip6table_filter ip6table_mangle ip6t_REJECT"
```

2. En el anfitrión, asegurarnos de que el módulo tun está disponible y cargado. Lo añadiremos al fichero /etc/modules para asegurarnos que que así es en cada inicio del sistema:

```
root@HServer01:~# lsmod | grep tun
root@HServer01:~# modprobe tun
root@HServer01:~# cat /etc/modules
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.
tun
```

3. En el anfitrión, damos permiso a netserver (ID=102) para usar dispositivos tun/tap:

```
root@HServer01:~# CTID=102
root@HServer01:~# vzctl set $CTID --devnodes net/tun:rw --save
root@HServer01:~# vzctl set $CTID --devices c:10:200:rw --save
root@HServer01:~# vzctl set $CTID --capability net_admin:on --save
root@HServer01:~# vzctl exec $CTID mkdir -p /dev/net
root@HServer01:~# vzctl exec $CTID mknod /dev/net/tun c 10 200
root@HServer01:~# vzctl exec $CTID chmod 600 /dev/net/tun
```

2.5.2. Para Proxmox 4 y posteriores (LXC)

1. En el anfitrión, añadimos al fichero */etc/pve/lxc/101.conf* la línea:

```
lxc.cgroup.devices.allow: c 10:200 rwm
```

```
root@HServer01:~# cat /etc/pve/lxc/101.conf
arch: amd64
cpulimit: 1
cpuunits: 1024
hostname: netserver
memory: 512
net0:
bridge=vmbr0,gw=192.168.7.1,hwaddr=32:63:35:31:62:61,ip=192.168.7.12/24,name=e
th0,type=veth
onboot: 1
ostype: debian
rootfs: vms:vm-101-disk-1,size=4G
swap: 512
lxc.cgroup.devices.allow: c 10:200 rwm
```

2. En el contenedor, editamos */etc/rc.local* para que genere un dispositivo *tun* al inicio y recargue el servicio OpenVPN:

```
root@netserver ~# cat /etc/rc.local
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
```

```
# By default this script does nothing.
####################
##  Prepara tun/tap para OpenVPN server
####################
if ! [ -d /dev/net ];then
 mkdir /dev/net
fi
if ! [ -c /dev/net/tun ]; then
 mknod /dev/net/tun c 10 200
 chmod 666 /dev/net/tun
fi
##
sleep 1
/etc/init.d/openvpn restart
##
exit 0
```

3. wserver2008: máquina KVM – Windows Server 2008R2

Aunque no entraremos en la instalación y configuración del sistema operativo, veremos en esta sección cómo sería el proceso para crear una máquina usando KVM y así poder virtualizar un sistema de Microsoft. Prepararemos un servidor virtual para instalar un Server 2008R2. El proceso es muy similar para todos los Windows, pero cada versión tiene sus particularidades y opciones que podemos ajustar para optimizar su funcionamiento como *guest*. Veremos algunas de ellas para nuestro caso.

3.1. Obtención de la ISO con el instalador del S.O

El primer paso que debemos realizar es disponer de la imagen con el instalador del sistema. Para ello podemos introducir un CD si el anfitrión tiene lector y luego seleccionarlo al crear la máquina, o como haremos nosotros, utilizar una imagen .iso. En nuestro laboratorio de pruebas están almacenadas en un servidor SAMBA de la red local, por lo que debemos trasladarlas al anfitrión y ubicarlas en un almacenamiento configurado como contenedor de imágenes. Usaremos el comando ***smbget*** para hacerlo desde la línea de comandos. Copiaremos la imagen con el instalador del sistema (wserver.iso) y otra imagen (virtio-win-0.1-100.iso) que contiene drivers que usaremos en la instalación

```
root@HServer01:~# cat /vz/template/iso/descargaIsos.sh
#!/bin/bash
smbget smb://192.168.10.11/01_Programas/SistemasO_isos/virtio-win-0.1-100.iso
smbget
smb://192.168.10.11/01_Programas/Windows_Y_Office/Windows_Server_2008_R2/wserv
er.iso
```

```
root@HServer01:~# ls -l /vz/template/iso/
total 3190436
-rwxr-xr-x 1 usuario usuario        183 abr  1 11:47 descargaIsos.sh
-rwxr-xr-x 1 root    root     69953536 abr  1 11:47 virtio-win-0.1-100.iso
-rwxr-xr-x 1 root    root   3197040640 abr  1 11:54 wserver2008.iso
```

Estos drivers los usaremos para mejorar el rendimiento de la máquina virtual (en la gestión del almacenamiento, red y memoria ram) aplicando técnicas de ***paravirtualización***, concepto que ya estudiamos en el capítulo 2. Las últimas versiones de estos drivers están publicadas en:

https://fedoraproject.org/wiki/Windows_Virtio_Drivers#Direct_download

3.2. Preparación de la máquina KVM

Para iniciar el proceso y crear nuestro servidor virtual Windows accedemos a la interfaz web de Proxmox, y en la esquina superior derecha, pulsamos el botón Crear VM.

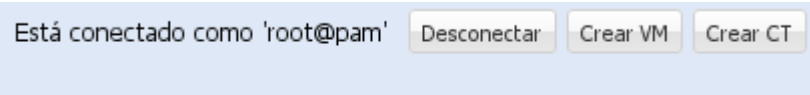


Ilustración 94: Inicio crear máquina virtual KVM

Arrancará el asistente que ya conocemos. Asignamos el primer ID libre y un nombre a nuestro equipo.

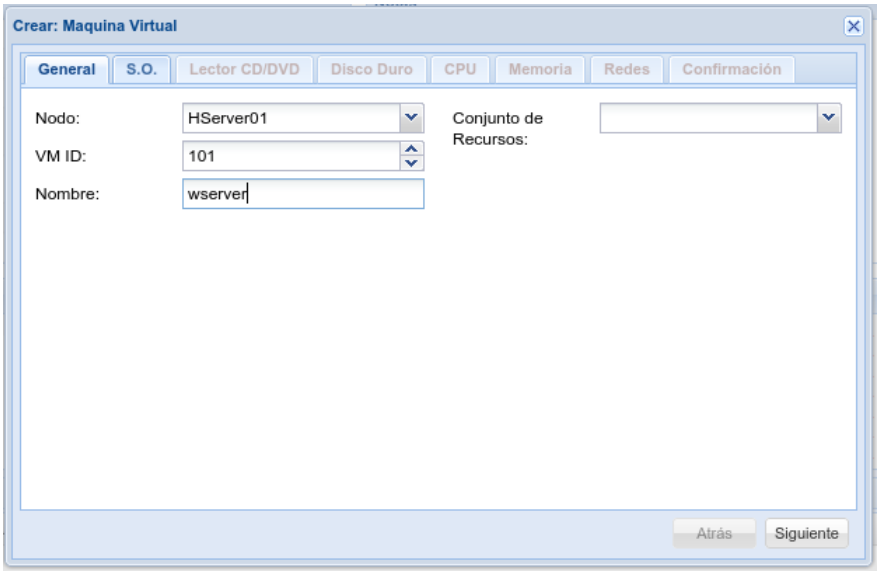


Ilustración 95: Creación de una máquina KVM – general

Fijamos el tipo de S.O. que contendrá

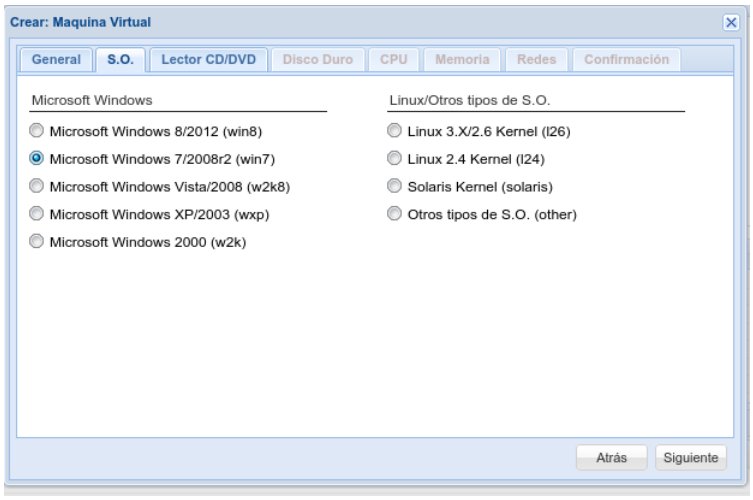


Ilustración 96: Creación de una máquina KVM – S.O.

El medio origen que contiene el instalador

Ilustración 97: Creación de una máquina KVM – medio de instalacón

y pasamos a configurar las prestaciones que asignaremos al servidor virtual. Importante seleccionar el tipo virtio siempre que sea posible para los buses y dispositivos.

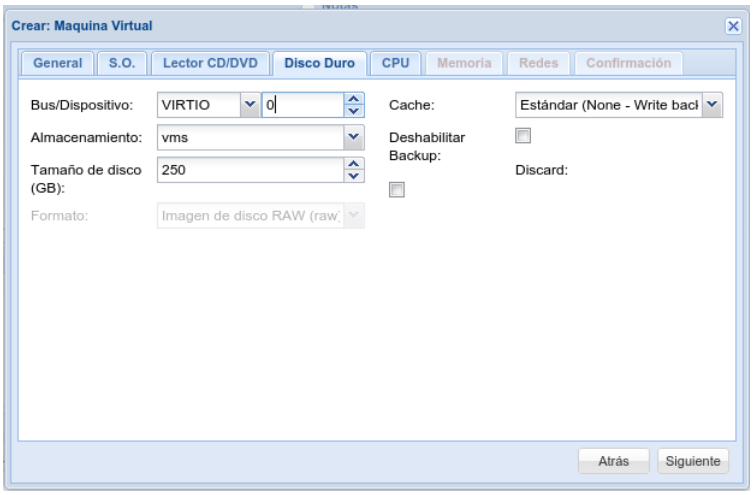


Ilustración 98: Creación de una máquina KVM – disco duro

Como procesador permitiremos el uso de los 4 núcleos disponibles y usaremos el tipo estándar (kvm64). Al usar el estándar de kvm nos abstraemos del hardware y facilitamos la portabilidad en caso de cambiar la plataforma hardware del anfitrión.

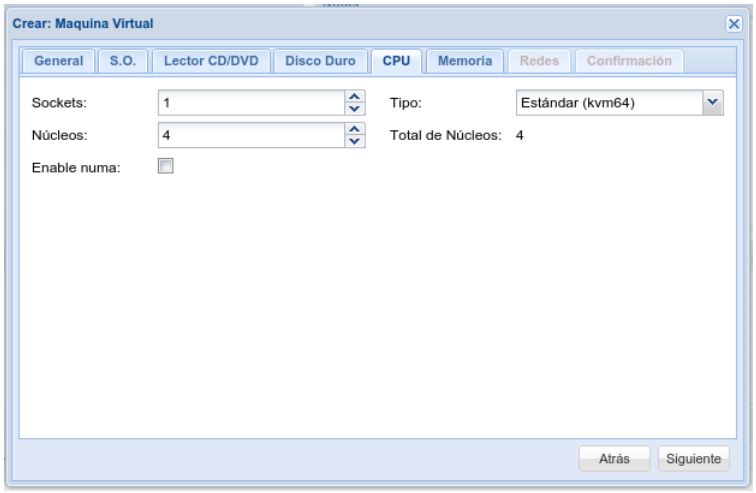


Ilustración 99: Creación de una máquina KVM – CPU

Como memoria RAM reservaremos de forma estática 3GB. La asignación dinámica es peligrosa porque suele ir de la mano del sobredimensionamiento y si en algún momento las máquinas exigen un uso cercano al máximo podemos encontrar problemas de estabilidad y bloqueos. Es preferible repartir de forma fija y controlada la memoria disponible y si se necesita más, ampliar la memoria añadiendo módulos al anfitrión.

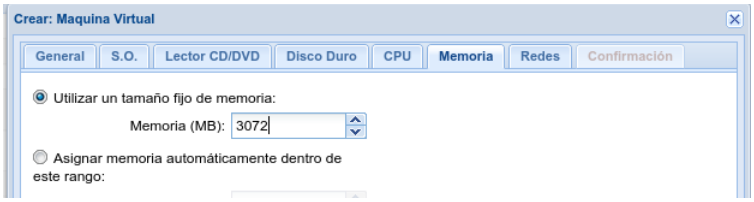


Ilustración 100: Creación de una máquina KVM – RAM

Por último configuraremos la interfaz de red de la máquina. La conectamos al puente virtual de nuestro anfitrión (vmbr0) y seleccionamos el tipo virtio. Observamos que, a diferencia de los contenedores LXC, en el caso de las máquinas QEMU-KVM la configuración IP no se ajusta aquí, sino que se realiza dentro del sistema invitado.

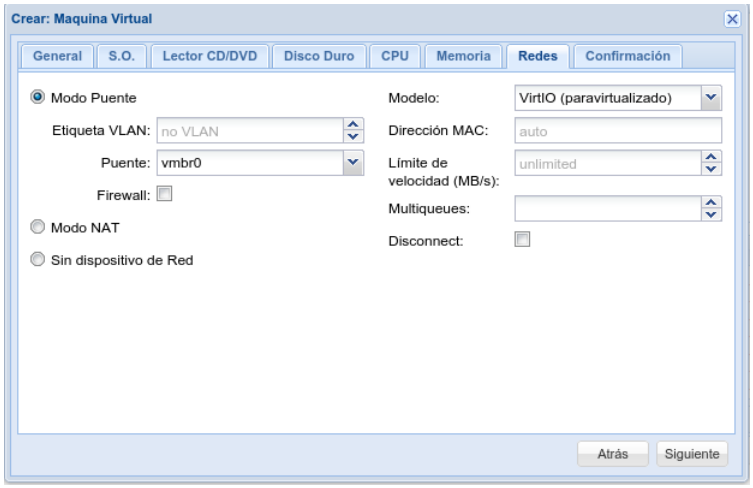


Ilustración 101: Creación de una máquina KVM – RED

Cuando terminamos el proceso y confirmamos la generación de la máquina nos queda un último paso antes de poder comenzar la instalación de Windows. Necesitamos conectar al equipo la segunda .iso que habíamos preparado con los drivers virtio para que estén disponible durante la instalación. El instalador requiere los controladores para detectar el disco duro y la interfaz de red. Para ello añadiremos un lector virtual de cdrom con conexión IDE o SATA y configuraremos la iso como si fuera un disco insertado.

Ilustración 102: Creación de una máquina KVM – añadir iso con drivers virtio

Una vez hecho todo esto, podremos arrancar la máquina en nuestra consola Proxmox e iniciar la instalación del sistema operativo invitado. En el siguiente enlace se mantiene una guía de buenas prácticas y recomendaciones para una configuración adecuada de Windows Server 2008 como máquina virtual en un entorno Proxmox. También hay publicadas muchas otras con información similar para el resto de versiones de Windows.

https://pve.proxmox.com/wiki/Windows_2008_guest_best_practices

Capítulo VI: Tareas de administración

1. Configuración de copias de seguridad

Como ya se ha discutido, para prevenir pérdidas de información causadas por problemas software (como borrados accidentales o infecciones de malware) la herramienta de que disponemos son las copias de seguridad. Esta fase de la configuración es tan importante como el resto, sin ella nuestro entorno de trabajo queda incompleto y es vulnerable a las amenazas más frecuentes.

Para implementar el sistema de copias en nuestro servidor Proxmox (teniendo en cuenta los objetivos marcados de economía, flexibilidad y seguridad) hemos planteado las siguientes opciones:

1. Añadir un tercer disco duro al anfitrión como almacenamiento local de backups.

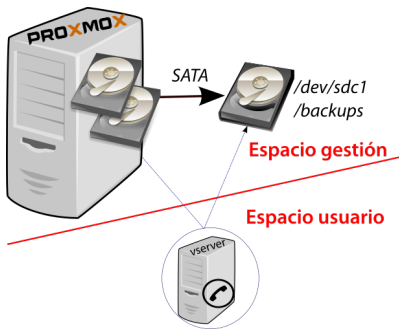


Ilustración 103: Backups en almacenamiento local

2. Configurar un NAS como almacenamiento remoto mediante NFS.

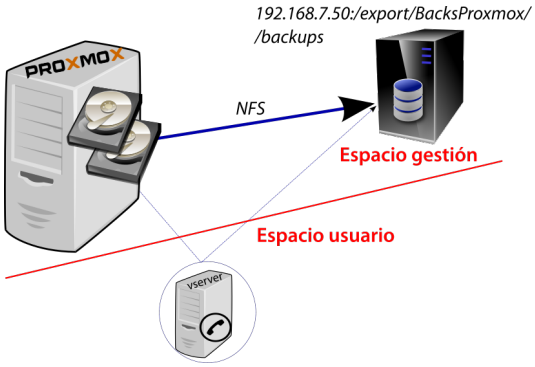


Ilustración 104: Backups en almacenamiento NFS

La primera opción es más económica, nos aporta velocidades de escritura y lectura mayores y las copias siguen estando aisladas del espacio de usuario, pues ese tercer disco duro sólo es accesible desde el propio anfitrión. Presenta el gran inconveniente de que comparte el mismo espacio físico que los datos principales: en caso de daños en el equipo anfitrión (problema eléctrico, incendio, robo, humedad) también podemos perder la copia de seguridad. Además, ocupa espacio de almacenamiento (ocupamos una bahía del equipo) que podría ser necesario más adelante.

La segunda opción requiere la compra y el montaje de un dispositivo externo. Esto nos permite ubicarlo en una zona alejada del servidor principal y resolver el problema del mismo emplazamiento. Sí dependemos, por tanto, de que exista conexión en red de buena velocidad para el tráfico de datos entre el anfitrión y el NAS. Hemos elegido NFS como protocolo de almacenamiento por su madurez, estabilidad, sencillez de configuración y seguridad. Veamos cómo realizar la configuración en los dos casos.

1.1. Destino de copias de seguridad en almacenamiento local

Una vez hayamos instalado el tercer disco duro, el proceso para habilitarlo como almacenamiento para nuestras copias consta de dos sencillos pasos:

1. Formatear y asignar un punto de montaje. Repetiremos el proceso por capas para crear un grupo de volúmenes LVM y ascender hasta una partición lógica de tipo ext4 montada en /backups. Hacerlo de este modo nos permitirá extender el espacio para las copias sumando dispositivos si lo necesitáramos.

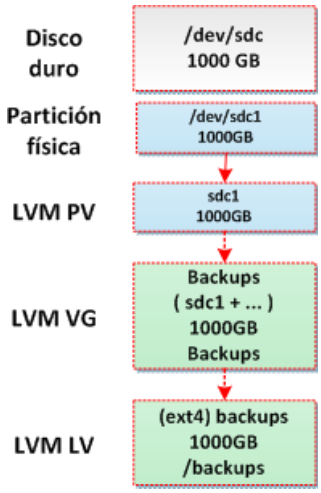


Ilustración 105: Particionado del tercer disco para backups

2. Añadir almacenamiento de tipo directorio en la interfaz de Proxmox y configurar la cantidad de copias almacenadas (esto nos definirá la rotación según la programación posterior).

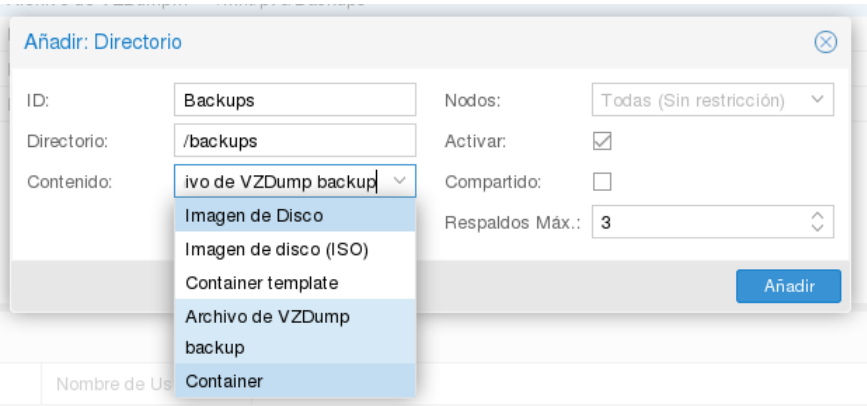


Ilustración 106: Añadir almacenamiento local para backups

1.2. Destino de copias en un NAS por NFS

En primer lugar deberemos elegir el equipo que usaremos como NAS. Existen infinidad de opciones en el mercado, y para elegir uno adecuado simplemente deberemos exigir dos características:

- Debe soportar NFS para exportar el almacenamiento
- Deberá tener capacidad suficiente para el tamaño y la cantidad de copias que queramos archivar.

Una vez cumpla con estos requisitos básicos, según el presupuesto del que dispongamos podemos buscar un equipo con mayor número de bahías, de mejor rendimiento o que integre

servicios más avanzados como gestión por SSH, configuraciones RAID de los discos, SAMBA, FTP y otros protocolos para compartir datos.

Por precio y prestaciones en nuestro caso hemos optado por un dispositivo DLINK DNS-340L. Tiene 4 bahías, admite configuración en JBOD y RAID 0 – 1 – 5 – 10, 2 interfaces de red gigabit, USB 3.0 y soporta los protocolos CIFS/SMB, iSCSI, NFS, AFP, DHCP Client, DDNS, NTP, FTP over SSL/TLS, FXP, HTTP/HTTPS, LLTD, PnP-X, UPnP AV, USB 3.0 , Bonjour o, WebDAV entre otros.

Ilustración 107: Dispositivo NAS DLINK DNS-340L

Las configuración que debemos realizar para integrarlo en nuestro sistema es mínima. Asignaremos una dirección IP estática, activaremos el servicio NFS y configuraremos el recurso exportado para la IP del servidor Proxmox. En el caso de las capturas realizadas, el NAS tiene la dirección 192.168.8.9/24 y el servidor la 192.168.8.10/24.

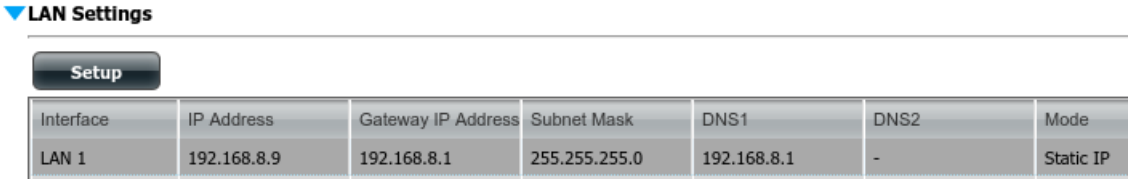


Ilustración 108: Configuración de red NAS DNS-340L

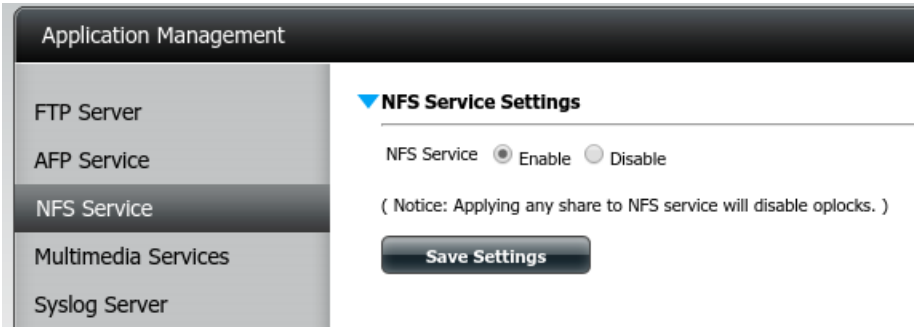


Ilustración 109: Configuración servicio NFS NAS DNS-340L

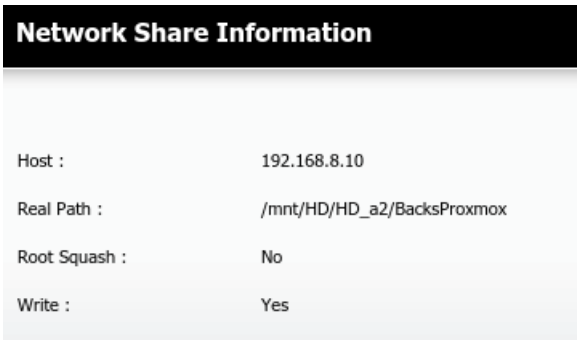


Ilustración 110: Configuración exportación NFS NAS DNS-340L

En el lado de nuestro servidor añadiremos un almacenamiento tipo NFS desde la interfaz de gestión de Proxmox. En cuanto escribamos la IP, Export mostrará el recurso disponible. En contenido deberemos activar VZDump backup. La cantidad de respaldos máximos es un bastante parámetro importante: define el máximo de copias de una misma máquina virtual que pueden almacenarse en ese destino. Si se va a realizar un backup automático y ya hay ese máximo, se borrará el más antiguo.

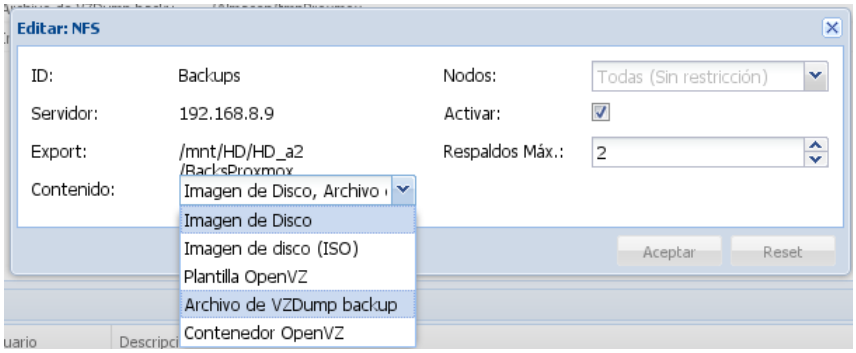


Ilustración 111: Añadir almacenamiento tipo NFS a Proxmox VE

Una vez hayamos activado el almacenamiento desde la interfaz podemos verificar su estado mediante la consola en el servidor. El comando ***mount*** nos muestra las unidades montadas en nuestro equipo:

```
usuario@HServer01:~$ mount |grep "type nfs"
192.168.8.9:/mnt/HD/HD_a2/BacksProxmox/ on /mnt/pve/Backups type nfs
(rw,relatime,vers=3,rsize=65536,wsize=65536,namlen=255,hard,proto=tcp,timeo=60
0,retrans=2,sec=sys,mountaddr=192.168.8.9,mountvers=3,mountport=51933,mountpro
to=udp,local_lock=none,addr=192.168.8.9)
```

1.3. Programación de las copias de seguridad

Ya hemos configurado un destino donde almacenaremos nuestros ficheros de backup. El siguiente paso es definir el tipo y la programación para realizarlos. En la imagen mostramos una configuración que realiza las copias del siguiente modo:

- Hace copia de las máquinas 100 (dataserver) y 103 (win7server).
- El modo de copia es Parar, esto quiere decir que los servidores virtuales son apagados y se salvan en estado de reposo. Para ello se envía una señal de apagado ACPI a las máquinas. Los backups de este modo son más fiables (el estado del servidor es más consistente). Si el sistema necesitara estar funcionando 24/7 deberíamos optar por una copia de tipo instantánea.
- El proceso se realiza las noches de martes y viernes (miércoles y sábado a las 00:00). Siempre se envía un email con el resultado de la operación a la dirección monitor@informaticasa.es.

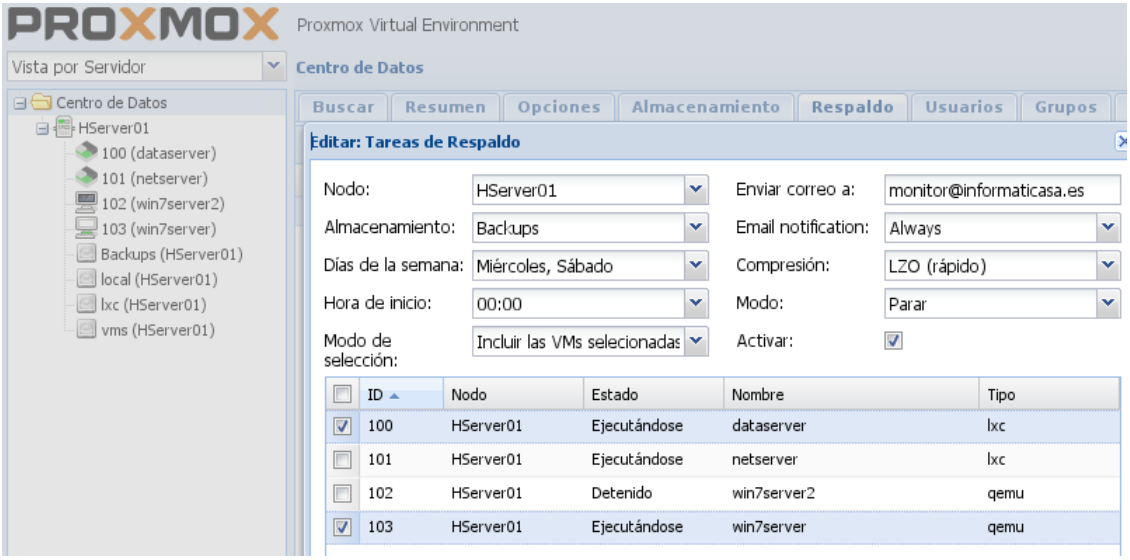


Ilustración 112: Programación de copias de seguridad

El email que recibimos tiene el aspecto siguiente:

Asunto: **vzdump backup status (HServer01.dominio) : backup successful**

VMID	NAME	STATUS	TIME	SIZE	FILENAME
100	dataserver	OK	01:33:54	194.65GB	/mnt/pve/Backups/dump/vzdump-lxc-100-2016_07_06-00_00_01.tar.lzo
103	win7server	OK	00:36:31	41.05GB	/mnt/pve/Backups/dump/vzdump-qemu-103-2016_07_06-01_33_55.vma.lzo
TOTAL			02:10:25	235.70GB	

Detailed backup logs:

```
vzdump 100 103 --mode stop --compress lzo --mailnotification always --mailto
monitor@informaticasa.es --storage Backups --node HServer01 --quiet 1
```

```
100: jul 06 00:00:01 INFO: Starting Backup of VM 100 (lxc)
100: jul 06 00:00:01 INFO: status = running
100: jul 06 00:00:01 INFO: backup mode: stop
100: jul 06 00:00:01 INFO: ionice priority: 7
100: jul 06 00:00:02 INFO: stopping vm
100: jul 06 00:00:17 INFO: creating archive '/mnt/pve/Backups/dump/vzdump-lxc-
100-2016_07_06-00_00_01.tar.lzo'
100: jul 06 01:33:20 INFO: Total bytes written: 231008225280 (216GiB, 40MiB/s)
100: jul 06 01:33:21 INFO: archive file size: 194.65GB
100: jul 06 01:33:21 INFO: delete old backup '/mnt/pve/Backups/dump/vzdump-
lxc-100-2016_06_22-00_00_04.tar.lzo'
100: jul 06 01:33:51 INFO: restarting vm
100: jul 06 01:33:55 INFO: vm is online again after 5633 seconds
100: jul 06 01:33:55 INFO: Finished Backup of VM 100 (01:33:54)

103: jul 06 01:33:56 INFO: Starting Backup of VM 103 (qemu)
103: jul 06 01:33:56 INFO: status = running
103: jul 06 01:33:57 INFO: update VM 103: -lock backup
103: jul 06 01:33:57 INFO: backup mode: stop
103: jul 06 01:33:57 INFO: ionice priority: 7
103: jul 06 01:34:01 INFO: stopping vm
103: jul 06 01:34:25 INFO: creating archive '/mnt/pve/Backups/dump/vzdump-
qemu-103-2016_07_06-01_33_55.vma.lzo'
103: jul 06 01:34:25 INFO: starting kvm to execute backup task
103: jul 06 01:34:27 INFO: started backup task '41ccc1bc-8175-4cd7-8541-
e00fd5075046'
103: jul 06 01:34:27 INFO: resume VM
103: jul 06 01:34:30 INFO: status: 0% (250871808/128849018880), sparse 0%
(89190400), duration 3, 83/53 MB/s
103: jul 06 01:34:56 INFO: status: 1% (1312620544/128849018880), sparse 0%
(100917248), duration 29, 40/40 MB/s
103: jul 06 01:35:39 INFO: status: 2% (2586509312/128849018880), sparse 0%
(124096512), duration 72, 29/29 MB/s

…

103: jul 06 02:09:57 INFO: status: 99% (127593873408/128849018880), sparse 52%
(67744210944), duration 2130, 84/0 MB/s
103: jul 06 02:10:18 INFO: status: 100% (128849018880/128849018880), sparse
53% (68999348224), duration 2151, 59/0 MB/s
103: jul 06 02:10:18 INFO: transferred 128849 MB in 2151 seconds (59 MB/s)
103: jul 06 02:10:18 INFO: archive file size: 41.05GB
103: jul 06 02:10:18 INFO: delete old backup '/mnt/pve/Backups/dump/vzdump-
qemu-103-2016_06_22-01_32_04.vma.lzo'
103: jul 06 02:10:26 INFO: vm is online again after 2185 seconds
103: jul 06 02:10:26 INFO: Finished Backup of VM 103 (00:36:31)
```

Si el sistema está funcionando correctamente podemos listar el contenido del almacenamiento Backups para ver las copias que hay almacenadas. En el nombre del fichero observamos a qué servidor virtual pertenece y cuándo se fue realizada.

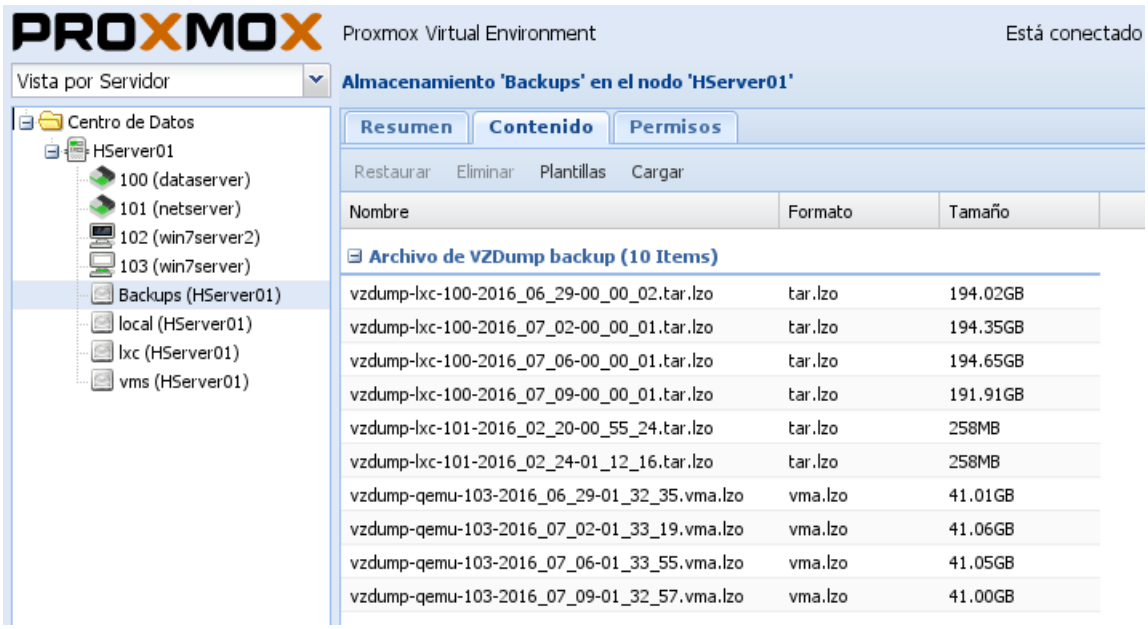


Ilustración 113: Listado de backups almacenados

2. Instalación en lugar de destino

Una vez hemos terminado de preparar el servidor Proxmox en el taller o laboratorio de pruebas llegará el momento de trasladarlo a las instalaciones del cliente. Para ponerlo en producción deberemos adaptarlo a su entorno de red y configurar los puestos para que trabajen con él. Esto se traduce en algunos cambios que hay que tener en cuenta. En primer lugar recordemos la configuración de partida que llevamos de nuestro taller.

```
HOST:         192.168.7.10
       Usuario:           usuario – clave
       Root:              root – claveRoot
       Acceso:            ssh puerto 1707 / proxmox web
 https://192.168.7.10:8006

CT Debian:    192.168.7.11       (dataserver)
       Usuario:           usuario – clave
       Root:              root – claveRoot
       Acceso:            ssh puerto 1707 / consola proxmox

CT Debian:    192.168.7.12       (netserver)
       Usuario:           usuario – clave
       Root:              root – claveRoot
       Acceso:            ssh puerto 1707 / consola proxmox

VM Windows:   192.168.7.13       (winserver)
       Usuario:           Usuario – usuario
       Acceso:            escritorio remoto / consola proxmox
```

Para adaptar nuestro sistema a la subred del cliente tendremos que modificar las siguientes configuraciones.

2.1. Cambios en el anfitrión Proxmox

- Editar **/etc/network/interfaces** para la reconfiguración IP.
- Editar ***/etc/hosts*** para actualizar la resolución de nombres a la nueva configuración IP.
- Editar ***/etc/postfix/main.cf*** para que postfix se ejecute correctamente en el entorno de red nuevo,
- Editar ***/usr/local/sbin/backupWinVM*** (si lo estamos usando) y reajustar con la nueva IP del servidor virtual Windows.

2.2. Cambios en los contenedores LXC

- Editar ***/etc/network/interfaces*** para la reconfiguración IP.
- Editar ***/etc/samba/smb.conf*** para SAMBA, grupo de trabajo, escucha en la nueva subred, etc.
- En el caso de netserver puede ser necesario editar ***/etc/openvpn/server.conf*** para "empujar" la ruta hacia la nueva subred a los clientes si no la dejamos preparada en la instalación inicial.

2.3. Cambios en las máquinas Qemu KVM

- Configuración IP de Windows.
- Grupo de trabajo y recursos compartidos en red del cliente.
- Reconectar unidades de red de dataserver.
- Instalación de impresoras.

2.4. Configuración de los puestos de trabajo

- Traspasar los scripts .bat para conectar las unidades de red en dataserver. Configurar su ejecución automática en cada inicio del equipo. Ejecutar y comprobar.
- Acceso de escritorio remoto a los puestos que lo requieran.

2.5. Configuraciones generales del entorno

2.5.1. Configuración del router de acceso a internet

Si vamos a trabajar con la VPN o queremos gestionar el servidor Proxmox desde el exterior no debemos olvidar configurar el router de la red. Necesitamos activar el reenvío de los puertos usados y salvar el NAT. Tendremos como mínimo que abrir los siguientes:

- Puerto 8006 TCP → IP del anfitrión PROXMOX, para gestión WEB.
- Puertos 5900-5999 TCP → IP del anfitrión PROXMOX, para consolas VNC.
- Puerto 3128 TCP → IP del anfitrión PROXMOX, para consolas spice.
- Puerto 1707 TCP → IP del anfitrión PROXMOX, para gestión por SSH (recordemos que habíamos modificado el puerto, el estándar sería el 22).
- Es recomendable verificar en https://pve.proxmox.com/wiki/Ports los puertos usados. Pueden cambiar según la versión de Proxmox VE.
- Puerto 1194 UDP → IP de netserver para la VPN.

2.5.2. Archivado de una imagen-clon con las particiones del anfitrión Proxmox

Un punto básico para la prevención de desastres en el diseño de nuestro entorno es la creación de una imagen de la instalación del anfitrión que permita su posterior restauración. Una vez hayamos trasladado el equipo a su entorno final y comprobado su correcto funcionamiento nos apoyaremos en la herramienta Clonezilla[52] (o alguna similar) para hacerlo. Esto tiene sentido porque la variabilidad de los datos en el sistema anfitrión será mínima. Salvo los registros de eventos y actualizaciones o cambios que hagamos de forma manual (y si son de importancia siempre podemos crear una nueva imagen), el trabajo del usuario está limitado a las máquinas virtuales. De ellas ya tenemos copias de seguridad periódicas.

Usando Clonezilla archivaremos una copia de las particiones ***/boot, / (raíz) y /var.*** Aunque su tamaño era de 1GB, 9GB y 4,5GB respectivamente, podemos observar que el uso del espacio en ellas es mínimo (1,8GB + 68MB + 499MB = 2,4GB aproximadamente), por lo que para almacenarla podríamos usar incluso un DVD.

```
usuario@HServer01:~$ df -h
S.ficheros                          Tamaño Usados  Disp Uso% Montado en
...
/dev/dm-0                             9,1G   1,8G  6,8G  21% /
```

52 http://clonezilla.org/

```
...
/dev/md0                                923M    68M  792M   8% /boot
...
/dev/mapper/SistemaHost-var             4,5G   499M  3,8G  12% /var
...
```

El coste de implementar esta medida es prácticamente nulo y nos será de gran utilidad en caso, por ejemplo, de un fallo simultáneo de los dos discos duros. Podremos restaurar el estado y las configuraciones del anfitrión a unos nuevos discos con un simple volcado de la imagen.

2.5.3. Documentación del entorno

Aunque lo más común es que se olvide (o que se posponga para más adelante...), una fase vital en una instalación profesional debe ser siempre la de **documentar**. La creación de un documento con diagramas, inventario hardware, configuraciones realizadas, usuarios y contraseñas, detalle y programación de las copias de seguridad, etc. es esencial. Facilitará las tareas de administración y permitirá a cualquier técnico trabajar en el entorno aunque no sea el instalador inicial.

3. Procedimientos de actuación

3.1. Sustitución de un disco duro averiado

En este proceso se describen los pasos que deberemos seguir para cambiar un disco duro, ya sea porque se ha estropeado o porque tiene demasiadas horas de uso y queremos reemplazarlo de forma preventiva. Las dos claves del proceso son la copia de la tabla de particiones a partir del disco que ha quedado activo y la restauración del RAID en espejo.

3.1.1. ¿Cómo sabemos si un disco ha fallado?

Para darnos cuenta de que un disco ha fallado o está a punto de hacerlo tenemos varios medios:

- Informes automáticos por email de mdadm (fallos en el RAID).
- Informes automáticos por email de smartmoontools (problemas físicos en el disco).
- Informes programados con el resultado del script estadoHDDs.
- Consulta manual de mensajes de error en el registro del sistema (***/var/log/messages*** y ***/var/log/syslog***).

- Consulta manual del estado del RAID en ***/proc/mdstat***. En el estado observaremos [U_] si alguno de los componentes ha fallado y el espejo está degradado.

```
root@HServer01:~$ cat /proc/mdstat
Personalities : [raid1]

md2 : active raid1 sda3[0] sdb3[2]
      937568064 blocks super 1.2 [2/2] [UU]

md1 : active raid1 sda2[0] sdb2[2]
      38052736 blocks super 1.2 [2/2] [UU]

md0 : active (auto-read-only) raid1 sda1[0] sdb1[2]
      975296 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

3.1.2. Configuración de partida

Recordemos la configuración del equipo. Tenemos dos discos duros instalados, ***/dev/sda*** *y* ***/dev/sdb***, particionados según el siguiente esquema:

```
/dev/sda1   |   /dev/sda2   |   /dev/sda3
/dev/sdb1   |   /dev/sdb2   |   /dev/sdb3
```

Y con esta configuración RAID:

```
(RAID1) /dev/sda1 + /dev/sdb1 => /dev/md0
(RAID1) /dev/sda2 + /dev/sdb2 => /dev/md1
(RAID1) /dev/sda3 + /dev/sdb3 => /dev/md2
```

Supongamos de aquí en adelante que el disco que ha fallado y que queremos reemplazar es **/dev/sdb.** Para /dev/sda el proceso sería similar.

3.1.3. Paso 1: marcar el disco como defectuoso

En primer lugar debemos etiquetar los bloques /dev/sdb1 /dev/sdb2 y /dev/sdb3 como dañados en la configuración de mdadm.

Si el disco sigue instalado:

```
mdadm --manage /dev/md0 --fail /dev/sdb1
mdadm --manage /dev/md1 --fail /dev/sdb2
mdadm --manage /dev/md2 --fail /dev/sdb3
```

Si el disco ya había sido extraído del equipo, el comando siguiente marcará los bloques desconectados como fallidos:

```
mdadm --manage /dev/md0 --fail detached
mdadm --manage /dev/md1 --fail detached
mdadm --manage /dev/md2 --fail detached
```

El contenido de /proc/mdstat ahora mostrará algo así:

```
cat /proc/mdstat
```

```
Personalities : [raid1]
md0 : active raid1 sda1[0] sdb1[2](F)
      24418688 blocks [2/1] [U_]
md1 : active raid1 sda2[0] sdb2[2](F)
      24418688 blocks [2/1] [U_]
md2 : active raid1 sda3[0] sdb3[2](F)
      24418688 blocks [2/1] [U_]
unused devices: <none>
```

3.1.4. Paso 2: eliminar /dev/sdb1[sdb2][sdb3] de los espejos /dev/md0[md1][md2]

```
mdadm --manage /dev/md0 --remove /dev/sdb1
```

```
mdadm: hot removed /dev/sdb1
```

```
mdadm --manage /dev/md1 --remove /dev/sdb2
```

```
mdadm: hot removed /dev/sdb2
```

```
mdadm --manage /dev/md2 --remove /dev/sdb3
```

```
mdadm: hot removed /dev/sdb3
```

El contenido de /proc/mdstat ahora será:

```
cat /proc/mdstat
```

```
Personalities : [raid1]
md0 : active raid1 sda1[0]
      24418688 blocks [2/1] [U_]
md1 : active raid1 sda2[0]
      24418688 blocks [2/1] [U_]
unused devices: <none>
```

3.1.5. Paso 3: apagar el equipo, extraer el disco deseado e instalar el nuevo

3.1.6. Paso 4: clonar la tabla de particiones

- Para tablas tipo **MBR – DOS**

Usaremos la utilidad **sfdisk**, que nos permite volcar la configuración de un disco y replicarla directamente en el otro.

```
sfdisk -d /dev/sda | sfdisk /dev/sdb
```

Podemos ejecutar después ***fdisk -l*** para comprobar si ahora ambos discos presentan el mismo esquema.

- Para tablas tipo **GPT.**

En este caso **sfdisk** no es compatible y debemos trabajar con **sgdisk**. La sintaxis es algo diferente y ***debemos ser muy cuidadosos***: el orden de los argumentos para el comando no es el típico. ***El origen con la tabla de particiones correcta se escribe en segundo lugar.***

Primero haremos una copia de seguridad de la tabla correcta:

```
sgdisk --backup=gptTableBack /dev/sda
```

A continuación replicamos la tabla al disco nuevo:

```
sgdisk -R[DESTINATION] [SOURCE]
sgdisk -R /dev/sdb /dev/sda
```

Por último debemos modificar los UIDS de las particiones para que sean diferentes a las de sda. Este comando los modifica por unos aleatorios:

```
sgdisk -Gg [DESTINATION]
sgdisk -Gg /dev/sdb
```

3.1.7. Paso 5: reactivar los espejos

Añadimos los bloques sdb1, sdb2 y sdb3 a su respectivo RAID.

```
mdadm --manage /dev/md0 --add /dev/sdb1
mdadm: re-added /dev/sdb1
```

```
mdadm --manage /dev/md1 --add /dev/sdb2
mdadm: re-added /dev/sdb2
```

```
mdadm --manage /dev/md2 --add /dev/sdb3
mdadm: re-added /dev/sdb3
```

En este momento los espejos empezarán a sincronizarse. En /proc/mdstat podemos observar el proceso para controlar cuándo termina la operación.

```
cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sda1[0] sdb1[1]
      24418688 blocks [2/1] [U_]
      [=>...................]  recovery =  9.9% (2423168/24418688)
finish=2.8min speed=127535K/sec
md1 : active raid1 sda2[0] sdb2[1]
      24418688 blocks [2/1] [U_]
      [=>...................]  recovery =  6.4% (1572096/24418688)
finish=1.9min speed=196512K/sec
md2 : active raid1 sda3[0] sdb3[1]
      24418688 blocks [2/1] [U_]
      [=>...................]  recovery =  6.1% (1572096/24418688)
finish=1.9min speed=196512K/sec
unused devices: <none>
```

3.1.8. Paso 6: reinstalar el cargador de arranque

Necesitamos recargar GRUB en el disco nuevo, para en caso de fallo ahora de sda, sdb también sea capar de arrancar el sistema. Con el siguiente comando actualizamos la configuración y recargamos en ambos discos:

```
grub-install /dev/sdb && grub-install /dev/sda && update-grub
```

3.2. Restauración de una copia de seguridad

Si hemos completado la configuración del entorno siguiendo este documento la carga de una copia de seguridad es realmente sencilla. Tenemos dos formas de hacerlo: accediendo desde el almacenamiento de backups o accediendo desde la gestión de la misma máquina que queremos restaurar. La primera está pensada para clonar una copia de un servidor virtual en una instancia nueva y la segunda para sobreescribir los datos actuales por los del backup.

3.2.1. Carga de un backup en una nueva máquina

Seleccionando en el menú de la izquierda el almacenamiento que contiene los backups y listando el contenido podremos activar el botón Restaurar. Simplemente elegimos en qué almacenamiento ubicaremos la nueva máquina y qué ID le asignaremos. Una vez se complete el proceso de extracción contaremos con un nuevo servidor virtual exactamente igual al de la fecha de la copia de seguridad.

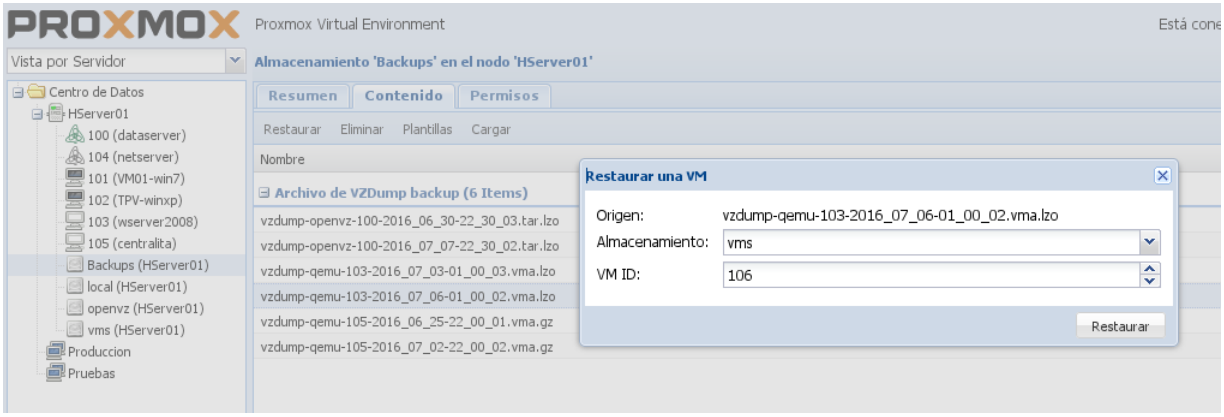


Ilustración 114: Carga de un backup en un nuevo servidor virtual

3.2.2. Carga de un backup sobreescribiendo el estado actual

Si por el contrario deseamos revertir el estado de nuestra máquina al que tuviera en un momento anterior, accederemos al proceso de restauración de la copia desde su página de gestión. Seleccionamos la que queramos restaurar en el menú del Centro de Datos y nos vamos a la pestaña Respaldo. Ahí aparecerán las diferentes copias disponibles.

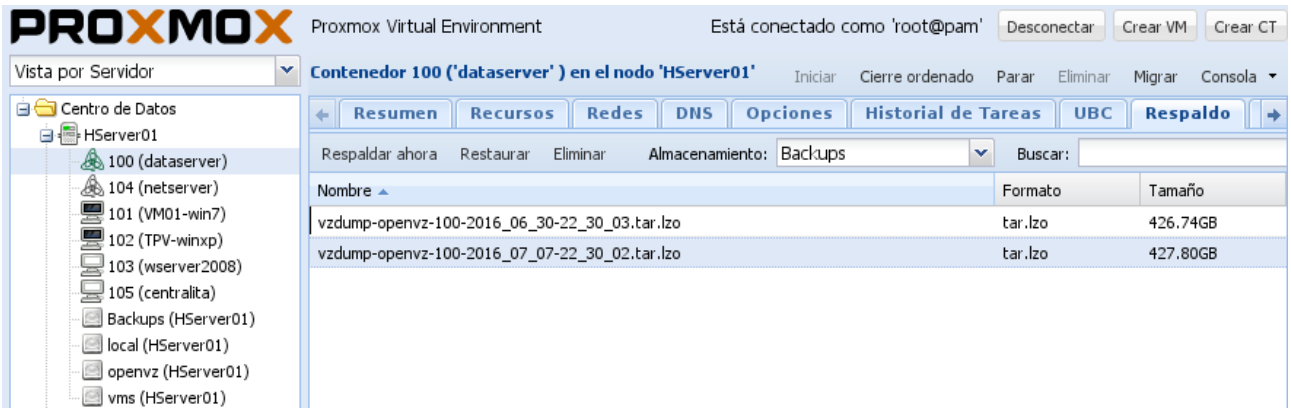


Ilustración 115: Restauración de un servidor virtual a un estado anterior

A continuación seleccionamos ***Restaurar*** y podremos comenzar el proceso. La siguiente captura muestra un ejemplo en el que restauraríamos el container LXC dataserver al estado que presentaba el día 09/07 a las 00:01 horas.

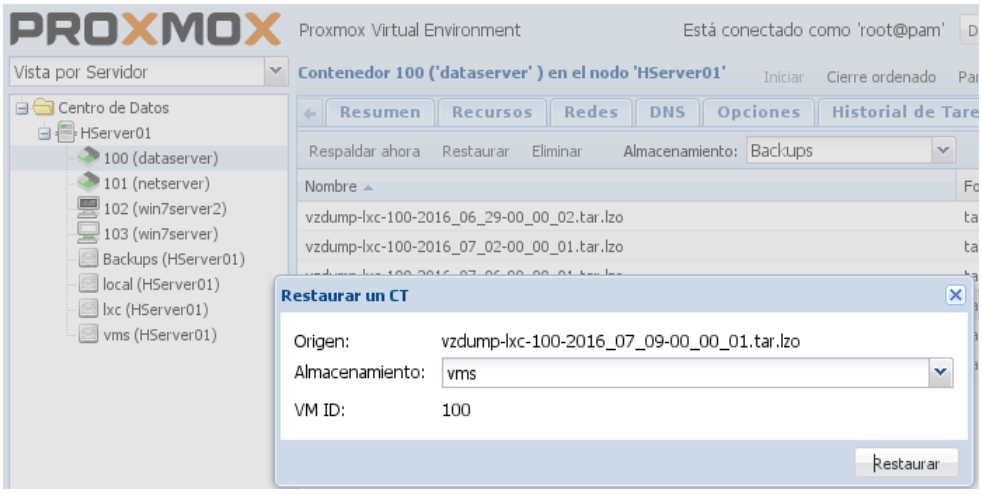


Ilustración 116: Restauración de un servidor virtual a un estado anterior II

3.3. Ampliación del espacio de almacenamiento

Si nuestra empresa agota el espacio planificado inicialmente, por ejemplo, los 600GB de dataserver, necesitaremos añadir discos duros para aumentar este tamaño. Gracias a LVM la tarea es simple: extender su disco virtual. Los discos duros habrá que añadirlos en parejas para no anular la protección del RAID en espejo. Para aumentar 2TB deberemos añadir dos unidades de 2TB. Es muy importante no olvidar esto, porque si se agregan volúmenes físicos desde un disco independiente a los grupos VirtDatos o SistemaHost añadimos un punto único de fallo. Para ampliar correctamente se debe configurar la nueva pareja con otro RAID1 y desde ahí añadir volúmenes físicos a los grupos LVM. El proceso será similar al seguido en la instalación inicial. Supongamos que los discos añadidos son mapeados en **/dev/sdd** y **/dev/sde** respectivamente.

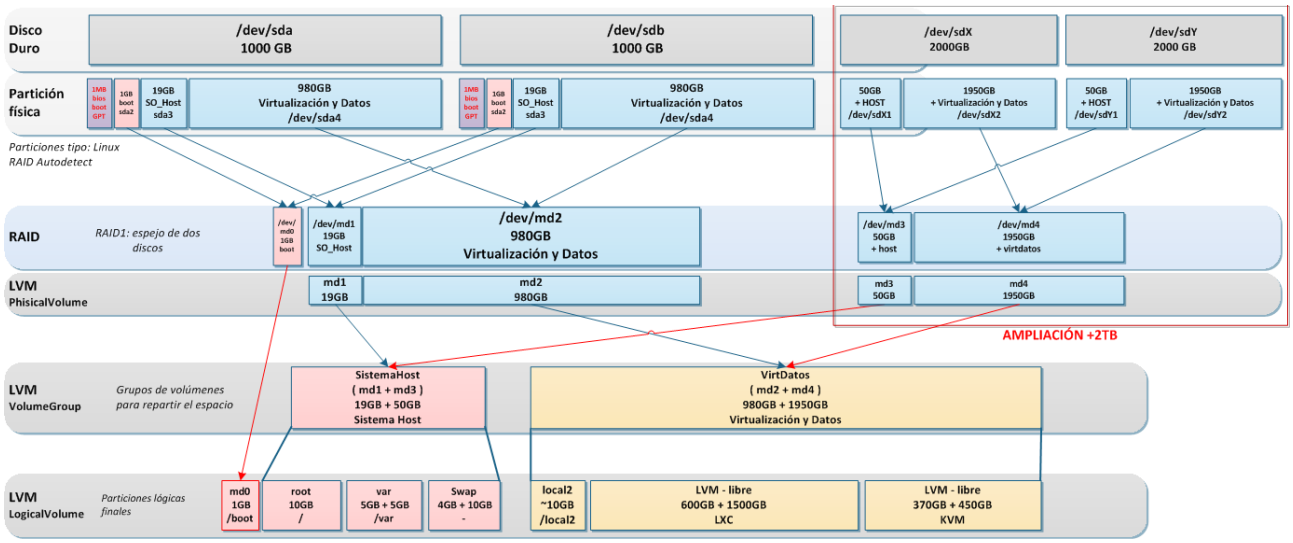


Ilustración 117: Esquema de particionado para ampliar almacenamiento

1. Particionar en bloques tipo linux RAID según el tamaño que queramos añadir a los grupos SistemaHost o Virtdatos. Si solo queremos añadir espacio a Virtdatos (que es lo más probable) solo necesitaremos uno. Si queremos más flexibilidad podemos crear varios más pequeños para repartir mejor el espacio. Lo haremos con **fdisk** en uno de los discos y clonaremos la tabla de particiones al otro. En nuestro ejemplo crearemos un único bloque para extender el espacio de dataserver.

```
fdisk /dev/sdd
```

```
n → crear nueva partición
P → tipo primaria
1 → partición número 1Next select the partition number as 1.
[Enter] → Aceptamos el tamaño completo
p → imprime la partición definida
L → lista los tipos de particiones conocidas
```

```
t → estableceremos el tipo de nuestra partición
valor fd → tipo linux RAID
w → guardar cambios y salir
```

Copiamos el esquema particionado a /dev/sde:

```
sfdisk -d /dev/sdd | sfdisk /dev/sde
```

2. Crear el espejo con los dispositivos anteriores:

```
mdadm --create --level=raid1 /dev/md3 --raid-devices=2 /dev/sdd1 /dev/sde1
```

3. Activar el nuevo volumen físico de LVM sobre md3:

```
pvcreate /dev/md3
```

4. Añadir el volumen físico al grupo que ganará el espacio. VirtDatos en nuestro caso:

```
vgextend VirtDatos /dev/md3
```

5. Extender el volumen lógico añadiendo el espacio deseado. Para dataserver tenemos que ampliar el volumen /dev/VirtDatos/vm-100-disk-1. Lo ampliaremos en 1500GB:

```
lvextend -L +1500G /dev/VirtDatos/vm-100-disk-1
```

6. Redimensionar el sistema de ficheros sobre el volumen lógico para que se adapte al nuevo tamaño de disco y el contenedor sea consciente del espacio añadido:

```
resize2fs /dev/VirtDatos/vm-100-disk-1
```

Capítulo VII: Conclusiones y líneas de trabajo futuro

1. Conclusiones

El objetivo que nos habíamos propuesto al inicio de este trabajo era cubrir una necesidad detectada en el entorno de las PYMES (podemos incluir aquí también administraciones públicas, asociaciones y otras organizaciones). En nuestro caso hemos observado el problema en la zona de la Sierra de Aracena y Picos de Aroche, pero parece lógico y de sentido común pensar que será algo generalizado en áreas rurales y pequeñas y medianas empresas con baja penetración digital de cualquier parte.

Como comentábamos en la introducción de este trabajo, en estas empresas la infraestructura informática y de comunicaciones suele ser muy deficiente, fuente de problemas constantes y se percibe más como un inconveniente que como una herramienta para optimizar el trabajo y ser más eficiente. Por otro lado, no cuentan con un presupuesto adecuado para mejorarlas y además, si lo tienen, ven esa inversión como algo secundario, innecesario y que no merece la pena. Ser capaz de transmitirles lo que pueden ganar con el gasto en una buena base informática no es ni mucho menos una tarea fácil.

Para tratar de resolver esta situación hemos intentado diseñar una infraestructura muy económica, multiusos, sólida, fiable, y adaptable a distintas situaciones. Para ello hemos aplicado técnicas de virtualización de última generación, múltiples herramientas de software libre y trabajado con conceptos avanzados de almacenamiento y redes. El objetivo ha sido acercar los

últimos avances tecnológicos en la telemática y crear herramientas realmente útiles para los usuarios más alejados y que menos se benefician de ellos. Aprovechar técnicas avanzadas adaptando la tecnología para crear un sistema sencillo, comprensible y sobre todo, utilizable por el usuario final, para que así, éste pueda realmente valorar y ser consciente de todo lo que le aporta.

Como resumen del diseño y como apoyo para transmitir lo que ofrece nuestro trabajo al potencial usuario se ha creado el siguiente esquema:

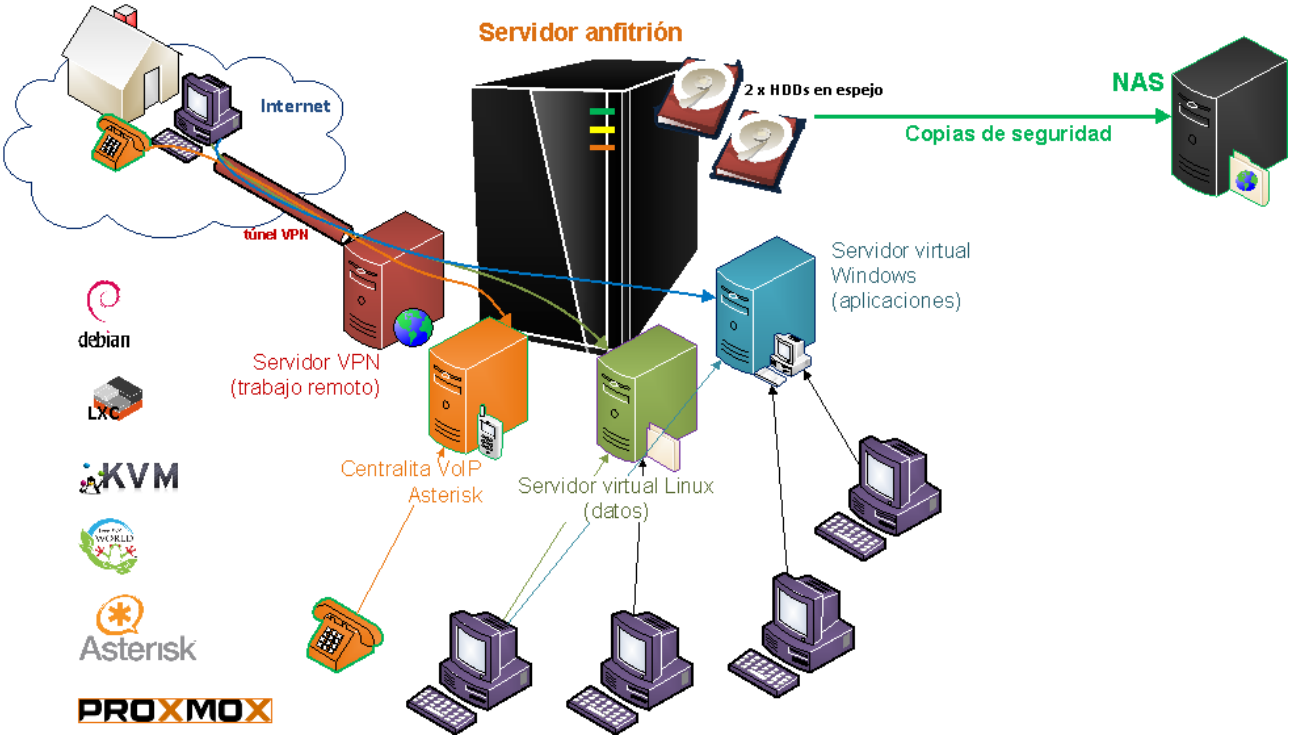


Ilustración 118: Resumen del diseño del entorno de trabajo

1.1. Presupuesto

Puesto que uno de los objetivos principales era que la solución fuera económica, lo más ajustada posible a la realidad actual de la empresa pequeña, veamos una estimación aproximada del coste real que nos supondría el sistema que hemos trabajado aquí (Servidor HP Proliant ML310 con un procesador de 4 núcleos, 8GB de RAM y dos discos duros de 1TB + NAS de dos bahías y 1TB para copias) . Así podremos valorar si realmente lo hemos conseguido.

Descripción	Cantidad	Precio unitario	Total
SERVIDOR HP Proliant ML310E G8 v2 (470065-800) Ya incluye 2 discos duros de 1TB	1	750€	750€
DISPOSITIVO PARA COPIAS DE SEGURIDAD NAS DLINK DNS-320L 2 BAHÍAS	1	85€	85€
DISCO DURO NAS HDD WD RED 1TB SATA3 NAS 3,5"	1	75€	75€
SAI SAI SALICRU SPS SOHO+ 1000VA 600W	1	140€	140€
ACCESORIOS DE MONTAJE LATIGUILLOS, CABLEADO, PERIFÉRICOS...	1	20€	20€
		TOTAL	**1070€**

Tabla 14: Estimación del coste del diseño propuesto

Este presupuesto contiene solo nuestros costes materiales. Para el usuario final siempre habrá que sumar las licencias de Windows y otro software de pago (si procede), el valor del servicio técnico por el diseño, preparación, configuración de los equipos y la instalación a domicilio. Este cálculo lo dejamos al criterio profesional de cada uno.

1.2. Verificación de la solución. ¿Qué hacer si…?

Un forma de poner a prueba y comprobar si la solución que hemos desarrollado es válida y cumple con los objetivos que nos habíamos propuesto de flexibilidad, redundancia, escalabilidad, o seguridad, es intentar anticipar las situaciones y los problemas que pueden surgir (y que surgirán) y cómo los afrontaremos gracias a nuestro diseño. Algunos de los procedimientos concretos para llevar a cabo las operaciones necesarias en esos casos ya los hemos visto detallados en el Capítulo 6. Veamos por tanto nuestro ***¿qué hacer si...?***

- Fallo de un disco duro → gracias a la configuración en RAID-1 no habrá pérdida de datos. Se sustituye el disco averiado por uno nuevo y se recompone el espejo.
- Fallo simultáneo de los dos discos duros → aunque es poco probable, entrará en juego la imagen clonada del sistema anfitrión que creamos y archivamos al completar la instalación en destino. Instalamos dos discos duros nuevos, cargamos la imagen y restauramos las últimas copias de seguridad disponibles.
- Fallo hardware completo del servidor (placa base, procesador…), suponiendo que no hay pieza de recambio y no es reparable. Buscamos un nuevo equipo de arquitectura y características similares y le instalamos los discos duros. GNU Linux utiliza un kernel

genérico que suele ser compatible con casi todas las plataformas, probablemente el sistema arrancará y funcionará. Si no es así siempre podemos reinstalar de nuevo el entorno Proxmox y luego cargar las máquinas virtuales desde los backups, recordemos que toda la información vital se gestiona en ellas y nunca en el host.

- Error humano, borrado de ficheros → recurrimos a las copias de seguridad.
- Fallo en algún puesto de trabajo → la información vital está en los servidores virtuales. Los puestos deberán ser clientes ligeros que podremos sustituir fácilmente con una mínima configuración.
- Nos quedamos sin espacio y necesitamos ampliar el almacenamiento → añadimos una pareja de discos con el espacio deseado siguiendo el procedimiento escrito en el capítulo anterior.
- La empresa cambia de software de gestión y requiere un nuevo sistema operativo (por ejemplo, trabajaban sobre XP y deben actualizar a Windows 10)→ creamos una nueva máquina virtual con el S.O. necesario y podemos hacer una migración suave, sin parar de trabajar con el servidor virtual antiguo hasta que completemos el proceso. Luego, si no se necesita más, podemos eliminar la antigua máquina.
- La empresa se expande y sus requisitos de rendimiento y fiabilidad aumentan → añadiremos otro servidor (y los que sean necesarios) con la potencia deseada al cluster. Si necesitamos que el entorno nunca pare aunque uno de los nodos falle, podemos integrar almacenamiento compartido de alto rendimiento y repartir la carga con balanceo y traspaso de máquinas en caliente.

Viendo la capacidad de respuesta que hemos logrado ante múltiples imprevistos y la flexibilidad para escalar y adaptarnos a futuros cambios, todo ello con un escaso presupuesto de 1070€, parece lógico concluir que hemos alcanzado de forma más que satisfactoria los objetivos marcados.

Quizás, una prueba más concluyente de que el desarrollo ha sido exitoso y el sistema cumple con las expectativas, es el hecho de que, a fecha de julio 2016, se han desplegado con acierto unos 15 servidores Proxmox basados en este trabajo. Los primeros prototipos llevan ya más de dos años en producción comportándose como se esperaba, y han resultado vitales para la recuperación de numerosos desastres y minimizar su impacto. Los usuarios han sabido

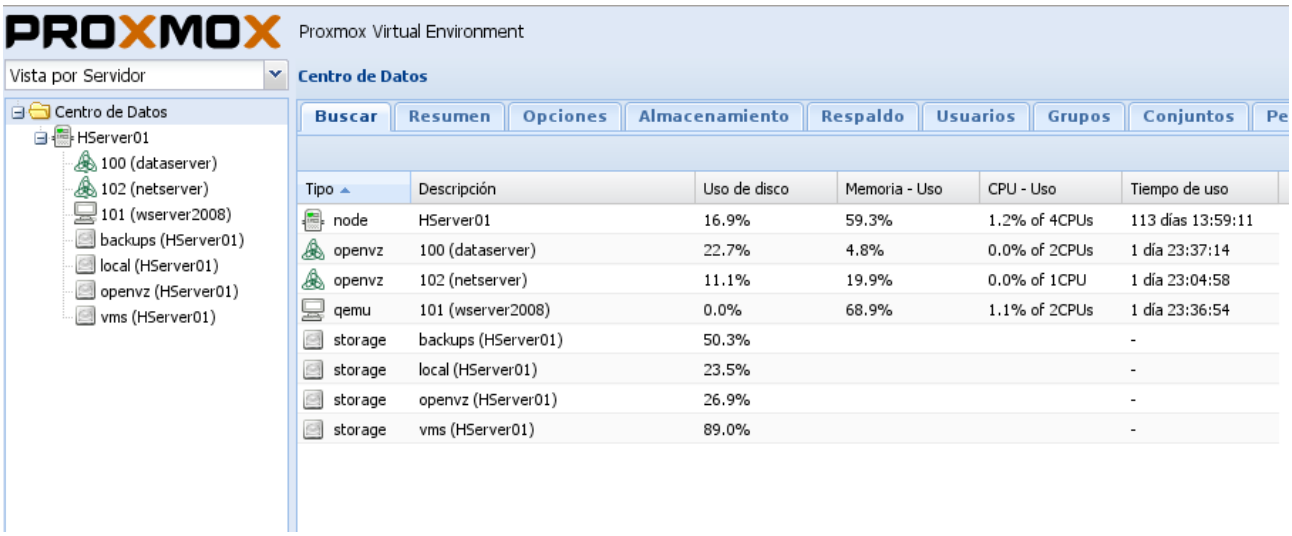


Ilustración 119: Servidor Proxmox en producción,113 días encendido

adaptarse al modo de trabajo propuesto y aprovechar sus ventajas. Las siguientes capturas muestran algunos de ellos en funcionamiento.

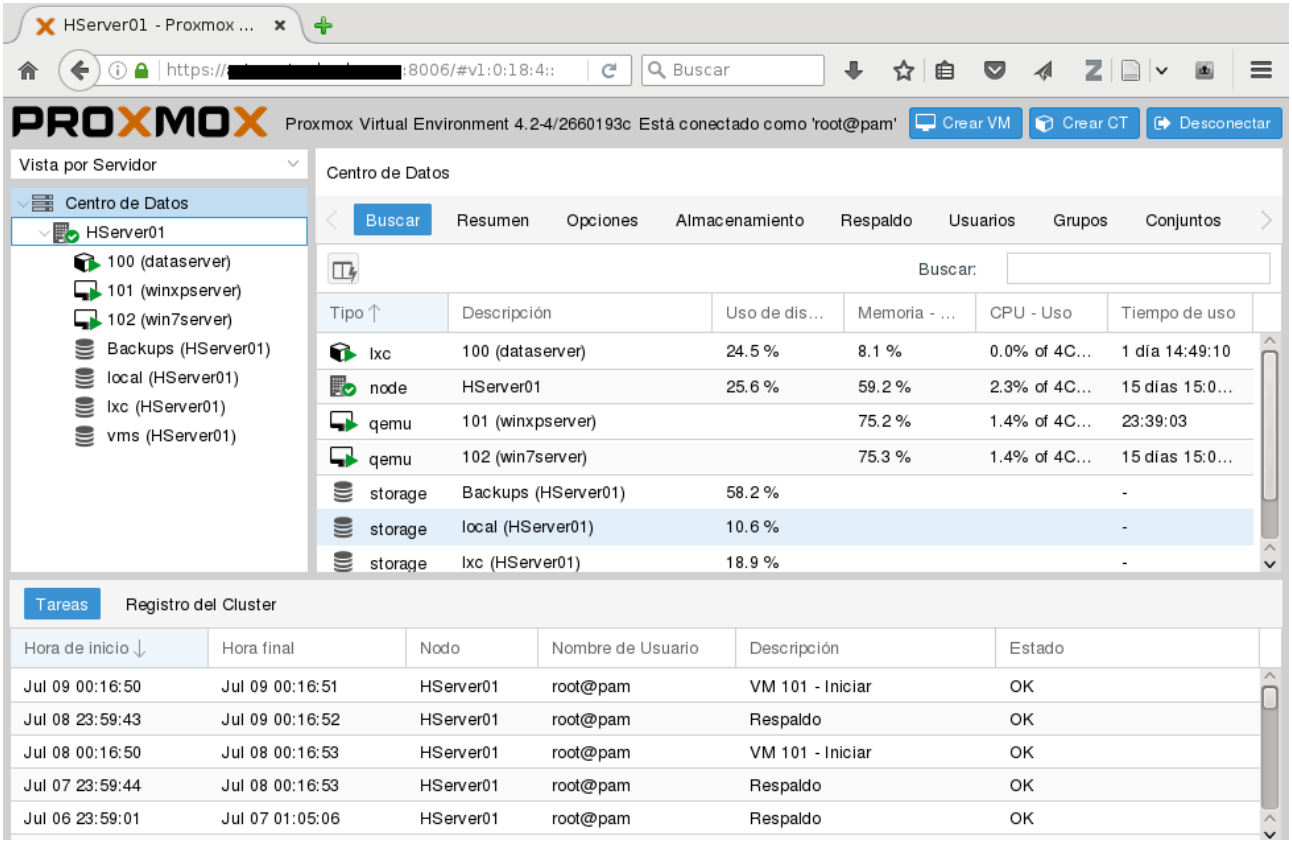


Ilustración 120: Servidor Proxmox en producción. Versión 4.2.4 con GUI actualizada

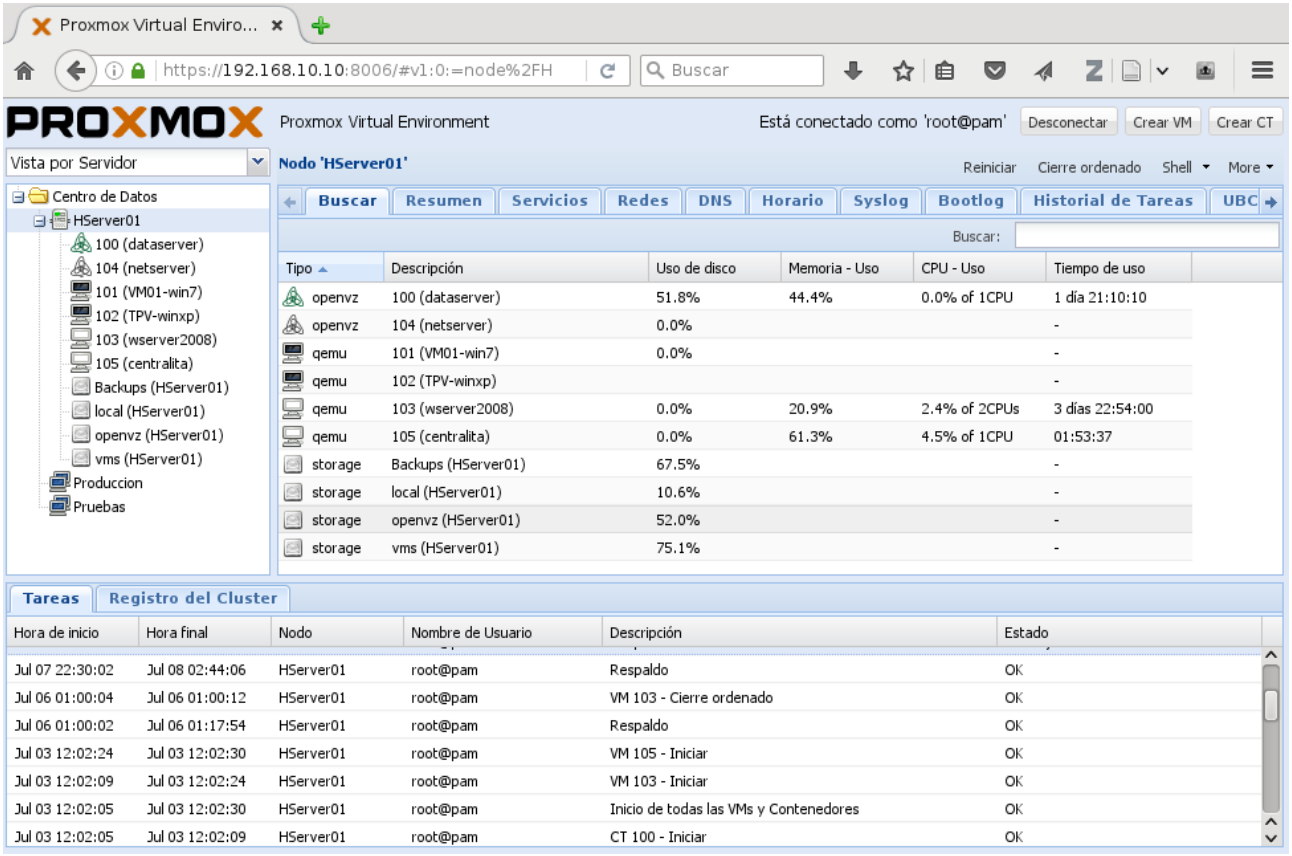


Ilustración 121: Servidor Proxmox en producción - Informaticasa

2. Líneas de trabajo futuro

Limitar la extensión del presente documento y saber ponerle fin ha sido un problema bastante importante que el autor ha tenido que enfrentar. Como el desarrollo de este trabajo, el diseño del sistema con sus pruebas, la investigación y la pelea diaria con Proxmox VE, se ha prolongado durante varios años, ha habido tiempo para comprender su potencial y observar todas las opciones que ofrece. Ha sido casi constante la introducción de modificaciones y mejoras conforme se iba profundizando en el manejo de las herramientas. Los cambios en las sucesivas versiones de Proxmox han ido abriendo nuevas vías muy tentadoras que amenazaban constantemente la finalización del proyecto. Algunas de estas "amenazas", nuevas aplicaciones e ideas que han ido surgiendo son las siguientes:

- Documentar la creación y configuración de una centralita virtual Asterisk para integrar las comunicaciones telefónicas de la empresa. Ahorraremos costes, modernizaremos el sistema con todas las ventajas de la voip y sacaremos más rendimiento a nuestro servidor.
- Mejorar el sistema de copias de seguridad que trae Proxmox de serie añadiendo la capacidad de realizar backups diferenciales. Así optimizaremos el uso del espacio en el dispositivo de copias y reduciremos el tiempo de creación. La URL siguiente lleva a un proyecto donde desarrollan parches para las distintas versiones de Proxmox y las instrucciones para implementar esta característica: https://ayufan.eu/projects/proxmox-ve-differential-backups/
- Trabajar con las configuraciones en clúster de Proxmox. Un buen sistema de alta disponibilidad o un simple clúster de dos servidores donde uno funciona como respaldo y almacén de copias de seguridad son soluciones que cubrirán los requisitos de infraestructuras más amplias y exigentes y nos permitirán abarcar más escenarios. En los libros Mastering Proxmox (Ahmed 2014) o Proxmox High Availability (Cheng 2014) podemos encontrar bastante información para profundizar en este aspecto.
- Investigar y sacar rendimiento de los últimos avances en los sistemas de almacenamiento distribuidos. En las últimas versiones de Proxmox (la imagen muestra un equipo con la 4.2.4) se soportan GlusterFS, RBD o ZFS sobre iSCSI entre otros.

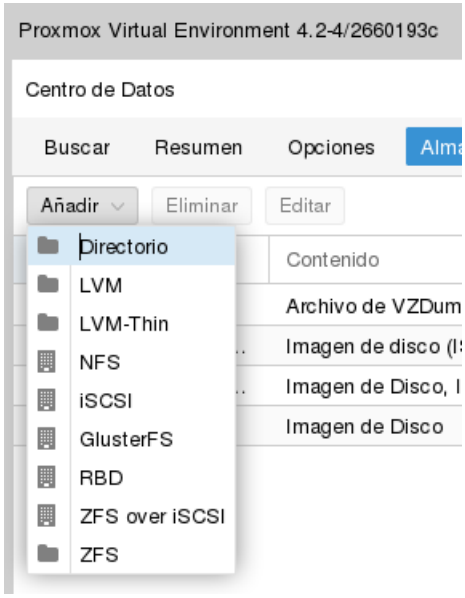


Ilustración 122: Almacenamientos soportados en Proxmox 4.2.4

- Analizar y configurar la herramienta ClusterSSH[53] y la autenticación en SSH mediante clave privada para facilitar/securizar la gestión remota y poder administrar varios servidores al mismo tiempo. Esta utilidad nos permite enviar la misma orden a múltiples equipos de una sola vez:

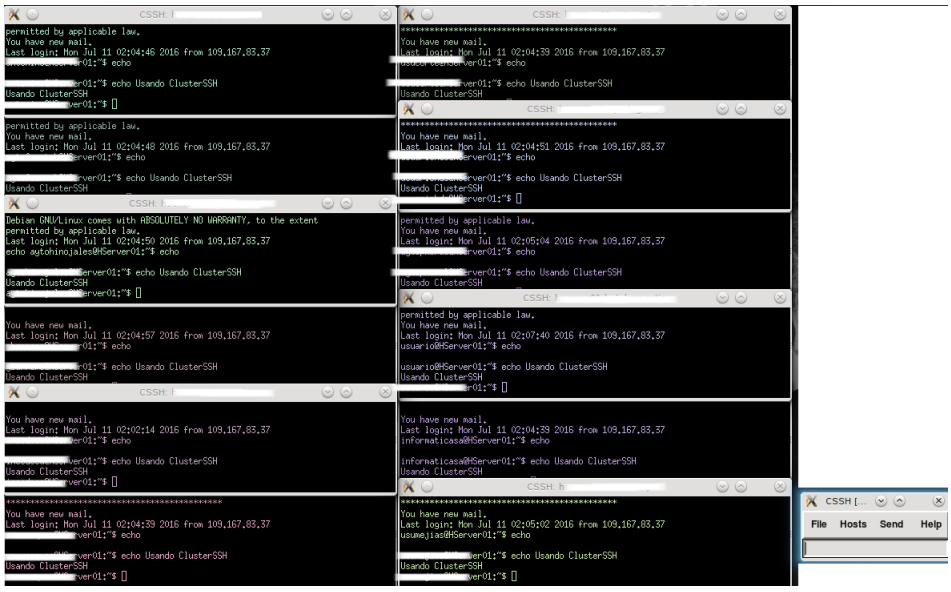


Ilustración 123: Ejemplo de uso de ClusterSSH

- Completar el desarrollo de los scripts de instalación del anfitrión y dataserver para hacerlos más interactivos y reducir aún más la configuración manual. Preparar un

53 https://sourceforge.net/projects/clusterssh/

conjunto de plantillas con máquinas KVM preconfiguradas para acelerar así el proceso de implementación.

- Seguir creando procedimientos de actuación para tareas administrativas. Diseñar una plantilla para generar la documentación de los entornos de trabajo configurados.

Bibliografía

AHMED, W., 2014. *Mastering Proxmox* [en línea]. S.l.: Packt Publishing. ISBN 978-1-78398-083-3. Disponible en: http://0-proquest.safaribooksonline.com.fama.us.es/9781783980826.

ALI, E., SUSANDRI y RAHMADDENI, 2015. Optimizing Server Resource by Using Virtualization Technology. *Procedia Computer Science* [en línea], vol. 59, pp. 320-325. ISSN 1877-0509. DOI 10.1016/j.procs.2015.07.572. Disponible en: http://www.sciencedirect.com/science/article/pii/S1877050915021018.

CHENG, S.M.C., 2014. *Proxmox High Availability* [en línea]. S.l.: Packt Publishing. ISBN 978-1-78398-089-5. Disponible en: http://0-proquest.safaribooksonline.com.fama.us.es/book/operating-systems-and-server-administration/virtualization/9781783980888.

KOVARI, A. y DUKAN, P., 2012. KVM amp; OpenVZ virtualization based IaaS open source cloud virtualization platforms: OpenNode, Proxmox VE. *2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics (SISY)*. S.l.: s.n., pp. 335-339. DOI 10.1109/SISY.2012.6339540.

MORRIS, R.J. y TRUSKOWSKI, B.J., 2003. The evolution of storage systems. *IBM systems Journal* [en línea], vol. 42, no. 2, pp. 205–217. Disponible en: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5386860.

MORRIS, R.J.T. y TRUSKOWSKI, B.J., 2003. The evolution of storage systems. *IBM Systems Journal*, vol. 42, no. 2, pp. 205-217. ISSN 0018-8670. DOI 10.1147/sj.422.0205.

PATTERSON, D.A., GIBSON, G.A. y KATZ, R.H., 1987. A Case for Redundant Arrays of Inexpensive Disks (RAID). [en línea]. S.l.: EECS Department, University of California, Berkeley. UCB/CSD-87-391. Disponible en: http://www.eecs.berkeley.edu/Pubs/TechRpts/1987/5853.html.

ROSEN, R., 2013. Resource management: Linux kernel Namespaces and cgroups. [en línea]. Disponible en: http://haifux.org/lectures/299/netLec7.pdf.

SANTANA, G.A.A., 2013. *Data Center Virtualization Fundamentals: Understanding Techniques and Designs for Highly Efficient Data Centers with Cisco Nexus, UCS, MDS, and Beyond* [en línea]. S.l.: Cisco Press.. ISBN 978-0-13-309643-9. Disponible en: http://0-proquest.safaribooksonline.com.fama.us.es/9780133096439.

Virtualización. En: Page Version ID: 84880893, *Wikipedia, la enciclopedia libre* [en línea], 2015. Disponible en: https://es.wikipedia.org/w/index.php?title=Virtualizaci%C3%B3n&oldid=84880893.

VMware Virtual Networking Concepts. [en línea], 2007. Disponible en: http://www.vmware.com/vmtn/resources/997.

WANG, A., IYER, M., DUTTA, R., ROUSKAS, G.N. y BALDINE, I., 2013. Network Virtualization: Technologies, Perspectives, and Frontiers. *Journal of Lightwave Technology*, vol. 31, no. 4, pp. 523-537. ISSN 0733-8724. DOI 10.1109/JLT.2012.2213796.

WOLF, C. y HALTER, E.M., 2005. *Virtualization: From the Desktop to the Enterprise*. Edición: Har/Cdr. Berkeley, CA : New York, NY: Springer. ISBN 978-1-59059-495-7.

Anexos

Anexo A. Scripts desarrollados para el anfitrión Proxmox

A.1. ConfiguraProxmox: Instalación inicial del sistema

```
#! /bin/bash
#
# SCRIPT: ConfiguraProxmox
# AUTHOR: Carlo de Cástulo Navarro Álvarez
# DATE: 01/12/2015
# REV: 1.1
#
# PURPOSE: realiza la configuración inicial de un servidor Debian +
#       Proxmox en función de las variables establecidas en la siguiente sección.
#          El script funciona en dos fases:
#          Fase1: realiza una configuración básica, instala la mayoría de los paquetes y
#         reinicia el servidor para arrancar con el kernel de Proxmox, necesario para
#         los siguientes pasos.
#          Fase2; termina la instalación de paquetes y la configuración pendiente.
#
# nota: Versión adaptada a Proxmox 4 - Debian Jessie (Cambio de OpenVZ por LXC en containers)
#
#####
# set -n # Uncomment to check script syntax, without execution.
#
# set -x # Uncomment to debug this shell script
##########################################################################


##########################################################################
## CONFIGURACIÓN
##########################################################################
# Nombre de empresa para mostrar información
NOMBRE_EMPRESA="NombreEmpresa"
# Versión de debian usada para la instalación y disponible en los repos de proxmox
# en dic de 2015 -> jessie
DEBIAN_VER=jessie
PROXMOX_VER=4.2.8-1
PROXMOX_KERNEL=4.2.8-1-pve
##########
# Nombre del servidor
NOMBRE_HOST="HServer01"
# Dominio, sufijo con info del lugar
DOMINIO="dominioEmpresa"
# Config de red
LAN_IFAZ=vmbr0
IP_ADDR=192.168.7.10
NETMASK=255.255.255.0
LAN_ADDR=192.168.7.0
LAN_CIDR=192.168.7.0/24
LAN_BCAST=192.168.7.255
GATEWAY=192.168.7.1
DNS_SERVER=192.168.7.1
##########
## Dirección a la que enviar un mail de prueba tras configurar POSTFIX
TEST_MAIL_RCV="carnavalv@gmail.com"
##########################################################################


## A partir de aquí NO suele ser necesario MODIFICAR NADA
# Directorio origen con los scripts personalizados
SCDIR="./Scripts"
[ ! -d "$SCDIR" ] && { echo "error: <${SCDIR}> no encontrado"; exit 1; }
# Destino para instalar los scripts
SCRIPTS_INSTALL="/usr/local/sbin"
# VARIABLES
THIS_SCRIPT=$(basename $0)
ECHO="echo -e"
```

```
FECHA=`date +%Y_%m_%d`
HORA=`date +%H:%M`
################################################################################


################################################################################
## FUNCIONES DE APOYO
################################################################################

########################################
########################################
uso() {
  $ECHO "\n +++++ $THIS_SCRIPT +++++"
  $ECHO " + Uso: $THIS_SCRIPT parte1 | parte2"
  $ECHO " + parte1: primera fase de la configuración. Al final se debe reiniciar."
  $ECHO " + parte2: segunda fase. Se debe ejecutar una vez se complete la fase 1 y"
  $ECHO " + se haya reiniciado el equipo con el kernel apropiado de proxmox. "
  $ECHO " ++++++++++++++++++++++\n"
}

########################################
########################################
do_saludos() {
  echo "+do_saludos: Configurando motd e issue.net..."

  local F_MOTD=/etc/motd
  local F_ISSUE=/etc/issue.net

  cat > ${F_MOTD} << EOM
****************************************
  Informaticasa servers: ${NOMBRE_EMPRESA}
*      Debian GNU/Linux + Proxmox VE        *
* Bienvenido a ${NOMBRE_HOST}:              *
*   Servidor host de máquinas virtuales     *
*                                           *
****************************************
EOM
  cat ${F_MOTD} > ${F_ISSUE}
  echo "+do_saludos: hecho."
}


########################################
########################################
do_network() {
  echo "+do_network: Configurando interfaces"
  local F_INTERFACES=/etc/network/interfaces

  #Instala las bridge-utils para que esto rule
  aptitude install bridge-utils

  cat > $F_INTERFACES <<EOM
## LOOPBACK
auto lo
iface lo inet loopback

## ETH0
iface eth0 inet manual
        ethernet-wol    g

## ETH1
iface eth1 inet manual
        ethernet-wol    g

## PUENTE LAN
auto ${LAN_IFAZ}
iface ${LAN_IFAZ} inet static
        address ${IP_ADDR}
        netmask ${NETMASK}
        network ${LAN_ADDR}
        broadcast ${LAN_BCAST}
        gateway ${GATEWAY}
        dns-nameservers ${DNS_SERVER}
        bridge_ports eth0 eth1
        bridge_stp off
        bridge_fd 0
```

```
EOM

  ifup ${LAN_IFAZ}
  echo "+do_network: hecho."
  echo ""
}


########################################
########################################
do_nombres() {
  echo "+do_nombres: configurando nombres de equipo..."
### Fichero hosts
  cat > /etc/hosts <<EOM

127.0.0.1       localhost
# 127.0.1.1    ${NOMBRE_HOST}.${DOMINIO} ${NOMBRE_HOST}
${IP_ADDR}      ${NOMBRE_HOST}.${DOMINIO} ${NOMBRE_HOST}

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

EOM
### Fichero mailname
  cat > /etc/mailname <<EOM
${NOMBRE_HOST}.${DOMINIO}
EOM

### Fichero muttrc
  cat > /root/.muttrc <<EOM
set realname="${NOMBRE_HOST}.${DOMINIO}"
set from="root@${NOMBRE_HOST}.${DOMINIO}"
set use_from=yes
EOM

  echo "+do_nombres: hecho."
  echo ""
}


########################################
########################################
do_crontab() {
  echo "+do_crontab: programando tareas..."
  cat >>/etc/crontab <<EOM

## A las 23:59 todos los días realiza el backup de la VM windows XP
# 59 23 * * 2,4,5       root    backupWinVM
## A las 8:00 y a las 18:00 de cada día notifica la IP pública
# 00 8,18       * * *           root    informa ip
## A las 15:00 los viernes envía un resumen de estado
# 00 15 * * 5   root    informa estado
## A las 00:00 del primer día de cada mes envía un informe completo de estado
# 00 00 1 * *           root    informa todo

EOM

  echo "+do_crontab: hecho."
  echo ""
}

########################################
########################################
do_sourcesList() {
  echo "+do_sourcesList: configurando APT..."
  cat > /etc/apt/sources.list <<EOM
#
deb http://ftp.de.debian.org/debian/ ${DEBIAN_VER} main non-free contrib
# deb-src http://ftp.de.debian.org/debian/ ${DEBIAN_VER} main non-free contrib
deb http://security.debian.org/ ${DEBIAN_VER}/updates main contrib non-free
# deb-src http://security.debian.org/ ${DEBIAN_VER}/updates main contrib non-free
# ${DEBIAN_VER}-updates, previously known as 'volatile'
deb http://ftp.de.debian.org/debian/ ${DEBIAN_VER}-updates main contrib non-free
```

```
# deb-src http://ftp.de.debian.org/debian/ ${DEBIAN_VER}-updates main contrib non-free

# PVE pve-no-subscription repository provided by proxmox.com
deb http://download.proxmox.com/debian ${DEBIAN_VER} pve-no-subscription

EOM

cat >/etc/apt/sources.list.d/pve-enterprise.list <<EOM
#/etc/apt/sources.list.d/pve-enterprise.list
# deb https://enterprise.proxmox.com/debian ${DEBIAN_VER} pve-enterprise
EOM

  echo "+do_sourcesList: hecho."
  echo ""
}


########################################
########################################
do_ssh_server() {
# TODO - por ahora configuración a mano
#  echo "+do_ssh_server: configura servicio SSH..."
#
#  cat >>/etc/ssh/sshd_config <<EOM
#
#EOM
#
#  echo "+do_ssh_server: hecho."
  echo ""
}

########################################
########################################
do_pcspkr() {
  echo "+do_pcspkr: bloquea módulo pitido spkr..."
  cat > /etc/modprobe.d/pcspkr-blacklist.conf <<EOM
blacklist pcspkr
EOM
  echo "+do_pcspkr: hecho."
  echo ""
}

########################################
########################################
do_scripts() {
  echo "+ do_scripts: instala scripts personalizados..."
  echo " Se instalarán en: [ $SCRIPTS_INSTALL ]"
  echo " "
  for script in `ls ${SCDIR}`
  do
        echo "Copiando <${script}>"
        cp "${SCDIR}/${script}" "${SCRIPTS_INSTALL}/"
        chown root:root "${SCRIPTS_INSTALL}/${script}"
        chmod +x "${SCRIPTS_INSTALL}/${script}"
  done
  echo "+do_scripts: hecho."
  echo ""
}

########################################
########################################
do_instalasw() {
  echo "+do_instalasw: instalando paquetes..."
  echo " "
  case "$1" in

  parte1)
        echo "+do_instalasw: PARTE 1"
        wget -O- "http://download.proxmox.com/debian/key.asc" | apt-key add -
        aptitude update
        aptitude safe-upgrade
        aptitude install proxmox-ve ntp ssh postfix ksm-control-daemon open-iscsi lvm2 ca-
certificates bootlogd
        # OLD aptitude install pve-firmware pve-kernel-${PROXMOX_KERNEL} pve-headers-$
{PROXMOX_KERNEL} resolvconf bridge-utils vim
    ;;
```

```
  parte2)
        echo "+do_instalasw: PARTE 2"
        # OLD aptitude install proxmox-ve-${PROXMOX_VER} ntp ssh lvm2 postfix ca-certificates
ksm-control-daemon vzprocps open-iscsi bootlogd
        aptitude install vim dmidecode lshw usbutils pciutils lsscsi ifstat sysv-rc-conf
ntfs-3g smbclient htop
        aptitude install logwatch lm-sensors hddtemp smartmontools
        cp /usr/share/logwatch/default.conf/logwatch.conf /etc/logwatch/conf/
        mkdir /var/cache/logwatch
        dpkg-reconfigure locales
        dpkg-reconfigure tzdata
        aptitude clean
        apt-get autoremove
        apt-get clean
    ;;

  *)
    echo "+do_instalasw: ERROR. No se hace nada. (falta parte1 / parte2)"
    ;;
esac

  echo "+do_instalasw: hecho."
  echo ""
}

########################################
########################################
do_postfix() {
  echo "+do_postfix: "
  echo " ++++++++++++++++++++++++++++++++++++++++++++++++++++ "
  echo " + Configuración del servidor de correo POSTFIX... "
  echo " + En el asistente siguiente deja todo por defecto excepto"
  echo " + el tipo de puesto. Debes escoger: SITIO DE INTERNET"
  echo " ++++++++++++++++++++++++++++++++++++++++++++++++++++ "
  echo " + Pulsa ENTER para continuar"
  echo " "
  read ab
  dpkg-reconfigure postfix

# Config de /etc/postfix/sasl/passwd
cat > /etc/postfix/sasl/passwd <<EOM
# /etc/postfix/sasl/passwd
[smtp.gmail.com]:587 uncorreodegmail@gmail.com:estaClaveEsFalsa
EOM

# Config de /etc/postfix/main.cf
cat > /etc/postfix/main.cf <<EOM
# /etc/postfix/main.cf
#
# See /usr/share/postfix/main.cf.dist for a commented, more complete version

# Debian specific:  Specifying a file name will cause the first
# line of that file to be used as the name.  The Debian default
# is /etc/mailname.
myorigin = /etc/mailname
smtpd_banner = \$myhostname ESMTP \$mail_name (Debian/GNU)
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no
readme_directory = no
# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:\${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:\${data_directory}/smtp_scache

### CONFIGURAR
myhostname = ${NOMBRE_HOST}.${DOMINIO}
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
### CONFIGURAR
mydestination = ${NOMBRE_HOST}.${DOMINIO}, localhost.${DOMINIO}, , localhost
### CONFIGURAR
```

```
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128 ${LAN_CIDR}
mailbox_command = procmail -a "\$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
inet_protocols = all
### CONFIGURAR
relayhost = [smtp.gmail.com]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl/passwd
smtp_sasl_security_options = noanonymous
smtp_use_tls = yes
smtp_tls_security_level = encrypt
smtp_tls_CAfile = /etc/postfix/cacert.pem

EOM

  chmod 600 /etc/postfix/sasl/passwd
  postmap /etc/postfix/sasl/passwd
  cat /etc/ssl/certs/Equifax_Secure_CA.pem > /etc/postfix/cacert.pem
  /etc/init.d/postfix reload

  sleep 1
  echo "Test mail from postfix on ${NOMBRE_HOST}.${DOMINIO}" | mail -s "Test Postfix" $
{TEST_MAIL_RCV}

  echo "+do_instalasw: hecho."
  echo ""
}

do_instruManuales() {
  echo " ++++++++++++++++++++++++++++++++++++++++++++++++++ "
  echo " A continuación se muestran unas instrucciones con varios"
  echo " pasos que se deben realizar a mano. "
  echo " ++++++++++++++++++++++++++++++++++++++++++++++++++ "
  echo "Pulsa ENTER para continuar"
  read ab

cat >&1 <<EOM

****************************************

--------------------------------------
1. Instalar un entorno gráfico mínimo para la administración local:

aptitude install --without-recommends xorg xserver-xorg-video-mga fluxbox tightvncserver
eterm feh
aptitude install --without-recommends gsmartcontrol smartmontools iceweasel iceweasel-l10n-
es-es openjdk-7-jre icedtea-7-plugin

Donde xserver-xorg-video-mga es el driver de la gráfica y deberás elegirlo en función
del hardware del equipo, ejecuta "lspci" para ver cual es.

--------------------------------------
2. Repasa fstab y verifica que todo está OK:

>$ nano /etc/fstab

--------------------------------------
3. Configurar correctamente GRUB:

  >$ nano /etc/default/grub

3.1. Desactivar los SUBMENUS para que se vean todos los kernels

  GRUB_DISABLE_SUBMENU=y

3.2 Establecer el kernel proxmox por defecto:
    (normalmente es el número 2)
  GRUB_DEFAULT=2

3.3 Actualizar la config e instalarlo en ambos HDDs
para asegurar el arranque si uno falla

  >$ update-grub
  >$ grub-install /dev/sda
```

```
  >$ grub-install /dev/sdb

-------------------------------------
4. Alertas desde mdadm (avisos de problemas en RAID / SMART):

MDADM:
Edita mdadm.conf:
  >$ nano /etc/mdadm/mdadm.conf
Y establece:
  # instruct the monitoring daemon where to send mail alerts
  MAILADDR monitor@informaticasa.es

SMART:
Para activar el demonio:
  >$ nano /etc/default/smartmontools
Establecer el mail de envio:
  >$ nano /etc/smartd.conf
  DEVICESCAN -d removable -n standby -m monitor@informaticasa.es -M exec
/usr/share/smartmontools/smartd-runner

-------------------------------------
5. Eliminar el warning de No Subcription en la web de proxmox:

  -- Editar el siguiente fichero alrededor de la línea 482:
  nano +482 /usr/share/pve-manager/ext4/pvemanagerlib.js
  -- Busca:
  if (data.status !== 'Active') {
  -- Y cámbialo por:
  if (false) {

-------------------------------------
6. Ejecutar sensors-detect y cargar los módulos deseados
sensors-detect

-------------------------------------
7. Configuración del servidor SSH:
  Cambia el puerto y/o la interfaz de escucha para mejorar la seguridad.

  >$ nano /etc/ssh/sshd_config

-------------------------------------
8. Ajustar crontab para programar los informes o las copias con la
   periodicidad deseada:

  >$ nano /etc/crontab

-------------------------------------
9. Si deseas una config avanzada de las vms:
/etc/vzdump.conf
/etc/vz/vz.conf
-------------------------------------

*****************************************
EOM

  echo "+do_instruManuales: hecho."
  echo ""
}


################################################################################
## FIN FUNCIONES DE APOYO
################################################################################

################################################################################
# MAIN1: configuración parte1
################################################################################
do_parte1() {

  # Mensaje de inicio
cat >&1 <<EOM
  *****************************************
  Autoconfiguración de HServer01 - PARTE 01.

  - Servidor de máquinas virtuales -
  Proxmox VE + Debian GNU Linux 64bits
```

```
  Los scripts personalizados deben estar en:
  $SCDIR
  ********************************************

EOM

  echo "Se van a realizar las primeras configuraciones básicas."
  echo "Pulsa ENTER para continuar"
  read ab

  echo " ++++++++ Configuraciones iniciales ++++++++ "
  echo " "
  # 1. RED
  do_network
  # 2. HOSTS, MAILNAME, MUTTRC
  do_nombres
  # 3. MOTD, ISSUE.NET
  do_saludos
  # 4. APT SOURCES.LIST
  do_sourcesList
  # 5. SERVIDOR SSH - NO, MANUAL POR AHORA
  # do_ssh_server
  # 6. OTROS
  do_pcspkr
  do_crontab

  echo "Se van a instalar los scripts personalizados."
  echo "Pulsa ENTER para continuar"
  read ab
  do_scripts

  echo "Se van a instalar paquetes del sistema."
  echo "Pulsa ENTER para continuar"
  read ab
  do_instalasw parte1

cat >&1 <<EOM

  ********************************************
  Ahora debes REINICIAR y arrancar con el kernel
  de proxmox:

  pve-kernel-${PROXMOX_KERNEL}

  Debes configurarlo por defecto en grub
  para no tener que cambiar a mano.

  Una vez se reinicie el equipo
  (comprueba que el kernel es correcto con uname -a)
  continúa con la configuración ejecutando este script
  para la parte 2:
  $> ${THIS_SCRIPT} parte2
  ********************************************

EOM

  echo "Pulsa ENTER para REINICIAR"
  read ab
  reboot
}
################################################################################

################################################################################
# MAIN2: configuración parte2
################################################################################
do_parte2() {
  cat >&1 <<EOM

  ********************************************
  Autoconfiguración de HServer01 - PARTE 02.
  ********************************************

EOM

  echo "Se van a instalar paquetes del sistema."
```

```
  echo "Pulsa ENTER para continuar"
  read ab
  do_instalasw parte2

  # Activa el módulo KVM para la virtualización
  modprobe kvm
  echo kvm >> /etc/modules

  echo " ++++++++++++++++++++++++++++++++++++++++++++++++++ "
  echo " + A continuación Activa/Desactiva los servicios que "
  echo " + quieres que arranquen automáticamente... "
  echo " ++++++++++++++++++++++++++++++++++++++++++++++++++ "
  echo " + Pulsa ENTER para continuar"
  echo " "
  read ab
  sysv-rc-conf

  echo "Se va a configurar POSTFIX."
  echo "Pulsa ENTER para continuar"
  read ab
  do_postfix

  # Terminando...
  do_instruManuales

  echo "******************************************"
  echo " FIN. La configuración inicial del servidor "
  echo " se ha completado. Ahora puedes empezar a trabajar"
  echo " con las máquinas virtuales accediendo a: "
  echo " https://${IP_ADDR}:8006"
  echo " mediante la interfaz web."
  echo "******************************************"
  echo " + Pulsa ENTER para continuar"
  echo " "
  read ab
}
############################################################
# BEGINNING OF MAIN
############################################################
# Valor de retorno: 0 OK, 2 ERROR
RETVAL=0
case "$1" in

  parte1)
    do_parte1
    ;;
  parte2)
    do_parte2
    ;;
  help|"")
    uso
    ;;
  *)
    uso
    RETVAL=2
    ;;
esac

exit $RETVAL
```

A.2. estadoHDDs: comprobación de estado de discos duros

```
#! /bin/bash
#
# SCRIPT: estadoHDDs
# AUTHOR: Carlo de Cástulo Navarro Álvarez
# DATE: 02/11/2014
# VERSION: 1.0
#
# PURPOSE:
#  Muestra un resumen con la info de estado de los discos duros
#
#
# DEPENDENCIAS: smartmontools, hddtemp
#
```

```
##########################################

for AA in a b c d; do
  HDD="/dev/sd$AA"
  if [ -b "${HDD}" ]; then
        echo "----------"
        echo "Dispositivo: < "${HDD}" >"
        echo "----------"
        smartctl -a ""${HDD}"" | \
          grep "Model\|Power_On_Minutes\|Power_On_Half_Minutes\|Power_On_Hours\|Serial
Number\|User Capacity\|SMART overall-health" | \
          awk '{ if(/Power/ ){\
                                split($10,sas,"h") ;\
                                dias = int( sas[1] / 24) ;\
                                year = dias / 365 ;\
                                yeari = int(year);\
                                mese = int ((year - yeari)*12 );\
                                print "\tHoras (Acumuladas):\t"$10 " || ", dias , "dias || ",
yeari, "años y ", mese ," meses //"}\
                           if (/Model Family/){\
                                gsub(/Model Family:/,"");\
                                print "\tFamilia:\t"$0}\
                           if (/Device Model/){\
                                gsub(/Device Model:/,"");\
                                print "\tModelo:\t\t"$0}\
                           if (/Serial Number/){\
                                gsub(/Serial Number:/,"");\
                                print "\tNSerie:\t\t "$0}\
                           if (/Capacity/){\
                                print "\tCapacidad:\t\t" $5 $6}\
                           if (/overall-health/){\
                                print "\tSalud general SMART:\t" $NF}\
                        }';
        TEMP=$(hddtemp -u c -q "${HDD}" |awk '{print $NF}')
        echo -e "\tTemperatura:\t\t${TEMP}"
        echo ""
  fi
done

exit 0
```

A.3. informa: envío de informe de estado por correo electrónico

```
#! /bin/bash
#
# SCRIPT: informa
# AUTHOR: Carlo de Cástulo Navarro Álvarez
# DATE: 17/02/2016
# REV: 1.6
#
# PURPOSE: envía informacion del estado del servidor
#          por correo electrónico.
#
#       Dependencias:
#
#       aptitude update && aptitude install dmidecode lshw usbutils pciutils lsscsi logwatch
lm-sensors ifstat
#
#####
# set -n # Uncomment to check script syntax, without execution.
#
# set -x # Uncomment to debug this shell script
##########################################################


############################################################
# CONFIGURACIÓN
############################################################
# Direcciones de correo a las que se envía el aviso
MAIL_ADDRS="monitor@informaticasa.es"
# Nombre descriptivo del servidor
S_NAME="HServer01.dominio"
# Info de red
IFAZ="vmbr0"
```

```
#############################################################
# VARIABLES
#############################################################
THIS_SCRIPT=$(basename $0)
ECHO="echo -e"
FECHA=`date +%Y_%m_%d`
HORA=`date +%H:%M`
HOST=`hostname`
KERNEL=`uname -a`
UPTIME=`uptime`
IPPUB=`wget -qO- checkip.dyndns.com|grep -oE "([0-9]+\.?){4}"` || IPPUB="error"
CUERPO_FILE="/tmp/cuerpo_mail"
ADJUNTO_FILE="/tmp/informe_${S_NAME}.txt"


#############################################################
# FUNCTIONS
#############################################################

uso () {
  $ECHO "\n +++++ $THIS_SCRIPT +++++"
  $ECHO " + Uso: $THIS_SCRIPT DEQUÉ"
  $ECHO " + Donde DEQUÉ será uno de los siguientes:"
  $ECHO " + estado | ip | todo | help "
  $ECHO " ++++++++++++++++++++++\n"
}

###############
# conexiones del equipo
###############
do_conexiones () {
  $ECHO "\n+++++++++++++++ Puertos | Conexiones ++++++++++++++++++"
  $ECHO  " ----> Puertos UDP escuchando <---- "
  netstat -nulp | tail -n+3
  $ECHO ""
  $ECHO  " ----> Puertos TCP escuchando <---- "
  netstat -ntlp | tail -n+3
  $ECHO ""
  $ECHO  " ----> Conexiones TCP establecidas <----"
  netstat -ntp |grep ESTABLISHED |awk '{print " LO: "$4"\tRE: "$5"\tAP: "$7}' | tail -n17 |
sort
  $ECHO ""
  # $ECHO  " ----> Conexiones TCP establecidas (INTERNET) <----"
  # netstat -ntp |grep ESTABLISHED |grep -Fv "${LAN_IP}:" |awk '{print " LO: "$4"\tRE:
"$5"\tAP: "$7}' | tail -n12
  # $ECHO ""
  # $ECHO  " ----> Conexiones TCP establecidas (RED LOCAL) <----"
  # netstat -ntp |grep ESTABLISHED |grep -E '192\.168\.7\.(1[0-9]+[0-9]?|[2-9]+[0-9]?[0-9]?)'
|awk '{print " LO: "$4"\tRE: "$5"\tAP: "$7}' | tail -n17
  # $ECHO ""
  $ECHO "+++++++++++++++ Puertos | Conexiones ++++++++++++++++++\n"
  return 0
} 2>&1


###############
# config de networking
###############
do_networking () {
  $ECHO "\n+++++++++++++++ Networking ++++++++++++++++++"
  $ECHO  " --> Configuración <--"
  ifconfig $IFAZ
  $ECHO  " --> Tráfico <--"
  ifstat -Wq -i $IFAZ 0.5 1
  $ECHO ""
  $ECHO  " ----> Tabla de rutas <----"
  route -n
  $ECHO ""
  $ECHO  " ----> Tabla ARP <----"
  arp -nv -a
  $ECHO "+++++++++++++++ Networking ++++++++++++++++++\n"
  return 0
} 2>&1


###############
```

```
# info "hardware"
##############
do_hardware () {

  FAB=`dmidecode -s system-manufacturer`
  PRO=`dmidecode -s system-product-name`
  VER=`dmidecode -s system-version`
  SER=`dmidecode -s system-serial-number`
  SKU=`dmidecode | grep -i sku| cut -c 14-`
  P_MOD=`grep -m 1 'model name' /proc/cpuinfo| cut -c 14-`
  P_ARQ=`lshw -C cpu | grep width |cut -c 15-`
  P_CACHE=`grep -m 1 'cache size' /proc/cpuinfo | cut -c 14-`
  P_CORES=`grep -m 1 'cpu cores' /proc/cpuinfo| awk '{print $4}'`
  RAM_KB=`grep 'MemTotal' /proc/meminfo|awk '{print $2}'`
  RAM=$((${RAM_KB}/1024))

  $ECHO "+++++++++++++++ Hardware Info ++++++++++++++++++"
  echo " "
  echo "+ Fabricante: < ${FAB} >"
  echo "+ Modelo: < ${PRO} >"
  echo "+ Número de serie: < ${SER} >"
  echo "+ Versión del producto: < ${VER} >"
  echo "+ SKU o P/N: < ${SKU} >"
  echo "------"
  echo "+ Procesador: < ${P_MOD} >"
  echo "+ Arquitectura: < ${P_ARQ} > - Núcleos: < ${P_CORES} > - Caché L2: < ${P_CACHE} >"
  echo "------"
  echo "+ Memoria RAM: < ${RAM}MB >"
  echo "------"
  echo "+ Almacenamiento - Particiones: "
  fdisk -l
  echo "------"
  echo "+ Dispositivos USB:"
  lsusb
  echo "------"
  echo "+ Dispositivos SCSI: "
  lsscsi
  echo "------"
  echo "+ Dispositivos PCI: "
  lspci
  echo " "
  $ECHO "+++++++++++++++ Hardware Info ++++++++++++++++++\n"

  return 0

} 2>&1

##############
# estado más "hardware"
##############
do_sistema () {
  $ECHO "\n+++++++++++++++ System Status ++++++++++++++++++"
  $ECHO  " ----> Uptime / Usuarios <---- "
  w
  $ECHO ""
  $ECHO  " ----> Programas más pesados <---- "
  ps axo comm,user,pcpu --sort -pcpu | head -n11 |tail -n12
  $ECHO ""
  $ECHO  " ----> Memoria RAM <---- "
  free -m |grep "buffers" |cut -c26-40
  $ECHO ""
  $ECHO  " ----> Espacio en disco <---- "
  df -h
  $ECHO ""
  $ECHO  " ----> Estado del RAID <---- "
  cat /proc/mdstat
  $ECHO ""
  $ECHO  " ----> Temperaturas <---- "
  sensors |grep temp1
  $ECHO "+++++++++++++++ System Status ++++++++++++++++++\n"
  return 0
} 2>&1

##############
# resumen de logs
# por logwatch
```

```
##############
do_resumen_logs () {
  logwatch
  return 0
} 2>&1

##############
# estado de las vms
# de proxmox
##############
do_proxmox_status() {
  $ECHO "\n+++++++++++++++ Proxmox VE status ++++++++++++++++++"
  echo " ----> LXC containers:"
  pct list
  echo " ----> Qemu/KVM machines:"
  qm list
  $ECHO "+++++++++++++++ Proxmox VE status ++++++++++++++++++\n"
  return 0
} 2>&1

do_almacenamiento(){
  $ECHO "\n+++++++++++++++ Almacenamiento ++++++++++++++++++"
  $ECHO ""
  $ECHO  " ----> Espacio en disco <---- "
  df -h
  $ECHO ""
  $ECHO  " ----> Estado del RAID <---- "
  cat /proc/mdstat
  $ECHO ""
  $ECHO  " ----> LVM2: Volúmenes físicos <---- "
  pvdisplay
  $ECHO ""
  $ECHO  " ----> LVM2: Grupos de volúmenes <---- "
  vgdisplay
  $ECHO ""
  $ECHO  " ----> LVM2: Volúmenes lógicos <---- "
  lvdisplay
  $ECHO ""
  $ECHO "+++++++++++++++ Almacenamiento ++++++++++++++++++\n"
  return 0
} 2>&1

##############
# Envía un mail notificando la ip
# pública del sistema
##############
do_informa_ip() {

  local ASUNTO="${S_NAME}: notificación IP"

  cat >> ${CUERPO_FILE} <<EOM

+ Informe de IP pública:
------------------------------------------
 - Equipo:              < ${S_NAME} >
 - Fecha:               < ${FECHA} >
 - Uptime:              < ${UPTIME} >
------------------------------------------
 - HostName:    < ${HOST} >
 - IP pública:  << $IPPUB >>
------------------------------------------

EOM

  # Envía el mail
  mutt -s "${ASUNTO}" ${MAIL_ADDRS} < "${CUERPO_FILE}"
  rm -f ${CUERPO_FILE}
  return 0
}

##############
# Envía un informe con el estado
# del sistema
##############
do_informa_estado() {
```

```
  local ASUNTO="${S_NAME}: informe resumen de estado"

cat >> ${CUERPO_FILE} <<EOM

+ Informe resumen de estado:
-----------------------------------------
 - Equipo:              < ${S_NAME} >
 - Fecha:               < ${FECHA} >
 - Uptime:              < ${UPTIME} >
-----------------------------------------
 - HostName:    < ${HOST} >
 - IP pública:  << $IPPUB >>
-----------------------------------------
** Ver adjunto.

EOM

  echo "" >> ${ADJUNTO_FILE}
  do_sistema 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_resumen_logs 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_proxmox_status 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}

  # Envía el mail
  mutt -s "${ASUNTO}" ${MAIL_ADDRS} -a "${ADJUNTO_FILE}" < "${CUERPO_FILE}"
  rm -f ${ADJUNTO_FILE}
  rm -f ${CUERPO_FILE}
  return 0

}

###############
# Envía un informe con toda la
# info de estado del sistema
###############
do_informa_todo() {

  local ASUNTO="${S_NAME}: informe completo de estado"

cat >> ${CUERPO_FILE} <<EOM

+ Informe completo de estado:
-----------------------------------------
 - Equipo:              < ${S_NAME} >
 - Fecha:               < ${FECHA} >
 - Uptime:              < ${UPTIME} >
-----------------------------------------
 - HostName:    < ${HOST} >
 - IP pública:  << $IPPUB >>
-----------------------------------------
** Ver adjunto.

EOM

  echo "" >> ${ADJUNTO_FILE}
  do_hardware 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_networking 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_conexiones 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_almacenamiento 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_sistema 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_resumen_logs 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}
  do_proxmox_status 2>&1 >> ${ADJUNTO_FILE}
  echo "" >> ${ADJUNTO_FILE}

  # Envía el mail
  mutt -s "${ASUNTO}" ${MAIL_ADDRS} -a "${ADJUNTO_FILE}" < "${CUERPO_FILE}"
  rm -f ${ADJUNTO_FILE}
  rm -f ${CUERPO_FILE}
```

```
  return 0
}


############################################################
# BEGINNING OF MAIN
############################################################

# Valor de retorno: 0 OK, 1 AlreadyRunning, 2 ERROR
RETVAL=0

case "$1" in

  todo)
    do_informa_todo
    ;;
  estado)
    do_informa_estado
    ;;
  ip)
    do_informa_ip
    ;;
  help|"")
    uso
    ;;
  *)
    uso
    RETVAL=2
    ;;
esac

exit $RETVAL
```

A.4. backupWinVM: apagado forzado por RPC y copia de seguridad de máquina virtual Windows (para Proxmox VE 3.XX)

```
#!/bin/bash
#
# Apaga una máquina virtual con Windows XP
# mediante RPC y realiza la copia de seguridad.
#
# Para que funcione es necesario que en la máquina remota
# se desactive el Uso compartido simple de ficheros:
# Desmarcar:
# Explorador/Opciones de carpeta/Ver/Utilizar uso compartido simple de archivos (recomendado)
#
# En el host se debe tener instalado smbclient:
# aptitude install smbclient
#
# Requiere el script auxiliar vzdumpHook.pl

################################################################
# CONFIGURACIÓN PARA EL APAGADO
################################################################
# Dirección IP de la máquina remota
VM_IPADDR=192.168.2.12
# Usuario con permisos de administrador en la máquina remota y su clave
VM_USER="Usuario"
VM_PASS="usuario"
# Texto que se muestra en el aviso de apagado
MSG_AVISO='Apagado remoto para copia de seguridad'
# Tiempo que dura el aviso previo al apagado, en segundos
T_WAIT=10
################################################################
# CONFIGURACIÓN PARA COPIA
################################################################
# id de la VM
VM_ID="101"
# Almacenamiento destino de proxmox para backups
BACKUP_STORAGE="backups"
# mail para enviar el resultado
MAILTO="monitor@informaticasa.es"
# script con el hook para iniciar la vm después del backup
HOOK="/usr/local/sbin/vzdumpHook.pl"
```

```
#############################################################

echo
echo "> Iniciando backup de la VM: < ${VM_ID} > con IP: < ${VM_IPADDR} >..."
# 1. Apaga la VM
echo ">> Enviando orden de apagado..."
net rpc shutdown -I ${VM_IPADDR} -U "${VM_USER}%${VM_PASS}" -f -t ${T_WAIT} -C "${MSG_AVISO}"
echo ">> Esperando el apagado..."
# 2. Espera 30s más del tiempo del aviso para que la vm se cierre
sleep $((${T_WAIT}+30))s
echo ">> Ejecutando backup..."
# 3. Ejecuta la copia
vzdump ${VM_ID} -compress lzo -storage ${BACKUP_STORAGE} -mailto ${MAILTO} -mode stop -script
${HOOK} -remove 1
# La espera y el rearranque se realizan desde el hook script de vzdump, para hacerlo al fin
del backup
# 4. Espera
# sleep 20s
# 5. Arranca de nuevo
# qm start ${VM_ID}
echo "> ¡Proceso de backup terminado!"
echo
```

```
#!/usr/bin/perl -w

# example hook script for vzdump (--script option)

use strict;

print "HOOK: " . join (' ', @ARGV) . "\n";

my $phase = shift;

if ($phase eq 'backup-end') {

    my $mode = shift; # stop/suspend/snapshot
    my $vmid = shift;

     # espera un poco y arranca la vm
        print ">> Iniciando VM < $vmid >... ";
        system ("sleep 10");
        system ("/usr/sbin/qm start $vmid -skiplock 1") == 0 ||
                        print ">> ERROR: el inicio de < $vmid > falló.\n";


} elsif ($phase eq 'job-start'  ||
                 $phase eq 'job-abort' ||
                 $phase eq 'job-end' ||
         $phase eq 'backup-start' ||
         $phase eq 'backup-abort' ||
         $phase eq 'log-end' ||
         $phase eq 'pre-stop' ||
         $phase eq 'pre-restart') {

    #my $vmtype = $ENV{VMTYPE}; # openvz/qemu
    #my $dumpdir = $ENV{DUMPDIR};
    #my $storeid = $ENV{STOREID};
    #my $hostname = $ENV{HOSTNAME};
        # print "HOOK-ENV: dumpdir=$dumpdir;storeid=$storeid\n";

    # tarfile is only available in phase 'backup-end'
    #my $tarfile = $ENV{TARFILE};

    # logfile is only available in phase 'log-end'
    #my $logfile = $ENV{LOGFILE};

    # print "HOOK-ENV:
vmtype=$vmtype;dumpdir=$dumpdir;storeid=$storeid;hostname=$hostname;tarfile=$tarfile;logfile=
$logfile\n";

    # example: copy resulting backup file to another host using scp
    #if ($phase eq 'backup-end') {
        #system ("scp $tarfile backup-host:/backup-dir") == 0 ||
```

```
        #    die "copy tar file to backup-host failed";
    #}

    # example: copy resulting log file to another host using scp
    #if ($phase eq 'log-end') {
        #system ("scp $logfile backup-host:/backup-dir") == 0 ||
        #    die "copy log file to backup-host failed";
    #}

} else {

    die "got unknown phase '$phase'";

}

exit (0);
```

Anexo B. Scripts desarrollados para dataserver

B.1. 00_instalar.sh: Configurador inicial

```
#!/bin/bash
################################################################################
#
# 00_instalar.sh
#
# Automatiza la configuración de un "dataserver" (servidor samba con varios
# usuarios preparados) en un container Debian en Proxmox. Este script
# junto con los directorios SCDIR y CFDIR se deberán copiar y ejecutar
# como root dentro del container recién creado.
#
# Fecha: 06/04/2016
#
################################################################################

################################################################################
## CONFIGURACIÓN
################################################################################
# Versión de debian usada para el container
# en junio de 2015 -> jessie
DEBIAN_VER=jessie

# Nombre de empresa para mostrar información
NOMBRE_EMPRESA="NombreEmpresa"
EMPRESA_CORTO="NomEmpre"

##### Usuarios y grupos unix
USU_PRI="usuarioPri"
USU_FULL_NAME="Nombre Usuario Pri"
USU_PRI_PASS="claveDelPri"
# Grupo común para los recursos compartidos de los usuarios del ayto
GRU_PRI="grupoPrincipal"

#
# Resto de usuarios que se crean para puestos remotos y comparticiones SAMBA
# estos usuarios no tienen acceso mediante SHELL.
# F_USUARIOS debe contener una lista con estos usuarios (uno por línea)
# donde se especifica su contraseña y grupo principal. El formato del fichero es por tanto:
# COL1  COL2    COL3
# user  clave   grupo
# pc01  pc011   grcps
#
USUARIOS="Usuario pc01 pc02 pc03 pc04 pc05 pc06 pc07 pc08 pc09 pc10"
F_USUARIOS="./usuarios.txt"

##########
# Grupo de trabajo SAMBA
SAMBA_WG="WORKGROUP"

# Config de red
## OJO:
# La configuración de la red en los LXC ya no es necesaria.
# Se realiza desde el HOST. Se mantiene esta sección por si hay que volver a OpenVZ.
#
###
IFAZ=eth0
IP_ADDR=192.168.7.11
NETMASK=255.255.255.0
LAN_ADDR=192.168.7.0
LAN_BCAST=192.168.7.255
GATEWAY=192.168.7.1
DNS_SERVER=192.168.7.1
##########

# Directorio origen con los ficheros de configuración para copiar (YA NO SE USA)
# CFDIR="./Conf_Files"
# Directorio origen con los scripts personalizados
```

```
SCDIR="./Scripts"
# Destino para instalar los scripts
SCRIPTS_INSTALL="/usr/local/sbin"
#############################################################################

#############################################################################
# *******************************************************************
# A PARTIR DE AQUÍ NO SUELE SER NECESARIO TOCAR NADA, PERO UN REPASO NUNCA VIENE MAL
# *******************************************************************
#############################################################################

#############################################################################
## FUNCIONES DE APOYO
#############################################################################
do_saludos() {
  echo "+do_saludos: Configurando motd e issue.net..."

  local F_MOTD=/etc/motd
  local F_ISSUE=/etc/issue.net

  cat > ${F_MOTD} << EOM
****************************************
  Informaticasa servers: ${NOMBRE_EMPRESA}
*         LXC  CT Debian GNU/Linux          *
* Bienvenido a DATASERVER:                  *
*    servidor virtual de almacenamiento     *
*    de datos                               *
****************************************
EOM
  cat ${F_MOTD} > ${F_ISSUE}
  echo "+do_saludos: hecho."
}
########################################

########################################
do_network() {
  echo "+do_network: Configurando red"
  local F_INTERFACES=/etc/network/interfaces

  cat > $F_INTERFACES <<EOM
## LOOPBACK
auto lo
iface lo inet loopback
## LAN IFAZ
auto ${IFAZ}
iface ${IFAZ} inet static
        address ${IP_ADDR}
        netmask ${NETMASK}
        network ${LAN_ADDR}
        broadcast ${LAN_BCAST}
        gateway ${GATEWAY}
        dns-nameservers ${DNS_SERVER}

EOM
  ifup ${IFAZ}
  echo "+do_network: hecho."
}
########################################

########################################
do_sources_list() {
  echo "+do_sources_list: Configurando APT..."

  local F_SOURCES=/etc/apt/sources.list

  cat > $F_SOURCES <<EOM
#### STABLE --> Debian 8 JESSIE (ha entrado en junio 2015)
# repos estándar
deb http://ftp.de.debian.org/debian/ ${DEBIAN_VER} main non-free contrib
# deb-src http://ftp.de.debian.org/debian/ ${DEBIAN_VER} main non-free contrib
# repos sec
deb http://security.debian.org/ ${DEBIAN_VER}/updates main contrib non-free
# deb-src http://security.debian.org/ ${DEBIAN_VER}/updates main contrib non-free
# actualizaciones stable
deb http://ftp.de.debian.org/debian/ ${DEBIAN_VER}-updates non-free contrib main
# deb-src http://ftp.fr.debian.org/debian/ ${DEBIAN_VER}-updates main contrib non-free
```

```
# deb http://ftp.de.debian.org/debian/ ${DEBIAN_VER}-proposed-updates non-free contrib main
# Versiones nuevas
# deb http://ftp.de.debian.org/debian/ ${DEBIAN_VER}-backports main non-free contrib

EOM

  echo "+do_sources_list: hecho."
}
########################################

########################################
do_samba_conf() {
  echo "+do_samba_conf: Configurando SAMBA..."
  local F_SMB_CONF_TMP=/tmp/smbconf
  local F_SMB_CONF_FILE=/etc/samba/smb.conf

# Configura el smb.conf temporal
### Sección global
cat > ${F_SMB_CONF_TMP} <<EOM

[global]
        unix charset = UTF8
        workgroup = ${SAMBA_WG}
        server string = dataserver
        netbios name = dataserver
        map to guest = Bad User
        obey pam restrictions = Yes
        pam password change = Yes
        passwd program = /usr/bin/passwd %u
        passwd chat = *Enter\snew\s*\spassword:* %n\n *Retype\snew\s*\spassword:* %n\n
*password\supdated\ssuccessfully* .
        unix password sync = Yes
        syslog = 0
        log file = /var/log/samba/log.%m
        max log size = 1000
        name resolve order = lmhosts host bcast wins
        dns proxy = No
        usershare allow guests = Yes
        usershare max shares = 10
        panic action = /usr/share/samba/panic-action %d
        idmap config * : backend = tdb
        use client driver = Yes
        hide dot files = Yes
        invalid user = root
        load printers = no
        printing = bsd
        printcap name = /dev/null
        disable spoolss = yes

[homes]
        comment = Home Directories
        valid users = %S
        read only = No
        create mask = 0660
        force create mode = 0660
        directory mask = 0770
        force directory mode = 0770
        browseable = No

[printers]
        comment = All Printers
        path = /var/spool/samba
        create mask = 0700
        printable = Yes
        print ok = Yes
        browseable = No

[print$]
        comment = Printer Drivers
        path = /var/lib/samba/printers

[${EMPRESA_CORTO}]
        path = /home/${USU_PRI}/${EMPRESA_CORTO}
        valid users = @${GRU_PRI}
        write list = @${GRU_PRI}
        force group = +${GRU_PRI}
```

```
        create mask = 0660
        force create mode = 0660
        directory mask = 2770
        force directory mode = 2770

[Publica]
        path = /home/${USU_PRI}/Publica
        read only = No
        create mask = 0666
        force create mode = 0666
        directory mask = 0777
        force directory mode = 0777
        guest only = Yes
        guest ok = Yes

[Informaticasa]
        path = /home/${USU_PRI}/Informaticasa
        read only = No
        create mask = 0666
        force create mode = 0666
        directory mask = 0777
        force directory mode = 0777
        guest only = Yes
        guest ok = Yes

EOM

  # comprueba y crea con testparm el fichero smb.conf final
  [[ -f $F_SMB_CONF_FILE ]] && mv ${F_SMB_CONF_FILE} "${F_SMB_CONF_FILE}.old"
  testparm -s $F_SMB_CONF_TMP >$F_SMB_CONF_FILE 2>/dev/null || {
        echo  "do_samba_conf: error de testparm creando el fichero smb.conf";
        ERR=2
  }
  # elimina el temporal
  rm -f $F_SMB_CONF_TMP

  echo "+do_samba_conf: hecho."
  return $ERR
}
########################################

########################################
do_permisos() {
  F_PAM=/etc/pam.d/common-session
  F_LOGIN=/etc/login.defs

  echo "+do_permisos: Configurando < ${F_PAM} > y < ${F_LOGIN} >..."
cat >>${F_PAM} <<EOM

# Umask por defecto para nuevos ficheros: rwxrwx---
session optional        pam_umask.so    umask=0007

EOM

  echo ""
  echo "OJO: Vamos a modificar login.defs"
  echo "Edita la variable UMASK para los nuevos ficheros"
  echo " la línea debe quedar así: (cambiar 022 por 027) "
  echo "UMASK           027"
  echo ""
  echo "Pulsa ENTER para continuar"
  read ab
  nano +146 /etc/login.defs
  echo "+do_permisos: hecho."
}
########################################

########################################
do_scriptPermisos() {

  F_SCPERMISOS="${SCRIPTS_INSTALL}/repararPermisos"

  echo "+do_scriptPermisos: Instalando < $F_SCPERMISOS >..."

cat >${F_SCPERMISOS} <<EOM
#!/bin/bash
```

```
EMPRESA_CORTO="${EMPRESA_CORTO}"
USUARIOS="${USUARIOS}"
USU_PRI="${USU_PRI}"
GRU_PRI="${GRU_PRI}"
SHARED_PRI="/home/${USU_PRI}/${EMPRESA_CORTO}"
SHARED_PUB="/home/${USU_PRI}/Publica"
SHARED_MANT="/home/${USU_PRI}/Informaticasa"

#### HOMEs usuarios
for U in \$USUARIOS
do
  rutaHome="/home/\${U}"
  echo "Directorio personal del usuario < \$U >:"
  echo " [ \${rutaHome} ] ..."
  chown -R \${U}:\${GRU_PRI} \${rutaHome}
  chmod -R u=rwx,og=rX \${rutaHome}
  echo "."
done

rutaHome="/home/\${USU_PRI}"
echo "Directorio personal del usuario principal < \$USU_PRI >:"
echo " [ \${rutaHome} ] ..."
chown -R \${USU_PRI}:\${GRU_PRI} \${rutaHome}
chmod -R u=rwx,og=rX \${rutaHome}
echo "."

echo "Directorio compartido privado: "
echo "< \$SHARED_PRI >"
chown -R \${USU_PRI}:\${GRU_PRI} \${SHARED_PRI}
chmod -R ug=rwx,o=rX \${SHARED_PRI}
echo "."

echo "Directorio compartido público: "
echo "< \$SHARED_PUB >"
chown -R \${USU_PRI}:\${GRU_PRI} \${SHARED_PUB}
chmod -R ugo+rwx \${SHARED_PUB}
echo "."

echo "Directorio de mantenimiento Informaticasa: "
echo "< \$SHARED_MANT >"
chown -R \${USU_PRI}:\${GRU_PRI} \${SHARED_MANT}
chmod -R ugo+rwx \${SHARED_MANT}
echo "."

EOM
  chmod u+rxw,og=r ${F_SCPERMISOS}
  chown root:root ${F_SCPERMISOS}
  echo "+do_scriptPermisos: hecho."
}
########################################

########################################
do_usuarios() {
  echo "+do_usuarios: configuración de usuarios, grupos y directorios..."

  echo " Pulsa ENTER para continuar"
  read ab

  ###################
  echo "Creando grupo de usuarios: < ${GRU_PRI} >..."
  addgroup ${GRU_PRI} || { echo "error: <addgroup> creando grupo ${GRU_PRI}"; exit 1; }
  echo "...OK"

  echo "A continuación se va a crear el usuario principal de dataserver..."
  echo "Pulsa ENTER para continuar"
  read ab

  echo "Creando usuario principal: < ${USU_PRI} > | Clave: < ${USU_PRI_PASS} > | Grupo: < $
{GRU_PRI} >..."
  useradd --home /home/${USU_PRI} --create-home -U -c "${USU_FULL_NAME}" ${USU_PRI} || { echo
"error: <useradd> creando usuario ${USU_PRI}"; exit 1; }
  echo ${USU_PRI}:${USU_PRI_PASS} | chpasswd || { echo "error: <chpasswd> estableciendo clave
de ${USU_PRI}"; exit 1; }
  usermod -g ${GRU_PRI} ${USU_PRI} || { echo "error: <usermod> estableciendo grupo de $
{USU_PRI}"; exit 1; }
```

```
  echo -ne "${USU_PRI_PASS}\n${USU_PRI_PASS}\n" | smbpasswd -s -a ${USU_PRI} || { echo
"error: <smbpasswd> estableciendo clave samba de ${USU_PRI}"; exit 1; }

  echo "A continuación se van a crear los usuarios para los puestos remotos..."
  echo "Pulsa ENTER para continuar"
  read ab

  while read -r USW PAW GRW; do
        echo "Creando usuario: < ${USW} > | Clave: < ${PAW} > | Grupo: < ${GRW} >..."
        useradd --home /home/${USW} --create-home --shell /bin/false -U ${USW} || { echo
"error: <useradd> creando usuario ${USW}"; exit 1; }
        echo ${USW}:${PAW} | chpasswd || { echo "error: <chpasswd> estableciendo clave de $
{USW}"; exit 1; }
        usermod -g ${GRW} ${USW} || { echo "error: <usermod> estableciendo grupo de ${USW}";
exit 1; }
        echo -ne "${PAW}\n${PAW}\n" | smbpasswd -s -a ${USW} || { echo "error: <smbpasswd>
estableciendo clave samba de ${USW}"; exit 1; }
        echo "...OK"
  done < ${F_USUARIOS}

  echo "A continuación se van a crear los directorios comunes y a ajustar los permisos..."
  echo "Pulsa ENTER para continuar"
  read ab
  # Crea los directorios para compartir con samba
  mkdir /home/${USU_PRI}/${EMPRESA_CORTO}
  mkdir /home/${USU_PRI}/Informaticasa
  mkdir /home/${USU_PRI}/Publica

  # Ajusta los permisos
  ${SCRIPTS_INSTALL}/repararPermisos

  echo "+do_usuarios: hecho."

}
########################################


##############################################################################
## INICIO DEL PROCESO
##############################################################################
WHOAMI=`whoami`

# Comprobaciones iniciales
if [ "${WHOAMI}" != "root" ]; then
  echo "¡Debes ser root para usar este script!"
  exit 1
fi
# [ ! -d "$CFDIR" ] && { echo "error: <${CFDIR}> no encontrado"; exit 1; }
[ ! -r "${F_USUARIOS}" ] && { echo "error: <${F_USUARIOS}> no encontrado"; exit 1; }
[ ! -d "$SCDIR" ] && { echo "error: <${CFDIR}> no encontrado"; exit 1; }

clear

cat >&1 <<EOM
############################################
############################################
Autoconfiguración de dataserver.

- Servidor de datos virtual SAMBA sobre Proxmox -
LXC container - Debian GNU Linux amd64

# Scripts personalizados en:
# $SCDIR

############################################
############################################

EOM


echo "Pulsa ENTER para continuar"
read ab

##############################################################################
## Paso 1: preparar ficheros de configuración
##############################################################################
```

```
echo " ++++++++++++++ "
echo " 1.  Configuración del sistema..."
echo " Pulsa ENTER para continuar"
read ab
echo " "
# OLD - do_network - EN LXC no es necesario
do_sources_list
do_saludos
do_permisos
echo ""
echo " ++++++++ ...hecho. ++++++++"
echo ""

#############################################################################
## Paso 2: copiar scripts personalizados
#############################################################################
echo " ++++++++++++++ "
echo " 2. Instalando scripts..."
echo " Se instalarán en: [ $SCRIPTS_INSTALL ]"
echo " Pulsa ENTER para continuar"
read ab
echo " "

do_scriptPermisos
for script in `ls ${SCDIR}`
do
  echo "Copiando <${script}>"
  cp "${SCDIR}/${script}" "${SCRIPTS_INSTALL}/"
  chown root:root "${SCRIPTS_INSTALL}/${script}"
  chmod u+rxw,og=r "${SCRIPTS_INSTALL}/${script}"
done
echo ""
echo " ++++++++ ...hecho. ++++++++"
echo ""

#############################################################################
## Paso 3: instalar software
#############################################################################
echo " ++++++++++++++ "
echo " 3. Instalando y configurando software..."
echo " Pulsa ENTER para continuar"
read ab
echo " "
aptitude update
aptitude safe-upgrade
aptitude install locales samba htop sysv-rc-conf vim clamav iftop ifstat sudo smbclient wget
ipcalc
dpkg-reconfigure locales
dpkg-reconfigure tzdata
aptitude clean
apt-get autoremove
apt-get clean
/etc/init.d/clamav-freshclam stop

#############################################################################
## Paso 4: usuarios, grupos y SAMBA
#############################################################################
echo " ++++++++++++++ "
echo " 4. Usuarios, grupos y SAMBA..."
echo " Pulsa ENTER para continuar"
read ab
echo " "

# Configura SAMBA
/etc/init.d/samba stop && sleep 1
do_samba_conf

# Usuarios
do_usuarios
echo ""
echo " ++++++++ ...hecho. ++++++++"
echo ""

#############################################################################
## Paso 5: configura servicios al inicio
#############################################################################
```

```
echo " +++++++++++++ "
echo " 4. Activa/Desactiva los servicios que deben autoarrancar...+ "
echo " Pulsa ENTER para continuar"
read ab
echo " "
sysv-rc-conf
echo ""
echo " ++++++++ ...hecho. ++++++++"
echo ""


echo ""
echo " ++++++++ FIN ++++++++"
echo ""
```

B.2. makeUniRedScripts.sh: genera scripts .bat para conectar unidades de red

```
#!/bin/bash
#
# dependiencias: paquete dos2unix
# Crea los ficheros .bat para conectar unidades de red de dataserver
#
# Autor: Carlo de C. Navarro Álvarez
# Fecha: 17/02/2016
#############################
USU_DEC="0 1"
USU_UNI="1 2 3 4 5 6 7 8 9"
SERVER="192.168.7.11"
COMPARTIDA="Compartida"
DIR_DEST="./UniRedScripts"
mkdir -p $DIR_DEST
#############################
for D in $USU_DEC
do
  for U in $USU_UNI
  do
        SCRIPT="ConectaUniRedPC${D}${U}.bat"

cat > ${DIR_DEST}/${SCRIPT} <<EOM

NET USE z: \\\\${SERVER}\pc${D}${U} /user:${SERVER}\pc${D}${U} pc${D}${U}${U}
NET USE y: \\\\${SERVER}\\${COMPARTIDA} /user:${SERVER}\pc${D}${U} pc${D}${U}${U}

EOM

  unix2dos ${DIR_DEST}/${SCRIPT}

  done
done
#############################
```

B.3. viruscan: analiza ficheros o directorios en busca de virus para Windows

```
#! /bin/bash

#
#  Script para escanear en busca
#  de virus para sistemas Windows.
#  Actualiza y busca recursivamente en el directorio/fichero
#  que se pasa como argumento.
#
#  Dependencias:
#
#  aptitude update && aptitude install clamav clamav-freshclam
#

########################
OBJ="$1"
# Comprobando argumentos
if [ "${OBJ}"x = x ]; then
  echo -e  "\E[1;37m - Uso: viruscan OBJETIVO - \033[0m"
  exit 2
fi
```

```
if [ ! -e "${OBJ}" ] ; then
  echo -e  "\E[1;37m - ¡El fichero/directorio indicado no existe! - \033[0m"
  exit 2
fi
########################

########################
RET_VAL=0

echo "....................................................."
echo " <<< Analizando en busca de virus: $OBJ >>>"
now=`date +%d_%m_%Y-%k_%M`
LOG="viruscan_report_${now}.log"
DIR_INFECTED="viruscan_infected_${now}"

# actualiza la db
freshclam --quiet
# y busca
clamscan -r -v --log="${LOG}" "${OBJ}"
CLAM=$?

if [ $CLAM -eq 1 ]; then
  echo " "
  echo " +++ Se han hallado posibles infecciones +++ "
  echo "¿Quieres reescanear y mover los ficheros sospechosos a $DIR_INFECTED? [s/n] "
  read RESP
  if [ "$RESP" = "s" ]; then
    mkdir $DIR_INFECTED
    clamscan -r --quiet --move="${DIR_INFECTED}" "${OBJ}"
  else
     echo " "
     echo " Ficheros sospechosos:"
     cat ${LOG} | grep "FOUND"
     echo " ¡Haz algo!"
  fi
  RET_VAL=1
fi

if [ $CLAM -eq 2 ]; then
  RET_VAL=2
  echo "Ha ocurrido algún error durante la búsqueda"
fi

echo " "
echo "....................................................."
echo " <<< Listo, informe creado en: $LOG >>>"
echo " ¿Quieres eliminarlo? [s/n] "
read RESP
if [ "$RESP" = "s" ]; then
  rm -f $LOG
fi
echo " "

exit $RET_VAL
```