



## Resumen

Este proyecto cubrirá la documentación e instalación desde cero de la infraestructura necesaria para desplegar un sistema de computación en la nube propia basado en tecnologías OpenStack. Además tendrá la particularidad de que los usuarios del mismo serán almacenados y gestionados desde una herramienta externa a este sistema de nube.

Para llevarlo a cabo además del ya mencionado software de computación en la nube, se realizarán todas las configuraciones necesarias para disponer de un sistema de autenticación basado en la implementación de Microsoft del protocolo LDAP, llamado comercialmente Directorio Activo.

Los usuarios y sus credenciales serán almacenados de forma centralizada en el servidor Windows y ambos sistemas serán configurados para realizar una integración con el agente encargado de la autenticación de la nube desplegada.

Con esto, usuarios almacenados y gestionados desde un ente externo y desconocido para OpenStack como es el Directorio Activo, podrán acceder y utilizar los recursos ofertados en la nube.

Daniel Fernandez Rodriguez  
“Integración de OpenStack con un Directorio Activo”

Palabras clave

OpenStack, Directorio Activo, virtualización, nube

Daniel Fernandez Rodriguez  
“Integración de OpenStack con un Directorio Activo”

# Índice general

## Tabla de contenidos

Capítulo 1. Memoria del Proyecto.....	10
¿Qué se pretende?.....	10
Desarrollo del Proyecto.....	10
Desarrollo de la Documentación.....	11
Apostando por herramientas abiertas.....	12
Disponibilidad del proyecto.....	13
Introducción.....	15
Aspectos teóricos.....	15
Planificación del Proyecto y Resumen de Presupuestos.....	15
Análisis.....	15
Diseño del sistema.....	15
Implementación del sistema.....	16
Problemas encontrados.....	16
Ampliaciones.....	16
Referencias bibliográficas.....	16
Capítulo 2. Introducción.....	17
Justificación del proyecto.....	17
Objetivos del Proyecto.....	17
Estudio de la Situación Actual.....	20
Evaluación de Alternativas.....	20
Eucalyptus.....	20
Apache CloudStack.....	20
VmWARE vCloud air.....	21
OpenNebula.....	21
Google Compute Engine.....	21
Microsoft Azure.....	22
Capítulo 3. Aspectos Teóricos.....	23
¿Qué es la nube?.....	23
Algunos pros y contras del modelo en la nube.....	24
Capítulo 4. Planificación del Proyecto y Resumen de Presupuestos.....	26
Planificación.....	26
Presupuesto.....	28
Descripción del servidor físico.....	29
Infraestructura necesaria.....	32
¿Qué se pretende?.....	32
Elección de Windows.....	33
Licencias.....	33
Decisión.....	36

Elección de OpenStack.....	36
Elección de la distribución GNU/Linux.....	37
Ubuntu (y derivadas).....	37
SuSe.....	39
RedHat (y derivadas).....	39
Requisitos del Sistema.....	41
Requisitos funcionales.....	41
Requisitos no funcionales.....	42
Identificación de Actores del Sistema.....	42
Administrador del sistema.....	42
Usuarios del sistema.....	42
Usuario 'Alumno'.....	42
Usuario 'Profesor'.....	43
Especificación de Casos de Uso.....	43
Arquitectura.....	46
Aprovisionamiento hardware.....	46
Windows Server.....	47
Scientific Linux.....	47
Configuraciones de Red.....	49
Herramientas y programas usados durante el desarrollo.....	50
Virtualización.....	50
Xen Server.....	50
Xen Center.....	50
Xen Orchestra.....	51
VirtualBox.....	52
Edición de archivos e imágenes.....	52
Libreoffice.....	52
Vim.....	53
Gedit.....	53
GIMP.....	53
Generación de diagramas.....	54
draw.io.....	54
Planner para Gnome.....	54
Otros.....	55
DreamSpark.....	55
Firefox.....	55
Git.....	56
NFS.....	56
Copias de seguridad.....	58
GitHub.....	59
ownCloud.....	59
Instalaciones.....	60
Preparativos necesarios para la instalación Windows Server.....	60
Instalación física.....	60
Instalación remota utilizando un almacén en red.....	61
Instalación de Debian.....	62
Instalación y configuración del almacén de imágenes NFS.....	66

Uso de NFS xenserver.....	71
Instalación de Windows Server.....	73
Instalación de Roles y Características necesarias.....	75
Instalación Xen Orchestra.....	77
Instalación manual.....	77
Instalación de Scientific Linux.....	84
Configuración de red.....	85
Instalación de OpenStack mediante RDO Packstack.....	88
Instalación y configuración de ownCloud en Debian.....	95
Instalación.....	95
Configuración HTTP/SSL en ownCloud.....	99
Configuración Apache.....	100
Integración OwnCloud Draw.io.....	103
OpenStack IceHouse.....	107
Historia del desarrollo.....	107
Versiones.....	107
Summits.....	109
Componentes.....	109
Nova (Computación).....	110
Swift (Almacenamiento de Objetos).....	110
Keystone (Identidad).....	111
Horizon (Interfaz Web).....	111
Glance (Imágenes).....	111
Cinder (Volúmenes).....	111
Neutron (Conectividad).....	112
Introducción.....	113
Directorio Activo de Microsoft y OpenStack.....	114
Autenticación en OpenStack.....	115
Creación de usuarios y tenants (proyectos).....	116
Añadir usuarios de servicio de OpenStack al Directorio Activo.....	121
Añadir roles.....	123
Creación la primera máquina en OpenStack.....	126
Descarga de la imagen.....	126
Falta de espacio en disco en el servidor OpenStack.....	131
Extendiendo el tamaño de raíz '/'.....	131
Error al borrar OU desde Active Directory User and Computers.....	135
Problemas con xapi.....	135
Heartbleed.....	138
Explicación.....	138
Solución.....	140
SeLinux activado el máquinas RH.....	140
Reparar la conexión a la biblioteca NFS.....	140
Actualización a la última versión de OpenStack.....	142
Diseño de un entorno multinodo.....	142
Añadir más nodos de computación.....	143
Libros.....	144
Referencias en Internet.....	144

Daniel Fernandez Rodriguez  
“Integración de OpenStack con un Directorio Activo”

Programas.....	144
Instalaciones.....	145



# Índice de figuras

## Tabla de imágenes

Imagen 1: Diagrama de proceso de edición de la documentación.....	12
Imagen 2: Tendencia de ofertas de trabajo relacionadas con OpenStack.....	18
Imagen 3: Diagrama de Gantt.....	26
Imagen 4: Rack en el que se aloja el servidor.....	29
Imagen 5: Diseño Openstack.....	37
Imagen 6: Diagrama de casos de uso para usuario administrador.....	43
Imagen 7: Diagrama de casos de uso para usuario alumno.....	44
Imagen 8: Diagrama de casos de uso para usuario Profesor.....	45
Imagen 9: Elección de idioma en el instalador de Debian.....	63
Imagen 10: Configuración de red.....	63
Imagen 11: Instalación de Debian.....	64
Imagen 12: Añadimos un nuevo almacén de tipo NFS ISO.....	72
Imagen 13: Instalación de Windows Server 2012.....	74
Imagen 14: Configuración de red Windows Server.....	75
Imagen 16: Asistente de instalación de roles y características.....	76
Imagen 17: Página de login de Xen Orchestra.....	83
Imagen 18: Pantalla de inicial de instalación Scientific Linux.....	84
Imagen 19: Scientific Linux proporciona varios tipos de instalación.....	85
Imagen 20: Instalación de la base de datos MySQL para ownCloud.....	98
Imagen 21: Pantalla de login de ownCloud.....	102
Imagen 22: Diálogo de configuración del cliente de escritorio onwCloud....	103
Imagen 23: Panel de activación de complementos en ownCloud.....	105
Imagen 24: Complemento de integración de ownCloud con draw.io.....	106
Imagen 23: Distintos servicios que componen OpenStack.....	110
Imagen 24: Esquema Escenario 1.....	114
Imagen 25: Esquema Escenario 2.....	115
Imagen 26: Estructura de Directorio Activo.....	117
Imagen 27: Diálogo de creación de usuario.....	123
Imagen 28: Error de Xen API desde la interfaz de consola de xenserver.....	136
Imagen 29: Sin espacio en disco en el servidor físico.....	137
Imagen 30: Explicación gráfica de HeartBleed.....	139

# Capítulo 1. Memoria del Proyecto

## ¿Qué se pretende?

Este proyecto cubrirá la documentación e instalación desde cero de la infraestructura necesaria para montar un sistema de nube propia en el que los usuarios serán almacenados y gestionados desde una herramienta externa. Para realizarlo se desplegará un sistema de autenticación basado en la implementación de Microsoft del protocolo LDAP llamado Directorio Activo.

Los usuarios, grupos y permisos serán configurados en Windows Server y se implementará una integración con el sistema de autenticación de la nube desplegada, para que dichos usuarios de estos puedan acceder y utilizar sus recursos.

## Desarrollo del Proyecto

Durante el desarrollo de este proyecto se ha intentado seguir el Principio del Menor Privilegio Posible también conocido por Principle of Least Privilege (término originalmente en inglés). Aplicar este principio a un proyecto como este significa que todas las configuraciones realizadas han intentado cubrir única y exclusivamente, aquello para lo que inicialmente han sido inicialmente pensadas y nada más. Es decir, a la hora de configurar reglas de acceso, permisos, grupos, usuarios, etc., se ha intentado proveer únicamente de los permisos mínimos y necesarios para desempeñar cada una de las tareas para las que han sido diseñadas desde un primer momento. A pesar de que la aplicación de esta práctica pueda parecer obvia y de sentido común, la realidad nos proporciona innumerables ejemplos de exactamente lo contrario y es por ello por lo que quiero hacer un mayor hincapié en este aspecto. Configuraciones donde por error u omisión, mala práctica o simplemente por desconocimiento, dejan expuesto al público más de lo que inicialmente se debería. Por tanto, una configuración de seguridad basada en mínimos siempre nos proporcionará mayor seguridad en caso de que nuestro sistema se vea comprometido.

Un ejemplo práctico de la aplicación de este principio en este proyecto lo tenemos en el caso de los accesos. Para las conexiones remotas a los equipos GNU/Linux, fundamentales debido la idiosincrasia del proyecto, se ha optado por accesos cifrados mediante el protocolo SSH con clave pública/privada. También se ha aplicado en el caso de los firewalls o cortafuegos, donde cada una de las excepciones, necesarias para el correcto funcionamiento de algunos servicios, han sido estudiadas y reducido su impacto únicamente

al uso desde ciertas máquinas controladas de nuestro entorno. Estos son sólo un par de ejemplos de cómo ha sido aplicado este principio.

Aunque en el uso diario, esto pueda parecer una práctica incómoda para el administrador, su no aplicación genera configuraciones adicionales y molestias para el usuario, a la larga está demostrado que genera sistemas software mucho más consistentes, seguros y con una menor exposición a ataques, errores y fallos.

## Desarrollo de la Documentación

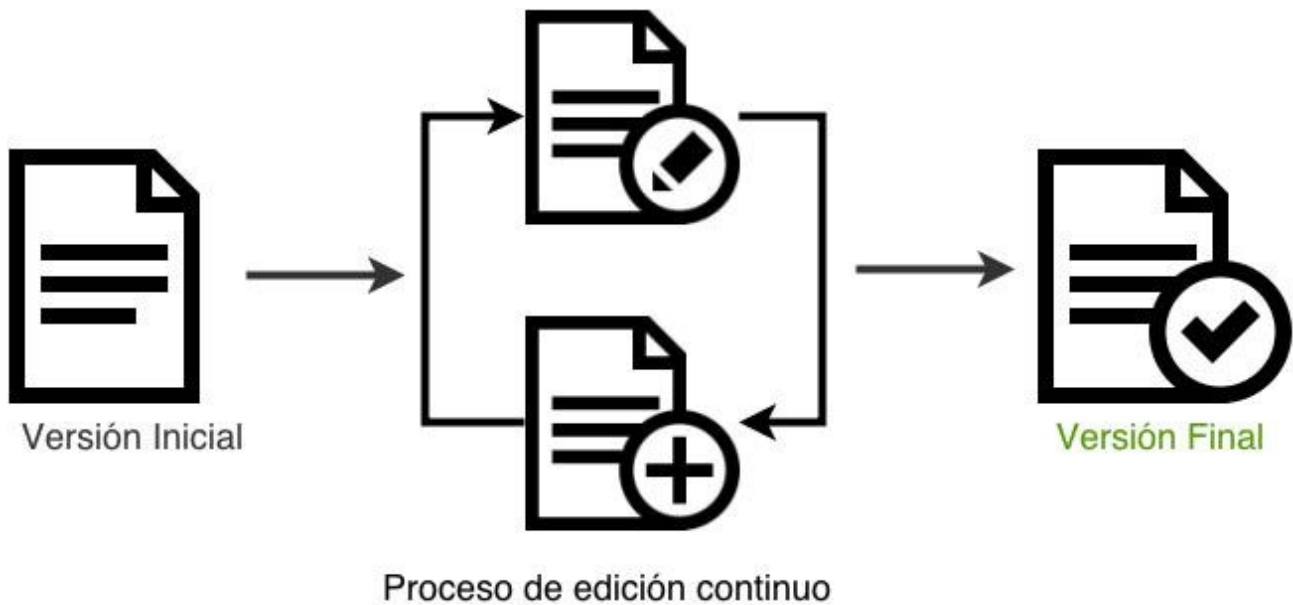
El ciclo de desarrollo de esta documentación también ha intentado seguir algunos los principios de las metodologías ágiles, con el objetivo principal de evitar el conocido -y temido- síndrome de la hoja en blanco. Durante el proceso de documentación se ha aplicado un modelo adaptado de revisiones, propio de las metodologías de desarrollo ágiles como SCRUM.

Para llevarlo a cabo en cada uno de los apartados de este documento, en un primer momento, justo antes de ponerse a escribir lo único que se tendrá claro es el tema del que va a tratar el párrafo pero no cómo se estructurará o se detallará la información. Ese paso de reestructuración y revisión vendrá después, en una de las futuras revisiones.

De esta forma en una primera fase se comienza a escribir prácticamente de una forma anárquica, sin cuidar las formas o el contenido del párrafo más allá de ceñirse al tema en concreto. Lo que se trata es volcar toda la información que se tenga sobre un tema determinado de la forma más pródiga posible. Así mismo y a medida que se escribe irán apareciendo nuevos temas o apartados considerados como interesantes o incluso cabe la posibilidad de subdividir los apartados si así se considera. En ese caso bastara con escribir el título del párrafo y continuar con lo que se estaba escribiendo anteriormente y así con todos los temas y apartados que vayan surgiendo.

En una segunda fase, o ciclo, se releerá toda esa información anteriormente escrita, se corregirán frases, errores ortográficos y posibles faltas ortográficas y se irá añadiendo información de una forma más refinada y cuidada. En esta segunda lectura podrán aparecer nuevos subapartados, reorganizaciones del texto o incluso el borrado de información por considerarse poco atractiva o interesante.

Una vez finalizada esta fase, el trabajo no termina ahí si no que se producirá de nuevo otra iteración en la que se repetirán los pasos descritos en el segundo punto. Este proceso se repetirá hasta llegar a una versión con la suficiente calidad, que pasará a ser considerada como definitiva.



*Imagen 1: Diagrama de proceso de edición de la documentación*

Gracias a este modelo de desarrollo, la documentación se produce prácticamente en tiempo real, evitando así el dejarla para el último momento. Cada nuevo cambio en el proyecto es reflejado en la documentación inmediatamente sin importar demasiado cómo. Poco a poco la documentación ira cogiendo la forma deseada a medida que las distintas iteraciones vayan pasando.

Como nota adicional, añadir que el contenido íntegro de este documento ha sido desarrollado siguiendo este proceso, incluido este mismo apartado.

## Apostando por herramientas abiertas

Otro de los retos accesorios que me presentaba este proyecto era la posibilidad de realizarlo íntegramente con herramientas y programas de código abierto. Personalmente, desde hace ya algún tiempo, tenía ganas de acometer una tarea como esta utilizando herramientas licenciadas como software libre y poder demostrar – y de paso demostrarme a mí mismo también, porque no- que éstas pueden ofrecer una calidad y resultados similares al que proporcionan sus homólogas privativas.

Una de las razones principales que me ha impulsado a embarcarme en este 'reto' es el hecho de que desde hace ya algunos años he ido mudándome paulatinamente a alternativas opensource.

Este proceso, comenzó con pequeños detalles en mi día a día informático como puede ser cambiando el navegador web del sistema o el reproductor de vídeo/música

predeterminado. Aunque a simple vista no pueda parecerlo, este proceso de migración, lleva bastante trabajo, auto-formación, así como búsqueda de alternativas a las herramientas ya conocidas. Aunque después de todo, considero que he conseguido realizarlo con bastante éxito y en el día a día no hay tarea que no pueda realizar usando software libre de las que realizaba desde su equivalente en funcionalidad de código cerrado.

Por tanto intentaré seguir esta filosofía de apostar por desarrollos y herramientas libres en el desarrollo de este proyecto y así verificar si es posible alcanzar unos objetivos y resultados que cumplan unos requisitos de calidad similares a los que podríamos conseguir con herramientas cerradas. La teoría nos dice que sí que es posible, pero muchas veces llevar esto a la practica no resulta tan sencillo.

A pesar de que hoy en día podríamos decir que los productos opensource están cada vez más en boga y existe una amplia variedad de herramientas que nos proporcionan alternativas de gran calidad y fiabilidad, en numerosas ocasiones su homólogo cerrado ofrece mayor sencillez de uso, fiabilidad ante errores, soporte y calidad del producto generado.

Por tanto, siendo inicialmente consciente de esta limitación, durante el desarrollo de el proyecto se intentarán utilizar en la medida de lo posible herramientas gratuitas, opensource y de código abierto. Así mismo se mantendrá una pequeña máquina virtual con un sistema Microsoft Windows de escritorio para realizar algunas de las configuraciones iniciales, que por circunstancias ajenas a este proyecto, sólo es posible realizar desde Windows. Esta máquina me servirá de apoyo durante todo el desarrollo y me brindará, en el caso de ser necesitada, la oportunidad de utilizar alguna herramienta únicamente disponible en entornos Microsoft Windows.

Está documentación, también seguirá la premisa anteriormente explicada y como prueba de ello está siendo escrita íntegramente utilizando la suite ofimática LibreOffice.

Una vez explicado lo anterior, me gustaría remarcar que en ningún momento el cumplimiento de esta premisa se convertirá en un dogma inquebrantable. Es decir, el uso de software y herramientas libres es un -deseable- posible camino y no un fin en sí mismo, por lo que en el caso de que no poder cumplir con el principal objetivo del proyecto, romperlo no supondrá un problema. En cualquier caso quiero dejar claro que en ningún momento se comprometerá la calidad final del trabajo o los resultados por mantenerlo vigente aunque sí haré un esfuerzo adicional para intentar cumplirlo.

## Disponibilidad del proyecto

Una vez finalizado este proyecto toda la documentación, ficheros de configuración, scripts

Daniel Fernandez Rodriguez  
“Integración de OpenStack con un Directorio Activo”

utilizados, lista de software disponible, así como las máquinas virtuales generadas durante este proyecto, se mantendrán en un repositorio público para su descarga, revisión y/o modificación por parte de la comunidad de usuarios.

# Resumen de todos los aspectos

## Introducción

Este primer capítulo servirá para ir introduciendo al lector en el proyecto, planteando el porqué del proyecto, lo que se pretende conseguir con él y la justificación de algunas de las decisiones más relevantes que se han tomado al comenzar este desarrollo.

## Aspectos teóricos

En este apartado se describirán brevemente aquellos conceptos, herramientas y tecnologías existentes en el proyecto que tengan una relación importante con el producto a realizado.

## Planificación del Proyecto y Resumen de Presupuestos

Describirá el plazo de ejecución del proyecto, mediante un diagrama de Gantt desde el que podrá verse el tiempo que se asigna a cada tarea, hitos, etc. en este apartado también se dedicará una pequeña tabla con el presupuesto del proyecto desglosado por fases.

## Análisis

Este capítulo tiene como objetivo la obtención de una especificación detallada del sistema de información que satisfaga las necesidades de información de los usuarios y sirva de base para el posterior diseño del sistema.

## Diseño del sistema

El objetivo del apartado de Diseño del sistema es la definición de la arquitectura del sistema y del entorno tecnológico que le va a dar soporte, junto con la especificación detallada de los componentes del sistema de información.

## **Implementación del sistema**

El objeto de este apartado es describir los elementos necesarios para la elaboración del sistema (herramientas, lenguajes, programas, etc.) así como detallar todos los conceptos que tengan relación con la creación del sistema.

## **Problemas encontrados**

Este apartado detallara algunos de los problemas encontrados a lo largo de este desarrollo.

## **Ampliaciones**

En él se detallarán las posibles ampliaciones que podrían ser interesantes de cara a un posible futuro desarrollo derivado de este.

## **Referencias bibliográficas**

Referencias utilizadas a lo largo de todo el ciclo de desarrollo del proyecto.



## Capítulo 2. Introducción

### Justificación del proyecto

Este proyecto busca conseguir la integración de OpenStack, un sistema de computación de nube con un Directorio Activo de Microsoft. Si bien se tratan de herramientas completamente distintas y sin a priori ninguna relación entre sí, existe la posibilidad de configurar la primera para que confíe en usuarios, contraseñas y otros elementos relacionados con la identificación y autenticación de un almacén externo.

El soporte de OpenStack con respecto al Directorio Activo es aún a día de hoy, bastante limitado y a pesar de que con las nuevas versiones del producto, ha mejorado en gran medida, su puesta en funcionamiento aún no resulta demasiado común. Además, encontrar documentación fiable y probada relacionada con este tema es bastante complicado y mucho menos en nuestro idioma.

A pesar de ésto, no sería extraño encontrar empresas u organizaciones a día de hoy que hayan llevado a la práctica lo propuesto en este proyecto. Aún así, considero que puede ser un ejercicio muy interesante el tener que lidiar desde prácticamente cero, con todos los actores involucrados en este sistema. Instalaciones, configuraciones, reglas, excepciones, parches... todos estos tendrán que ser revisados y colocados en su sitio para que todas las piezas de este puzzle a construir encajen de la mejor forma posible. Además, es necesario tener en cuenta que para llevarlo a cabo se necesitan unos conocimientos de herramientas de ámbitos completamente opuestos como son Windows GNU/Linux por lo que creo que su realización me dará unos conocimientos de gran utilidad que podrán ser aplicados no solo durante este desarrollo si no también durante mi carrera profesional.

### Objetivos del Proyecto

Independientemente de la experiencia previa que se pueda tener con los conceptos que rodean a este proyecto, el objetivo principal de éste es bastante claro. Como se puede leer en el propio título de este proyecto, el desarrollo buscará principalmente la integración de un OpenStack con un Directorio Activo de Microsoft.

A pesar de ser éste el objetivo primordial, existen varios secundarios que también será necesario tener en cuenta para así poder entender bien cómo ha sido realizado el proyecto y su magnitud.

En primer lugar, todos los componentes de este proyecto se desplegarán sobre un único servidor físico en el cual descansará toda la estructura virtual necesaria para poder llevarlo a cabo. Debido a esto, a lo largo del desarrollo de este proyecto se adquirirán capacidades de uso de sistemas de virtualización. Además las propias máquinas virtuales utilizadas, harán uso de sistemas distintos como son Windows y GNU/Linux por lo que adquirir capacidades relacionadas con uso y entender las bases de su funcionamiento, también será uno de los objetivos. Personalmente creo que estos dos últimos puntos, son muy interesante y puede ayudar a completar y enriquecer el perfil de un profesional dedicado a la Informática hoy en día.

Otro objetivo que podemos extraer del principal, es el de poder conocer y utilizar uno de los proyectos de software libre más prometedores y con mayor crecimiento de los últimos años, el proyecto OpenStack.

El crecimiento y uso desde su lanzamiento en el año 2010 ha sido exponencial y su interés a nivel mundial se ha visto multiplicado año tras año. Como se puede apreciar en la siguiente gráfica confeccionada por el portal de empleo en línea indeed.com, se puede observar la creciente demanda de puestos de trabajo relacionados con éste producto así como su evolución con respecto a sus alternativas en el mercado.



*Imagen 2: Tendencia de ofertas de trabajo relacionadas con OpenStack*

A pesar de que los datos expuestos en el gráfico sólo cubren hasta mediados del año 2012, se ha considerado interesante utilizarlos para así poder mostrar a simple vista cual es la tendencia que sigue este desarrollo.

## Estudio de la Situación Actual

A pesar de que desde el principio de este proyecto se tenía decidido que se utilizaría OpenStack como sistema de computación en la nube, en la actualidad existen multitud de alternativas a éste en el mercado. Cada una de estas presenta un enfoque completamente válido, por lo que puede resultar provechoso tenerlas en cuenta.

Aún así, es importante recalcar que cada opción aquí presentada, aunque similares en concepto, viene a cubrir unos casos de usos particulares por lo que optar por una u otra alternativa puede conllevar a ventajas o inconvenientes frente a otras.

## Evaluación de Alternativas

### Eucalyptus

Eucalyptus es un producto software gratuito y de código abierto con el que poder desplegar nubes de carácter privado así como híbrido. Esta herramienta se diferencia del resto de opciones, en que pone el acento en proporcionar la máxima compatibilidad posible con la nube de Amazon, denominada Amazon Web Services (AWS). Aunque no lo parezca a simple vista, su nombre se trata del acrónimo en inglés de “Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems”.

Su desarrollo comenzó en el año 2008 y tras una gran crecimiento en sus primeros compases fue adquirida a mediados de 2014 por la empresa americana Hewlett-Packard. La última versión estable disponible para el gran público es la 4.1.0.

Como en numerosos casos dentro del software libre, Eucalyptus es ofrecida en tres distintas versiones del producto: una versión comunitaria y totalmente gratuita y otras dos de pago por licencia más orientadas a entornos profesionales donde en las que se añaden configuraciones adicionales y soporte directo por parte de la empresa.

### Apache CloudStack

CloudStack es la solución software de computación en la nube que propone la Fundación Apache. Como en otros casos, CloudStack ofrece soporte a las ampliamente utilizadas interfaces de Amazon Web Services así como a Open Cloud Computing Interface (OCCI) de la Open Grid Forum lo que la convierte en una apuesta muy versátil.

CloudStack comenzó su andadura en el año 2010 desarrollada por la empresa americana Cloud.com, para años más tarde ser comprada por el gigante Citrix que a su vez, unos años

pasaría el testigo del proyecto a manos de la Fundación Apache en donde ha permanecido hasta la actualidad. Apache CloudStack esta desarrollado en Java y licenciado bajo Apache License 2. Su última versión disponible, de la rama 4.X, fue lanzada al público en enero de 2015.

## VmWARE vCloud air

vCloud air es el nombre del producto para la nube privada de la empresa VmWare. Está desarrollado íntegramente sobre el también producto de la compañía, vSphere por lo que su orientación tiende más hacia clientes familiarizados con los productos de virtualización de VmWare. Además, vCloud air proporciona características que facilitan en gran medida su puesta en marcha y manejo si ya se dispone de una infraestructura basada en los productos de virtualización de VmWare.

Como en otros desarrollos de productos similares, con la idea de ampliar espectro de posibles usuarios, VmWare ofrece distintos planes de precios adaptados a las diferentes necesidades de los usuarios, pudiendo seleccionar características y funcionalidades a la carta.

En contraposición con otras opciones aquí detalladas, la licencia de VmWARE vCloud air no es de código abierto.

## OpenNebula

OpenNebula es una plataforma de cloud computing gratuita y de código abierto desarrollada y mantenida íntegramente por la Comunidad de Usuarios. Este desarrollo ofrece soporte a nubes públicas, privadas e híbridas así como a las distintas interfaces ofrecidas por Amazon Web Services (AWS), Open Cloud Computing Interface (OCCI), vCloud y una de carácter propio. Como en el caso de Apache CloudStack, OpenNebula ofrece una gran flexibilidad.

En cuanto al soporte a distintos hipervisores, OpenNebula tampoco se queda atrás y ofrece compatibilidad con tecnologías Xen, KVM y VMWare.

OpenNebula es completamente código abierto y está licenciado como Apache License versión 2. Para terminar, simplemente comentar que este desarrollo ha gozado de una popularidad enorme en el pasado, aunque a día de hoy se ha visto desplazada por otras alternativas como OpenStack.

## Google Compute Engine

Como no podría ser de otro modo, el gigante americano del buscador también proporciona

su propio servicio de computación en la nube, llamado Google Compute Engine o GCE.

Si bien este desarrollo tiene una particularidad que lo diferencia enormemente de la mayoría de alternativas vistas en esta sección: y es que su utilización está orientada únicamente y exclusivamente hacia el uso de los propios servidores de Google y no a su instalación y despliegue en servidores de carácter particular.

Gracias a esto Google, proporciona de una forma rápida y sencilla acceso a multitud de software así como diversos sistemas operativos para que desplegar el servicio no suponga ningún tipo de problemas a empresas de cualquier tamaño.

Además, aparte de la anteriormente mencionada hay otras características que lo diferencian de otros servicios similares como la alta capacidad a nivel de hardware de sus servidores, balanceo de carga global y como no, el precio por servicio. Sin duda la la amplísima infraestructura mundial con la que cuenta Google, juega a favor del coloso del buscador para poder ofrecer un servicio de gran potencial e imbatible en muchos aspectos.

## Microsoft Azure

Azure es un caso parecido al de Google pero esta vez con el sello de confianza que brinda Microsoft. Como en el caso de Google, Azure cuenta con todas las ventajas de tener una gran empresa detrás y además en este caso añade completa integración de la mayoría de los servicios Microsoft como, la suite ofimática de Microsoft, bases de datos SQL, Active Directory, servidor de correo Exchange o incluso Share Point.

Sin duda esta es la gran baza del producto de Redmond para la competir en un mercado tan reñido a día de hoy como es el de la virtualización y los servicios en la nube.

A día de hoy la infraestructura informática de muchas empresas utiliza herramientas de Microsoft por lo que migrar hacia una solución de computación en la nube que hace uso de estas tecnologías puede ser la solución menos traumática tanto para los usuarios como para los administradores de la plataforma.

Por último, y para enfrentarlo con otras alternativas aquí nombradas, a pesar de que Microsoft parece estar cambiando su política interna con relación al software libre, ninguna de las tecnologías relacionadas con Azure los son, ni está previsto que lo sean en el corto medio plazo. Aún así, la nueva política de negocio de Microsoft, hace que sea posible probar y utilizar muchas de estas herramientas sin coste alguno por lo que puede ser una fantástica solución para ciertos casos.

## Capítulo 3. Aspectos Teóricos

En este apartado se va a proceder a explicar algunos de los conceptos base para entender bien las partes significativas del proyecto y así poder comprender mejor es lo que se pretende realizar en este desarrollo.

Cada término está explicado de manera independiente para introducir al lector en los conceptos que se han considerado claves. No pretende ser una guía exhaustiva de cada término sino más bien un pequeño texto en el que se explique de manera fácil y accesible a todos los posibles lectores (independientemente de sus conocimientos en el tema), algunos de los términos básicos que se van a manejar durante toda la documentación del proyecto y sobre los que se sustenta este proyecto.

### ¿Qué es la nube?

La “nube” es el término coloquial que se le suele dar al modelo de entrega de recursos informáticos, aplicaciones y contenido software bajo demanda a través de Internet (Amazon).

Desde un punto de vista no técnico podríamos considerarla como un conjunto de recursos informáticos disponibles en alguna parte de la que desconocemos su ubicación, pero de los cuales podemos utilizar todo aquello que necesitemos, en el preciso momento que lo necesitemos. Además una vez que dejemos de utilizar dicho recurso, el sistema de “nube” nos permitirá devolverlo para que otros clientes del servicio puedan reutilizarlo a su vez.

Para entender este concepto de forma más clara, pongamos un ejemplo sencillo con el que posiblemente todos estemos familiarizados. A día de hoy, prácticamente cualquier usuario de Internet posee una cuenta de correo, ya sea de carácter personal o profesional, con la que poder recibir y enviar correos electrónicos.

Normalmente cada una de estas cuentas, dispone de un espacio limitado para almacenar todos los archivos relacionados con el correo y su manejo: bandeja de entrada, mensajes enviados, borradores, etc. Así, en el caso de recibir un correo que contenga algún fichero adjunto, como fotografías, vídeo o incluso documentos de texto, es posible acceder a ese mismo correo desde diversas localizaciones y del mismo modo seguir teniendo acceso al contenido del mensaje y sus archivos adjuntos.

No importa si ingresamos desde nuestro teléfono, tableta, ordenador personal o desde casa de un amigo o familiar, siempre y cuando accedamos a nuestra cuenta de correo,

tendremos disponibles nuestros archivos. Pero entonces, ¿dónde es que se almacenan estos archivos? ¿En nuestros dispositivos? ¿En el propio navegador? No. La respuesta es: en 'la nube'.

Obviamente nuestros archivos se encuentran alojados en los servidores físicos en algún lugar del mundo. Para este ejemplo concreto, estarán almacenados en los servidores de nuestro proveedor de servicio de correo, esperando a ser accedidos por los usuarios del mismo. De esta forma no ocupan lugar en nuestros dispositivos y estarán siempre accesibles desde cualquier ubicación siempre y cuando se disponga de una conexión a Internet con la que poder conectarse.

Una vez entendido el ejemplo anterior, ampliemos este concepto no sólo a almacenamiento si no también a capacidad de cálculo o memoria RAM de la que disponemos para realizar tareas. Es decir, podríamos adquirir cierta capacidad de proceso de manera puntual para desempeñar un trabajo concreto y una vez finalizado, simplemente devolverlo. Y lo más importante, sin preocuparnos de tiempos de envío, retrasos, montajes, instalación, etc.

Los recursos de la nube están siempre disponibles y a completa disposición de los usuarios para ser utilizados.

Esto genera un modelo de negocio donde hay empresas que se dedican a ceder su infraestructura de en la nube a organizaciones, otras empresas o incluso a particulares por periodos limitados de tiempo, por una cantidad económica previamente pactada.

## Algunos pros y contras del modelo en la nube

Como todo, el modelo en la nube tiene aspectos positivos y otros negativos. En este apartado veremos algunos de ellos.

Es difícil negar que la nube nos proporciona algunas características que nos hacen la vida mas cómoda y fácil. Algunos ejemplos, como el simple hecho de poder acceder a nuestros datos desde cualquier ubicación o desde cualquier dispositivo con una conexión a Internet, es posiblemente el más destacado de todos ellos.

Gracias a la nube, esa *tediosa* tarea de compartir contenido en formato físico entre tus conocidos en CD/DVS o lápices USB ha terminado. Hoy en día, bastará con almacenar dichos datos en tu cuenta en la nube, enviar un enlace al interesado y de forma inmediata dispondrá acceso a dichos archivos para poder descargarlos o simplemente revisarlos en línea. Así de fácil.

Además, debido a que nuestros datos no se encuentran almacenados directamente en nuestros dispositivos, podemos dedicar ese espacio sobrante para otras cosas. Es más, dependiendo de nuestro proveedor de servicios de nube es muy posible que incluso se



realice una copia de seguridad automática de nuestros datos, por lo que además podemos despreocuparnos totalmente de tener que realizar duplicados, etiquetarlos y almacenarlos...

Por último otros de los factores claves, el precio. Con el paso de los años los servicios basados en la nube están haciéndose cada vez más y más populares y en consecuencia el coste del servicio baja haciendo este tipo de servicios aún más atractivos si cabe para un mayor número de personas.

Como no todo podía ser perfecto, este modelo también conlleva una serie de contras que debemos tener en cuenta antes de lanzarnos a la nube.

En primer lugar debemos ser conscientes de que existe una pérdida del control sobre aquellos datos que confiamos a la nube. Resulta bastante obvio que guardando nuestra vida digital en la nube, ganamos en comodidad pero perdemos conocimiento de dónde se encuentran, quien los tiene alojados o incluso de quien o quienes tienen acceso a ellos. Esto se hace más acuciante con los escándalos de los últimos tiempos relacionados con espionaje y filtraciones por parte de gobiernos. Parece que todo lo almacenado en la nube es propenso a ser espiado por los servicios de inteligencia de algunos países u organizaciones, alegando motivos de seguridad. Esta preocupación surge a raíz de la restricción del propio modelo de tener que confiar en una tercera parte para almacenar tus datos. Dentro de este sistema ya no tenemos control completo sobre dónde y cómo se almacenará nuestros datos, por lo que su utilización no está indicada para todos los casos de uso.

Otro de los inconvenientes es la necesidad de estar siempre conectados a Internet, lo cual puede resultar un gran problema en determinados entornos. Siempre y cuando dispongamos de una conexión estable y con velocidades decentes no habrá problema, pero en caso contrario, acceder a nuestro datos pueden convertirse en una tarea tediosa y en el peor de los casos, imposible.

# Capítulo 4. Planificación del Proyecto y Resumen de Presupuestos

## Planificación

En este apartado se muestra el diagrama de Gantt, que se corresponde con la planificación del proyecto. El diagrama ha sido realizado mediante la herramienta multiplataforma Planner, disponible para el entorno de escritorio Gnome, así como para Windows .

Debido a las grandes dimensiones del diagrama y a la imposibilidad de ser mostrado de forma completa en este documento, se recomienda revisar el gráfico en directamente a través de su archivo digital y sin las restricciones de tamaño aquí expuestas.



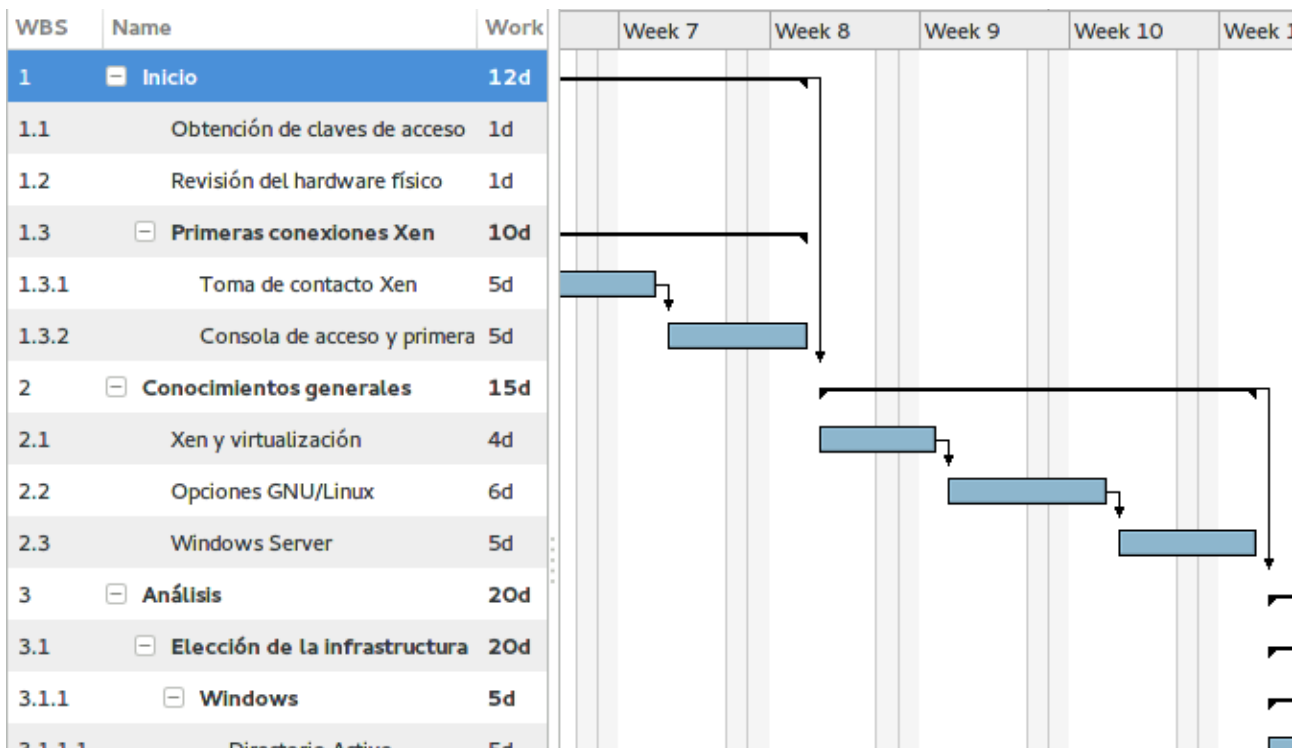


Imagen 3: Diagrama de Gantt

En cuanto al diseño de esta planificación, el desarrollo del proyecto se ha compaginado con una jornada laboral a tiempo completo, por lo que dependiendo de la semana y época del año el tiempo dedicado al proyecto ha sido variable. Debido a esto y para facilitar su posterior revisión y lectura se condensará toda la planificación en un pedido fijo de 6 meses. De esta forma se conseguirá obtener una imagen global de cada fase, gracias a la ponderación temporal de cada uno de los hitos.

La duración del proyecto va desde el 1 de febrero hasta el día 1 de julio de 2015. La planificación se ha ido revisando cada vez que alcanzaba un hito, quedando finalmente tal y como se puede apreciar en el diagrama de Gantt.

La planificación está dividida en cinco tareas principales y éstas a su vez se encuentran divididas en varias subtarefas. En el siguiente párrafo se comentará de forma breve cada una de estas tareas, para así obtener una visión global de la planificación del proyecto.

La primera tarea ha sido denominada Inicio, es la temporalmente más corta de toda la planificación y comprende la primera toma de contacto con el proyecto. La segunda tarea se denomina Conocimientos generales, tiene una duración de 15 días y en ella se comenzaron a estudiar los conceptos básicos que tenían que ver con el proyecto, también en este punto se tomaron las primeras decisiones importantes, como cuales serían los sistemas operativos a utilizar.

La siguiente tarea se corresponde con el Análisis, en ella se comienza a investigar sobre cómo será el sistema y qué se podrá esperar de él. Una vez terminado el Análisis, llega el Diseño, desde el que se continúa el trabajo iniciado en la fase anterior y se complementa.

Después de estas dos fases con aspectos teóricos, comienza la de Implementación, en la que siguiendo las directrices generadas en el apartado de Diseño se comenzará a llevar a cabo las instalaciones y configuraciones necesarias.

## Presupuesto

En la tabla siguiente podemos observar el presupuesto que se ha realizado para este desarrollo. Cada fase está desglosada según su precio y la cantidad de tiempo necesaria para llevarla a cabo.

Item	SubItem	Concepto	Horas	Precio unid	Total
01		Desarrollo del proyecto:			
	001	Análisis	65	50,00 €	3.250,00 €
	002	Diseño	80	60,00 €	4.800,00 €
	003	Implementación	200	40,00 €	8.000,00 €
	004	Pruebas	75	55,00 €	4.125,00 €
				Subtotal	20.175,00 €
				IVA (21%)	4.236,75 €
				TOTAL	24.411,75 €

## Capítulo 5. Análisis

### Descripción del servidor físico

Para llevar a cabo este proyecto la Facultad de Ingeniería Informática de Oviedo me cedió uno de sus servidores sobre el que poder implementar, desplegar y probar libremente mi entorno de desarrollo al completo.



*Imagen 4: Rack en el que se aloja el servidor*

El grupo de Cloud Computing me proporcionó acceso completo a uno de los servidores físicos alojados en uno de los armarios de la sala de máquinas de la Facultad de Ingeniería Informática de Oviedo.

El servidor cedido se corresponde a un modelo Dell PowerEdge R710 con 2Tb espacio en

disco y 128Gb de RAM.

Desde el grupo de Cloud me también me proporcionaron las credenciales del usuario administrador 'root' así como acceso remoto SSH posibilitando su configuración a distancia.

El servidor, está gobernado por Xen Server en su versión 6.2. Xen es una distribución GNU/Linux derivada de RedHat Enterprise Linux (RHEL) y es sobre la cual se sustenta toda la infraestructura necesaria para el funcionamiento de este proyecto.

Para finalizar, me gustaría remarcar que tanto la instalación de Xen Server 6.2 en el servidor físico, como la configuración de acceso de mi usuario por SSH, fue realizada directamente por los miembros del grupo de Cloud Computing de la Escuela.

A continuación una lista a modo de resumen de la configuración del servidor:

- Nombre de la máquina: **xenserver-05**
- Hardware
  - Fabricante: Dell Inc.
  - Modelo: de servidor: PowerEdge R710
- Información de BIOS
  - Fabricante: Dell Inc
  - Versión: 3.0.0
- Procesador
  - Modelo: Intel(R) Xeon(R) CPU E5504
  - Velocidad de reloj: 2.00GHz
  - CPUs físicas: 2
  - CPUs lógicas : 8
  - CPUs Sockets: 2
- Memoria
  - Disponible : 122880 MB
  - Sockets de memoria: 18
- Controladores de almacenamiento
  - Interfaz IDE: Dell PowerEdge R710 SATA IDE



- Interfaz RAID: Controlador de BUS RAID: Dell PERC 6/i Integrated RAID
- Interfaces de Red disponibles
  - Interfaz de red principal
    - Nombre: eth0
    - MAC: 00:26:b9:85:5a:26
    - Velocidad: Gigabit
  - Interfaz de red secundarias
    - Nombre: eth1
      - MAC: 00:26:b9:85:5a:28
      - Velocidad: Gigabit
    - Nombre eth2
      - MAC: 00:26:b9:85:5a:2a
      - Velocidad: Gigabit
    - Nombre: eth3
      - MAC: 00:26:b9:85:5a:26
      - Velocidad: Gigabit

En cuanto a la configuración de Red:

- MAC: 00:26:b9:85:5a:26
- IP: 156.35.94.186
- Máscara: 255.255.254.0
- Puerta de enlace: 156.35.94.201

Sistema Operativo:

- Xen Server versión 6.2.0-70446c (xenenterprise)

Así mismo el grupo de Cloud de la Facultad de Informática me proporcionó un rango de

direcciones IP con salida directa a Internet (sin proxies ni saltos intermedios) con las que poder configurar las máquinas alojadas sobre el xenserver-05. Estas IPs se encuentran dentro del rango 156.35.95.20-156.35.95.24 con máscara de red de 24 bits.

Lista de IPs disponibles:

- 156.35.95.20, 156.35.95.21, 156.35.95.22, 156.35.95.23, 156.35.95.24

Máscara de red:

- 255.255.255.0

## Infraestructura necesaria

En este apartado veremos como fue instalado el sistema sobre el que se desplegaría este proyecto, así como la descripción de algunos de los problemas que he ido encontrando mientras realizaba el proceso. En el Capítulo 9 de esta documentación, se dedica un apartado completo a repasar algunos de los problemas encontrados que se han considerados como más importantes.

## ¿Qué se pretende?

Esta es posiblemente la pregunta más importante qué uno debe realizarse antes de iniciar un proyecto de una envergadura como esta. No se pretende comenzar a desplegar máquinas sin ton ni son, duplicando funcionalidades, configuraciones y muy probablemente, futuros dolores de cabeza...

En primer lugar antes de meterse manos a la obra con las instalaciones propiamente dichas es necesario realizar un análisis de qué es lo que se pretende, objetivos que se intentan alcanzar y qué elementos serán necesarios para llevarlos a cabo: herramientas, requisitos funcionales, requisitos hardware, etc.

Este trabajo está titulado “Integración de OpenStack con Active Directory” por lo sin necesidad de darle demasiadas vueltas, resulta bastante obvio que como mínimo, el proyecto consistirá en al menos dos máquinas bien diferenciadas.

La primera de ellas será un Windows Server que adoptará el rol de Directorio Activo mientras que la segunda corresponderá a una máquina Linux con los componentes de OpenStack desplegados y configurados para su correcto funcionamiento.

A continuación veremos la justificación de algunas de las decisiones tomadas relacionadas con este apartado.

## Elección de Windows

El caso de la decisión en la parte Windows, no se presentaban muchas dudas mas allá de la elección del tipo de licencia a utilizar.

Desde un primer momento se tuvo muy claro que se optaría por un sistema operativo Windows en su versión disponible orientada para servidores, llamada comercialmente Microsoft Windows Server.

En relación a esto, en el momento de escribir líneas, la última versión disponible en el mercado de este producto se trata de Windows Server 2012 R2. Desafortunadamente para este proyecto, para su instalación únicamente se disponía de una licencia de estudiante que sólo cubría la instalación de la versión anterior, llamada comercialmente Windows Server 2012.

Microsoft siempre añade nuevas funcionalidades así como configuraciones en cada una de sus versiones, por lo que a veces, utilizar una versión un tanto desactualizada puede suponer algunos quebraderos de cabeza adicionales. De todas formas, a pesar de no ser la última versión disponible en el mercado, es importante recalcar que para el cometido de este proyecto prácticamente cualquiera de las versiones actuales de Windows Server cumple de forma más que suficiente.

Desde hace ya varias décadas, los productos de Microsoft destinados a servidores han mejorado notablemente, afianzándose con el paso de los años como solución más extendida entre las pequeñas y medianas empresas (PYMES) e incluso entre las más grandes. La integración con las otras herramientas propias de la empresa, su sistema de formación y certificaciones, el amplio soporte de los fabricantes hardware y comunicaciones, posicionan a Microsoft Windows Server como una de las opciones más consolidadas para su uso empresarial, así como en proyectos de prácticamente cualquier tipo.

Por todo esto, considero que conocer y estar familiarizado algunas de las configuraciones básicas así como conceptos de administración relacionados con sistemas Windows es algo muy importante que complementa y enriquece el perfil de un profesional dedicado a la informática.

## Licencias

En cuanto al tipo de licencia elegida, la compañía de Redmond suelen optar por crear distintas licencias para un mismo producto. Dependiendo de la inicialmente elegida unas u otras opciones estarán posteriormente disponibles en el producto. A juicio de Microsoft, este modelo hace sus productos más atractivos de cara a los distintos perfiles de cliente final. Ellos mismos son conscientes de lo complicado que puede llegar a ser entender los

diferentes tipos de licencia y dejan a disposición de los usuarios un documento con la descripción detallada de cada una de las versiones.

Por razones obvias relacionadas con la funcionalidad final del proyecto, en este caso sólo me centraré en aquellas versiones que ofrecen como mínimo la característica de Directorio Activo. El resto de ellas, aunque igualmente válidas para otros usos, no cumplen el cometido básico de este proyecto por lo que no serán tenidas en cuenta.

De todas las distintas versiones ofrecidas en el catálogo de Microsoft en su versión Windows Server 2012, serán evaluadas únicamente las siguientes:

- Windows Server 2012 Datacenter
- Windows Server 2012 Standard
- Windows Server 2012 Essentials
- Windows Server 2012 Foundation

Para facilitar la comparación entre ellas, a continuación se mostrarán las distintas versiones enfrentadas por características en formato de tabla.

	Versiones de Windows Server 2012			
	Datacenter	Standard	Essentials	Foundation
Limitaciones de versión				
Número máximos de usuarios	Por licencia	Por licencia	25	15
RAM (máxima)	4TB	4TB	64GB	32GB
¿Capaz de unirse a un dominio?	Sí	Sí	En caso de migración	En caso de migración
Roles de servidor presentes				
Servicio de Certificados de Directorio Activo	Sí	Sí	Sí	Sí
Servicio de Dominio de Directorio Activo	Sí	Sí	Requerido	Sí (opcional)
Servicio de Federación de Directorio Activo	Sí	Sí	Sí	Sí
Servicios ligeros de Directorio Activo	Sí	Sí	No	Sí
Servidor de Aplicaciones	Sí	Sí	Sí	Sí
Servidor DHCP	Sí	Sí	Sí	Sí
Servidor DNS	Sí	Sí	Sí	Sí
Servidor de archivos	Sí	Sí	Sí	Sí
Características de servidor presentes				

Controlador de dominio sólo de lectura	Sí	Sí	No	No
--	----	----	----	----

## Decisión

Una vez estudiada la comparativa, siendo realista y teniendo siempre presente el que será el alcance final del proyecto, todas las versiones anteriormente descritas nos proporcionan una experiencia parecida por su similitud de características ofrecidas, por lo que es más que probable que todas cumplan con creces los requerimientos básicos de este proyecto.

Además como ya había comentado de forma breve anteriormente, el precio de la licencia, muchas veces determinante, no supone un problema puesto que en este proyecto se utilizará las licencias gratuitas de estudiante proporcionadas por la plataforma de la propia Microsoft, DreamSpark.

Teniendo en cuenta todo lo anteriormente comentado, mi decisión final se inclinó por Windows Server 2012 en su versión Standard, simplemente por cuestión de simplicidad.

La versión Standard incluye prácticamente todas las funcionalidades que se requieran hoy en día en un servidor de uso general.

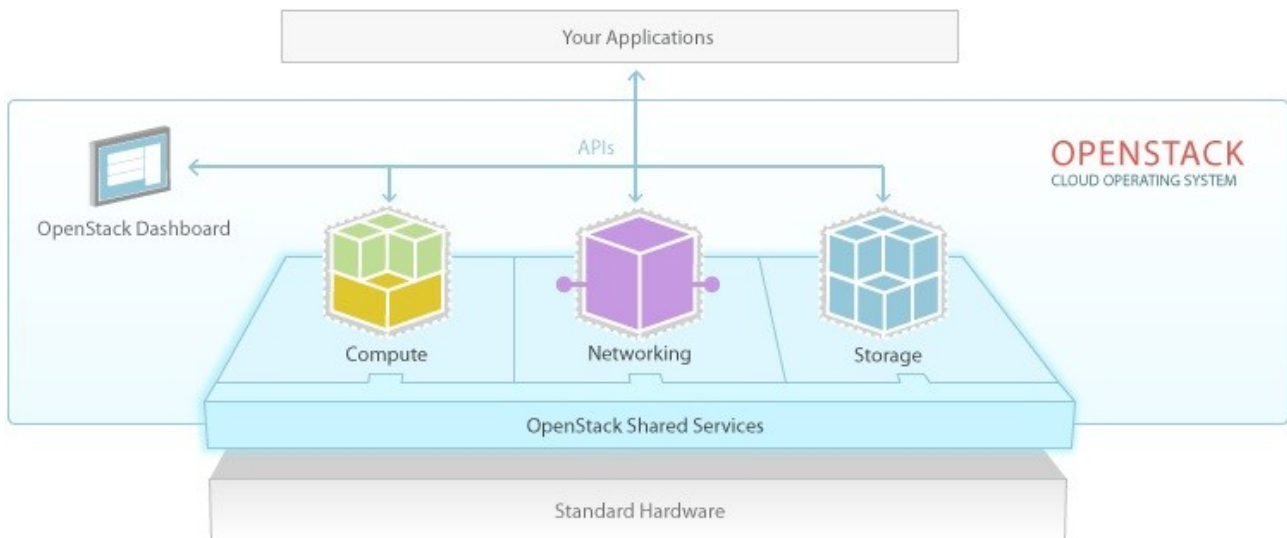
En un entorno de producción real, con costes reales, es más que probable que podría haberse optado por una licencia no tan genérica, con el objetivo de reducir el gasto del presupuesto en licencias del proyecto.

## Elección de OpenStack

Como en el caso anterior, la elección de OpenStack como el sistema de computación en la nube a utilizar en este proyecto no resultó demasiado compleja. Como se expuso anteriormente en el apartado de dedicado al estudio de la situación actual, a día de hoy existen en el mercado distintas opciones con características similares y que podrían utilizarse exitosamente para un caso similar al aquí planteado. A pesar de todas estas alternativas, considero que OpenStack es el más extendido así como aquel que tiene una mayor proyección de futuro.

OpenStack es un proyecto joven con una comunidad de usuarios y desarrolladores en expansión y que ya está siendo utilizada en grandes organizaciones como Hewlett-Packard, RackSpace, Yahoo!, Ebay, Paypal, el CERN o la NASA, entre otras.

OpenStack es instalado como una capa software adicional sobre el sistema operativo en la que se despliegan sus distintos módulos y servicios.



*Imagen 5: Diseño Openstack*

Por tanto para poder instalar OpenStack, previamente debemos elegir sobre qué sistema operativo lo desplegaremos.

## Elección de la distribución GNU/Linux

En este apartado veremos cuales han sido las distintas opciones barajadas así como la escogida finalmente, dado que en contraposición al caso de Windows, sí que han habido varios firmes candidatos de procedencia muy diversa desde un primer momento.

Cada una de las siguientes distribuciones será juzgada en base a los siguientes puntos:

- Grado de actualización de sistema VS madurez y estabilidad
- ¿Es soportada oficialmente por OpenStack?
- ¿Distribución desarrollada por empresa/fundación o comunitaria?
- Tamaño de comunidad de usuarios

Aunque antes de comenzar con este apartado me gustaría remarcar que prácticamente cualquiera de las opciones aquí barajadas, podrían haber sido válidas para este desarrollo, si bien es cierto que la experiencia OpenStack que nos hubiera brindado cada una de ellas diferiría en algunos aspectos de las de las otras.

## Ubuntu (y derivadas)

Ubuntu es posiblemente una de las tres distribuciones GNU/Linux más conocidas y

comentadas a nivel mundial. Basada en la robusta Debian, ésta distribución fue creada en el año 2004 por Canonical, una empresa con capital íntegramente privado que es la que se encarga al completo de su desarrollo y mantenimiento. Además, debido al apoyo por parte de Canonical, Ubuntu es una de las distribuciones soportadas oficialmente por el proyecto OpenStack.

Gracias a su sistema de repositorios (llamados PPAs) es fácil disponer de una versión actualizada y adaptada a nuestro sistema, sin que ello suponga demasiado esfuerzo por parte del usuario final. Además, Ubuntu dispone de una amplia documentación disponible en la red, así como de una gran comunidad de usuarios.

Otra característica a tener muy en cuenta es que Canonical brinda a disposición de los usuarios algunas versiones especiales denominadas LTS (o Long Term Support) con soporte completo oficial extendido.

En el caso de Ubuntu 14.04 LTS, numerosos parches y mejoras han sido lanzadas desde su lanzamiento en abril 2014, haciendo de esta distribución una opción muy estable y madura.

Por otra parte, desde mediados de 2008 Ubuntu -en sus distintas versiones- ha sido la distribución que he utilizado en mi ordenador personal. Por lo que a pesar de que mi conocimiento sobre ella no va mas allá del simple uso en el día, sí que parto de una base que con otras distribuciones no tendría.

## Debian

Debian es la distribución 'madre' de muchas otras grandes conocidas del mundo GNU/Linux como la propia Ubuntu o algunos sabores de Linux Mint.

Esta distribución es ampliamente utilizada en servidores y ordenadores de uso general en todo el mundo. El proyecto Debian es un proyecto cien por cien de código abierto, centrado primordialmente en brindar al usuario una experiencia lo más estable y robusta posible.

Es decir, apostando por Debian sin ningún tipo de añadidos o configuración de repositorios adicionales, sólo encontremos aquellas versiones de paquetes con una larga trayectoria de uso, pruebas y verificaciones. Como es obvio, este fuerte empeño en la estabilidad hace que muchos de los paquetes que la conforman no estén muy actualizados y se encuentren disponibles en versiones antiguas.

Como en todo proyecto software la estabilidad del sistema base es algo muy importante, aun así para este caso concreto, optar por Debian no supone una opción debido al objetivo del mismo de desplegar un OpenStack completamente funcional. En el caso de hacerlo en Debian se tendría que contar con repositorios adicionales y en muchos casos otros denominados como 'testing' con los actualizar una gran cantidad de paquetes del sistema debido las numerosas dependencias derivadas que una instalación de este tipo supone.



Además, a pesar de que Debian es una excelente distribución, no hay que olvidar que se trata de una versión comunitaria sin una empresa u organización detrás dándole soporte, por lo que quizá para un desarrollo como este sea posible encontrar otras opciones más acordes.

## SuSe

Otra de las opciones aquí barajadas fue la proporcionada por la empresa alemana Novel con su sistema operativo SUSE y las diversas variantes de éste. Suse es la distribución madre y sobre ella está basada la versión empresarial denominada SUSE Linux Enterprise Server, llamada comúnmente, SLES.

SLES es un SUSE con añadidos muy interesantes dirigidos al sector profesional y con el soporte y ayuda por parte de Novel. A su vez, partiendo de la base de SLES surge OpenSuSe, proporcionando todas la ventajas de SLES pero sin coste para el usuario final y obviamente sin el soporte empresarial de Novel.

Relacionado con este proyecto, en el año 2012 Novel lanzaba públicamente la primera versión comercial de su implementación de OpenStack en Suse Linux bajo el nombre de SuSe Cloud versión 1.0 y basada en una de las versiones iniciales de OpenStack denominada Essex. Con cada nueva versión de OpenStack, Novel lanza a su vez una nueva versión de su producto actualizándolo y dotándolo de las nuevas características disponibles.

SuSe Cloud es un producto muy sólido que goza de unas excelentes críticas entra la comunidad, pero su elevado precio y su orientación hacia un perfil más profesional, la descartan para el uso en este proyecto.

Aún así, esta solución es ideal para un entorno empresarial puro en el que se necesite desplegar y mantener un sistema en la nube de forma rápida y fiable a la vez que cuenta con un amplio soporte y experiencia por parte de la gente de SuSe.

## RedHat (y derivadas)

Posiblemente el candidato más fuerte de esta comparativa junto con la ya comentada Ubuntu Linux. Como en el caso de Ubuntu existe una ingente cantidad de documentación en línea así como una amplia comunidad de usuarios. Dentro de esta rama encontramos distintos sabores, cada uno de ellos con sus particularidades: Fedora, RedHat Enterprise Linux, CentOS y Scientific Linux.

En las primeras fases del desarrollo de este proyecto, a principios de 2014 CentOS la edición comunitaria de RedHat Enterprise Linux (RHEL) fue integrada oficialmente bajo el paraguas protector de la empresa de sombrero rojo.

En cuanto a Fedora es una versión que llevo utilizado a diario en mi puesto de trabajo y en líneas generales estoy bastante satisfecho con su rendimiento y desempeño. Fedora es la rama de pruebas de RHEL e incorpora actualizaciones de paquetes y aplicaciones de una forma muy rápida. No en vano, Fedora se suele considerar la rama de pruebas de lo que serán las nuevas versiones de RHEL. Es por este motivo por la que no la he tenido finalmente en cuenta para este proyecto. Para el uso diario en un entorno de trabajo en un ordenador de escritorio, Fedora cumple las expectativas del día a día, pero no es recomendable para un entorno en el que necesitemos la mayor tasa posible de fiabilidad y estabilidad.

En el caso de RHEL, cumple con todas las premisas deseadas en un proyecto como este. Es soportada por una gran empresa, RedHat, una gran comunidad de usuarios, orientada al sector profesional, actualizaciones, así como sus derivadas, además RHEL es una de las distribuciones soportadas oficialmente por el proyecto OpenStack. El único problema, es que su uso no es gratuito, por lo que para utilizar una versión en este desarrollo tendría que desembolsar el precio de la licencia (799 dólares americanos al año para una única instalación). A pesar de ser una fantástica candidata, debido a esto, esta versión fue descartada.

Otra posible candidata es Scientific Linux. Ésta es una distribución orientada a la comunidad científica auspiciada por Fermilab y CERN. Está fuertemente basada en CentOS por lo que hereda todas sus grandes ventajas, más el buen hacer de Fermilab y CERN. Además, por ti todo lo anterior fuera poco, su uso es completamente gratuito.

Debido a esto, Scientific Linux ha sido la versión Linux finalmente elegida sobre la que montar OpenStack.

A continuación se le dedicará un apartado completo a esta distribución para así entender de donde viene, que es lo que busca, así como sus principales objetivos.

## **Scientific Linux**

Al tiempo de escribir estas líneas, la última versión disponible en el mercado está basada en CentOS 6.6. Scientific Linux, prácticamente se trata de una CentOS recompilada y ajustada para cumplir los altos estándares de calidad fiabilidad del CERN y Fermilab. Desarrollada conjuntamente por estos dos grandes centros de investigación científica, Scientific Linux nos proporciona un sistema estable, seguro y claramente destinado a un entorno de producción donde la estabilidad sea una de las bazas más importantes atender en cuenta. En su versión de escritorio, Scientific Linux utiliza el antiguo, probado y ampliamente configurable, Gnome 2.30.

Para que nos hagamos una idea de forma gráfica y bastante inmediata de lo comprometidos que están en esta distribución con la estabilidad, podemos echar un vistazo a algunos de

sus componentes claves. Por ejemplo en el caso del kernel, Scientific Linux utiliza una versión Long Term es decir una versión del núcleo con soporte extendido basado en el kernel 2.6.32.X.

Durante el desarrollo de este proyecto se utilizó este: 2.6.32-504.1.3.el6.x86\_64 lo que significa que tiene al menos 504 revisiones (correcciones, arreglos y mejoras) de diversa índole. Además del núcleo, prácticamente todos los paquetes siguen este principio por lo que nos puede dar una idea aproximada del grado de madurez del proyecto.

Aunque la parte negativa de esta filosofía, como se comentaba anteriormente para el caso de Debian, es que no encontraremos paquetes muy actualizados o novedosos, por lo que si se trata de probar nuevas funcionalidades es posible que los oficialmente soportados no nos proporcione la mejor experiencia.

## Requisitos del Sistema

### Requisitos funcionales

- Los usuarios podrán identificar e ingresar en el sistema OpenStack gracias a las credenciales almacenadas en un Directorio Activo de Microsoft.
- Cada usuario dispondrá de distintos permisos y funciones dentro del proyecto de OpenStack acordes con su rol en el Directorio activo.
- Existirán dos niveles distintos de acceso según los grupos configurados en el Directorio Activo: 'admin' y 'member'.
- Todo usuario del sistema tendrá un proyecto propio en donde podrá crear y probar sus máquinas de forma privada.
- Los usuarios podrán subir sus propias imágenes al sistema a partir de las cuales se podrán crear instancias de máquinas virtuales.
- Los usuarios dispondrán de una serie de imágenes generadas con antelación para poder crear instancias rápidamente a partir de ellas.
- Toda configuración realizada será correctamente documentada para poder ser imitada por una persona distinta al autor de este proyecto. La documentación estará escrita de forma clara y sencilla, explicando cada uno de los puntos. Cualquier persona con unos conocimientos mínimos de redes, sistemas operativos y virtualización será capaz de replicar un sistema similar al expuesto, en cualquier otro entorno a partir de esta documentación.

## Requisitos no funcionales

- El sistema estará alojado en uno de los servidores físicos de la Facultad de Informática de la Universidad de Oviedo.
- El sistema estará accesible desde cualquier lugar a través de una dirección IP pública perteneciente a la Red de la Universidad de Oviedo.
- Para el desarrollo se dispondrá de un único servidor físico en el que se alojarán todas las máquinas virtuales que formarán parte del proyecto.
- El servidor físico sobre el cual se montará toda la infraestructura, estará gobernado por el sistema operativo Xen.
- El sistema soportará el acceso simultáneo de varios usuarios a la vez.
- Multiplataforma, el producto final será accesible desde cualquier dispositivo con un navegador moderno capaz de interpretar código html y javascript.
- Así mismo se proporcionará acceso simultáneo a través de interfaz de órdenes (CLI) a través de los clientes de los módulos de OpenStack.

## Identificación de Actores del Sistema

### Administrador del sistema

Este usuario será destinado a las tareas de administración puras y por consiguiente cumplirá con el rol clásico de administrador total de la aplicación. Tendrá acceso completo a todos los niveles de configuración de la aplicación y entre sus funciones constará el conceder/retirar permisos a otros usuarios.

### Usuarios del sistema

Los usuarios serán aquellos elementos que se encargarán del uso de la aplicación. Para la aplicación práctica de este proyecto se han identificado dos tipos de usuarios; Profesor y Alumno.

### Usuario 'Alumno'

Este usuario representa al usuario clásico de la aplicación que será el encargado de utilizar la aplicación. Su funciones no se encuentran más allá de utilizar el sistema dentro los

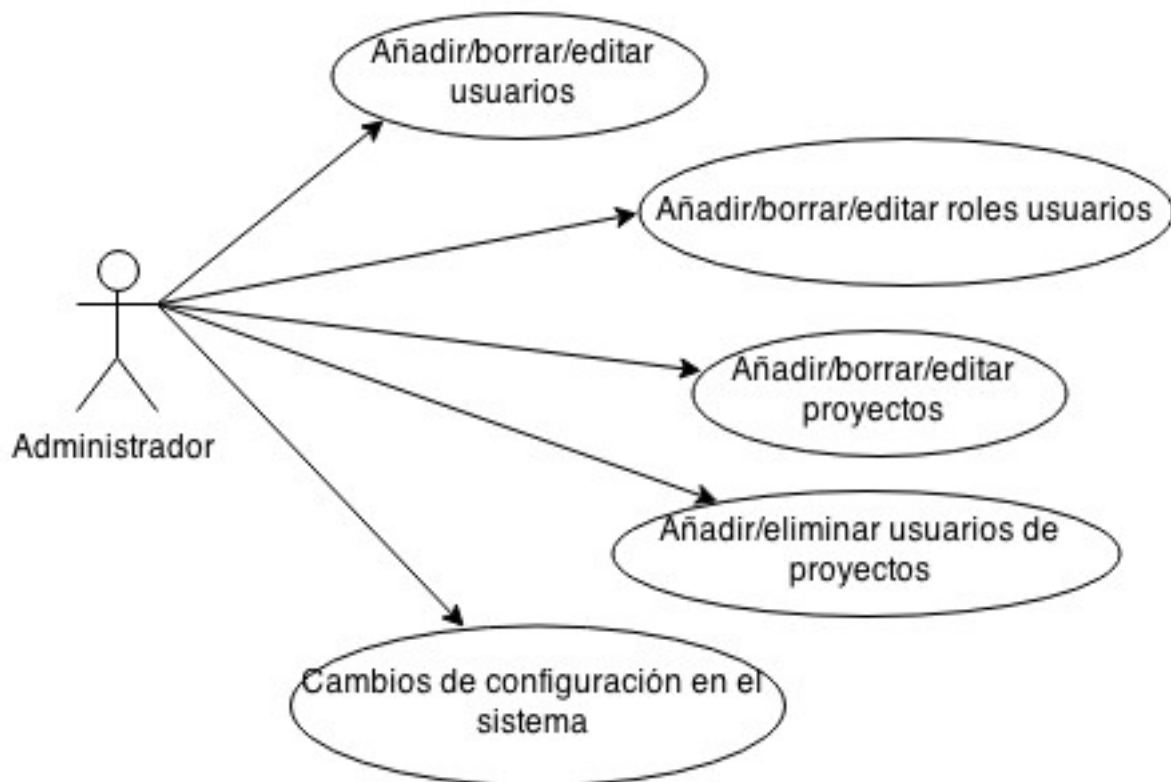
límites propuestos por el administrador. Tiene acceso limitado al las partes que realizan modificaciones y/o ediciones sobre cualquier parte de la aplicación.

## Usuario 'Profesor'

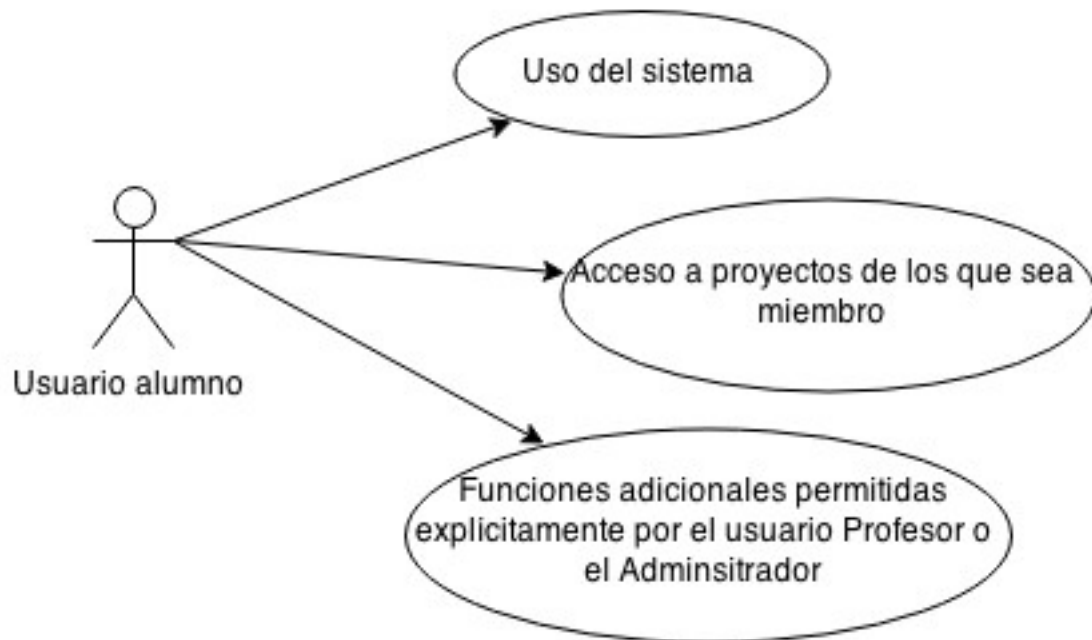
Este usuario tendrá todas las habilidades del Usuario Alumnos y además de estas, gozará de unos permisos especiales sobre ciertas partes del sistema que le permitirán modificar y adaptar el comportamiento de ellas.

Por ejemplo, un usuario con rol de profesor podrá ser administrador de uno o varios proyectos sobres los que podrá realizar las funciones básicas de crear, obtener, actualizar y borrar para imágenes, instancias, volúmenes, etc.

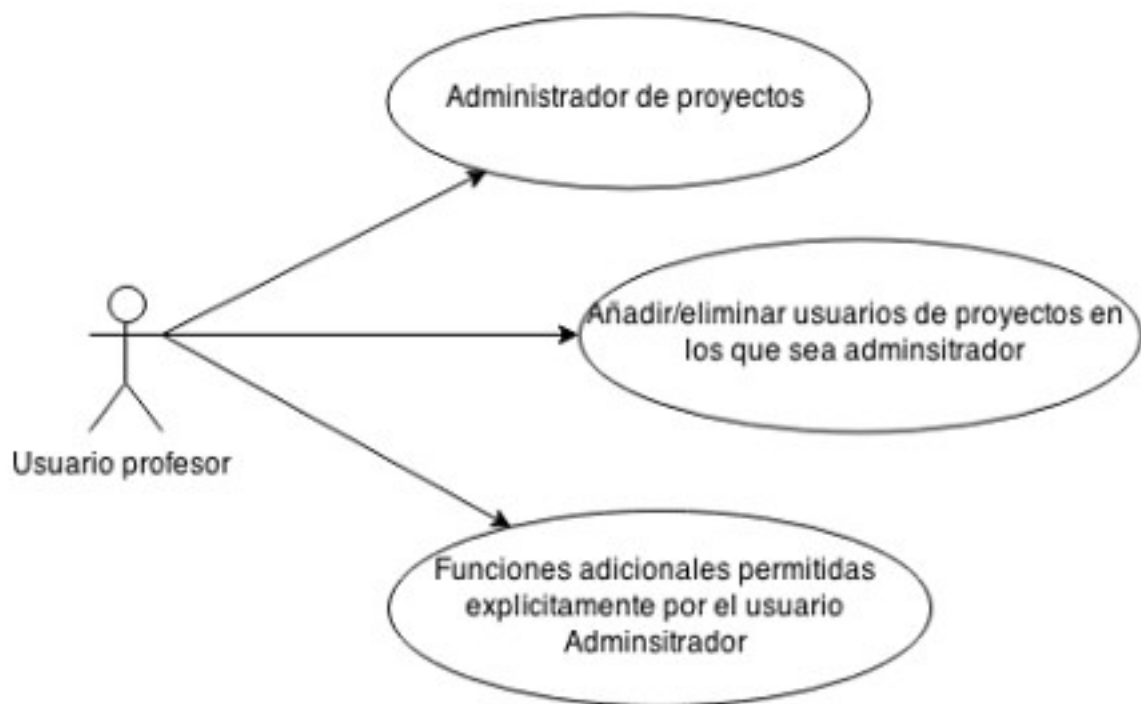
## Especificación de Casos de Uso



*Imagen 6: Diagrama de casos de uso para usuario administrador*



*Imagen 7: Diagrama de casos de uso para usuario alumno*



*Imagen 8: Diagrama de casos de uso para usuario Profesor*

## Capítulo 6. Diseño del Sistema

### Arquitectura

#### Aprovisionamiento hardware

Una vez decididos los sistemas operativos que serán instalados, en este capítulo estará centrado en definir cuales serán las necesidades hardware que requerirá cada una de las máquinas desplegadas. Como se comentó anteriormente, todo el sistema será desplegado sobre un entorno de virtualización puro y gracias a ello, no supondrá demasiada dificultad aprovisionar nuestras máquinas con nuevos recursos posteriormente.

Aún así se ha considerado muy provechoso el realizar poder este ejercicio de reflexión previo y dedicar un momento a pensar cuales son los requisitos hardware que debería cumplir cada una de las máquinas para desempeñar su actividad solventemente.

Gracias a esto, se intentará evitar verse obligado a realizar modificaciones importantes en la configuración hardware de nuestras máquinas durante su ciclo de vida una vez puestas en producción.

Antes de comenzar, recordemos el hardware físico que tenemos disponible y que como es obvio limitará irremediabilmente el desempeño virtual:

xenserver-05	
Procesador (núcleos)	8
RAM	128Gb
Disco	1000Gb
Interfaces red	4

A continuación veremos la configuración hardware elegida para cada una de las máquinas virtuales



## Windows Server

Windows Server (Requisitos mínimos)	
Procesador	Procesador de 64 bits o compatible
RAM	512Mb
Disco	32Gb
Interfaz de red	1

Con toda esta información y sabiendo que el uso que se le dará a lo largo de proyecto será sólo el de alojar un Directorio Activo con usuarios y grupos de prueba se ha decidido dotarla con lo siguiente:

Windows Server	
Procesador (núcleos)	1
RAM	8192Mb
Disco	200Gb disco
Interfaz de red	1

## Scientific Linux

En el caso de Scientific Linux al ser una versión fuertemente basada en la distribución CentOS Linux, los requisitos mínimos serán exactamente los mismos que para ésta. En este caso los podemos encontrar aquí: <http://wiki.centos.org/About/Product>

Scientific Linux (Requisitos mínimos)	
Procesador	Procesador de 32 bits o compatible
RAM	• 392Mb sin entorno gráfico

• 512Mb con entorno gráfico	
Disco	2Gb
Interfaz de red	1 (opcional)

La información anterior es sólo aplicable si quisiéramos realizar una instalación de Scientific Linux y nada más. En este caso Scientific Linux será la base para nuestra configuración OpenStack tenemos que tener en cuenta los requisitos mínimos de este:

<http://openstack.redhat.com/Quickstart>

Por lo que en este caso de mínimo tendríamos:

OpenStack (Requisitos mínimos)	
Procesador (núcleos)	1 *
RAM	2Gb
Disco	5Gb disco

\* imprescindible que éste sea compatible con técnicas de virtualización

Como en el caso anterior, sabiendo esta información eligiéremos una configuración inicial capaz de desempeñar con éxito nuestras exigencias sin tener que modificarla en el corto plazo.

También hay que tener en cuenta que este sistema incluirá en sí mismo las nuevas instancias virtuales creadas en el futuro, por lo que debemos ser generosos en la configuración si queremos poder montar a su vez varias de ellas en nuestra entorno de nube.

Scientific Linux + OpenStack	
Procesador (núcleos)	8*
RAM	81920Mb
Disco	500Gb disco

\* utilizando sobre-aprovisionamiento

## Configuraciones de Red

Como ya se comentó de forma breve anteriormente, el grupo de Cloud Computing de la Facultad de Informática de Oviedo proporcionó a este proyecto cinco IPs públicas dentro del rango de red la Universidad de Oviedo para ser asignadas a las máquinas virtuales que formarían parte del proyecto.

Debido a esto, la intención inicial fue la de repartir cada uno de los distintos servicios utilizados en este proyecto, en hasta cinco máquinas distintas. Desafortunadamente esto no pudo llevarse a cabo tal y como se había concebido originalmente, dado que producía demasiadas pérdidas de recursos.

Debido a esto, para solucionar el problema se optó por una solución más pragmática, agrupando aquellos servicios accesorios al proyecto y de carácter secundario en una única máquina virtual. De esta forma ahorramos IPs, esfuerzos y recursos, sobretodo núcleos de procesador, muy necesarios en este tipo de entornos.

Con todo, la tabla de asignación de IPs, máquinas virtuales y servicios desplegados queda de la siguiente forma.

Máquina	Servicio	IP	Descripción
alexandria	Servidor NFS	156.35.95.20	Debian. Biblioteca NFS con las ISO de instalación del resto de sistemas.
lesoleil	CD+AD	156.35.95.21	Windows Server 2012 con funciones de controlador de dominio y directorio activo.
lanuage	OpenStack	156.35.95.22	Scientific Linux 6.6 con RDO OpenStack instalado y los distintos modulos configurados y funcionando
alexandria	OwnCloud	156.35.95.20	Debian con ownCloud para alojar las copias de seguridad
alexandria	Xen Orchestra	156.35.95.20	Debian con Xen Orchestra para configurar y nuestras máquinas virtuales Xen.

Como nota adicional, comentar que los nombres de cada una de las máquinas expuestas en la tabla anterior han sido elegidos en el momento de su instalación y no se corresponden con ningún tipo de restricción de uso o funcionamiento de los servicios que albergan.

## Herramientas y programas usados durante el desarrollo

Además de las dos más obvias, una distribución GNU/Linux con OpenStack y Microsoft Windows Server con Directorio Activo, para el desempeño de este proyecto se han utilizado una serie de aplicaciones/herramientas que se pasarán a introducir a continuación de forma breve. Para facilitar su lectura se han agrupado por funcionalidad.

### Virtualización

#### Xen Server

Xen Server es un hipervisor de tipo uno desarrollado en el año 2003 como proyecto universitario en el seno de la Universidad de Cambridge. Desde su lanzamiento al público el Xen fue ganando notoriedad poco a poco hasta que en el año 2007, la compañía americana Citrix terminó por adquirir al completo su desarrollo. En el año 2013, Citrix se alía con la Fundación Linux para liberar su código bajo la licencia GNU GPL versión 2 utilizando el nombre de “Proyecto Xen”.

En la actualidad Xen Server soporta diversas arquitecturas y es utilizado en como hipervisor en una gran cantidad de servidores de todos el mundo.

En este desarrollo se utilizó la versión 6.2 de Xen Server y sobre él se hospedaron todas las máquinas virtuales desplegadas en este proyecto.

#### Xen Center

Xen Center es una aplicación exclusiva para sistemas Microsoft Windows, que proporciona una interfaz gráfica de usuario para administrar remotamente servidores instalados mediante el hipervisor Xen Server. Gracias a esta herramienta, es posible realizar la mayoría de sus configuraciones remotas de una forma más amigable e intuitiva, de lo que se podría hacer utilizando una ventana de terminal de comandos. Xen Center es posiblemente una de las mejores opciones iniciales para un usuario no experimentado. Desde mediados del año 2013, Xen Center es completamente opensource y su código está licenciado como BSD 2-Clause. Xen Center está desarrollado utilizando el lenguaje de

programación C#.

A día de hoy, a pesar de ser considerado como un proyecto maduro y ya no recibir grandes cambios en su código, Xen Center sigue actualizándose periódicamente. Estas actualizaciones, están más orientadas hacia el arreglo de fallos conocidos y normalmente sirven para aumentar la estabilidad y fiabilidad general del programa.

Es posible echarle un vistazo al código del proyecto es incluso hacer una copia del mismo desde su página en Github: <https://github.com/xenserver/xenadmin>

Xen Center fue utilizada en los primeros compases de este proyecto para definir y crear la infraestructura virtual del mismo.

## Xen Orchestra

Xen Orchestra es una herramienta que nos proporciona una completa y moderna interfaz Web para administrar y configurar Xen Server. Xen Orchestra es una aplicación opensource desarrollada por Vates, una empresa francesa especializada en el desarrollo de productos de código abierto. Una de las ventajas que ofrece frente a otras opciones de funcionalidad similar, es que al tratarse de una aplicación web es posible acceder a ella desde cualquier navegador web moderno, independientemente del sistema operativo instalado. Para conectarse, bastará con una conexión a Internet y un navegador capaz de interpretar código HTML5 y javascript, que son los principales lenguajes utilizados en su desarrollo

Xen Orchestra nos proporciona una interfaz limpia, sencilla y amigable para administrar nuestro servidor Xen de forma remota.

Desde la página web del proyecto es posible descargarse una imagen Xen en formato xva, preparada para ser importada en nuestro entorno Xen Server. Esta imagen que distribuyen directamente los desarrolladores de la aplicación, no es más que una Debian Wheezy 7 con los servicios de Xen Orchestra corriendo en el puerto 80. Una vez desplegada la imagen en nuestro servidor sólo tendremos que abrir un navegador e introducir la IP de la máquina XOA.

Al tratarse de un desarrollo joven y opensource el proyecto tiene un ritmo de actualizaciones bastante elevado, incorporando mejoras, nuevas características y correcciones muy rápido.

Como la misión de Xen Orchestra es la de brindarnos un frontend web de configuración para nuestro Xen Server, y esto no afecta de manera notoria a la parte intrínseca del proyecto, se ha decidido incorporar las actualizaciones al proyecto a medida que estas fueran siendo liberadas a pesar de que el producto ya estaba siendo utilizado en “producción”. Al no disponer de un entorno de desarrollo a pruebas definido como tal, como medida de cautela

adicional, cada nueva actualización ha sido probada en una máquina virtual antes de ser instalada. Una vez que la actualización en el entorno de pruebas, es considerada como satisfactoria entonces se procedía con la instalación en el servidor.

La primera versión utilizada en este proyecto en lo que podría considerarse como “producción” fue la 3.1 liberada el 14 de febrero de 2014. A lo largo del todo el desarrollo han salido nuevas versiones que han sido incorporadas satisfactoriamente a nuestro sistema en producción.

## VirtualBox

Es un software hipervisor de tipo dos gratuito y de código abierto desarrollado originalmente por Sun Microsystems. Tras la compra de Sun por parte de Oracle a principios de 2010, el gigante de americano de las bases de datos ha sido el encargado de mantener activo el desarrollo de VirtualBox. Sencillo, intuitivo y multiplataforma es una buena alternativa gratuita y opensource a la presentadas por VMware o Parallels entre otras.

Durante este proyecto VirtualBox ha sido utilizado para mantener en mi portátil personal una modesta máquina virtual con una instalación de Microsoft Windows de escritorio. El objetivo de ésta era la de poder realizar todas aquellas configuraciones que requerían inevitablemente el uso de entornos Windows.

## Edición de archivos e imágenes

### Libreoffice

LibreOffice es posiblemente la suite ofimática de código abierto por antonomasia y que incluye herramientas como procesador de textos, hoja de cálculo, presentaciones, herramientas para el dibujo vectorial y base de datos. Está disponible para un gran número de las plataformas más extendidas en la actualidad. Además, soporta numerosos formatos de archivo, incluyendo como predeterminado el formato estándar ISO/IEC OpenDocument (ODF), entre otros formatos comunes.

Libreoffice está basada en OpenOffice.org que a su vez tiene como base inicial a StarOffice, una suite ofimática desarrollada por StarDivision y adquirida por Sun Microsystems en agosto de 1999. Tras la compra de esta última por parte de Oracle, en el año 2010 se produjo una bifurcación del código del proyecto que fue bautizado como LibreOffice.

Prácticamente condenada al olvido sin una compañía fuerte que lo sustente, contra todo pronóstico, Libreoffice ha mejorado enormemente durante sus últimas versiones. La comunidad de desarrolladores del proyecto y la fundación The Document Foundation está

haciendo un trabajo realmente formidable.

El desarrollo de esta suite ofimática han colaborado muchas compañías muy conocidas como Novell, RedHat, RedFlag CH2000, IBM, Google, entre otras. El código fuente de la aplicación está disponible bajo la Licencia pública general limitada de GNU (LGPL).

Aún existe una versión de OpenOffice mantenida por la Apache Foundation con un funcionamiento óptimo, aunque si bien es cierto que la gran mayoría de las distribuciones GNU/Linux han optado por migrar su suite ofimática a LibreOffice.

LibreOffice ha sido utilizado para escribir toda la documentación de este proyecto.

## Vim

Uno de los editores de texto en modo consola más famosos. Vim fue concebido como una versión mejorada del mítico editor vi y ha resultado una herramienta tremendamente potente y configurable. Para los usuarios más noveles puede resultar algo complicado en su manejo, pero con algo de tiempo de uso es sencillo llegar a entender el porqué en es uno de los editores de consola para el mundo GNU/Linux que mas alabanzas y usuarios recoge.

En este proyecto fue utilizado para editar y escribir cada uno de los archivos editados en los servidores virtuales. Los ejemplos usando en esta documentación usan vim como referencia.

## Gedit

Gedit es el editor de texto por defecto del entorno de escritorio de Gnome. Sencillo, rápido y con características tan utiles como sintaxis resaltada, búsqueda y reemplazo y pestañas.

A lo largo de este proyecto, ha sido utilizado para abrir y realizar pequeñas ediciones de archivos de texto en mi ordenador personal desde el que se ha desarrollado la mayor parte de este proyecto.

## GIMP

GIMP (GNU Image Manipulation Program) es un programa de edición de imágenes digitales en forma de mapa de bits, dibujos y fotografías. Es un programa libre y gratuito. Está englobado en el proyecto GNU y disponible bajo la Licencia pública general de GNU.

Es el programa de manipulación de gráficos disponible en más sistemas operativos, algunos de ellos son: Unix, GNU/Linux, Windows y Mac OS X. Además de esto, GIMP es incluido en un gran número de distribuciones GNU/Linux, como la herramienta de edición de imágenes

predeterminada.

Para el desarrollo de este proyecto ha sido utilizado en multitud de ocasiones en las que se necesitaba realizar alguna edición de una imagen. Muchos de los ejemplos utilizados en las pruebas han sido generados con esta herramienta, así como un gran parte de las imágenes de este documento.

Este programa puede descargarse de forma libre y gratuita desde su página oficial: <http://www.gimp.org/downloads/>

## Generación de diagramas

### draw.io

Draw.io es un editor de diagramas en línea completo y muy sencillo de usar. Está disponible a través de la dirección con su propio nombre <http://www.draw.io> y para su manejo no es necesario ningún tipo de registro o instalación adicional.

Además de su sencillez y facilidad en su manejo, draw.io ofrece la opción de ser enlazado con diversos proveedores de almacenamiento externo, como Dropbox o Google Drive para poder guardar los diagramas automáticamente en dichas localizaciones. También dispone de integración con herramientas como Confluence y JIRA muy utilizadas en desarrollos profesionales. Todo diagrama generado en draw.io es un fichero XML por lo que favorece la interoperabilidad entre plataformas y su posterior exportación.

Todos los diagramas presentes en esta documentación han sido generados utilizando dicha herramienta. Como nota adicional, remarcar que a pesar de que su uso es completamente gratuito para el usuario tanto profesional como amateur, el desarrollo no está licenciado como código abierto.

### Planner para Gnome

Planner fue desarrollado originalmente por Richard Hult y Mikael Hallendal, dos trabajadores de una empresa llamada Imendio, cuya finalidad principal consistía en el desarrollo de software para el entorno de escritorio Gnome. Actualmente Planner se encuentra integrado en el proyecto de herramientas de gestión de Gnome.

Planner es una herramienta para realizar planificaciones y realizar seguimientos de proyectos de cualquier tipo. Se trata de un proyecto de código abierto licenciado bajo GPL, está escrito en lenguaje C y entre sus metas se encuentra el intentar convertirse en una alternativa tan válida y capaz como las distintas herramientas propietarias disponibles en este campo.



Una de las características de Planner es la posibilidad de utilizar dos formatos de archivos distintos para almacenar los detalles del proyecto, uno basado en XML y otro en una base de datos PostgreSQL.

Entre sus características principales se encuentran: creación de diagramas de Gantt, gestión de tareas, manejo de dependencias, definición y asignación de recursos, subtareas, calendarios de proyecto y vacaciones. Los proyectos pueden ser exportados a formato PDF y a HTML para facilitar su visualización en un navegador.

Actualmente existen versiones disponibles tanto para GNU/Linux como para Microsoft Windows. Este programa puede descargarse libre y gratuitamente desde <http://live.gnome.org/Planner/Downloads>

## Otros

### DreamSpark

DreamSpark (<https://www.dreamspark.com/>) es una iniciativa surgida en el seno de Microsoft, que tiene como objetivo dotar con licencias software a estudiantes para que así puedan utilizar gratuita y legalmente algunos de sus productos. El programa fue anunciado públicamente por Bill Gates en el año 2008. Desde su creación la lista de instituciones y países elegibles no ha parado de crecer. Para beneficiarse del programa sólo será necesario darse de alta en el mismo utilizando una cuenta de correo de una institución educadora válida.

Ésta es una gran iniciativa por parte de Microsoft que proporciona a estudiantes de todo el mundo la oportunidad de acceder a software al que normalmente no tendría acceso debido al elevado coste de licencias.

La lista de software ofertado es muy amplia y es posible encontrar desde sistemas operativos de escritorio, servidores, a entornos de desarrollo, software de virtualización, etc.

Todas las herramientas de Microsoft que requerían licencia comercial utilizadas en este proyecto, han sido activadas utilizando el programa DreamSpark.

### Firefox

Es difícil que a día de hoy Mozilla Firefox necesite presentación. Firefox, es uno de los navegadores web más famosos del mundo. Desarrollado sin ánimo de lucro por la Fundación Mozilla, su código fuente está basado en el mítico NetScape. De código abierto y multiplataforma, en la actualidad sólo se ve superado en número de usuarios por el

navegador web de Google, Chrome.

El navegador del panda rojo, es ampliamente utilizado en sistemas Windows, Machintosh y GNU/Linux cumple con los últimos estándares referentes a la navegación Web.

A lo largo de este desarrollo fue utilizado para todo lo referente a búsquedas, configuraciones y acceso a través de la web UI de los distintos servicios implementados.

## Git

Git es posiblemente el software de control de versiones más conocido y utilizado en la actualidad. Git fue diseñado y desarrollado originalmente en el año 2005 por Linus Torvalds, con el objetivo de facilitar todas las tareas relacionadas con el mantenimiento del Kernel de Linux.

El código de Git está licenciado como GNU General Public License v2 y desde su creación, su crecimiento en términos de uso y expansión ha sido exponencial. A día de hoy, el uso de Git está tan extendido, que prácticamente ha anulado por completo al resto de opciones del mercado. Algunas de estas alternativas de código abierto existentes como Mercurial o Apache Subversion, se han visto forzosamente relegadas a un segundo plano, aunque aún siguen siendo utilizadas a día de hoy en algunos proyectos importantes.

Durante este proyecto se ha utilizado la promoción llamada “Pack de desarrolladores de GitHub” en la que por ser estudiante se proporciona acceso gratuito a hasta cinco repositorios de visibilidad privada en el portal GitHub. Uno de estos repositorios privados fue utilizado para guardar documentos y archivos considerados clave de este desarrollo.

## NFS

En el apartado capítulo de Implementación del Sistema se detallará la instalación y posterior configuración del servidor NFS, por lo que he considerado oportuno dedicarle unos párrafos a este longevo protocolo en este apartado para así poder entender un poco mejor que es lo se pretende usándolo y cómo funciona.

### ¿Qué es NFS?

NFS es un protocolo de sistema distribuido de ficheros desarrollado originariamente en 1984 por la desaparecida Sun Microsystems. NFS permite el acceso remoto de ficheros desde un ordenador cliente a un servidor a través de la red. A lo largo de los años el protocolo ha sufrido varias revisiones dotándolo de mayor seguridad y compatibilidad con un amplio número de sistemas operativos.

La versión actual es la 4.1 lanzada en enero del año 2010. La versión 4.2 está actualmente

siendo desarrollada, aunque no se espera su lanzamiento en un futuro cercano. (<https://tools.ietf.org/html/draft-ietf-nfsv4-minorversion2-29>).

## **¿Qué sistemas operativos los soportan?**

Actualmente, NFS es soportado por la gran mayoría de los sistemas operativos modernos y su uso en la actualidad está bastante extendido. Como se comentó anteriormente, el protocolo NFS se ha utilizado para poder crear un almacén en red en donde guardar imágenes ISO de sistemas operativos y así poder crear una biblioteca de imágenes con las que poder instalar nuestras máquinas en Xen.

En este caso y debido a la restricción de tener que partir de un sistema operativo que tuviera instalador en red, se ha optado por Debian. Como vimos anteriormente, Debian ofrece toda la flexibilidad de las distribuciones GNU/Linux y además una fiabilidad, estabilidad y seguridad digna de los entornos de producción de carácter más empresarial. A pesar de mis experiencias previas con Ubuntu, Debian se trataba de una distribución completamente nueva para mí y me parecía interesante afrontar el aprendizaje de la instalación y configuración de un servidor de estas características basado en Debian.

## Copias de seguridad

En este apartado trataremos uno de los temas más importantes que hay que tener en cuenta al desarrollar un proyecto como éste, las copias de seguridad.

Llevar a cabo un desarrollo de esta envergadura no sería posible sin un sistema de respaldo apropiado y como no podía ser de otro modo en este proyecto, las copias de seguridad serán almacenadas en la nube. Todos los archivos accesorios a este proyecto (documentación, ficheros de configuración, bibliografía, etc) se guardarán de forma local en mi ordenador personal (desde el cual se realizará principalmente el proyecto) y de manera simultánea, se mantendrá una copia de dichos archivos en dos ubicaciones distintas: un servidor ownCloud y un repositorio privado en GitHub.

Para el caso de las máquinas virtuales sobre las que se sustenta el proyecto, no se mantendrán copias de seguridad en ubicaciones distintas. Se ha tenido que llegar a este compromiso, no sólo por una cuestión lógica de falta de espacio si no también por una limitación física en el ancho de banda ofrecido por la línea de comunicación.

Desafortunadamente, mover decenas o en algunos casos incluso cientos de GB de información entre dos ordenadores conectados por WAN no es algo asumible en términos temporales con las conexiones de datos actuales. Llegados al punto en el que fuera necesario utilizarlas, es más que probable que se invirtiese menor tiempo en reinstalar todo el sistema de nuevo, que en transferir y restaurar la imagen desde otra ubicación.

Donde sí se mantendrán una copia exacta de las máquinas virtuales será de forma local dentro del propio servidor xenserver-05. Así mismo, se mantendrán una serie de instantáneas o snapshots de cada una de las máquinas. Estas instantáneas serán creadas como medida precaución antes de realizar cualquier cambio considerado como relevante en la configuración de las mismas.

Así, en el caso de fallo en alguna de las máquinas virtuales, será posible devolverla en apenas segundos, a un estado de configuración anterior gracias a la instantánea creada previamente. Si este proceso de restauración no funcionase, se mantiene la posibilidad de restaurarla a partir de un clon completo de la imagen. La restauración de este clon completo, devuelve la máquina a un estado de configuración muy temprano, en el cual ya se hubieran realizado algunas de los pasos más tediosas como pueden ser la instalación del sistema operativo, la configuración de red y reglas acceso. De esta forma en apenas unos minutos podremos volver a disponer de las máquinas en un estado en el que sólo será necesario realizar instalaciones y reconfigurar algunos archivos. Una vez finalizado el proyecto se crearán nuevas copias completas de dichas máquinas virtuales para poder volver al estado final de este proyecto en el futuro.

Como se comentó anteriormente, soy consciente de que el método aquí descrito no es el más adecuado en caso de catástrofe aunque considero esta solución como necesaria debido al peso de la máquina y las conexiones a Internet actuales. Mantener una copia completa en otra ubicación fuera del entorno de la Universidad y restaurarla, llevaría muchísimo más tiempo que el hecho de reinstalarla y reconfigurarla una vez que se poseen los conocimientos y los ficheros de configuración pertinentes.

## GitHub

Aunque GIT no es un software de copias de seguridad propiamente dicho, su sistema de control de versiones puede resultar muy útil y de gran ayuda en caso de catástrofe. Gracias a GIT podemos almacenar archivos y documentos en repositorios externos a nuestro entorno, desde los que será posible recuperar incluso una versión antigua concreta de nuestros ficheros.

Como se comentó en el apartado de Herramientas y programas usados durante el desarrollo, en este proyecto se ha utilizado un repositorio privado de GitHub para aquellos archivos y documentos de pequeño tamaño relacionados con el mismo.

## ownCloud

ownCloud es un software desarrollado por la empresa homónima ownCloud Inc. que nos proporciona de forma sencilla y automatizada nuestro propio servidor en el que poder almacenar cualquier tipo de datos. Además de la parte destinada al servidor, ownCloud proporciona un cliente de escritorio multiplataforma con el que podemos mantener nuestros archivos sincronizados en nuestros ordenadores y dispositivos móviles de una forma muy cómoda.

Podríamos considerar ownCloud como un Dropbox en el que el propio usuario es el encargado de gestionar el almacén donde se guardarán los archivos. De esta forma desaparecen por completo los posibles problemas relacionados con la privacidad o el acceso sin consentimiento por parte de terceros a nuestros datos. Con ownCloud los datos se encuentran siempre almacenados en un servidor en el que el propio usuario es quien lo controla.

En cuanto al modelo de negocio de la empresa que está detrás de ownCloud, es muy similar al de otras compañías del sector opensource. Se ofrece un producto sin ningún tipo de coste de licencia por uso y posteriormente venden soporte o ediciones “empresariales” de ese mismo, en las que se añaden y mejoras y características orientados al mundo de la empresa.

## Capítulo 7. Implementación del Sistema

Uno de los primeros problemas que se tuvo que afrontar relacionados con la implementación del sistema, fue la instalación remota del entorno de trabajo sobre el que se desplegaría todo proyecto. La teoría dice que a día de hoy y gracias a la extraordinaria ubicuidad que nos proporciona Internet, no disponer de acceso físico a una máquina no supone un problema serio. Esto es así *en teoría* y me gustaría remarcar bien lo de 'en teoría' porque como se verá a continuación, durante el desarrollo de este proyecto se encontraron algunas dificultades, fruto de la falta de acceso físico a la máquina y la propia inexperiencia con Xen y entornos de virtualización.

Este capítulo cubrirá todo aquello relacionado con la implementación del sistema en sus correspondientes máquinas virtuales.

### Instalaciones

#### Preparativos necesarios para la instalación Windows Server

Las instalaciones de Microsoft Windows suelen bastante sencillas y gracias a los asistentes en forma de diálogos que nos proporciona el sistema, cualquier persona con unos mínimos conocimientos de informática puede ser capaz de completarlas con prácticamente éxito garantizado.

Es este caso, el principal problema residía en la dificultad que entraña instalar Microsoft Windows, un sistema operativo que carece oficialmente de instalador en red, sin tener acceso físico a la máquina. En Internet pueden encontrarse imágenes no oficiales modificadas por usuarios con las que se puede llegar a suplir esta característica. Pero en mi caso no disponía de alguna de estas imágenes y descargarla de una fuente no verificada en Internet, no parecía la mejor opción.

Para solucionar este inesperado problema se tuvieron en cuenta las siguientes alternativas.

#### Instalación física

La primera es posiblemente la más obvia, no se disponía de acceso físico, pero sería posible contactar con alguna persona o grupo de personas que sí tuviera y que pudiera prestar su ayuda en esta fase del proyecto. como el grupo de Becarios de la Facultad de Informática.

Con esta ayuda, podría ser posible que se dejara conectado un CD/DVD o un lápiz USB con una imagen preparada para iniciar la instalación. Así, remotamente sería posible conectarme y completar la instalación de manera más o menos al uso. Posiblemente ésta fuera la opción más directa y fácil, pero por otro lado, implicaba movilizar a terceras personas y creaba cierta dependencia en el caso de encontrarme con una más que probable, situación similar en el futuro.

Descartada la primera, la segunda opción que se barajó fue la de preparar una imagen de Windows Server que admitiera instalación en red. El problema de esto es que el Acuerdo de Licencia de Usuario Final (EULA por sus siglas en inglés) no permite ningún tipo de modificación de las imágenes de Windows, por lo que prácticamente antes de empezar ya estaría incurriendo en una violación de la licencia del producto. Además este tipo de modificaciones requieren a su vez de software especial, en muchos casos de pago.

Tras investigar esta vía, decidí rechazarla por todas las razones comentadas anteriormente.

## Instalación remota utilizando un almacén en red

La tercera idea barajada resultaba bastante sencilla sobre el papel. En algún lugar accesible desde el servidor físico, se podría montar una carpeta compartida en la que almacenar las imágenes ISO de los sistemas operativos. Esta opción utiliza un almacén en red desde el cual se pueda acceder y montar los distintos archivos de instalación de los sistemas operativos. A efectos prácticos esta solución funciona como un almacenamiento externo de gran tamaño conectado por red con el servidor físico.

Tras consultar la documentación, Xen Server da soporte a dos tipos de almacén en red desde los que posteriormente poder iniciar máquinas virtuales, SAMBA/CIFS y NFS.

De cara a su funcionamiento una vez completada su configuración, ambas opciones resultaban bastante parecidas y decantarse por una en favor de la otra respondía más que nada a una cuestión de preferencia personal. Debido a mi completa inexperiencia con este tipo de tecnologías, pregunté al grupo de Becarios de la Facultad de Informática, en busca de alguna respuesta sobre cómo habían afrontado ellos este problema en el pasado. En su contestación, me comentaron que ellos utilizaban un servidor NFS, así que finalmente me decidí por esta opción.

## Biblioteca NFS ISO

Una vez tomada la decisión que se utilizaría esta última opción de almacenamiento de imágenes con las que se instalarían los sistemas operativos, se valoraron a su vez dos variantes:

1. Que el grupo de Becarios de la Facultad de Informática me proporcionase acceso a

uno de sus servidores NFS.

2. Instalación de mi propio servidor NFS, partiendo de un sistema operativo que sí tuviera instalador en red.

En el caso de la primera opción, el problema que planteaba era que en los servidores NFS albergaban software con licencias a los que, por motivos ajenos al grupo de de Becarios de la Facultad de Informática, un usuario externo al grupo no podía tener acceso. En los correos electrónicos que intercambié durante aquella época tampoco parecían muy seguros de querer configurar un acceso especial que permitiera acceder sólo a un subconjunto del servidor NFS.

Descartada la opción anterior, la segunda alternativa consistía en desplegar un servidor con un almacén en red NFS en el que guardar las imágenes ISO de los sistemas operativos. La parte negativa de esta opción residía en la consecuente pérdida de recursos hardware que supondría su despliegue, además del necesario tiempo y esfuerzo dedicado a un tema completamente accesorio al proyecto.

A pesar de estos contras, en aras de de disponer de un sistema solvente que solucionase la restricción que suponía no poder instalar ningún sistema operativo que no dispusiera de opción de instalación en red se optó por llevar adelante esta opción.

## Instalación de Debian

Para ello, se tomó la decisión de instalar una máquina Debian. En el apartado de Elección de la distribución GNU/Linux, fueron valoradas varias alternativas y a pesar de que Ubuntu y sus derivadas contaban con unas características muy positivas, en la decisión final no fueron las opciones elegidas. Debido a esto, y también de cara a darle mayor diversidad al proyecto, se ha elegido Debian para este nuevo servidor de funcionalidad miscelánea.

Como primer paso a realizar, utilizaremos el asistente que nos proporciona Xen Center para generar una máquina con una configuración hardware suficiente para este fin.

Debian (Jessie)	
Procesador (núcleos)	1
RAM	8192Mb de RAM
Disco	150GB*

\* espacio considerado como suficiente para almacenar imágenes ISO de los sistemas operativos que



queremos instalar en nuestro entorno

Una vez en el asistente, seleccionamos instalación desde URL, para este caso y dado que por el momento no disponemos de ninguna otra opción, sólo necesitaremos una configuración básica con una configuración de red con IP fija.

En las opciones de instalación seleccionamos instalación por URL para seguidamente especificar la dirección pública en la que se encuentra el instalador. Para el caso de Debian podemos utilizar una de sus numerosos *mirrors*, o replicas, repartidos por todo el mundo.

```
http://ftp.es.debian.org/debian/
```

Arrancamos la máquina y realizamos la instalación normal de Debian a través del asistente en modo consola que nos proporciona Xen Center.

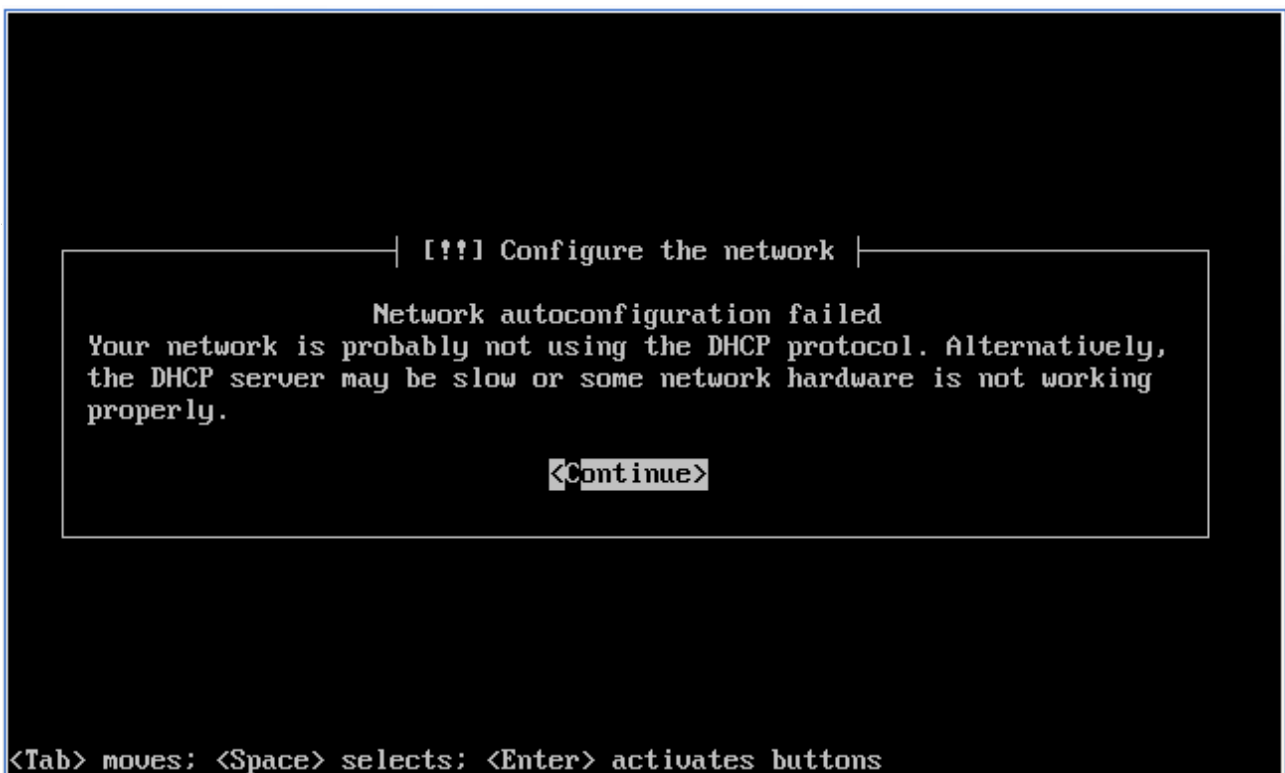
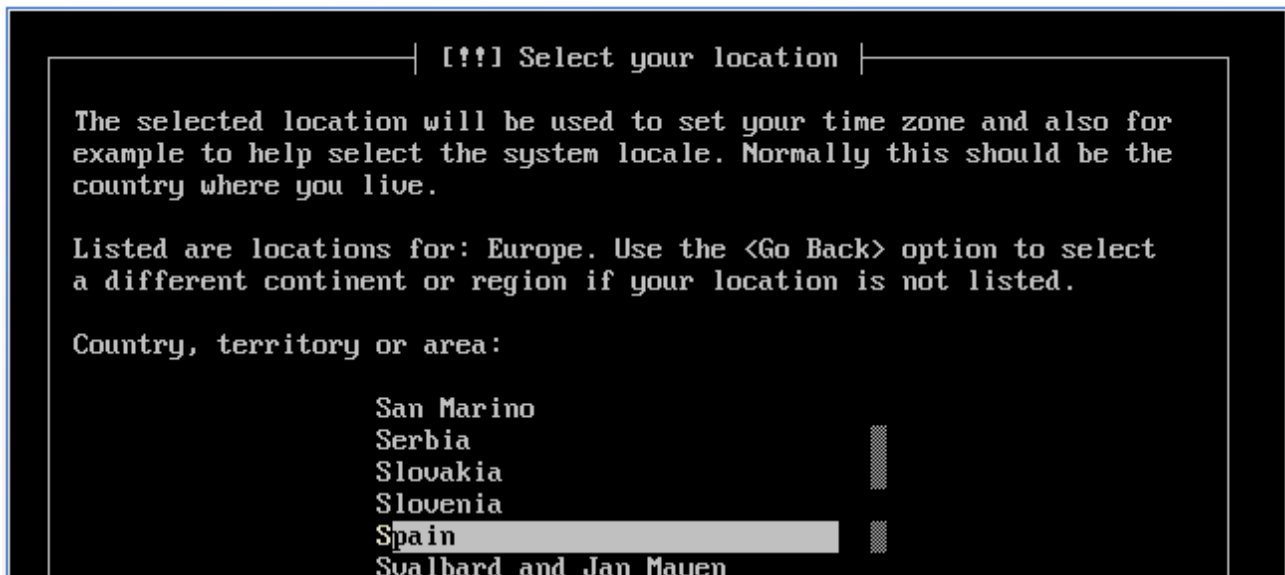
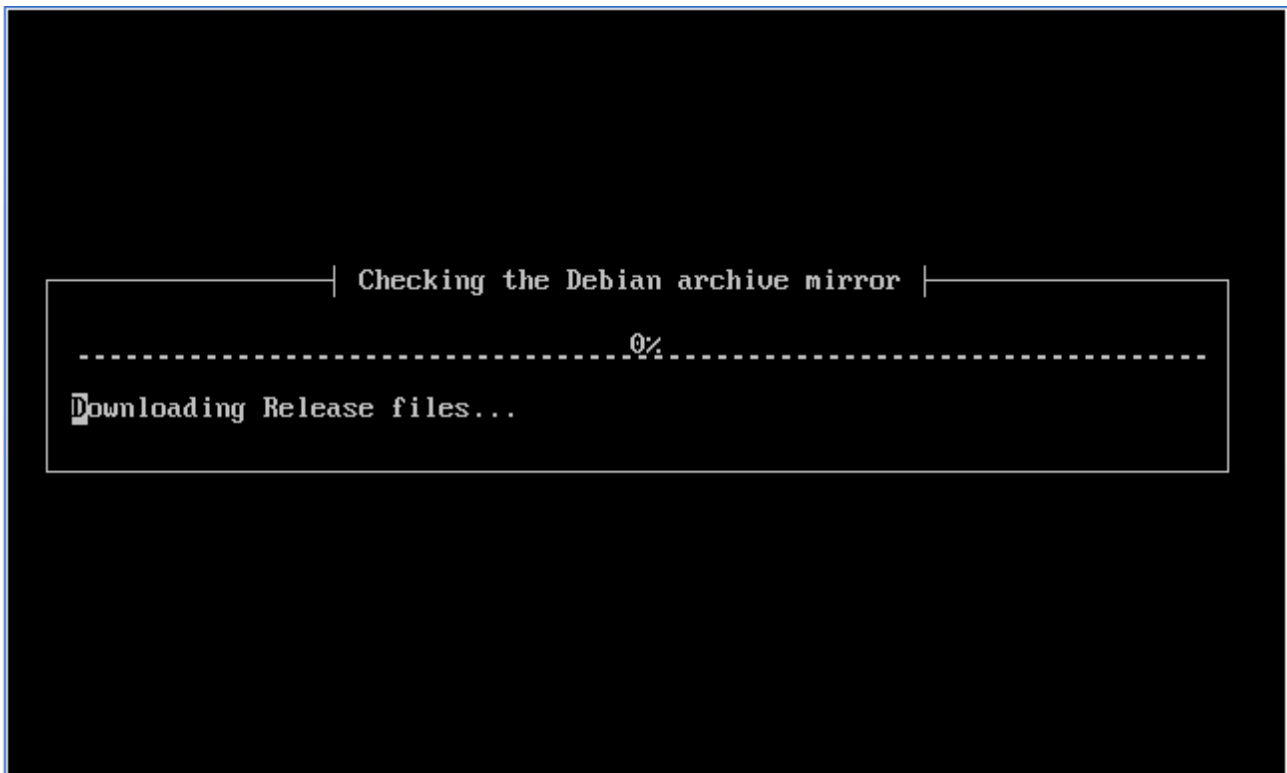


Imagen 10: Configuración de red

Será necesario editar manualmente la configuración de red si se requiere el disponer de acceso a Internet durante la instalación.

```
address 156.35.95.20
netmask 255.255.254.0
network 156.35.94.0
broadcast 156.35.95.255
gateway 156.35.94.201
dns-nameservers 156.35.14.2 156.35.14.3
dns-search lenuage.tfg
```

Hecho lo anterior se procederá con la instalación propiamente dicha del sistema. Dependiendo de la velocidad de la conexión a Internet utilizada, el proceso de instalación podrá tomar más o menos tiempo.



*Imagen 11: Instalación de Debian*

Una vez instalado el sistema base, asignada la contraseña del usuario root y habilitadas las conexiones remotas por SSH gracias al asistente que nos proporciona el instalador, el siguiente paso es conectarse a la máquina recién creada mediante el SSH a través de la dirección IP asignada durante el proceso de instalación.

```
dafero@archenvy ~-> ssh root@156.35.95.20
root@156.35.95.20's password:
Linux alexandria 3.2.0-4-amd64 #1 SMP Debian 3.2.65-1+deb7u2 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 7 18:54:27 2015 from gnontos.cern.ch
root@alexandria:~#
```

Realizado satisfactoriamente el acceso en el sistema, comenzaremos con la creación de lo que será la almacén NFS para las imágenes ISO.

## Instalación y configuración del almacén de imágenes NFS

Para ello utilizaremos la máquina Debian recién instalada en el apartado anterior.

Nombre	IP	Hardware	Descripción
alexandria	156.35.95.20	<ul style="list-style-type: none"><li>• 1 núcleo</li><li>• 8 Gb de RAM</li><li>• 150 Gb de disco</li></ul>	Debian. Biblioteca NFS que almacenará las imágenes ISO del resto de sistemas operativos.

## Instalación de los paquetes necesarios

Antes de comenzar con el proceso, es recomendable verificar que todos los paquetes del sistema se encuentran correctamente actualizados. Para este caso concreto dado que se acaba de realizar la instalación del sistema, esta comprobación no sería estrictamente necesaria puesto que normalmente, siempre y cuando haya acceso a Internet durante el proceso de instalación, el propio instalador del sistema se encargará de descargar y actualizar todos los paquetes a la última versión disponible.

Aun así nunca está de más verificar que todas las dependencias están correctamente satisfechas y que el sistema se encuentra completamente actualizado. Para asegurarnos de esto, en Debian y distribuciones derivadas podemos ejecutar el siguiente comando:

```
root@alexandria:~# sudo apt-get update && sudo apt-get upgrade
Hit http://download.opensuse.org Release.gpg
Hit http://download.opensuse.org Release
Hit http://security.debian.org wheezy/updates Release.gpg
Hit http://cdn.debian.net wheezy Release.gpg
Hit http://download.opensuse.org Packages
Hit http://security.debian.org wheezy/updates Release
Hit http://cdn.debian.net wheezy-updates Release.gpg
Hit http://security.debian.org wheezy/updates/main Sources
Hit http://security.debian.org wheezy/updates/main amd64 Packages
Hit http://security.debian.org wheezy/updates/main Translation-en
Hit http://cdn.debian.net wheezy Release
Hit http://cdn.debian.net wheezy-updates Release
Ign http://download.opensuse.org Translation-en_GB
Ign http://download.opensuse.org Translation-en
Hit http://cdn.debian.net wheezy/main Sources
Hit http://cdn.debian.net wheezy/main amd64 Packages
Hit http://cdn.debian.net wheezy/main Translation-en
```

```
Hit http://cdn.debian.net wheezy-updates/main Sources
Hit http://cdn.debian.net wheezy-updates/main amd64 Packages/DiffIndex
Hit http://cdn.debian.net wheezy-updates/main Translation-en/DiffIndex
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages have been kept back:
  owncloud
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
root@alexandria:~#
```

Una vez terminado el proceso anterior, se procederá con la instalación de los dos paquetes necesarios para iniciar la creación de nuestra biblioteca NFS:

- nfs-kernel-server
- nfs-common

```
root@alexandria:~# apt-get install nfs-kernel-server nfs-common -y
```

Con el comando de instalación anterior, se utilizará la opción `-y` para responder afirmativamente el diálogo de confirmación por parte del usuario.

## Configuración NFS

En primer lugar se creará el directorio en donde se archivarán las imágenes en formato ISO que formarán parte de la biblioteca. Para este caso y por simplicidad, se creará un nuevo directorio llamado 'myshare' en '/home', aunque obviamente tanto la ruta como el nombre del mismo pueden ser alterados libremente para cada caso concreto.

Comenzaremos con la creación del directorio donde se almacenarán las imágenes que quedarán compartidas en red. Para realizarlo, utilizaremos el comando 'mkdir' de la siguiente forma:

```
root@alexandria:~# mkdir /home/myshare
```

Además de la creación de este nuevo directorio, para configurar el acceso a nuestra almacén NFS será imprescindible añadir una línea adicional al fichero 'exports' en la que se indicará desde qué redes será posible acceder al almacén en red. Este paso es muy importante puesto que se pretende evitar que ningún servicio ajeno a este proyecto se conecte y pueda ver utilizar o modificar el contenido de nuestro directorio en red.

El formato de cada una de estas líneas del fichero 'exports' es siempre el mismo, en primer lugar será necesario especificar la ruta absoluta al directorio que será exportado y justo a continuación y separado por un espacio, la lista de direcciones o sub-redes de los clientes a

los que les será permitido montar dicho directorio.

Finalmente, y siempre en la misma línea, añadimos de una serie de opciones o flags que serán las que determinen los permisos aplicados sobre los clientes conectados.

```
#/ruta/al_directorio_a/exportar IP/MASCARA_DE_RED(opcion1, opcion2...)  
IP2/MASCARA_DE_RED_2(opcion1, opcion2...) ...
```

```
/home/example 192.168.0.0/255.255.255.0(ro,no_subtree_check)
```

Una vez comprendida la explicación anterior, se procederá a editar el fichero 'exports' localizado en el directorio '/etc/exports'.

A lo largo de esta documentación utilizaremos el programa 'vim' (vi improved) como nuestro editor de archivos, pero queda a disposición del lector utilizar otro editor que considere conveniente.

```
root@alexandria:~# vim /etc/exports
```

Una vez abierto, se añadirá la siguiente línea.

```
/home/myshare 156.35.94.186(ro,no_subtree_check)
```

En esta caso y tal y como había comentado anteriormente, por motivos de seguridad únicamente se exportará el directorio 'myshare' contenido en /home para la IP 156.35.94.186 que es la correspondiente al servidor físico xenserver-05. Respecto a las opciones, se habilitarán únicamente permisos de lectura (ro= read only) así como la variable para que no compruebe los sub-árboles de directorios (no\_subtree\_check).

## Permitiendo conexiones entrantes

Una vez configurado lo anterior editamos el archivo '/etc/hosts.allow' para añadir la IP o los rangos de IPs para las que se quiere permitir la conexión.

La sintaxis del archivo 'host.allow' es la siguiente, nombre\_servicio: IP/Máscara. Para este caso y dado que sólo se pretende que el servidor NFS sea accesible por la máquina xenserver-05 con IP 156.35.94.186, únicamente se configurará acceso para dicha dirección explícitamente, sin necesidad de aplicar ningún tipo de máscaras o rangos.

De esta forma nos aseguramos que sólo tendrán acceso desde esa IP concreta y limitaremos la entrada a un posible atacante malicioso.

Teniendo en cuenta todo lo anterior, la información relevante del fichero '/etc/hosts.allow' nos quedaría así:

```
root@alexandria:~# vim /etc/hosts.allow  
#
```

```
portmap: 156.35.94.186
lockd: 156.35.94.186
rquotad: 156.35.94.186
mountd: 156.35.94.186
statd: 156.35.94.186
```

## Firewall

En el caso de tener un firewall o corta fuegos activado en el sistema (una opción altamente recomendable debido que la máquina estará expuesta directamente a Internet) será necesario abrir en éste ciertos puertos para garantizar la correcta comunicación entre los clientes y el propio servidor NFS.

Para ello editamos el archivo '/etc/default/nfs-common':

```
root@alexandria:~# vim /etc/default/nfs-common
```

Y añadimos las siguientes líneas:

```
# Options for rpc.statd.
STATDOPTS="-p 32765 -o 32766"

# vi /etc/default/nfs-kernel-server
# Options for rpc.mountd.

RPCMOUNTDOPTS="--manage-gids -p 32767"

#
```

Por último, sólo nos queda crear la reglas necesarias en nuestro cortafuegos. En el caso de Debian el firewall instalado en el sistema es **iptables**. Su sintaxis puede resultar algo compleja y difícil de recordar, por lo que para su manejo se utilizará la cómoda interfaz que nos proporciona Uncomplicated Firewall (ufw).

En el caso de no tener 'ufw' instalado en el sistema tecleamos lo siguiente:

```
root@alexandria:~# sudo apt-get install ufw -y
```

Una vez instalado veremos que tanto añadir, modificar o eliminar regla con 'ufw' es una tarea sumamente sencilla.

```
ufw allow from IP to IP_2 port XXXX proto protocolo
```

Como en los ejemplos anteriores, por una cuestión de seguridad sólo abriremos los puertos estrictamente necesarios y únicamente para la IP del servidor físico xenserver-05.

El resto de puertos los mantendremos cerrados por el momento, a la espera de ser

requeridos para habilitar el funcionamiento de otros servicios.

```
root@alexandria:~# ufw allow from 156.35.94.186 to any port 111 proto tcp
root@alexandria:~# ufw allow from 156.35.94.186 to any port 111 proto udp
root@alexandria:~# ufw allow from 156.35.94.186 to any port 2049 proto tcp
root@alexandria:~# ufw allow from 156.35.94.186 to any port 2049 proto udp
root@alexandria:~# ufw allow from 156.35.94.186 to any port 32765:32767 proto tcp
root@alexandria:~# ufw allow from 156.35.94.186 to any port 32765:32767 proto udp
```

Una vez añadidas todas estas reglas, es posible verificar que han sido añadidas correctamente en el firewall tecleando 'ufw' con la opción 'status'.

```
root@alexandria:~# ufw status
Status: active

To Action From
--
22 ALLOW Anywhere
111/tcp ALLOW 156.35.94.186
111/udp ALLOW 156.35.94.186
2049/tcp ALLOW 156.35.94.186
32765:32767/tcp ALLOW 156.35.94.186
32765:32767/udp ALLOW 156.35.94.186
2049/udp ALLOW 156.35.94.186
22 ALLOW Anywhere (v6)
root@alexandria:~#
```

Para comprobar que todo está funcionando correctamente utilizaremos el comando 'exportfs'. La salida de este comando mostrará aquellos directorios que han sido exportados junto con la lista de IPs habilitadas para su acceso.

Con todo, la salida del comando debería ser similar a la a continuación mostrada:

```
root@alexandria:~# exportfs
/home/myshare 156.35.94.186
root@alexandria:~#
```

Sólo nos queda reiniciar los servicios para que se apliquen las nuevas configuraciones.

```
root@alexandria:~# /etc/init.d/nfs-common restart
[ ok ] Stopping NFS common utilities: idmapd statd.
[ ok ] Starting NFS common utilities: statd idmapd.
root@alexandria:~# /etc/init.d/nfs-kernel-server reload
[ ok ] Re-exporting directories for NFS kernel daemon....
```



```
# /etc/init.d/nfs-kernel-server restart
[ ok ] Stopping NFS kernel daemon: mountd nfsd.
[ ok ] Unexporting directories for NFS kernel daemon....
[ ok ] Exporting directories for NFS kernel daemon....
[ ok ] Starting NFS kernel daemon: nfsd mountd.
root@alexandria:~#
```

Una vez completado todo este proceso satisfactoriamente ya deberíamos tener nuestro servidor NFS correctamente configurado en la máquina virtual Debian y listo para recibir conexiones desde el servidor físico xenserver-05.

Para probarlo, podemos copiar o descargar algunas imágenes de prueba en formato ISO en nuestro recién directorio exportado, en este caso: '/home/myshare' y comprobar posteriormente si el servidor físico desde el que queremos acceder, tiene acceso a ellas y puede cargar su contenido.

En este caso probaremos a descargar una de las imágenes públicas Scientific Linux que más tarde utilizaremos para realizar la instalación de lo que será OpenStack.

```
root@alexandria:~# cd /home/myshare/
root@alexandria:/home/myshare# wget
http://ftp1.scientificlinux.org/linux/scientific/6x/x86_64/iso/SL-66-
x86_64-2014-11-09-LiveCD.iso
root@alexandria:/home/myshare#
```

## Uso de NFS xenserver

En este punto, si todo ha ido bien, ya tendremos nuestro servidor NFS correctamente configurado y listo para su uso. El siguiente paso será añadirlo como un nuevo volumen al servidor físico xenserver-05 para que podamos arrancar fácilmente máquinas desde los propios archivos ISO almacenados en la biblioteca.

## Añadir servidor NFS ISO a nuestros servidor Xen Center

Para realizar este paso utilizaremos una vez más la interfaz gráfica que nos proporciona Xen Center seleccionado la opción de NFS ISO.

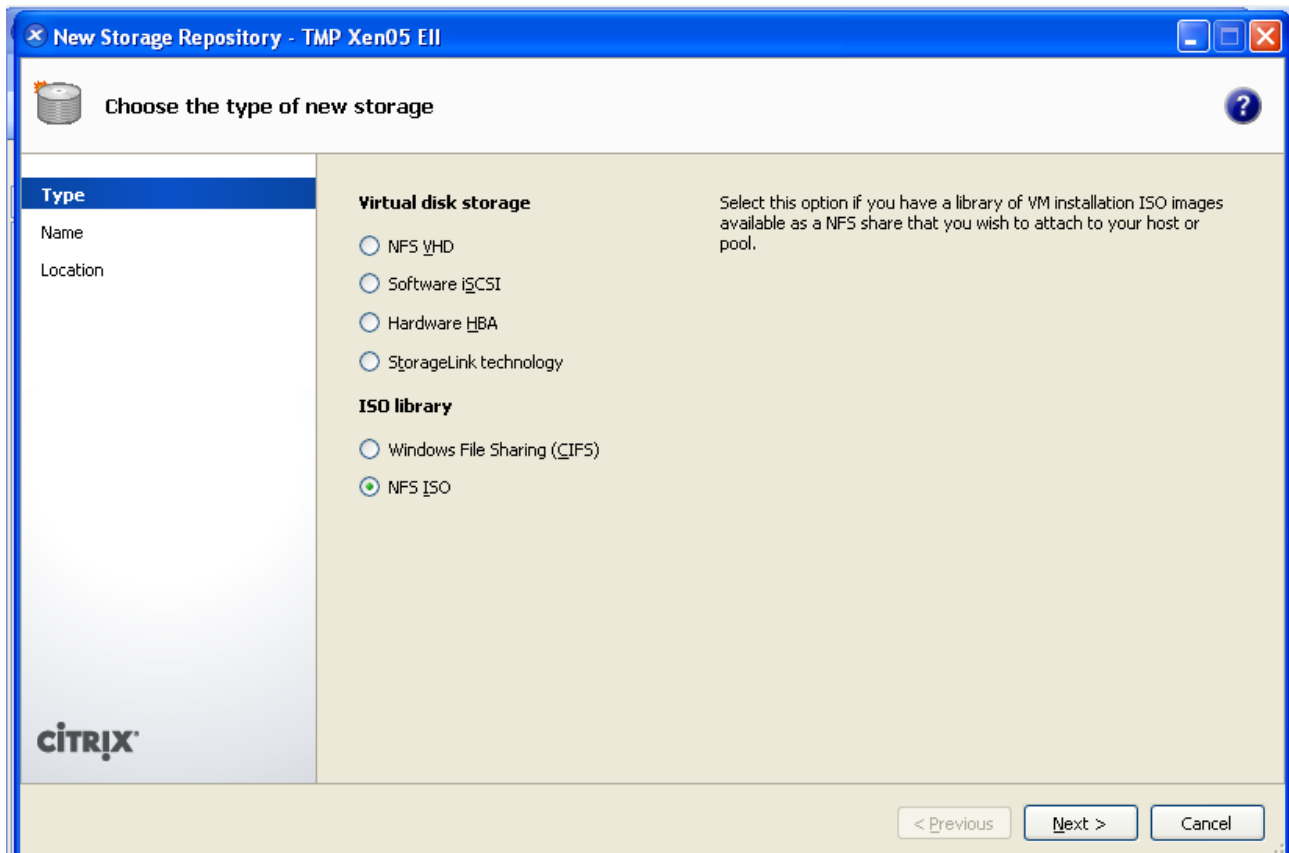


Imagen 12: Añadimos un nuevo almacén de tipo NFS ISO

A través del asistente, se seleccionará el nombre con el que se catalogará este nuevo repositorio en red y en la dirección tendremos que poner `IP_SERVIDOR_NFS:/DIRECTORIO_EXPORTADO`.

Para este caso y tal cómo se explicó en el apartado anterior añadiremos lo siguiente.

```
156.35.95.20:/home/myshare
```

Una vez terminados con todos los pasos del asistente, ya es posible comenzar a utilizar la librería.

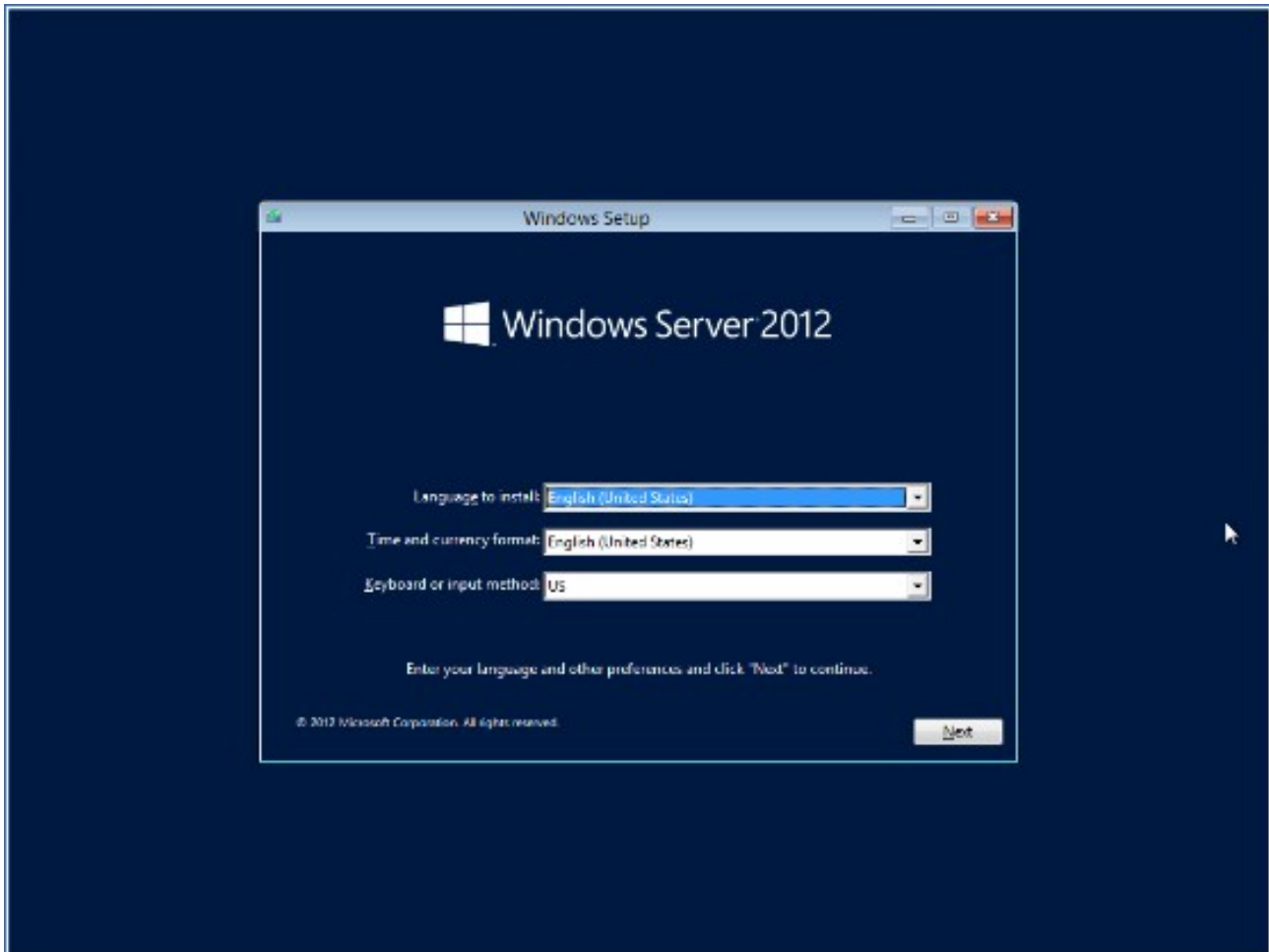
## Instalación de Windows Server

Una vez solucionado el problema descrito en el apartado anterior, ya podremos comenzar la instalación de lo que será el servidor de Directorio Activo. Para la instalación de Windows Server utilizaremos la consola de acceso gráfico que nos proporciona Xen Center.

Nombre	IP	Hardware	Descripción
leciel	156.35.95.21	<ul style="list-style-type: none"><li>• 1 núcleo</li><li>• 8 Gb de RAM</li><li>• 150 Gb de disco</li></ul>	Windows Server 2012 + Directorio Activo

Se creará una nueva máquina virtual con una interfaz de red y las características hardware expuestas en la tabla y a la hora de seleccionar el fichero de instalación, eligiéremos la imagen ISO que previamente guardamos en el almacén en red para imágenes NFS.

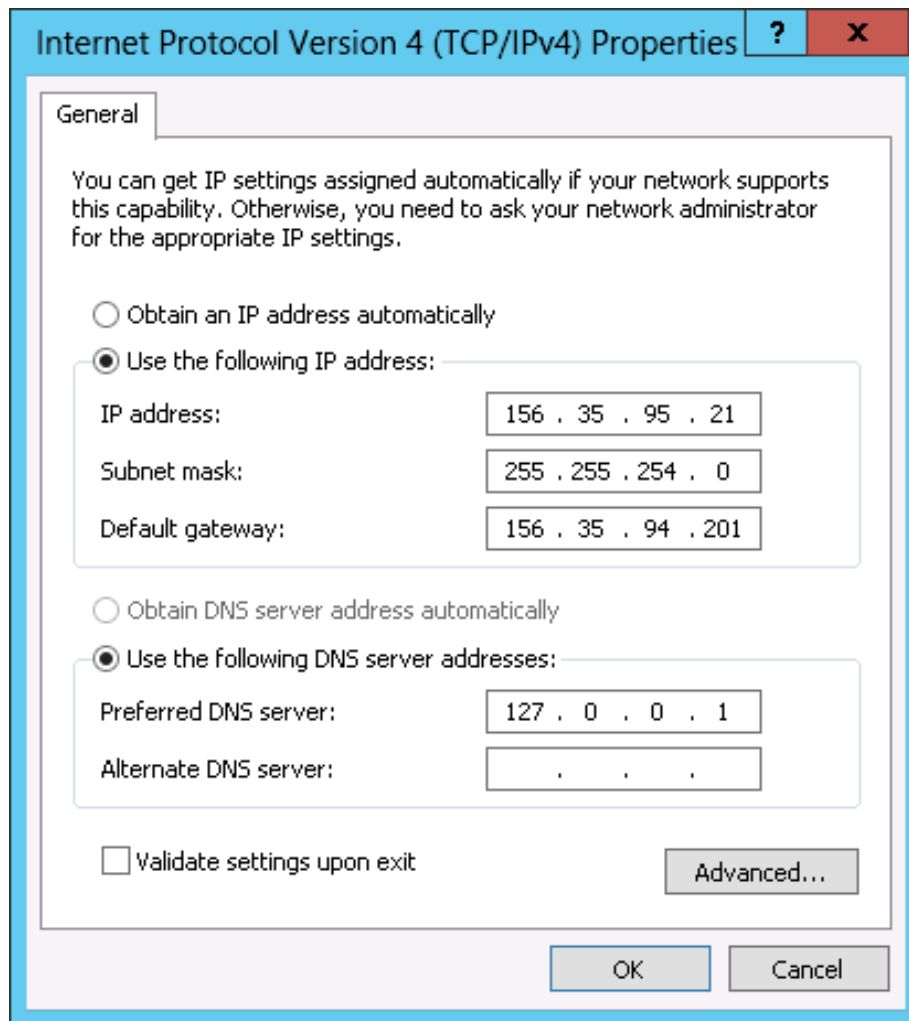
La instalación en sí misma carece de complicación. Bastará con seguir los pasos del asistente y proporcionar aquellos valores requeridos en cada paso.



*Imagen 13: Instalación de Windows Server 2012*

Una vez haya finalizado el proceso de instalación, reiniciaremos la máquina para completar la misma. Tras este primer reinicio, se configurará las credenciales de la cuenta de administrador del sistema y así como la configuración de red. Ambas son cruciales para el buen funcionamiento futuro del sistema por lo que debemos prestar especial atención en recordar claves de acceso así como los valores de red.

Para el caso de la configuración de red, utilizaremos el asistente de configuración que proporciona Windows para añadir los valores deseados.



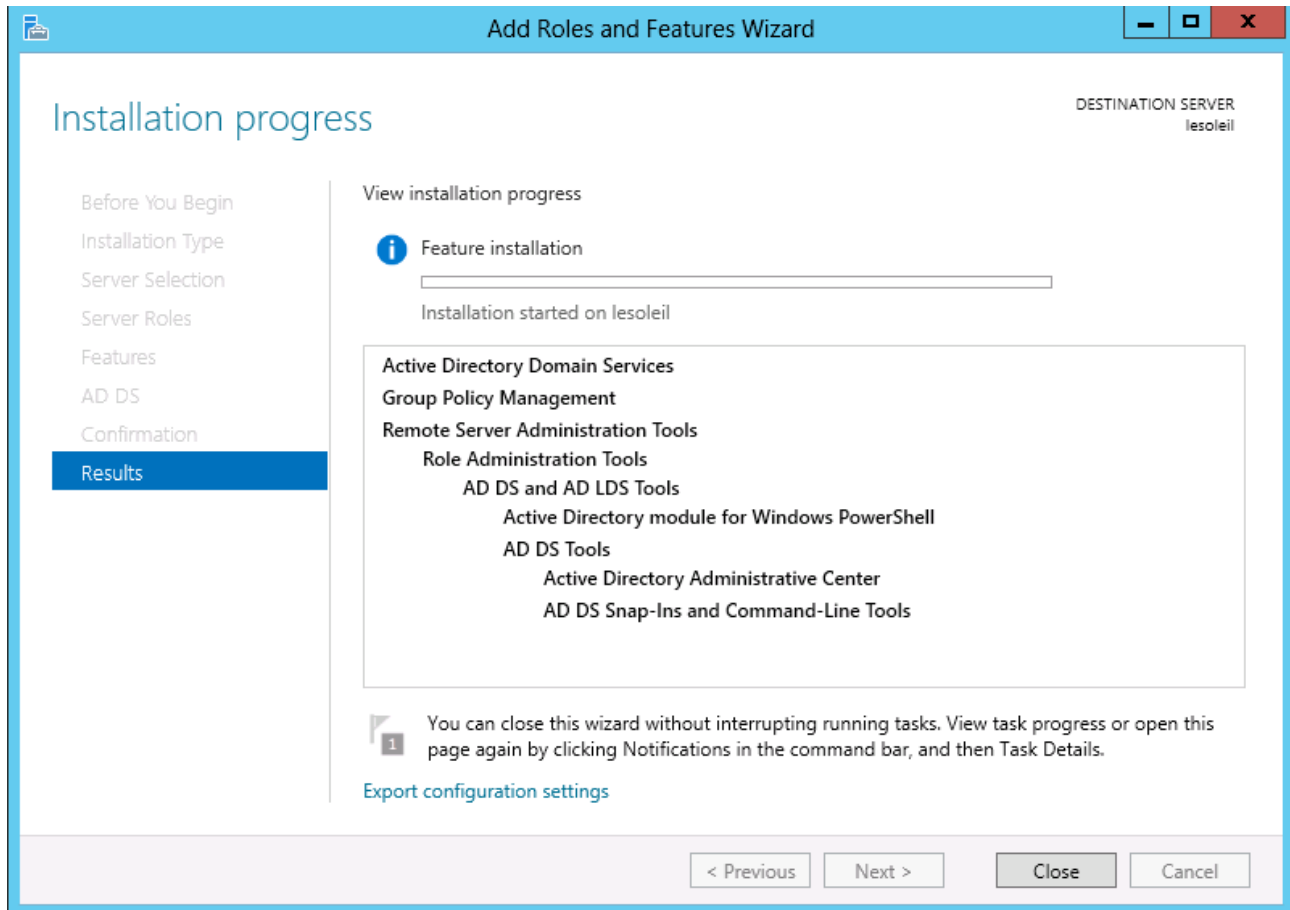
*Imagen 14: Configuración de red Windows Server*

Una vez confirmado el diálogo anterior, será necesario reiniciar la máquina una vez más con el objetivo de que se apliquen correctamente los cambios realizados. A partir de este momento ya será posible conectarse remotamente a la máquina utilizando la IP escogida anteriormente, por lo que no será necesario seguir utilizando la limitada consola de Xen Center.

## Instalación de Roles y Características necesarias

Finalizados los pasos iniciales de la instalación y su posterior configuración de red, pasaremos entonces a la instalación de los roles y características necesarios. La instalación de estos no podría ser más sencilla y gracias a los asistentes de instalación gráficos que nos proporciona Windows Server, la configuración de estos no supondrá mayor problema.

Para ello abrimos la aplicación de Windows Manager y desde ella agregaremos todas las opciones requeridas.



*Imagen 16: Asistente de instalación de roles y características*

Para este caso añadiremos el rol Active Directory Domain Services así como todas aquellas dependencias de éste requeridas por el sistema. El propio asistente de instalación nos irá guiando con el proceso y para completarlo será necesario reiniciar la máquina.

Una vez finalizados los pasos del asistente ya tendremos instalado el servidor Windows con un Directorio Activo.

## Instalación Xen Orchestra

Xen Orchestra (XOA) es un herramienta diseñada para administrar servidores Xen Server cómodamente a través de una interfaz Web. A lo largo de este apartado nos centraremos en los pasos necesarios para su completar instalación y configuración inicial.

Desde la página oficial del proyecto se puede encontrar dos tipos de instalación; Recomendada y manual. Ambas son completamente válidas para entornos de producción, aunque como el lógico cada una tiene sus propias ventajas e inconvenientes respecto a la otra.

En el caso de la etiquetada como recomendada, desde la página oficial del desarrollo se proporciona una imagen de un sistema operativo con un servidor Xen Orchestra ya instalado y correctamente configurado para ser desplegado y usado.

Desafortunadamente, esta opción implica utilizar una máquina exclusiva para su uso con Xen Orchestra por lo que en este desarrollo se optó por instalarlo como un servicio más en el servidor destinado a alojar los servicios accesorios al desarrollo del proyecto.

Servidor utilizado:

Nombre	IP	Hardware	Descripción
alexandria	156.35.95.20	<ul style="list-style-type: none"><li>• 1 núcleo</li><li>• 8 Gb de RAM</li><li>• 150 Gb de disco</li></ul>	Debian con Xen Orchestra para configurar y nuestras máquinas virtuales Xen.

## Instalación manual

La instalación manual está pensada para ser utilizada de manera que permita configurar un mayor número de opciones. En este caso y dado que no queremos destinar una máquina completa para esta funcionalidad, aprovecharemos la ya existente utilizada como servidor de imágenes ISO.

## Prerequisitos y paquetes

Para poder comenzar con la instalación será necesario disponer de las librerías de Node.js así como algunos paquetes de carácter estándar relacionados con el desarrollo software y

la construcción de paquetes como, 'build essentials', 'curl', 'wget', 'git' o 'python' entre otros.

## NodeJS

Tal y como recomiendan en la página oficial del desarrollo de Xen Orchestra utilizaremos el gestor de binarios de Node.js llamado 'n' para instalar esta librería.

Antes de comenzar, será necesario asegurar dependencias y verificar que los paquetes 'curl' y 'wget' están instalados en nuestro sistema.

Como en otras ocasiones, junto con el comando de instalación utilizaremos la opción '-y' para contestar afirmativamente al diálogo de confirmación de instalación por parte del usuario.

```
root@alexandria:~# apt-get install wget curl -y
```

Una vez instalados ambos paquetes, procedemos con la descarga del archivo de instalación de 'n' que guardaremos en el directorio '/usr/local/bin/n'.

```
root@alexandria:~# curl -o /usr/local/bin/n
https://raw.githubusercontent.com/visionmedia/n/master/bin/n
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
Current                                  Dload  Upload   Total   Spent    Left
Speed
 100 8393 100 8393    0     0 12991      0 --:--:-- --:--:--
--:--:-- 15956
root@alexandria:~#
```

Finalizada la descarga del archivo, debemos asignarle permisos de ejecución mediante el comando 'chmod +x' para así poder instalar la versión etiquetada como 'stable'.

```
root@alexandria:~# chmod +x /usr/local/bin/n
root@alexandria:~# n stable

install : v0.10.33
mkdir   : /usr/local/n/versions/0.10.33
fetch   : http://nodejs.org/dist/v0.10.33/node-v0.10.33-linux-
x64.tar.gz
installed : v0.10.33

root@alexandria:~#
```

Para verificar que todo ha ido como esperamos, podemos ejecutar el siguiente comando que, en el caso de estar todo bien configurado, nos devolverá el número de la versión instalada:



```
root@alexandria:~# node -v
v0.10.33
root@alexandria:~#
```

Realizado lo anterior, nos centraremos en la instalación de los paquetes requeridos por Xen Orchestra:

```
root@alexandria:~# apt-get install build-essential redis-server libpng-dev git python-minimal -y
```

Una vez finalizado el proceso de actualización estamos listo para descargar el código del proyecto desde los repositorios de GitHub.

```
root@alexandria:~/xoa# git clone http://github.com/vatesfr/xo-server
root@alexandria:~/xoa# git clone http://github.com/vatesfr/xo-web
```

Xen Orchestra está dividido en dos partes, server y web. Para obtener una instalación completamente funcional será necesario instalar ambas.

## XO-Server

Dentro de este apartado, comenzaremos por la instalación de dependencias del proyecto. Para ello utilizaremos el comando 'npm install' desde el directorio en el que se encuentra descargado el código de xo-server.

```
root@alexandria:~/xoa# cd xo-server/
root@alexandria:~/xoa/xo-server# npm install
```

Terminado el proceso de instalación de la parte servidor, será necesario copiar la plantilla de configuración que se encuentra en el directorio 'xo-server', a un archivo oculto en el mismo directorio con el nombre 'xo-server.xml'.

```
root@alexandria:~/xoa/xo-server# cp sample.config.yaml .xo-server.yaml
```

Una vez copiada, editamos la plantilla para añadir la configuración necesaria de nuestro entorno. Para este caso concreto, modificaremos el puerto donde escuchará la aplicación así como el uso de https en detrimento de http.

```
# Basic HTTPS.
# -
#   # The only difference is the presence of the certificate and
the #   # key.
    host: '156.35.95.20'
    port: 443
```

```
# # File containing the certificate (PEM format).
# #
# # Default: undefined
certificate: './certificate.pem'

# # File containing the private key (PEM format).
# #
# # If the key is encrypted, the passphrase will be asked at
# # server startup.
# #
# # Default: undefined
key: './key.pem'
# List of files/directories which will be served.
mounts:
  '/': '../xo-web/dist/'
```

Como se puede apreciar en las líneas resaltadas en amarillo, para que esta configuración pueda ser aplicada será necesario crear los archivos correspondientes al certificado y a la clave privada. En este caso y por motivos de simplicidad se optará por generar nuestro propio certificado auto-firmado que una vez que accedamos desde el navegador tendremos que aceptar.

```
root@alexandria:~/xo/xo-server# openssl req -x509 -newkey rsa:2048
-keyout key.pem -out certificate.pem -days 365

Generating a 2048 bit RSA private key
.....+++
.....
writing new private key to 'key.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]: ES
State or Province Name (full name) [Some-State]: Asturias
Locality Name (eg, city) []: Oviedo
Organization Name (eg, company) [Internet Widgits Pty Ltd]: EII
Organizational Unit Name (eg, section) []: IT
Common Name (e.g. server FQDN or YOUR name) []: alexandria.tfg
Email Address []: daferoes@gmail.com
```

Una vez terminado el proceso de generación de certificados verificamos que todos archivos han sido creados correctamente.

```
root@alexandria:~/xo/xo-server# ls -la
total 48
4 drwxr-xr-x 2 root root 4096 Dec 4 15:30 bin
4 -rw-r--r-- 1 root root 1411 Dec 4 17:21 certificate.pem
4 -rw-r--r-- 1 root root 1844 Dec 4 15:30 coffeelint.json
4 -rw-r--r-- 1 root root 162 Dec 4 15:30 index.js
4 -rw-r--r-- 1 root root 1834 Dec 4 17:21 key.pem
4 drwxr-xr-x 56 root root 4096 Dec 4 15:31 node_modules
4 -rw-r--r-- 1 root root 2224 Dec 4 15:30 package.json
4 -rw-r--r-- 1 root root 1164 Dec 4 15:30 README.md
4 -rwxr-xr-x 1 root root 561 Dec 4 15:30 run-tests
4 -rw-r--r-- 1 root root 2693 Dec 4 15:30 sample.config.yaml
4 drwxr-xr-x 4 root root 4096 Dec 4 15:30 src
4 -rw-r--r-- 1 root root 218 Dec 4 15:30 xo-server.service
```

## XO-Web

La instalación de este componente es muy similar al caso anterior, aunque esta vez instalaremos aquellas dependencias necesarias para poder desplegar la interfaz Web de la aplicación. Situados en el directorio xo-web ejecutamos lo siguiente.

```
root@alexandria:~/xo/xo-web# npm install
```

Para finalmente ejecutar

```
root@alexandria:~/xo/xo-server# ./gulp --production
```

Y si todo va como es esperado, podremos proceder a ejecutar el archivo que despliega el servidor mediante el comando.

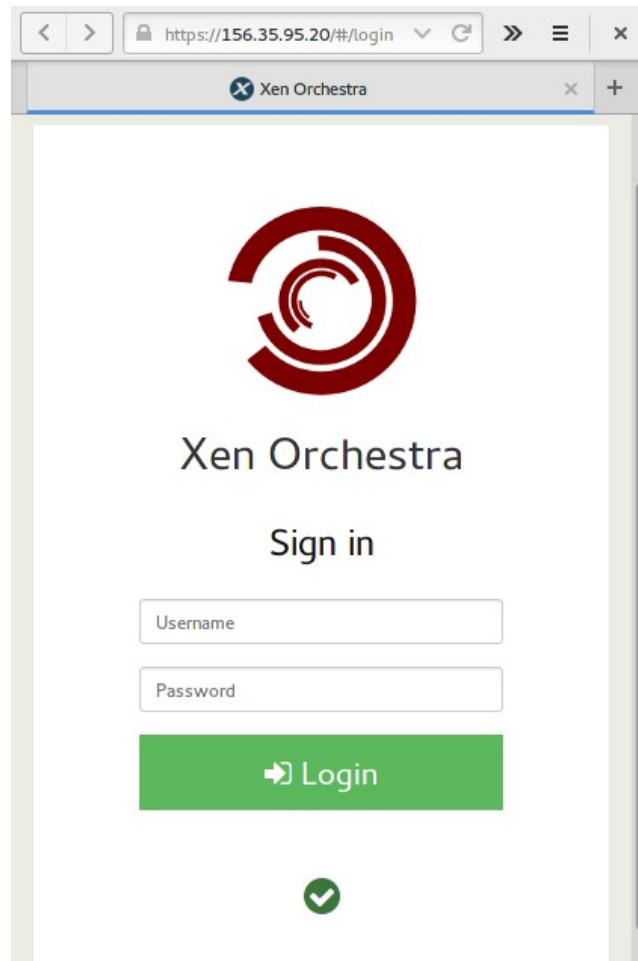
```
root@alexandria:~/xo/xo-server# ./bin/xo-server
xo:main Starting... +0ms
xo:main Configuration loaded. +54ms
[WARN] Web server could not listen on undefined
Address already in use.
xo:xapi root@156.35.94.186:443: session.login_with_password(...)
+213ms
xo:main Initializing connection to Xen servers... +9ms
xo:main Setting up / → ../xo-web/dist/ +1ms
xo:xapi Logged in root@156.35.94.186:443 (session =
OpaqueRef:2f3add03-d05b-6627-64c1-1b9665d5452e) +66ms
xo:xapi root@156.35.94.186:443: system.listMethods(...) +0ms
xo:xapi root@156.35.94.186:443: pool.get_all_records(...) +74ms
xo:xapi root@156.35.94.186:443: subject.get_all_records(...) +25ms
xo:xapi root@156.35.94.186:443: role.get_all_records(...) +1ms
```

```
xo:xapi root@156.35.94.186:443: task.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: pool_patch.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VM.get_all_records(...) +3ms
xo:xapi root@156.35.94.186:443: VM_metrics.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VM_guest_metrics.get_all_records(...)
+1ms
xo:xapi root@156.35.94.186:443: VMPP.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VM_appliance.get_all_records(...)
+0ms
xo:xapi root@156.35.94.186:443: DR_task.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: host.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: host_crashdump.get_all_records(...)
+1ms
xo:xapi root@156.35.94.186:443: host_patch.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: host_metrics.get_all_records(...)
+0ms
xo:xapi root@156.35.94.186:443: host_cpu.get_all_records(...) +2ms
xo:xapi root@156.35.94.186:443: network.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VIF.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VIF_metrics.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: PIF.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: PIF_metrics.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: Bond.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VLAN.get_all_records(...) +0ms
xo:xapi root@156.35.94.186:443: SM.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: SR.get_all_records(...) +3ms
xo:xapi root@156.35.94.186:443: VDI.get_all_records(...) +0ms
xo:xapi root@156.35.94.186:443: VBD.get_all_records(...) +2ms
xo:xapi root@156.35.94.186:443: VBD_metrics.get_all_records(...) +2ms
xo:xapi root@156.35.94.186:443: PBD.get_all_records(...) +2ms
xo:xapi root@156.35.94.186:443: crashdump.get_all_records(...) +0ms
xo:xapi root@156.35.94.186:443: console.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: blob.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: message.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: secret.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: tunnel.get_all_records(...) +0ms
xo:xapi root@156.35.94.186:443: PCI.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: PGPU.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: GPU_group.get_all_records(...) +1ms
xo:xapi root@156.35.94.186:443: VGPU.get_all_records(...) +1ms
xo:xapi Error from root@156.35.94.186:443: MESSAGE_REMOVED +369ms
xo:xo 1917 objects fetched from root@156.35.94.186 +2s
xo:xo objects inserted into the database +767ms
xo:xapi root@156.35.94.186:443: event.register(...) +1ms
xo:xapi root@156.35.94.186:443: event.next(...) +22ms
```

Obviamente para que tengamos acceso a la interfaz de Xen Orchestra desde el navegador, será necesario abrir el puerto en el que se está ejecutando la aplicación y que hemos configurado en nuestro archivo de propiedades '`xo-server.yaml`'. Para este caso de ejemplo se ha configurado el servicio el puerto 443, por lo que procederemos a abrir este puerto

para todas las conexiones entrantes. Como en anteriores ocasiones utilizaremos la interfaz para gestionar 'iptables' llamada 'ufw'.

```
root@alexandria:~# ufw allow from any to any port 443 proto tcp
```



*Imagen 17: Página de login de Xen Orchestra*

## Instalación de Scientific Linux

En este apartado veremos el proceso de instalación del sistema que nos servirá como anfitrión para OpenStack así como su algunos pasos necesarios para su configuración inicial. con estos parámetros.

Nombre	IP	Hardware	Descripción
lenuage	156.35.95.22	<ul style="list-style-type: none"><li>• 8 núcleos</li><li>• 80 Gb de RAM</li><li>• 500 Gb de disco</li></ul>	Scientific Linux + OpenStack

La instalación de Scientific Linux no tiene demasiadas complicaciones si seguimos uno a uno los pasos del asistente. Para facilitar las cosas a los usuario menos avanzados, el proceso es de instalación en Scientific Linux resulta completamente guiado a través de su asistente gráfico de instalación.

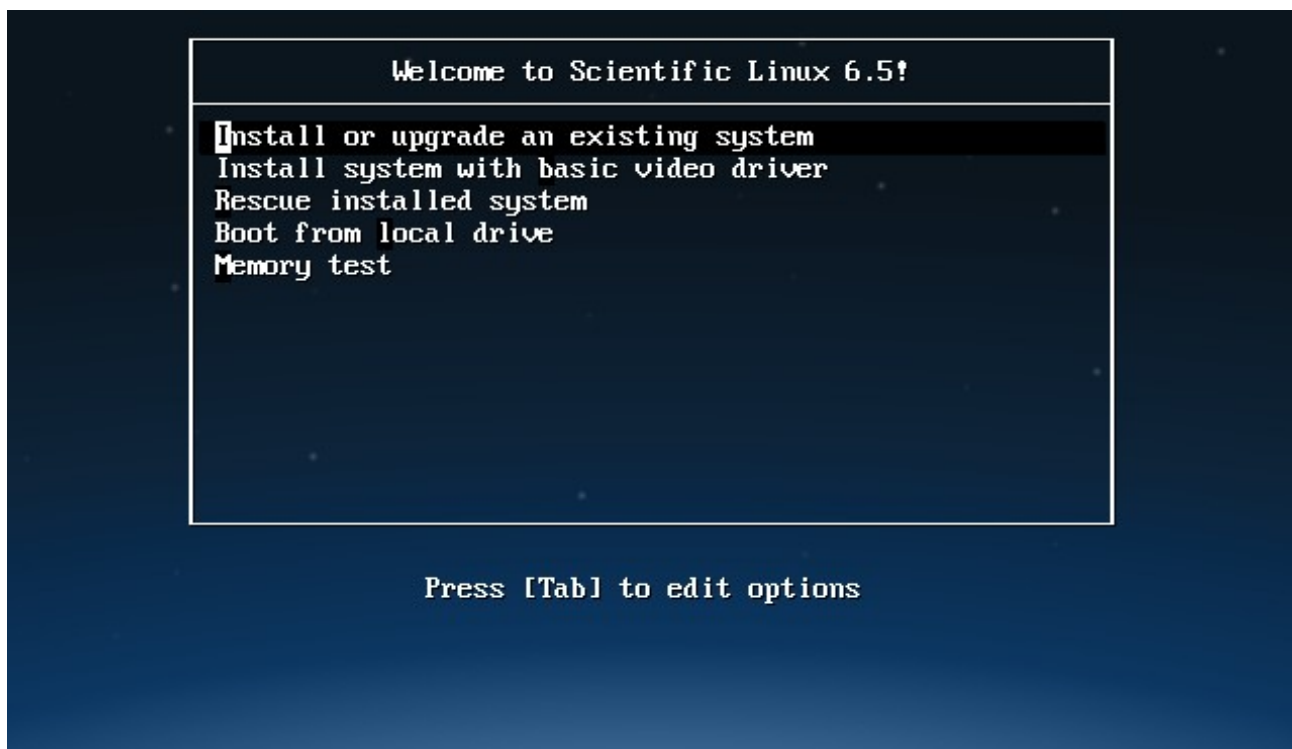


Imagen 18: Pantalla de inicial de instalación Scientific Linux

Así mismo a lo largo del proceso de instalación el propio configurador nos dará opción para asignar la contraseña que utilizaremos para el usuario administrador, editar la configuración de red y discos así como el nombre de la máquina.

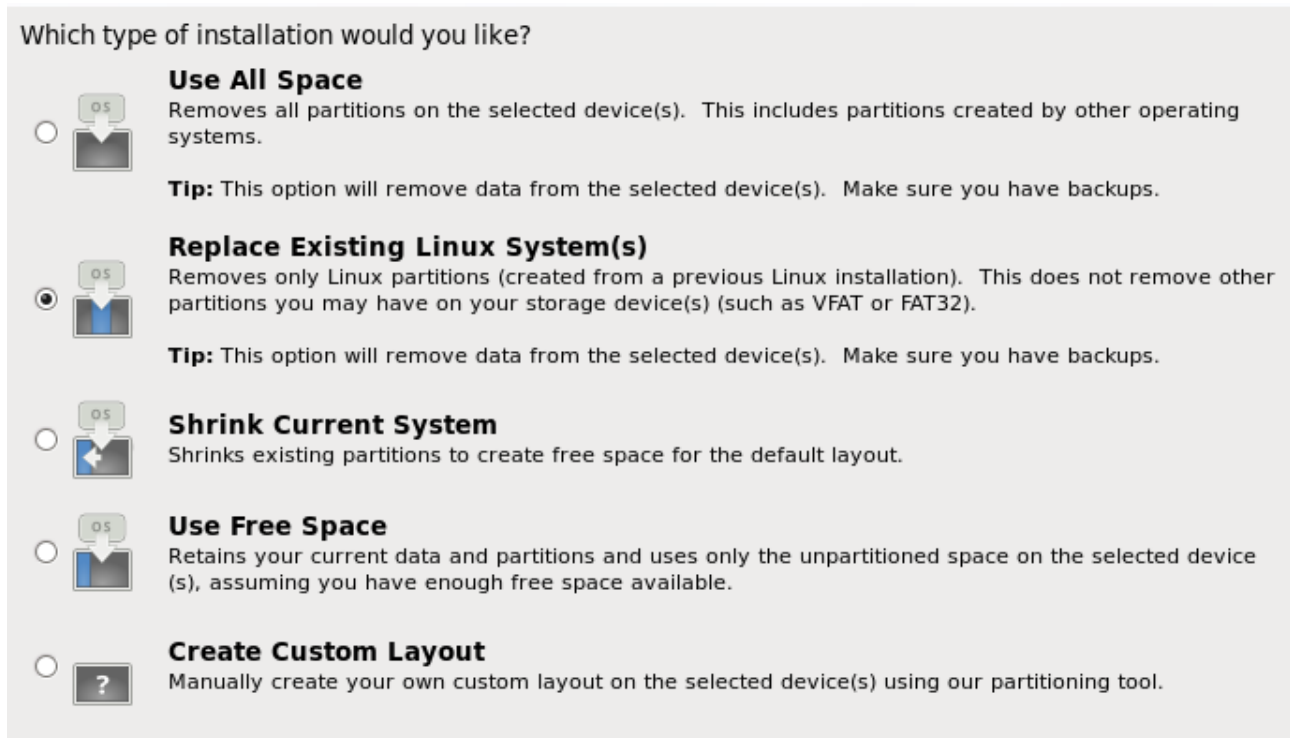


Imagen 19: Scientific Linux proporciona varios tipos de instalación

Finalizada la instalación y una vez reiniciado el sistema, utilizaremos la consola que nos proporciona Xen Center para asignar a nuestra máquina recién creada, una dirección IP estática. Éste será un paso necesario para que pueda comenzar a comunicarse con el resto de instancias desplegadas en el sistema.

## Configuración de red

Al tratarse de una distribución basada en CentOS6 que a su vez está basada en RedHat Enterprise Linux6 el procedimiento de configuración será idéntico al requerido para cualquiera de estas.

Para ello editaremos los siguientes ficheros

- /etc/sysconfig/network
- /etc/sysconfig/network-scripts/ifcfg-NOMBRE\_ADAPTADOR\_DE\_RED
- /etc/resolv.conf

En el primero bastará con especificar el nombre FQDN (Fully Qualified Domain Name o nombre que incluye el nombre de la máquina y el nombre de dominio asociado a ese equipo) de la máquina , la dirección IP de la puerta e enlace y una variable lógica que determinará si la conexión está habilitada o no.

Con todo esto y para nuestro caso particular, editamos el archivo ...

```
[root@lenuage ~]# vim /etc/sysconfig/network
```

... y añadimos lo siguiente.

```
NETWORKING=yes  
HOSTNAME=lenuage.tfg  
GATEWAY=156.35.94.201
```

Antes de editar el segundo archivo de los mencionados anteriormente, debemos averiguar primero cual es el nombre de la interfaz de red activa. Aunque normalmente la interfaz de red suele responder a la etiquetada con el nombre 'eth0', esto no es siempre así pudiendo darse el caso de usar otros nombres distintos en el caso de utilizar otro tipo de adaptadores, como por ejemplo inalámbricos.

En cualquier caso, para evitar problemas utilizaremos la siguiente orden, que devolverá el nombre de la interfaz activa que esta siendo utilizada.

```
[root@lenuage ~]# ip addr | grep "state UP" | awk '{print "Interfaz de Red: " $2}'  
Interfaz de Red: eth0:  
[root@lenuage ~]#
```

Como era previsible, la interfaz de red utilizada en este caso se corresponde con 'eth0'. Confirmado este dato, nos centraremos pues en editar el fichero almacenado en '/etc/sysconfig/network-scripts/ifcfg-eth0'.

```
[root@lenuage ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
```

Y completamos las variables, 'ONBOOT', 'BOOTPROTO', 'HWADDR' y 'IPADDR'.

```
DEVICE=eth0  
HWADDR=3E:70:26:0F:CE:11  
TYPE=Ethernet  
UUID=ba18bda3-5921-46c3-928e-73d09ef44325  
ONBOOT=yes  
NM_CONTROLLED=yes  
BOOTPROTO=static  
IPADDR=156.35.95.22  
NETMASK=255.255.254.0
```



Para identificar la dirección MAC de la tarjeta de red, es posible consultar su valor tecleando la siguiente orden.

```
[root@lenuage ~]# ip addr
...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP qlen 1000
    link/ether 3e:70:26:0f:ce:11 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::3c70:26ff:fe0f:ce11/64 scope link
        valid_lft forever preferred_lft forever
...
```

Para el caso del tercer archivo a editar, añadiremos la información relacionada con los servidores DNS y el dominio de búsqueda.

Para este proyecto el contenido de '/etc/resolv.conf' será el siguiente:

```
nameserver 156.35.14.2
nameserver 156.35.14.3
search ccu.uniovi.es
```

Ya por último, para que se apliquen los cambios realizados, será necesario reiniciar el servicio de red así como verificar que el mismo esté configurado para ser arrancado automáticamente al inicio del sistema.

```
[root@lenuage ~]# chkconfig network on
[root@lenuage ~]# service network restart
```

Una vez realizados estos cambios, comprobaremos la conectividad con el exterior realizando 'ping' a uno de los servidores públicos DNS de Google.

```
[root@lenuage ~]# ping -c 4 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=45 time=45.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=45 time=45.2 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=45 time=45.4 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=45 time=45.3 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3049ms
rtt min/avg/max/mdev = 45.214/45.430/45.798/0.268 ms
[root@lenuage ~]#
```

En caso de éxito, probaremos que nuestro servidor DNS resuelve correctamente.

```
[root@lenuage ~]# ping -c 4 www.google.es
PING www.google.es (74.125.230.56) 56(84) bytes of data.
64 bytes from mad01s25-in-f24.1e100.net (74.125.230.56): icmp_seq=1
ttl=53 time=7.85 ms
```

```
64 bytes from mad01s25-in-f24.1e100.net (74.125.230.56): icmp_seq=2
ttl=53 time=7.64 ms
64 bytes from mad01s25-in-f24.1e100.net (74.125.230.56): icmp_seq=3
ttl=53 time=7.65 ms
64 bytes from mad01s25-in-f24.1e100.net (74.125.230.56): icmp_seq=4
ttl=53 time=7.66 ms

--- www.google.es ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3012ms
rtt min/avg/max/mdev = 7.644/7.704/7.858/0.124 ms
[root@lenuage ~]#
```

Una vez finalizada la configuración la conectividad con el exterior, revisaremos que SSH está activado para así dejar de utilizar la limitada consola web que nos proporciona Xen Center.

```
[root@lenuage ~]# service sshd status
openssh-daemon (pid 1360) is running...
[root@lenuage ~]#
```

El servicio está siendo ejecutado en segundo plano tal y como esperábamos. Ahora comprobaremos si el cortafuegos de la máquina está configurado para recibir peticiones a través del puerto 22 (por defecto el puerto que usa SSH)

```
[root@lenuage ~]# iptables -S | grep '\-\-dport 22'
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
[root@lenuage ~]#
```

\*en el caso de arriba utilizamos el comando 'grep' para filtrar la salida de 'iptables' y que así sólo nos muestre la información que nos interesa

Vemos que el puerto 22 está abierto y escuchando, sólo queda conectarse directamente a la máquina por ssh.

```
Welcome to fish, the friendly interactive shell
Type help for instructions on how to use fish
dafero@archenvy ~/g/tfg> ssh root@156.35.95.22
root@156.35.95.22's password:
Last login: Mon Apr 13 19:48:40 2015 from 85-169-74-
185.rev.numericable.fr
[root@lenuage ~]#
```

## Instalación de OpenStack mediante RDO Packstack

Completados los pasos iniciales de la instalación del sistema y su configuración de acceso, procederemos con la instalación de OpenStack. Para ello utilizaremos una herramienta

llamada RDO Packstack desarrollada por la empresa RedHat y que facilitará enormemente la instalación de los distintos módulos que componen OpenStack.

A través de un archivo de respuestas que generaremos al inicio, packstack se encargará de instalar todos los servicios allí especificados así como algunos de los pasos necesarios para su configuración inicial, como las configuraciones de red, excepciones en los cortafuegos, plantillas de configuración, etc.

En primer lugar, antes de comenzar con la instalación en sí, es muy recomendable verificar que el todos los paquetes del sistema se encuentran actualizados a la última versión disponible en los repositorios.

```
[root@lenuage ~]# yum update -y
Loaded plugins: priorities, security
Setting up Update Process
epel/metalink | 27 kB
00:00
foreman-plugins | 2.9 kB
00:00
sl | 3.6 kB
00:00
sl-security | 2.9 kB
00:00
sl-security/primary_db | 2.4 MB
00:11
sl6x | 3.6 kB
00:00
sl6x-security | 2.9 kB
00:00
sl6x-security/primary_db | 2.4 MB
00:11
No packages marked for update
[root@lenuage ~]#
```

Para comenzar con el uso de packstack, instalaremos un paquete RPM específicamente preparado por RedHat, que nos configurará los repositorios necesarios para la posterior instalación del programa.

```
[root@lenuage ~]# yum install http://rdo.fedorapeople.org/openstack-icehouse/rdo-release-icehouse.rpm
Loaded plugins: changelog, kernel-module, priorities, rpm-warm-cache, security, tsflags, versionlock
Setting up Install Process
rdo-release-icehouse.rpm
| 13 kB 00:00
Examining /var/tmp/yum-root-4eN623/rdo-release-icehouse.rpm: rdo-release-icehouse-4.noarch
Marking /var/tmp/yum-root-4eN623/rdo-release-icehouse.rpm to be installed
```

```
202 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package rdo-release.noarch 0:icehouse-4 will be installed
--> Finished Dependency Resolution
Beginning Kernel Module Plugin
Finished Kernel Module Plugin

Dependencies Resolved

=====
Package           Arch      Version           Repository        Size
=====
Installing:
  rdo-release     noarch    icehouse-4        /rdo-release-icehouse 10 k
=====

Transaction Summary
=====
Install      1 Package(s)

Total size: 10 k
Installed size: 10 k
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : rdo-release-icehouse-4.noarch
1/1
  Verifying   : rdo-release-icehouse-4.noarch
1/1

Installed:
  rdo-release.noarch 0:icehouse-4

Complete!
[root@lenuage ~]#
```

Una vez instalados los repositorios procederemos con la instalación de la herramienta.

```
[root@lenuage ~]#yum install openstack-packstackLoaded plugins:
changelog, kernel-module, priorities, rpm-warm-cache, security,
: tsflags, versionlock
Setting up Install Process
openstack-icehouse | 2.9 kB
00:00
331 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package openstack-packstack.noarch 0:2014.1.1-0.31.5.dev1280.el6
```

```
will be installed
--> Processing Dependency: openstack-packstack-puppet = 2014.1.1-
0.31.5.dev1280.el6 for package: openstack-packstack-2014.1.1-
0.31.5.dev1280.el6.noarch
--> Processing Dependency: python-netaddr for package: openstack-
packstack-2014.1.1-0.31.5.dev1280.el6.noarch
--> Processing Dependency: openstack-puppet-modules for package:
openstack-packstack-2014.1.1-0.31.5.dev1280.el6.noarch
--> Running transaction check
--> Package openstack-packstack-puppet.noarch 0:2014.1.1-
0.31.5.dev1280.el6 will be installed
--> Package openstack-puppet-modules.noarch 0:2014.1.1-4.el6 will be
installed
--> Package python-netaddr.noarch 0:0.7.5-4.el6 will be installed
--> Finished Dependency Resolution
Beginning Kernel Module Plugin
Finished Kernel Module Plugin

Dependencies Resolved

=====
Package                Arch    Version                               Repository    Size
=====
Installing:
openstack-packstack          noarch 2014.1.1-0.31.5.dev1280.el6    openstack-
icehouse 214 k
Installing for dependencies:
openstack-packstack-puppet  noarch 2014.1.1-0.31.5.dev1280.el6    openstack-
icehouse 44 k
openstack-puppet-modules    noarch 2014.1.1-4.el6                openstack-
icehouse 1.3 M
python-netaddr              noarch 0.7.5-4.el6                   slc6-os
1.0 M

Transaction Summary
=====
Install      4 Package(s)

Total size: 2.6 M
Total download size: 1.4 M
Installed size: 8.9 M
Is this ok [y/N]: y
Downloading Packages:
(1/2): openstack-packstack-puppet-2014.1.1-0.31.5.dev12 | 44 kB
00:00
(2/2): openstack-puppet-modules-2014.1.1-4.el6.noarch.r | 1.3 MB
00:00
-----
Total                                     267 kB/s | 1.4 MB
```

```
00:05
warning: rpmts_HdrFromFdno: Header V4 RSA/SHA1 Signature, key ID
0e4fbd28: NOKEY
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-RD0-Icehouse
Importing GPG key 0x0E4FBD28:
  Userid : rdo-icehouse-sign <rdo-info@redhat.com>
  Package: rdo-release-icehouse-4.noarch (@/rdo-release-icehouse)
  From   : /etc/pki/rpm-gpg/RPM-GPG-KEY-RD0-Icehouse
Is this ok [y/N]: y
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : python-netaddr-0.7.5-4.el6.noarch
1/4
  Installing : openstack-packstack-puppet-2014.1.1-
0.31.5.dev1280.el6.noarch 2/4
  Installing : openstack-puppet-modules-2014.1.1-4.el6.noarch
3/4
  Installing : openstack-packstack-2014.1.1-0.31.5.dev1280.el6.noarch
4/4
  Verifying   : openstack-puppet-modules-2014.1.1-4.el6.noarch
1/4
  Verifying   : openstack-packstack-2014.1.1-0.31.5.dev1280.el6.noarch
2/4
  Verifying   : openstack-packstack-puppet-2014.1.1-
0.31.5.dev1280.el6.noarch 3/4
  Verifying   : python-netaddr-0.7.5-4.el6.noarch
4/4

Installed:
  openstack-packstack.noarch 0:2014.1.1-0.31.5.dev1280.el6

Dependency Installed:
  openstack-packstack-puppet.noarch 0:2014.1.1-0.31.5.dev1280.el6
  openstack-puppet-modules.noarch 0:2014.1.1-4.el6
  python-netaddr.noarch 0:0.7.5-4.el6

Complete!
[root@lenuage ~]#
```

Una vez completada la instalación podemos ejecutar el comando 'packstack' que nos mostrará un gran número de preguntas con las que poder generar una instalación lo más acorde posible a nuestras necesidades.

```
[root@lenuage ~]# packstack
Welcome to Installer setup utility
Enter the path to your ssh Public key to install on servers
[/root/.ssh/id_rsa.pub] :
Should Packstack install MariaDB [y|n] [y] : y
```

```
Should Packstack install OpenStack Image Service (Glance) [y|n] [y] :
y
Should Packstack install OpenStack Block Storage (Cinder) service [y|n]
[y] : y
Should Packstack install OpenStack Compute (Nova) service [y|n] [y] :
y
Should Packstack install OpenStack Networking (Neutron) service [y|n]
[y] : n
Should Packstack install OpenStack Dashboard (Horizon) [y|n] [y] : y
Should Packstack install OpenStack Object Storage (Swift) [y|n] [y] :
y
Should Packstack install OpenStack Metering (Ceilometer) [y|n] [y] : n
Should Packstack install OpenStack Orchestration (Heat) [y|n] [n] : n
Should Packstack install OpenStack client tools [y|n] [y] : y
Enter a comma separated list of NTP server(s). Leave plain if Packstack
should not install ntpd on instances.:
Should Packstack install Nagios to monitor OpenStack hosts [y|n] [y] :

...
```

Al finalizar el este proceso se generara un fichero de texto con una serie de variables que se corresponderán con la configuración deseada.

```
[general]

# Path to a Public key to install on servers. If a usable key has not
# been installed on the remote servers the user will be prompted for a
# password and this key will be installed so the password will not be
# required again
CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub

# Set to 'y' if you would like Packstack to install MySQL
CONFIG_MYSQL_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Image
# Service (Glance)
CONFIG_GLANCE_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Block
# Storage (Cinder)
CONFIG_CINDER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Compute
# (Nova)
CONFIG_NOVA_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Networking (Neutron)
CONFIG_NEUTRON_INSTALL=n
```

. . .

Para instalar OpenStack a través del recién generado fichero de respuestas, bastará con ejecutar lo siguiente.

```
packstack --answer-file=<ANSWER_FILE>
```

Packstack internamente utiliza manifiestos de Puppet con aquellos recursos necesarios para proceder con los parámetros de instalación especificados. El proceso de instalación suele ser bastante largo y en mi pruebas solía tardar en torno a los 30 minutos.

Una vez completado el proceso, se mostrará un mensaje con información útil relacionada con la instalación. Así mismo packstack generará automáticamente un fichero con credenciales para el usuario 'admin' que podemos utilizar para conectarnos por primera vez al sistema.



## Instalación y configuración de ownCloud en Debian

En este apartado cubriremos el proceso de instalación y configuración de nuestro almacén de copias ownCloud.

Como en el caso del Xen Orchestra se ha optado por reutilizar el mismo servidor inicialmente dedicado a proporcionarnos la biblioteca de imágenes ISO.

Nombre	IP	Hardware	Descripción
alexandria	156.35.95.20	<ul style="list-style-type: none"><li>• 1 núcleo</li><li>• 8 Gb de RAM</li><li>• 150 Gb de disco</li></ul>	Debian con ownCloud para alojar las copias de seguridad

### Instalación

En primer lugar, antes incluso de proceder con la instalación en sí misma, descargamos y añadimos la clave pública del repositorio de ownCloud a nuestro almacén de claves del sistema. Este paso es importante dado que en caso contrario, simplemente añadiendo el repositorio y actualizando, nos saltaría el siguiente error de clave GPG:

```
http://download.opensuse.org Release: The following signatures
couldn't be verified because the public key is not available: NO_PUBKEY
977C43A8BA684223
```

Por tanto, para añadirla tecleamos como root:

```
root@alexandria:~# cd /tmp/
root@alexandria:/tmp# wget
http://download.opensuse.org/repositories/isv:ownCloud:community/Debian_
7.0/Release.key
--2014-12-10 19:42:50--
http://download.opensuse.org/repositories/isv:ownCloud:community/Debian_
7.0/Release.key
Resolving download.opensuse.org (download.opensuse.org)...
195.135.221.134, 2001:67c:2178:8::13
Connecting to download.opensuse.org (download.opensuse.org)|
195.135.221.134|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location:
http://download.opensuse.org/repositories/isv:ownCloud:/community/Debian
_7.0/Release.key [following]
--2014-12-10 19:42:51--
http://download.opensuse.org/repositories/isv:ownCloud:/community/Debian
```

```
_7.0/Release.key
Reusing existing connection to download.opensuse.org:80.
HTTP request sent, awaiting response... 301 Moved Permanently
Location:
http://download.opensuse.org/repositories/isv:/ownCloud:/community/Debian_7.0/Release.key [following]
--2014-12-10 19:42:51--
http://download.opensuse.org/repositories/isv:/ownCloud:/community/Debian_7.0/Release.key
Reusing existing connection to download.opensuse.org:80.
HTTP request sent, awaiting response... 200 OK
Length: 1003 [application/pgp-keys]
Saving to: `Release.key'

100%[=====>] 1,003      --.-K/s   in 0s

2014-12-10 19:42:51 (88.4 MB/s) - `Release.key' saved [1003/1003]

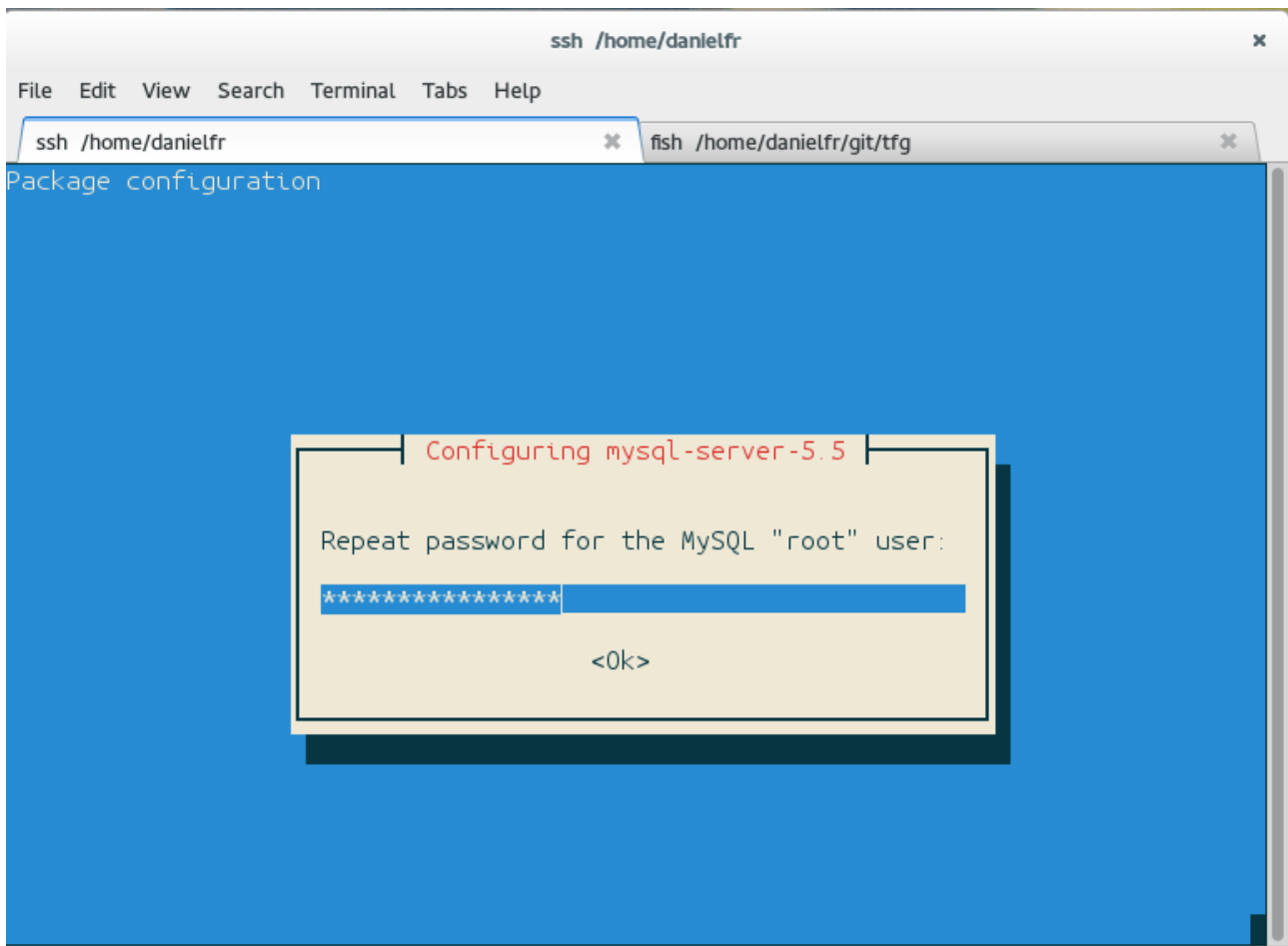
root@alexandria:/tmp# apt-key add - < Release.key
OK
root@alexandria:/tmp# echo 'deb
http://download.opensuse.org/repositories/isv:/ownCloud:/community/Debian_7.0/ /' >> /etc/apt/sources.list.d/owncloud.list
root@alexandria:/tmp#
```

Una vez realizado lo anterior actualizamos el sistema y procedemos con la instalación del paquete ownCloud y sus dependencias asociadas.

```
root@alexandria:/tmp# apt-get update && apt-get install owncloud
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  apache2 apache2-mpm-prefork apache2-utils apache2.2-bin apache2.2-common
  dbus fontconfig-config fonts-droid ghostscript gsfonts
  imagemagick-common libaio1 libapache2-mod-php5 libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap libavahi-client3 libavahi-common-data
  libavahi-common3 libcups2 libcupsimage2 libdbd-mysql-perl libdbi-perl
  libdbus-1-3 libffi5 libfontconfig1 libgd2-xpm libgl2.0-0 libgl2.0-data
  libgs9 libgs9-common libhtml-template-perl libice6 libicu48 libijs-0.35
  libjasper1 libjbig0 libjbig2dec0 libjpeg8 liblcms2-2 liblqr-1-0 libltdl7
  libmagickcore5 libmagickwand5 libmcrypt4 libmysqlclient18 libonig2
  libpaper-utils libpaper1 libpq5 libqdbm14 libsm6 libsystemd-login0
  libtiff4 libxpm4 libxt6 mysql-client-5.5 mysql-common mysql-server
  mysql-server-5.5 mysql-server-core-5.5 php-pear php-xml-parser php5
  php5-cli php5-common php5-curl php5-gd php5-imagick php5-intl php5-mcrypt
  php5-mysqld php5-pgsql php5-sqlite poppler-data shared-mime-info ssl-cert
  ttf-dejavu-core x11-common
Suggested packages:
  apache2-doc apache2-suexec apache2-suexec-custom dbus-x11
```

```
ghostscript-cups ghostscript-x hpijs cups-common libgd-tools libipc-
sharedcache-perl libjasper-runtime liblcms2-utils libmagickcore5-extra
  libmcrypt-dev mcrypt libterm-readkey-perl tinyca clamav clamav-daemon
smbclient libav-tools ffmpeg libreoffice-writer php5-dev poppler-utils
fonts-japanese-mincho fonts-ipafont-mincho fonts-japanese-gothic
  fonts-ipafont-gothic fonts-arphic-ukai fonts-arphic-uming fonts-
unfonts-core openssl-blacklist
Recommended packages:
  php5-apc
The following NEW packages will be installed:
  apache2 apache2-mpm-prefork apache2-utils apache2.2-bin apache2.2-
common dbus fontconfig-config fonts-droid ghostscript gsfonts
imagemagick-common libaiol libapache2-mod-php5 libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap libavahi-client3 libavahi-
common-data libavahi-common3 libcups2 libcupsimage2 libdbd-mysql-perl
libdbi-perl libdbus-1-3 libffi5 libfontconfig1 libgd2-xpm libglib2.0-0
  libglib2.0-data libgs9 libgs9-common libhtml-template-perl libice6
libcups2 libicu48 libijs-0.35 libjasper1 libjbig0 libjbig2dec0 libjpeg8 liblcms2-
2 liblqr-1-0 libltdl7 libmagickcore5 libmagickwand5 libmcrypt4
  libmysqlclient18 libonig2 libpaper-utils libpaper1 libpq5 libqdbm14
libsm6 libsystemd-login0 libtiff4 libxpm4 libxt6 mysql-client-5.5 mysql-
common mysql-server mysql-server-5.5 mysql-server-core-5.5 owncloud
  php-pear php-xml-parser php5 php5-cli php5-common php5-curl php5-gd
php5-imagick php5-intl php5-mcrypt php5-mysqlnd php5-pgsql php5-sqlite
poppler-data shared-mime-info ssl-cert ttf-dejavu-core x11-common
0 upgraded, 81 newly installed, 0 to remove and 11 not upgraded.
Need to get 80.6 MB of archives.
After this operation, 332 MB of additional disk space will be used.
Do you want to continue [Y/n]?
```

Durante la instalación se nos abrirá un diálogo para configurar el acceso a la base de datos SQL que servirá de backend de la aplicación. Para la versión aquí instalada de ownCloud se utilizará una base de datos MySQL v5.5.



*Imagen 20: Instalación de la base de datos MySQL para ownCloud*

Completamos todos los pasos del asistente hasta completar el proceso instalación. Una vez finalizado ya estaremos listos para utilizar ownCloud a través de la siguiente dirección:

- <http://156.35.95.20/owncloud/index.php>

Aunque antes de comenzar realizaremos algunos ajustes en la configuración para poder utilizar HTTP/SSL (importantísimo si no queremos que nuestras credenciales viajen en texto plano) y cambiar el puerto donde se ejecuta el servicio dado que es más que probable que en el 80/443 ya estemos ejecutando otros servicios.

Como se puede observar del apartado anterior echar a andar una configuración básica de ownCloud es un proceso muy directo y sencillo. A pesar de esto, una vez instalado es recomendable al menos configurar el acceso por HTTP/SSL para que así todo acceso a la aplicación web sea cifrado y aquellos datos sensibles como, nombres de usuario, contraseñas o archivos no viajen por la red en texto plano.

## Configuración HTTP/SSL en ownCloud

En primer lugar y como cuando confirmamos el acceso HTTPS para XenOrchestra, crearemos nuestros propios certificados. Aunque si bien es cierto que en este caso seguiremos un proceso muy similar aunque ligeramente distinto debido a Apache, utilizado por ownCloud como servidor Web. Como en otras ocasiones, en aras de la simplicidad, almacenaremos los certificados auto-generados en un directorio interno que crearemos para este fin.

### Generación de Certificados

En primer lugar añadiremos un directorio en el que almacenaremos el certificado en el directorio donde se encuentra la configuración del servidor web Apache.

```
root@alexandria:~# cd /etc/apache2/  
root@alexandria:~# mkdir Cert0wncloud  
root@alexandria:~# cd Cert0wncloud
```

Una vez dentro del directorio ejecutaremos las siguientes órdenes para generar las claves con las que crearemos el certificado:

```
root@alexandria:/etc/apache2/Cert0wncloud# openssl genrsa -out  
owncloud.key 2048  
Generating RSA private key, 2048 bit long modulus  
....+++  
.....+++  
e is 65537 (0x10001)  
Una vez generada, a partir de esta crearemos los ficheros .key y .csr  
root@alexandria:/etc/apache2/Cert0wncloud# openssl req -new -key  
owncloud.key -out owncloud.csr
```

Será a partir de ellos estos dos archivos mediante los que crearemos nuestro certificado:

```
root@alexandria:/etc/apache2/Cert0wncloud# openssl x509 -req -days 365  
-in owncloud.csr -signkey owncloud.key -out owncloud.crt
```

Una vez realizado lo anterior, bastará con copiar cada archivo en aquellos directorios que más tarde especificaremos en la configuración de Apache. Por lo que conviene no perderlos de vista.

En nuestro caso los dejaremos en '/etc/ssl/certs' para el certificado y '/etc/ssl/private' para la clave privada.

```
root@alexandria:/etc/apache2/Cert0wncloud# cp owncloud.crt  
/etc/ssl/certs  
root@alexandria:/etc/apache2/Cert0wncloud# cp owncloud.key  
/etc/ssl/private
```

## Configuración Apache

En el caso de la configuración de Apache sólo será necesario añadir un archivo el cual contenga la configuración que veremos a continuación. Para ello, creamos un nuevo fichero llamado 'owncloud.https' dentro del siguiente directorio '/etc/apache2/sites-available/'

```
root@alexandria: vim /etc/apache2/sites-available/owncloud.https
```

En el que definiremos las siguientes líneas.

```
NameVirtualHost *:4433
<VirtualHost *:4433>
DocumentRoot /var/www/owncloud
# Configuración de SSL
SSLEngine On
SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
# Ruta a los Certificados
SSLCertificateFile /etc/ssl/certs/owncloud.crt
SSLCertificateKeyFile /etc/ssl/private/owncloud.key
</VirtualHost>
```

Importante remarcar el puerto de escucha, en este caso se usará el 4433 para no generar conflicto con otros servicios, así como la ruta especificada en DocumentRoot y los directorios donde previamente guardamos el certificado así como la clave privada.

Dado que estamos utilizando un puerto de los denominados 'no estándar', cambiaremos la configuración del puerto de Apache, por defecto en el 80. Para ello abrimos el archivo 'ports.conf' dentro del directorio de configuración de Apache y modificamos el valor al que hace referencia la etiqueta Listen por el puerto especificado anteriormente, 4433.

Como posteriormente activemos el uso del modulo mod\_ssl.c, necesario para la activación de SSL, también se modificará el puerto de escucha del apartado que hace referencia a la configuración del módulo.

```
root@alexandria:~# vim /etc/apache2/ports.conf
```

```
# If you just change the port or add more ports here, you will likely
also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default
# This is also true if you have upgraded from before 2.2.9-3 (i.e. from
# Debian etch). See /usr/share/doc/apache2.2-common/NEWS.Debian.gz and
# README.Debian.gz

NameVirtualHost *:80
Listen 4433
```

```
<IfModule mod_ssl.c>
    # If you add NameVirtualHost *:443 here, you will also have to
change
    # the VirtualHost statement in /etc/apache2/sites-
available/default-ssl
    # to <VirtualHost *:443>
    # Server Name Indication for SSL named virtual hosts is currently
not
    # supported by MSIE on Windows XP.
    Listen 4443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>
```

Una vez modificados ambos valores en el archivo anterior, sólo debemos activar algunos módulos en Apache y reiniciar el servicio para que estos últimos cambios realizados sean aplicados correctamente.

```
root@alexandria:~# a2enmod ssl
root@alexandria:~# a2ensite owncloud.https
root@alexandria:~# apachectl configtest
root@alexandria:~# service apache2 restart
[ ok ] Restarting web server: apache2 ... waiting .
root@alexandria:~#
```

Con esto ya hemos finalizado con Apache, aunque antes de comenzar a probar su funcionamiento será necesario añadir una regla con el puerto definido anteriormente en nuestro cortafuegos. Como en anteriores ocasiones, utilizaremos la interfaz para 'iptables', llamada 'ufw'.

```
root@alexandria:~# ufw allow 4433
```

Ahora sí, lo único que nos quedaría es probarlo para verificar su correcto funcionamiento. Abrimos un navegador y tecleamos:

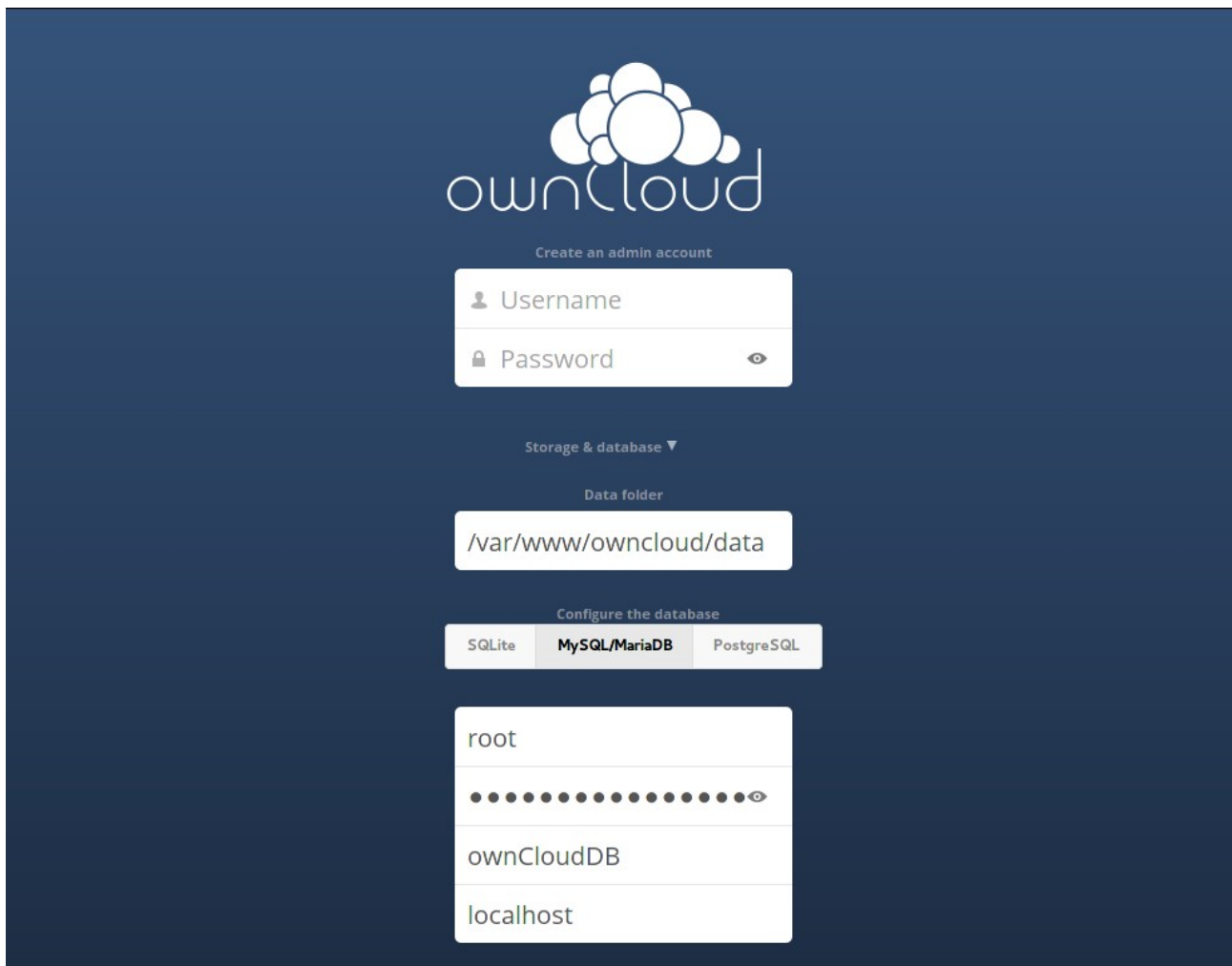
- <https://156.35.95.20:4433>

Y si todo va bien veremos la pantalla de ingreso de ownCloud en la que sólo nos quedará definir un usuario administrador de la aplicación así como la configuración de la base de datos.









*Imagen 21: Pantalla de login de ownCloud*

Una vez instalada la versión de servidor, instalaremos el cliente de escritorio que nos permitirá mantener nuestros archivos sincronizados en todo momento.

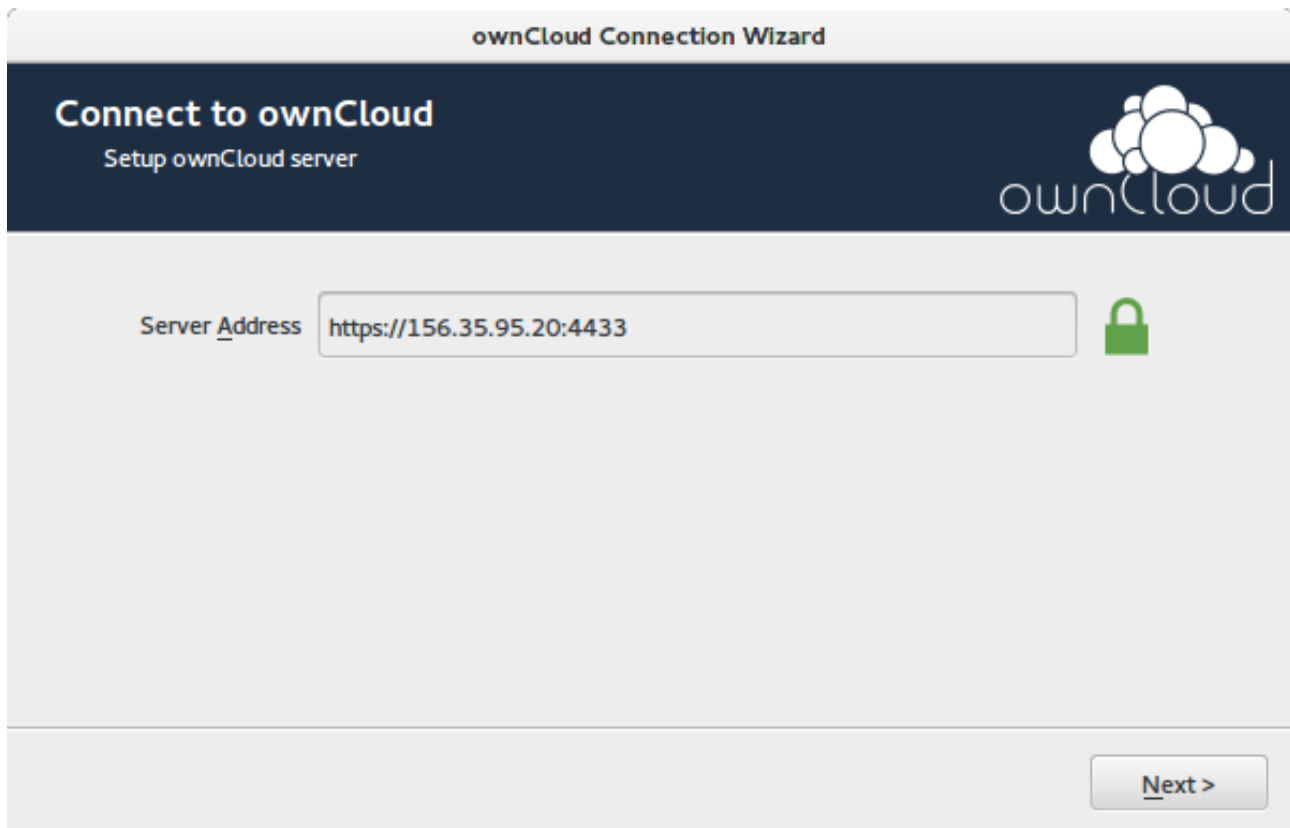
Desde la pagina oficial del proyecto proporcionan los binarios necesarios para la instalación del cliente de ownCloud para los sistemas operativos más comunes del mercado: Windows, Mac y GNU/Linux. En el caso de Linux nos proporciona los binarios para algunas de las distribuciones mas extendidas CentOS/RHEL, Fedora, openSUSE, Ubuntu y Debian. Así mismo y dado que está licenciado bajo una licencia de código abierto, e posible descargar directamente los archivos fuente del programa, para poder instalarlo desde tu propia compilación.

En mi caso concreto y dado que utilizo Arch Linux en mi ordenador personal, he optado por instalarlo desde los repositorios AUR (Arch User Repository) en lo que se puede encontrar la ultima versión compatible con este sistema, lista para ser instalada.

Para instalarla abrimos una terminal de órdenes y escribimos.

```
dafero@archenvy ~-> sudo yaourt -S owncloud-client --no-confirm
```

Una vez instalado el cliente, sólo queda completar todos los pasos del asistente de configuración para que se conecte a nuestro servidor.



*Imagen 22: Diálogo de configuración del cliente de escritorio onwCloud*

## Integración OwnCloud Draw.io

Draw.io, la herramienta gratuita de creación de diagramas en línea, proporciona por defecto integración con distintos servicios de almacenamiento externo. Gracias a esto es posible guardar los ficheros generados con la herramienta en sistemas de almacenamiento en la nube como por ejemplo Google Drive, Dropbox o Microsoft OneDrive.

Además a través de un complemento disponible desde la página oficial del proyecto, podemos configurar el servidor ownCloud para que este pueda funcionar como almacén para los diagramas draw.io. De esta forma guardaremos los archivos en nuestro servidor ownCloud e incluso podremos abrirlos directamente.

En primer lugar y como es lógico, será necesario descargar el complemento desde la sección de aplicaciones de la propia ownCloud:

<https://apps.owncloud.com/content/show.php/drawio?content=166206>

Para instalarlo lo único que hará falta es descomprimir su contenido en el directorio apps de la instalación de ownCloud en nuestro servidor.

```
root@alexandria:~# apt-get install unzip -y
root@alexandria:~# unzip 166206-files_drawio.zip
/var/www/owncloud/apps/
```

De acuerdo con la documentación oficial del complemento, una vez descomprimido será necesario cambiar los permisos de ejecución de los archivos recién instalados.

```
root@alexandria:~# chmod -R 755 /var/www/owncloud/apps/files_drawio
root@alexandria:~# ll /var/www/owncloud/apps/
total 100
drwxr-xr-x 11 root root 4096 Dec 10 19:52 activity
drwxr-xr-x  6 root root 4096 Dec 10 19:52 admin_dependencies_chk
drwxr-xr-x 12 root root 4096 Dec 10 19:52 bookmarks
drwxr-xr-x 12 root root 4096 Dec 10 19:52 calendar
drwxr-xr-x 11 root root 4096 Dec 10 19:52 contacts
drwxr-xr-x 11 root root 4096 Dec 10 19:52 documents
drwxr-xr-x 10 root root 4096 Dec 10 19:52 external
drwxr-xr-x 11 root root 4096 Dec 10 19:52 files
drwxr-xr-x  5 root root 4096 Jul 15 2014 files_drawio
drwxr-xr-x 13 root root 4096 Dec 10 19:52 files_encryption
drwxr-xr-x 11 root root 4096 Dec 10 19:52 files_external
drwxr-xr-x  6 root root 4096 Dec 10 19:52 files_pdfviewer
drwxr-xr-x 10 root root 4096 Dec 10 19:52 files_sharing
drwxr-xr-x  7 root root 4096 Dec 10 19:52 files_texteditor
drwxr-xr-x 10 root root 4096 Dec 10 19:52 files_trashbin
drwxr-xr-x  9 root root 4096 Dec 10 19:52 files_versions
drwxr-xr-x  7 root root 4096 Dec 10 19:52 files_videoviewer
drwxr-xr-x  9 root root 4096 Dec 10 19:52 firstrunwizard
drwxr-xr-x  9 root root 4096 Dec 10 19:52 gallery
drwxr-xr-x  9 root root 4096 Dec 10 19:52 search_lucene
drwxr-xr-x 11 root root 4096 Dec 10 19:52 templateeditor
drwxr-xr-x  9 root root 4096 Dec 10 19:52 updater
drwxr-xr-x  5 root root 4096 Dec 10 19:52 user_external
drwxr-xr-x 11 root root 4096 Dec 10 19:52 user_ldap
drwxr-xr-x  5 root root 4096 Dec 10 19:52 user_webdavauth
Como siempre al realizar un cambio de este tipo, será necesario
reiniciar el servicio.
root@alexandria:~# service apache2 restart
[ ok ] Restarting web server: apache2 ... waiting .
root@alexandria:~#
```

En este punto solo será necesario activamos el plugin desde la interfaz web

- <https://156.35.95.20:4433/index.php/settings/apps?installed>

Una vez hecho lo anterior veremos como una nueva opción con la etiqueta de 'Editar diagrama' está disponible.

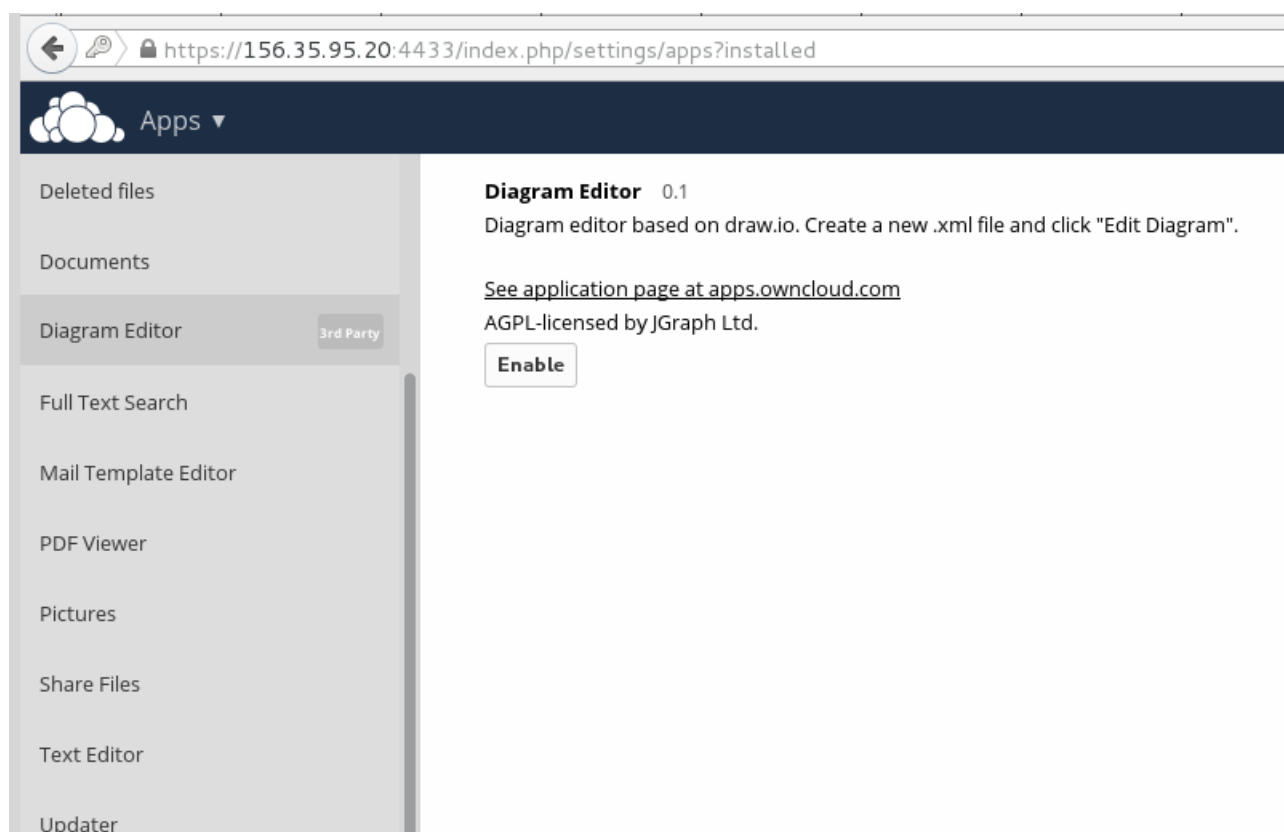


Imagen 23: Panel de activación de complementos en ownCloud

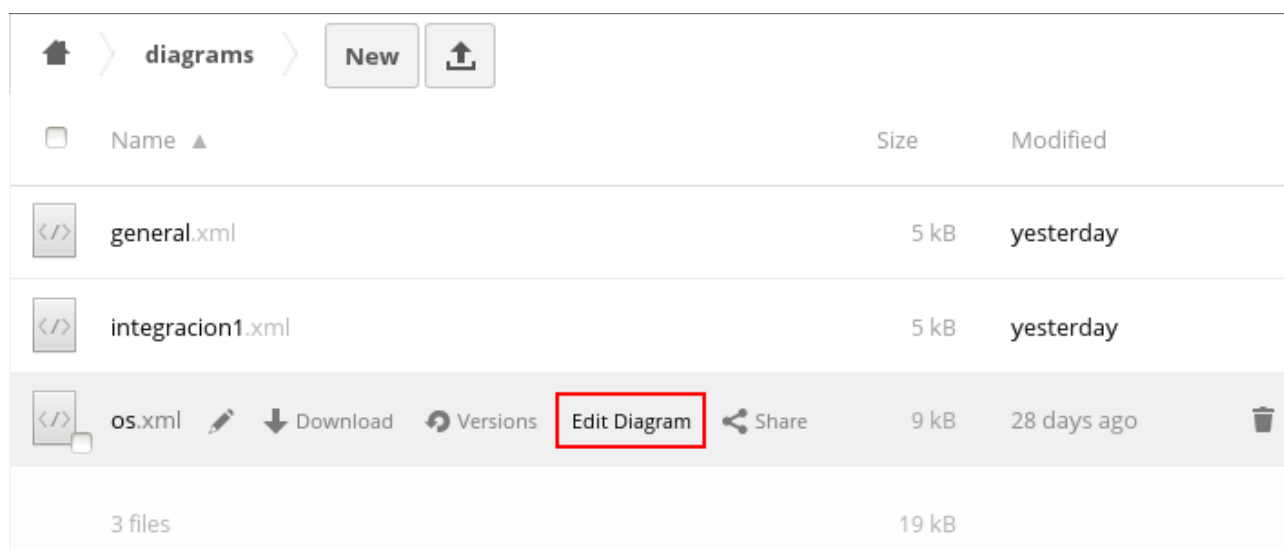


Imagen 24: Complemento de integración de ownCloud con draw.io



## OpenStack IceHouse

OpenStack IceHouse es el noveno lanzamiento de OpenStack desde que comenzó oficialmente su desarrollo en la segunda mitad del año 2010. Ésta versión fue liberada al público en mayo de 2014 y desde su lanzamiento y hasta la actualidad han salido cuatro revisiones en las que se han arreglado errores y fallos, mejorando su estabilidad y fiabilidad.

### Historia del desarrollo

OpenStack comenzó su andadura de la mano de la unión de dos proyectos pertenecientes a dos grandes compañías, la primera Rackspace una empresa americana de computación y alojamiento en la nube y la segunda, NASA (Agencia Espacia Norteamericana).

A principios del año 2010, Rackspace liberó bajo licencia de código abierto su conocido sistema operativo basado en tecnologías de nube, Swift. Por otro lado, la NASA por aquel entonces estaba utilizando un proyecto libre de desarrollo propio llamado OpenNebula, aunque distaba mucho de ser una solución madura.

De la unión de estos dos proyectos surgió OpenStack para dar una respuesta libre a la ya por aquel entonces ampliamente utilizada, Amazon EC2 (Amazon Elastic Cloud Computing).

### Versiones

Desde su creación en el año 2010, diversas versiones de OpenStack han visto la luz. OpenStack tienen un modelo de desarrollo muy rápido y ágil, con el lanzamiento de nuevas versiones cada aproximadamente 6 meses. Como en otros proyectos software, el sistema de nombrado de versiones sigue un orden alfabético, comenzando por la A.

Con esto, tenemos los siguientes nombres clave para las distintas versiones: Austin, Bexar, Cactus, Diablo, Essex, Grizzly, Havana, Icehouse, Juno y la futura Kilo.

Versión	Nombre	Lanzamiento	Componentes incluidos
1	Austin	Octubre 2010	Nova, Swift
2	Bexar	Febrero 2011	Nova, Glance, Swift



Daniel Fernandez Rodriguez  
 “Integración de OpenStack con un Directorio Activo”

3	Cactus	Abril 2011	Nova, Glance, Swift
4	Diablo	Septiembre 2011	Nova, Glance, Swift
5	Essex	Abril 2012	Nova, Glance, Swift, <b>Horizon, Keystone</b>
6	Folsom	Septiembre 2012	Nova, Glance, Swift, Horizon, Keystone, <b>Quantum (ex Neutron), Cinder</b>
7	Grizzly	Abril 2013	Nova, Glance, Swift, Horizon, Keystone, Quantum (ex Neutron), Cinder
8	Havana	Octubre 2013	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, <b>Ceilometer, Heat</b>
9	IceHouse	Abril 2014	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Ceilometer, Heat, <b>Trove</b>
10	Juno	Octubre 2014	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Ceilometer, Heat, Trove, <b>Sahara</b>
11	Kilo	Abril 2015	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Ceilometer, Heat, Trove, Sahara, <b>IroniC</b>

\* Lanzamientos de OpenStack desde sus inicios en 2010

Para facilitar la vida a los desarrolladores el sistema, el nombrado de paquetes es un poco distinto y en lugar de utilizar el nombre en clave se utiliza la nomenclatura: componente/proyecto, año, lanzamiento, revisión.

Nombre del componente/proyecto: Keystone, glance, nova...

Año: año de lanzamiento, 2014, 2015..

Lanzamiento: Se liberan dos versiones anuales por lo que este campo unicamente contendrá los valores 1 o 2.

Revisión: Distintas revisiones para un mismo paquete.

Con todo esto para un nombre de paquete:

- openstack-keystone-2014.1.2

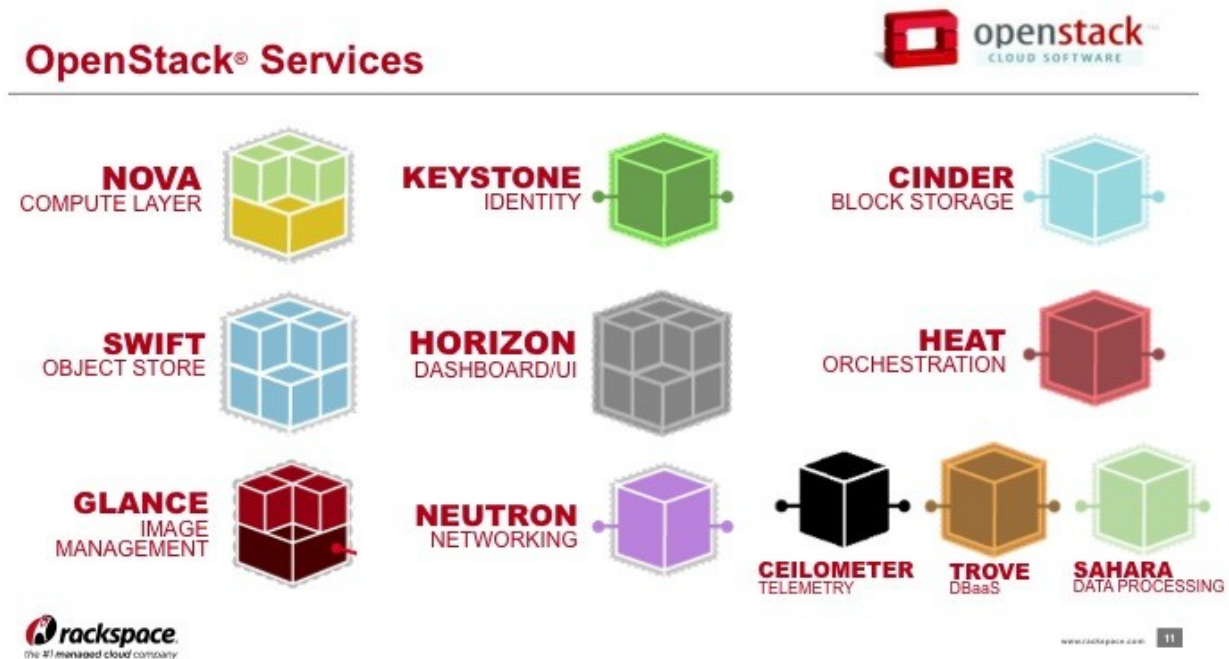
Tenemos el componente keystone, lanzado en la primera mitad del año 2014, nombre en clave de versión IceHouse y revisión número 2.

## Summits

Otra característica importante dentro de la comunidad OpenStack son los llamados summits. Estos summits son una serie de conferencias de unos 5 días de duración que se celebran aproximadamente cada 6 meses en distintas ciudades del mundo. Las conferencias proporcionan un punto de encuentro entre desarrolladores, clientes y usuarios finales durante las que se discute que decisiones tomará el proyecto durante los próximos ciclos. Este tipo de eventos contribuyen sin duda a publicitar el uso de OpenStack y ayudan a la -cada año creciente- comunidad de usuarios y desarrolladores.

## Componentes

Como puede parecer lógico, OpenStack dista mucho de ser un desarrollo monolítico debido a su gran tamaño. El proyecto está dividido en distintos módulos agrupados por funcionalidad y con cada nuevo lanzamiento, nuevos componentes aparecen en escena aumentando así las opciones que nos brinda el sistema. Este diseño modular permite un rápido crecimiento y evolución de cada uno de estos componentes individuales y por consiguiente de OpenStack como elemento agregador de estos.



*Imagen 23: Distintos servicios que componen OpenStack*

De todos los módulos disponibles, algunos de ellos pertenecen al denominado núcleo o “core” de Openstack y otros que podrían ser considerados como más accesorios. La configuración elegida en cada caso dependerá de las características particulares del proyecto y el tipo de despliegue realizado.

A continuación describiremos algunos de los componentes más importantes que forman parte del proyecto.

## Nova (Computación)

Nova es el nombre en clave del módulo destinado a aglutinar todas las tareas relacionadas con computación en OpenStack. Por ejemplo, a través de Nova es posible administrar las instancias o servidores, configuraciones de red, así como los tamaños o sabores de éstas. La interacción con Nova se produce a través de una interfaz API REST y es compatible con con el resto de API de los otros proyectos de OpenStack y con EC2 API.

## Swift (Almacenamiento de Objetos)

Swift es uno de los componentes iniciales que formaba la base de OpenStack en aquella primera versión en 2012. Este módulo funciona como una capa intermedia encargada de acceder y recuperar los datos guardados en los volúmenes de almacenamiento.

A nivel de arquitectura, Swift está dividido en dos partes: la primera es una API con un proxy y la segunda hace referencia a toda la parte destinada al almacenamiento en la que se guardan los datos.

## Keystone (Identidad)

Keystone es el componente encargado de los procesos de Identidad en Openstack. En él se definen los usuarios, proyectos, roles y tokens. Es uno de los servicios clave dentro de OpenStack y el resto de servicios está directa o indirectamente relacionado con este llegando a poder funcionar como un registro para otros servicios.

Como en otros casos, Keystone utiliza una interfaz REST con la que es posible interaccionar fácilmente desde el exterior. Keystone se apoya sobre tres diferentes backends, uno destinado a almacenar para los tokens, otro destinado a almacenar usuarios y sus contraseñas y otro para los proyectos.

## Horizon (Interfaz Web)

Horizon es el componente destinado a ofrecer una interfaz de usuario Web para el manejo de OpenStack. Desde él, se proveen asistentes para algunas de las opciones consideradas más comunes, como la creación de instancia, control del ciclo de vida de un instancia, manejo de volúmenes etc.

Horizon es una aplicación Python WSGI (Web Server Gateway Interface) desarrollada utilizando el framework de Django. Carece de estado y toda la información mostrada es obtenida utilizando las API de los distintos servicios.

## Glance (Imágenes)

Glance es utilizado como el registro para imágenes que posteriormente servirán para iniciar las instancias. Además a parte del almacenado de imágenes, Glance guarda una serie de metadatos sobre que ayudan a su catalogado y ordenación y búsqueda,. A través de añadidos, Glance da soporte a multitud de backends en donde almacenar las imágenes. Algunos de estos son; swift, grusteds, ceph o incluso un sistema de ficheros tradicional.

## Cinder (Volúmenes)

El proyecto Cinder proporciona todo lo relacionado con la gestión de volumen e instantáneas de de las instancias. Gracias a los volúmenes podemos añadir almacenamiento externo a nuestras máquinas e incluso poder iniciarlas desde éste. En el caso de las instantáneas, Cincer creará una imagen de una instancia en un punto concreto en el tiempo

desde la cual será posible crear otras derivadas de estas. Como en el caso de Glance, Cinder ofrece numerosas opciones para ser utilizadas como backends: lvm, iSCSI, Gluster Ceph, etc.

## Neutron (Conectividad)

Neutron es el componente destinado a la conectividad y comunicaciones dentro de Openstack. Basado es el antiguo servicio de rd integrado en Nova nova-network, ofrece la posibilidad de muy interesantes y demandadas como configuración de rangos de IPs así como y sub-redes para las instancias, manejo de routers y switches virtuales, balanceo de carga y VPN como servicio entre otra otras. A pesar de esto el estado de este componente es aún a día de hoy un poco verde y para según que escenarios su uso no está desaconsejado en favor de nova-network.

## Capítulo 8. Integración y pruebas

### Introducción

Conectar el Servicio de Identidad de OpenStack, para que sea capaz de autenticarse contra un Directorio Activo no es una tarea extremadamente compleja en sí misma. De hecho, es prácticamente seguro que a día de hoy ya existan empresas u organizaciones que tengan una solución funcionando en producción parecida a la aquí propuesta. El problema principal reside en que no todas las soluciones documentadas para abordar este problema, funcionan y/o hacen referencia a un escenario similar al aquí presentado.

Además, en la mayoría de los casos optar por una de estas configuraciones implica necesariamente poseer unos amplios conocimientos del funcionamiento interno del Directorio Activo, así como del propio código de OpenStack para poder entender realmente qué puede estar pasado en el más que probable caso de encontrar problemas.

Como ya se explicó en el apartado de objetivos del proyecto, se pretende enlazar un directorio Activo de Microsoft con el Servicio de Identidad de OpenStack. De esta forma usuarios y así como sus respectivas credenciales, serán gestionadas externamente desde en el Directorio Activo.

Sobre el papel esta funcionalidad vino soportada por primera vez en OpenStack Havana (versión 8), lanzada en la segunda mitad del año 2013. Me gustaría recalcar el hecho de “sobre el papel” porque a pesar de que oficialmente el código nos proporciona esta característica, para ponerlo en funcionamiento es necesario ser consciente de muchas de sus limitaciones y estar dispuesto a enfrentarse a funcionalidad incompleta y errores no documentados. Afortunadamente para este desarrollo, muchos de estos problemas se solucionaron en la siguiente revisión, IceHouse, la cual se está utilizando como base de este proyecto.

A pesar de esto, el código de IceHouse sigue conteniendo algunos bugs que de no estar familiarizado con ellos pueden resultar bastante frustrantes y de difícil entendimiento.

En mi caso concreto algunos de estos errores anteriormente comentados me dieron muchos dolores de cabeza dado que para una gran parte de ellos, arreglarlos únicamente servía para que aparecieran otros derivados o incluso en muchos casos, no derivados.

La parte positiva es que la tratarse de un proyecto abierto con una comunidad muy extensa siempre me resultó sencillo reportar los errores, los funcionamientos no esperados o

simplemente no implementados. En algunos casos mis pequeños parches al código, comentarios y pruebas realizadas también resultaron útiles a la comunidad.

## Directorio Activo de Microsoft y OpenStack

Para llevar a cabo la integración, podemos optar por varias formas de abordarlo dependiendo de como esté realizado el diseño de nuestra infraestructura. De inicio, no resulta difícil identificar al menos dos enfoques distintos.

La primera opción consiste en almacenar los roles y las asignaciones en una base de datos SQL interna en contraposición a los usuarios y sus credenciales que se guardarán de el Directorio Activo. Tanto la base de datos MySQL como el Directorio Activo serán accedidos directamente por Keystone. De esta forma, los datos relacionados con OpenStack se almacenarán en OpenStack y aquellos relacionado con los usuarios se quedarán en el Directorio Activo. La mayoría de las empresas están utilizando este enfoque puesto que es sencillo de manejar y en su configuración mínima sólo requiere de acceso de lectura sobre el Directorio Activo.

Además en un entorno empresarial de producción, normalmente el grupo humano destinado a administrar aquello relacionado con OpenStack y máquinas virtuales suele ser distinto al que se encarga de las tareas relacionadas con el Directorio Activo, por lo que mantenerlo separado es una buena opción.

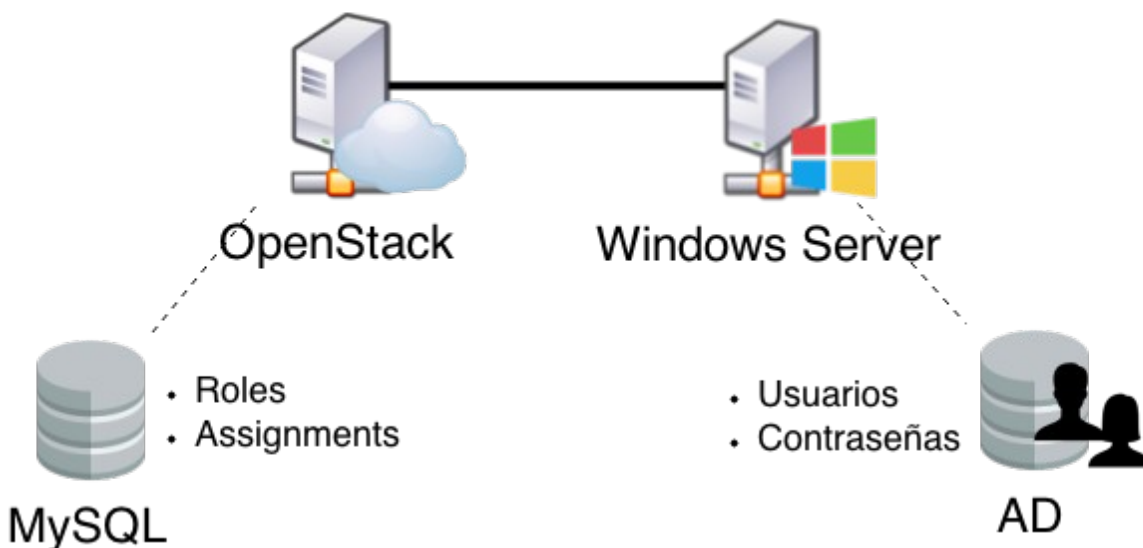
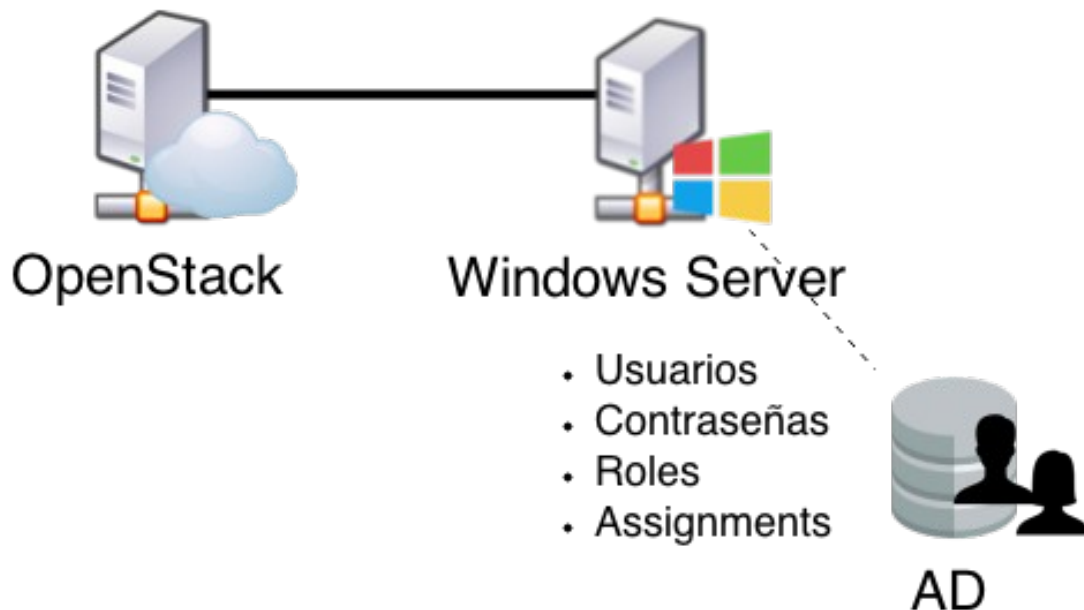


Imagen 24: Esquema Escenario 1

El segundo enfoque, al contrario que el anterior, guarda en el Directorio Activo todos los actores necesarios para hacer la autenticación posible. Es decir, roles, asinaciones y usuarios vivirán en el Directorio Activo, mientras que Keystone, el servicio de identidad de OpenStack, será el encargado de leer y consultar esa información para llevar a cabo el proceso de autenticación.



*Imagen 25: Esquema Escenario 2*

Al tratarse de una prueba de concepto, en este proyecto se ha optado por poner en marcha los dos enfoques anteriormente explicados. Aunque como es evidentemente, sólo uno de ellos podrá estar activo al mismo tiempo.

## Autenticación en OpenStack

En el proceso de autenticación participan la terna, usuario, rol y asignación a proyectos. Es decir: un usuario, pertenecerá a un proyecto en el que tendrá un rol asignado para realizar distintas tareas. Para que OpenStack funcione correctamente, dicha terna deberá estar bien formada y configurada.

El encargado de verificar dicha configuración es el componente Keystone, el servicio de Identidad de OpenStack. Toda ésta configuración se almacena en un único archivo llamado 'keystone.conf'. Cualquier cambio en la configuración pasará necesariamente por la modificación de ese archivo.



## Creación de usuarios y tenants (proyectos)

En OpenStack cada usuario puede estar asignado a uno o varios proyectos en los que podrá trabajar con las distintas instancias de las máquinas virtuales. Estos proyectos son conocidos dentro del entorno OpenStack como 'tenants'.

Cada usuario o grupos de usuarios podrán tener sus propios 'tenants' personales en donde desempeñarán su trabajo de forma privada y uno o varios compartidos en donde se pondría en conocimiento su trabajo con otros usuarios acreditados.

## Modificaciones en Windows: Crear la estructura necesaria en AD

Antes de empezar a modificar los archivos de configuración en OpenStack, será necesario crear la estructura de usuarios, unidades organizativas y demás elementos necesarios en el Directorio Activo. Para comenzar, será necesario tener configurado como mínimo lo siguiente:

1. Una cuenta que nos sirva de enlace entre el Directorio Activo y OpenStack.
2. Seleccionar una unidad organizativa en la que guardaremos los usuarios
3. Creación de usuarios clave OpenStack: nova, cinder y glance.

Además en el caso de querer almacenar todos los actores implicados en el proceso de autenticación en el Directorio Activo, será necesario generar la estructura necesaria para guardar los roles y las asignaciones de proyectos.

Dado que el escenario número uno es contiene en sí mismo al escenario dos, inicialmente se comenzará explicando el proceso de configuración de éste y más adelante se ampliará su funcionalidad con el objetivo de englobar al segundo.

### Escenario 1

En primer lugar crearemos la cuenta de enlace que nos servirá para conectar OpenStack con el Directorio Activo. Para ello abrimos la aplicación Active Directory Users and Computers desde el servidor Windows y dentro de contenedor 'Users' creamos un nuevo usuario llamado 'ldapbinding' con una contraseña al gusto lo suficientemente segura.

Ante de finalizar con la creación del usuario, seleccionaremos la opción que evita el cambio de contraseña por parte del usuario así como la opción que evita que la contraseña caduque.

Una vez hecho lo anterior, crearemos una nueva Unidad Organizativa que tendrá como objetivo albergar los usuarios del OpenStack. Para ello justo debajo de la raíz del dominio creamos la OU con nombre 'openstack' y a su vez, otra dentro de ella, otra llamada 'users'.

Dentro de ésta última crearemos algunos usuarios de ejemplo, en este caso en la imagen de ejemplo se pueden apreciar los dos usuarios creados **alumno01** y **alumno02**.

Una vez hecho esto sólo quedará confirmar correctamente el módulo de identidad de OpenStack para que apunte a nuestro Directorio Activo como se verá en el próximo apartado.

## Escenario 2

Como pre-requisito, será necesario haber realizado los pasos que se detallan en la descripción de Escenario 1 y además se añadirán dos nuevas Unidades Organizativas dentro de openstack: 'roles' y 'projects'

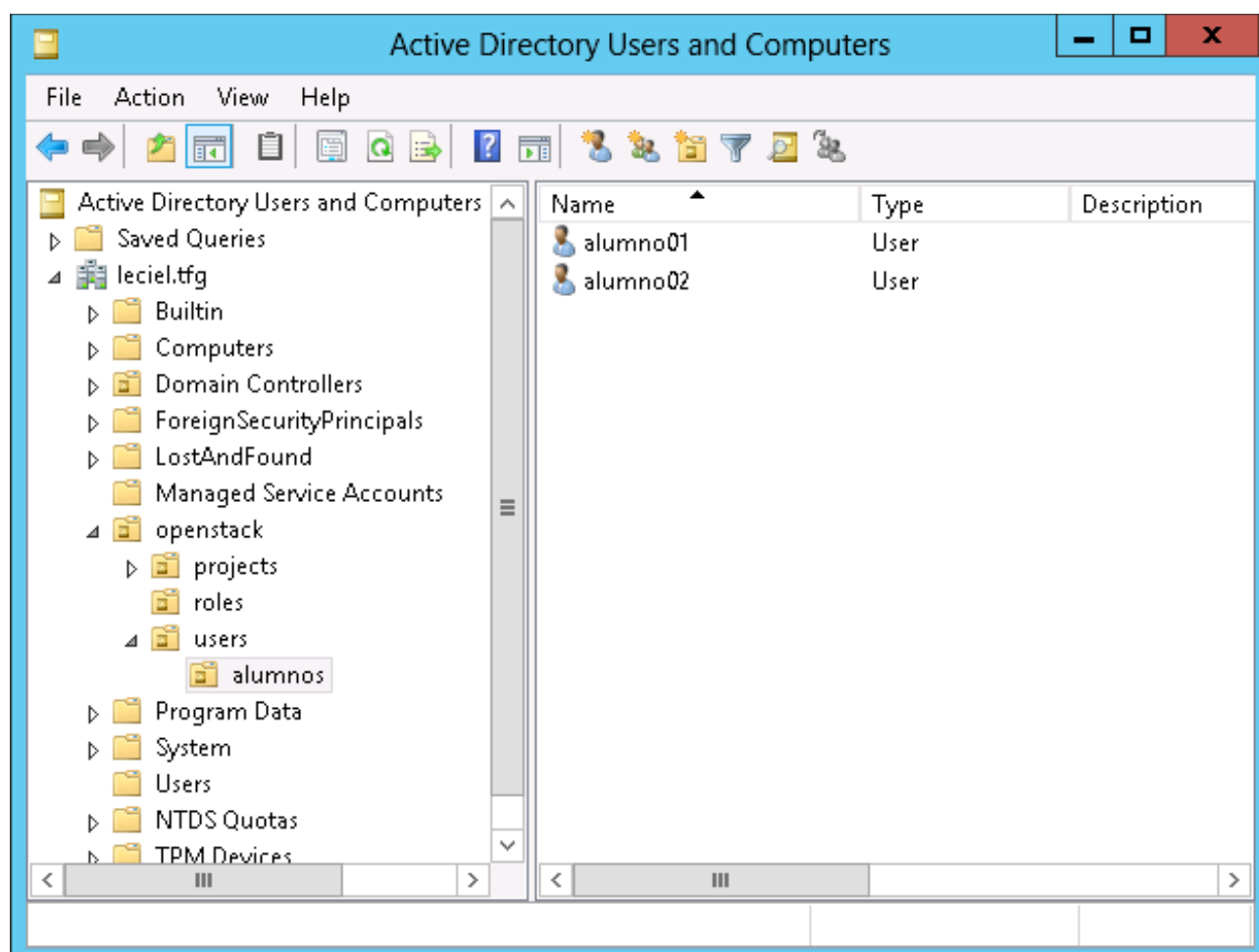


Imagen 26: Estructura de Directorio Activo

Una vez hecho, nuestro árbol debería quedar más o menos similar al aquí descrito:

leciel.tfg

```
openstack
  projects
  roles
  users
    alumnos
      alumno01
      alumno02
      ....
```

Recordemos que en este caso además de los usuarios, almacenaremos los roles y las asignaciones de proyectos en el Directorio Activo. El proceso en sí mismo no debería resultar complicado pero sí habrá que realizar cada uno de los pasos aquí detallados de forma correcta.

El problema principal reside en que a pesar de que Keystone utiliza una serie de objetos de clases y atributos que podríamos considerar como estándar dentro del mundo Linux/LDAP, éstos no son ampliamente utilizados en Directorio Activo de Microsoft por lo que, si queremos hacer uso de ellos, deberemos añadir previamente algunas de estas clases de forma manual.

Para realizarlo necesitaremos una herramienta especial llamada ADSI Edit y que deberemos tener instalada en el servidor. Para este caso esto no debería ser motivo de preocupación dado que viene incluida junto con el resto de software necesario para la configuración de Directorio Activo y que ya ha sido instalado.

Para ejecutarlo, podemos teclear ADSIEDIT.msc desde la ventana de ejecución de órdenes o simplemente utilizar el buscador de aplicaciones del sistema, presionando la tecla de Windows y tecleando como clave de búsqueda la cadena “ADSI”.

Una vez abierto añadiremos los objetos de tipo 'organizationalRole' dentro de la Unidad Organizativa 'roles'. Es primero de ellos será el que dedicaremos para el rol de administrador mientras que el segundo identificará a los usuarios comunes de la aplicación.

El proceso de creación es sencillo y bastarán una serie de clics de ratón para dejarlo todo configurado correctamente. En primer lugar, realizaremos botón derecho sobre la OU roles y seleccionaremos, Nuevo – Objeto.

En el diálogo de creación seleccionaremos 'organizationalRole' para posteriormente

asignarle el nombre 'admin'. En el último paso del asistente, haremos clic en la opción 'Más atributos' y añadiremos a nuestro nuevo objeto la propiedad de 'roleOccupant'. En el campo 'distinguishedName' añadiremos la ruta completa al contenedor donde se encuentra el usuario que queremos que sea administrador de la aplicación para este caso:

```
cn=admin,cn=users cn=openstack,dc=leciel,dc=tf
```

Pasaremos a continuación a realizar los mismos pasos para el rol de usuario normal o no administrador. Igualmente, como en el caso anterior seleccionaremos 'organizationalRole' aunque en esta ocasión elegiremos el nombre 'member'.

Antes de finalizar añadiremos la propiedad 'roleOccupant' en el apartado de 'Más Atributos' y a diferencia de lo anterior añadiremos la ruta hacia los alumnos.

```
cn=alumnos,cn=users cn=openstack,dc=leciel,dc=tf
```

Una vez creados los roles de usuario, continuaremos con la configuración de proyectos. Para ello crearemos dos Unidades Organizativas nuevas dentro de las ya existente 'projects'. Estas dos nuevas unidades tendrán el nombre de 'admin' y 'service'.

A su vez dentro de 'admin' crearemos dos objetos, el primero similar a los ya creados anteriormente y de tipo 'organizationalRole', llamado 'admin', y el segundo de tipo 'groupOfNames' con nombre 'adminUsers'. Como en el caso anterior, para el caso de 'admin' la propiedad de 'roleOccupant' será la ruta hasta nuestro usuario 'admin'. Para el caso de 'adminUser' idéntico resultado.

Para finalizar sólo restará habilitar las Unidades Organizativas recién creadas que representan a los proyectos admin y service, modificando su propiedad 'extensionName' con el valor 'TRUE'.

## Modificaciones en Openstack: 'keystone.conf'

Como en el caso anterior dividiremos las modificaciones según los dos distintos escenarios que queremos llevar a cabo.

### Escenario 1

Editaremos el archivo 'keystone.conf' que se encuentra en el directorio '/etc/keystone'. Este fichero es muy extenso por lo que para una edición lo más óptima posible será necesario utilizar el buscador de cadenas de nuestro editor. En primer lugar localizaremos la cadena "[identity]". Esto nos llevará directamente al apartado dentro del fichero de configuración donde se especifica sobre que actor recaerá la tarea de identidad. Como buscamos que de ésta se encargue el Directorio Activo, editaremos la variable 'driver' por lo siguiente.

```
driver = keystone.identity.backends.ldap.Identity
```

A continuación volveremos a hacer uso del buscador para en este caso encontrar la cadena "[ldap]" y añadiremos el bloque de código mostrado a continuación.

```
[ldap]

#
# Options defined in keystone
#
query_scope = sub
# URL for connecting to the LDAP server. (string value)
url = ldap://156.35.95.21
# User BindDN to query the LDAP server. (string value)
user = cn=ldapbinding,cn=Users,dc=leciel,dc=tfg
password = PasswordS3cret0:)
suffix = dc=leciel,dc=tfg
use_dumb_member = True
allow_subtree_delete = True
dumb_member = cn=ldapbinding,cn=Users,dc=leciel,dc=tfg

user_tree_dn = cn=Users,dc=leciel,dc=tfg
user_objectclass = organizationalPerson
#user_filter = (memberof=cn=openstack,dc=leciel,dc=tfg)
user_id_attribute = cn
user_name_attribute = sAMAccountName
user_mail_attribute = mail
#user_pass_attribute =
user_enabled_attribute = userAccountControl
user_enabled_mask = 2
user_enabled_default = 512
user_attribute_ignore = password,tenant_id,tenants,lower
user_allow_create = False
user_allow_update = False
user_allow_delete = False

tenant_tree_dn = ou=admin,ou=projects,ou=openstack,dc=leciel,dc=tfg
#tenant_filter =
tenant_objectclass = organizationalUnit
tenant_id_attribute = ou
tenant_member_attribute = member
tenant_name_attribute = ou
tenant_desc_attribute = description
tenant_enabled_attribute = extensionName
tenant_attribute_ignore = description,businessCategory,extensionName
tenant_allow_create = True
tenant_allow_update = True
tenant_allow_delete = True

role_tree_dn = ou=roles,ou=openstack,dc=leciel,dc=tfg
```

```
#role_filter =  
role_objectclass = organizationalRole  
role_id_attribute = cn  
role_name_attribute = cn  
role_member_attribute = roleOccupant  
#role_attribute_ignore =  
role_allow_create = True  
role_allow_update = True  
role_allow_delete = True
```

## Escenario 2

Para este caso sera necesario realizar las mismas modificaciones detalladas en el apartado anterior más las aquí explicadas.

Para comenzar buscamos la cadena "[assignment]" y remplazaremos el valor de la variable driver por lo siguiente:

```
[assignment]  
driver = keystone.assignment.backends.ldap.Assignment
```

Independientemente del caso en el que nos encontremos, tras realizar cambios en el archivo de configuración de keystone, sera imprescindible reiniciar el servido para que las modificaciones sean aplicadas.

```
[root@lenuage ~]# service openstack-keystone restart  
Stopping keystone: [ OK ]  
Starting keystone: [ OK ]  
[root@lenuage ~]#
```

## Añadir usuarios de servicio de OpenStack al Directorio Activo

Una vez realizado el cambio anterior ya será posible autenticarnos contra el Directorio Activo de Microsoft pero desgraciadamente aún no será posible utilizar OpenStack de forma normal. Para comenzar a utilizarlo será necesario añadir a nuestro soporte de Directorio Activo aquellos usuarios claves que hacen uso interno de OpenStack. El nombre de estos usuarios se corresponden con el nombre de los componentes que representan. Dependiendo de nuestra instalación de OpenStack deberemos añadir unos u otros usuarios. Para esta configuración concreta añadiremos los siguientes usuarios a nuestro Directorio Activo: nova, cinder y glance

En este sentido es muy importante que la contraseña de dichos usuarios concuerde con la especificada en los archivos de configuración de OpenStack, en caso contrario el sistema nos devolverá un error 403 para aquellos comandos relacionados con cada usuario.

```
[root@lenuage ~(keystone_myadmin)]# nova image-list
```

```
ERROR: This server could not verify that you are authorized to access
the document you requested. Either you supplied the wrong credentials
(e.g., bad password), or your browser does not understand how to supply
the credentials required. (HTTP 401) (Request-ID: req-fdd1fd08-501a-
4ef6-9b9c-4ffd8ccf4b54)
```

Es decir, debemos revisar el contenido de los siguientes archivos y verificar el valor de la propiedad "admin\_password".

- /etc/nova/nova.conf
- /etc/cinder/cinder.conf
- /etc/glance/glance-api.conf

Ese valor será el la contraseña que debemos configurar para cada usuario correspondiente en nuestro Directorio Activo.

Podemos utilizar la orden 'grep' para buscar fácilmente todas las contraseñas contenidas en los archivos de configuración de estos servicios mediante el siguiente comando

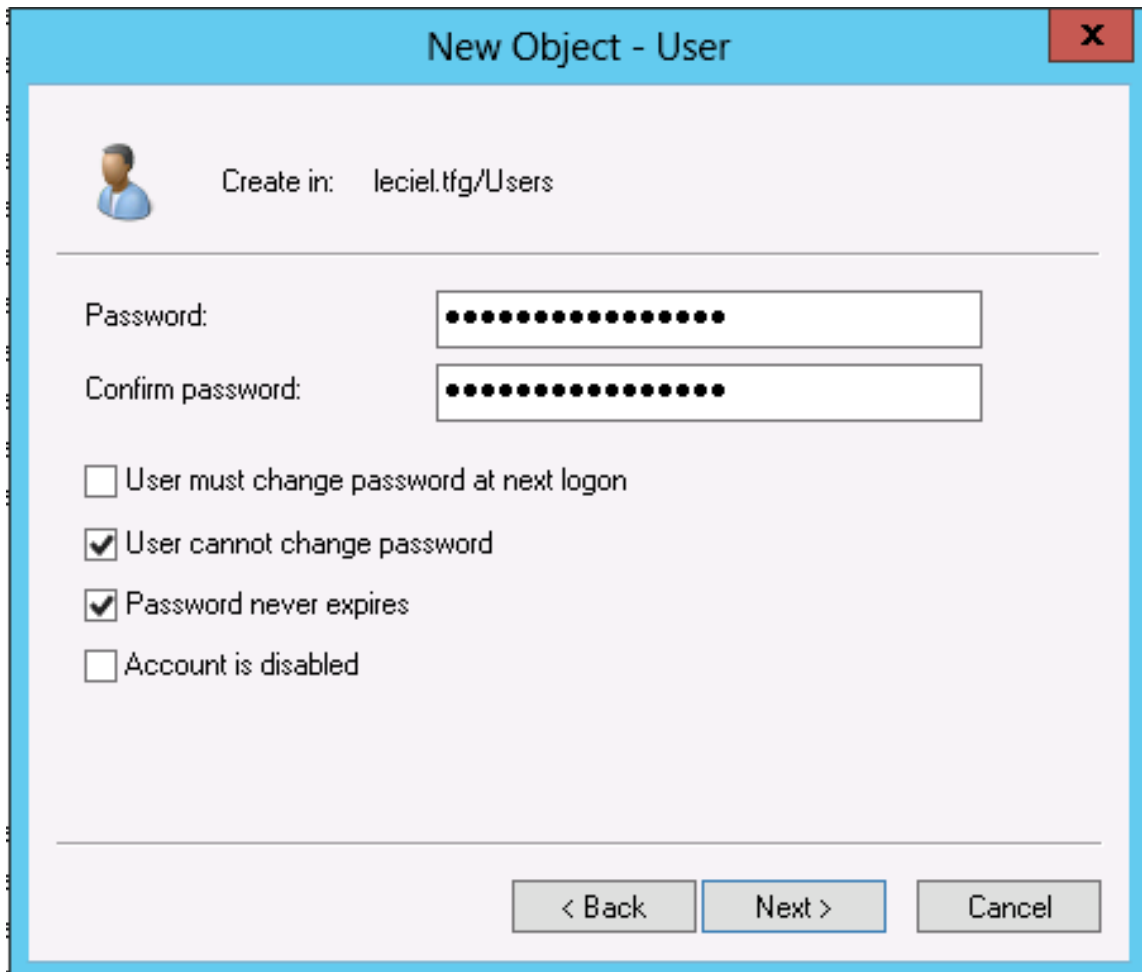
```
grep -R "cadena_a_buscar" <directorio>
```

En nuestro caso concreto el objetivo es localizar la cadena "admin\_password" dentro del directorio '/etc' que es donde se encuentran todos los archivos de configuración de OpenStack:

```
[root@lenuage ~(keystone_myadmin)]# grep -R "admin_password" /etc/
/etc/nova/nova.conf:#neutron_admin_password=<None>
/etc/nova/nova.conf:#admin_password=%SERVICE_PASSWORD%
/etc/nova/nova.conf:admin_password=d6487810e1874f27
/etc/swift/proxy-server.conf:admin_password = c20a92650ce14f99
/etc/cinder/api-paste.ini:admin_password=d9899d3442004fa3
/etc/cinder/cinder.conf.rpmnew:# `admin_user` and `admin_password`
instead. (string value)
/etc/cinder/cinder.conf.rpmnew:#admin_password=<None>
/etc/cinder/cinder.conf:#admin_password=<None>
/etc/glance/glance-api.conf:#admin_password=%SERVICE_PASSWORD%
/etc/glance/glance-api.conf:#admin_password=%SERVICE_PASSWORD%
/etc/glance/glance-api.conf:admin_password=8cfa0b752b884b0d
/etc/glance/glance-scrubber.conf:# admin_password = %SERVICE_PASSWORD%
/etc/glance/glance-cache.conf:# admin_password = %SERVICE_PASSWORD%
/etc/glance/glance-cache.conf:admin_password = 8cfa0b752b884b0d
/etc/glance/glance-registry.conf:#admin_password=%SERVICE_PASSWORD%
/etc/glance/glance-registry.conf:admin_password=8cfa0b752b884b0d
```

Revisando la salida del comando anterior nos será muy fácil identificar las contraseñas de las cuentas.

Una vez que tenemos las contraseñas de las cuentas de servicio de OpenStack, será necesario crear dichas cuentas en nuestro Directorio Activo. Para ello, dentro de la unidad organizativa en la que guardamos los usuarios, añadimos los usuarios de OpenStack:



*Imagen 27: Diálogo de creación de usuario*

Como vimos anteriormente es importante marcar la opción que evita el cambio de contraseña a posteriori por parte del usuario así como la opción que evita que la contraseña caduque. De esta forma nos evitaremos posibles dolores de cabeza relacionados con este tema en el futuro.

## Añadir roles

Una vez completado lo anterior ya tendremos configurados los usuarios mínimos que OpenStack necesita para su correcto funcionamiento en nuestro Directorio Activo. A pesar de esto, si comenzamos a utilizar el usuario 'admin' recién creado para por ejemplo mostrar la lista de proyectos obtendremos una desagradable sorpresa:



```
[root@lenuage ~(keystone_admin)]# source keystone_admin
[root@lenuage ~(keystone_admin)]# keystone tenant-list
Invalid user / password (HTTP 401)
[root@lenuage ~(keystone_admin)]#
```

Esto es debido a que aunque el usuario 'admin' sea un usuario reconocido por el servicio de identidad de OpenStack, éste no tiene ningún tipo de permisos sobre los proyectos existentes.

Para solucionarlo debemos añadir al usuario 'admin' al proyecto 'admin' con el rol 'admin'. De esta forma configuraremos nuestra terna y realmente daremos permisos de administrador a nuestro usuario 'admin'.

```
[root@lenuage ~]# keystone user-list
+-----+-----+-----+-----+
| id      | name  | enabled | email  |
+-----+-----+-----+-----+
| admin   | admin | True    |        |
| cinder  | cinder | True    |        |
| glance  | glance | True    |        |
| nova    | nova  | True    |        |
+-----+-----+-----+-----+
[root@lenuage ~]# keystone role-list
+-----+-----+
| id                  | name  |
+-----+-----+
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
| fb354cf6d767441d8f53589464d90b54 | admin   |
+-----+-----+
[root@lenuage ~]# keystone tenant-list
+-----+-----+-----+
| id                  | name  | enabled |
+-----+-----+-----+
| 4e6c1002c09d41c595a55fc2a040681f | admin  | True    |
| 54244f8ede8944e4ba0fa8aa7cdd5798 | services | True    |
+-----+-----+-----+
[root@lenuage ~]# keystone user-role-add --user admin --role
fb354cf6d767441d8f53589464d90b54 --tenant
4e6c1002c09d41c595a55fc2a040681f
[root@lenuage ~]#
```

Una vez realizados los pasos anteriores, configurado el usuario admin correctamente, ya tendremos acceso. Para probarlo ejecutamos lo siguiente:

```
[root@lenuage ~(keystone_admin)]# keystone tenant-list
+-----+-----+-----+
| id                  | name  | enabled |
+-----+-----+-----+
| 4e6c1002c09d41c595a55fc2a040681f | admin  | True    |
+-----+-----+-----+
```

```
| 54244f8ede8944e4ba0fa8aa7cdd5798 | services | True |  
+-----+-----+-----+-----+  
[root@lenuage ~(keystone_admin)]#
```

Como en nuestro caso particular queremos que el usuario 'admin' sea un usuario administrador clásico, es decir que pueda acceder a todos los servicios de este y manejarlos y configurarlos a su antojo, tendrá el role admin para todos los proyectos.

Finalmente será necesario realizar lo anteriormente explicado para el resto de usuarios de uso interno de OpenStack: nova, cinder y glance.

## Nova

```
[root@lenuage ~(keystone_admin)]# nova list  
ERROR: Unauthorized (HTTP 401)
```

Este ejemplo es idéntico al anterior con la salvedad de que deberos añadir el usuario nova en el proyecto services y no al de 'admin'.

```
[root@lenuage ~(keystone_admin)]# keystone user-role-add --user nova  
--role fb354cf6d767441d8f53589464d90b54 --tenant  
54244f8ede8944e4ba0fa8aa7cdd5798  
  
[root@lenuage ~(keystone_admin)]# nova list  
+-----+-----+-----+-----+-----+-----+  
| ID | Name | Status | Task State | Power State | Networks |  
+-----+-----+-----+-----+-----+-----+  
+-----+-----+-----+-----+-----+-----+  
  
[root@lenuage ~(keystone_admin)]#
```

## Glance

Igual que en el caso de nova pero aplicado sobre el usuario de 'glance'.

```
[root@lenuage ~(keystone_admin)]# glance image-list  
Request returned failure status.  
Invalid OpenStack Identity credentials.  
  
[root@lenuage ~(keystone_admin)]# keystone user-role-add --user glance  
--role fb354cf6d767441d8f53589464d90b54 --tenant  
54244f8ede8944e4ba0fa8aa7cdd5798  
  
[root@lenuage ~(keystone_admin)]# glance image-list  
+-----+-----+-----+-----+-----+-----+  
| ID | Name | Disk Format | Container Format | Size | Status |  
+-----+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+

[root@lenuage ~(keystone_admin)]
```

## Cinder

Por último realizaremos los mismos pasos para el usuario de 'cinder'.

```
[root@lenuage ~(keystone_admin)]# cinder list
ERROR: Unauthorized (HTTP 401)

[root@lenuage ~(keystone_admin)]# keystone user-role-add --user cinder
--role fb354cf6d767441d8f53589464d90b54 --tenant
54244f8ede8944e4ba0fa8aa7cdd5798

[root@lenuage ~(keystone_admin)]# cinder list
+-----+-----+-----+-----+-----+-----+
| ID | Status | Display | Size | Volume Type | Bootable | Attached to |
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+

[root@lenuage ~(keystone_admin)]#
```

## Creación la primera máquina en OpenStack

A continuación se explicará como crear la primera máquina virtual en nuestro recién instalado entorno de virtualización OpenStack. Para llevar a cabo esta tarea utilizaremos, tres de los componentes instalados anteriormente, 'nova', 'glance' y 'keytone'. Las dos primeras serán usadas a través de sus command line interface (CLI) mientras que 'keystone' será usado indirectamente para temas relacionados con identidad y autenticación.

## Descarga de la imagen

En primer lugar deberemos descargar el archivo de la imagen que queremos utilizar en OpenStack para que una vez descargado, sea posible integrarlo en el sistema y de esta forma posibilitar el iniciar instancias a partir de ella.

Para este ejemplo se utilizará una imagen de un sistema GNU/Linux llamada CirrOS. CirrOS es una pequeña distribución que se puede encontrar gratuitamente en la red, preparada para ser desplegada y que destaca por su ligereza llevada al extremo. CirrOS es una distribución Linux muy simplificada y contenida en una imagen de menos de 10 Mb de peso.

Este sistema nos proporciona una instalación mínima de un sistema GNU/Linux con el

objetivo de que podamos probar nuestra configuración de la forma más rápida posible. Además su instalación por defecto viene con el puerto 22 abierto y con un usuario no administrador ya creado, para que conectarse remotamente por ssh no suponga ningún problema.

Para su descarga, apuntaremos hacia sus repositorios oficiales en cirros-cloud.net y descargaremos la imagen en nuestro sistema utilizando directamente el comando 'wget':

```
[root@lenuage images(keystone_admin)]# wget --no-check-certificate
https://download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img
--2014-12-11 18:26:02-- https://download.cirros-
cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img
Resolving download.cirros-cloud.net... 69.163.241.114
Connecting to download.cirros-cloud.net|69.163.241.114|:443...
connected.
WARNING: cannot verify download.cirros-cloud.net's certificate, issued
by `/CN=download.cirros-cloud.net':
Self-signed certificate encountered.
HTTP request sent, awaiting response... 302 Found
Location: http://cdn.download.cirros-cloud.net/0.3.3/cirros-0.3.3-
x86_64-disk.img [following]
--2014-12-11 18:26:04-- http://cdn.download.cirros-
cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img
Resolving cdn.download.cirros-cloud.net... 2.22.63.66, 2.22.63.56,
2a02:26f0:64::170e:5d39, ...
Connecting to cdn.download.cirros-cloud.net|2.22.63.66|:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 13200896 (13M) [application/octet-stream]
Saving to: `cirros-0.3.3-x86_64-disk.img'

100%[=====>] 13,200,896 9.62M/s
in 1.3s

2014-12-11 18:26:05 (9.62 MB/s) - `cirros-0.3.3-x86_64-disk.img' saved
[13200896/13200896]

[root@lenuage images(keystone_admin)]#
```

Una vez completada, pasaremos a importar dicha imagen haciendo uso de 'glance', el componente de gestión imágenes integrado en OpenStack. Para ello usaremos el comando 'glance image-create' proporcionado por el propio cliente de OpenStack con las siguientes opciones.

```
[root@lenuage images(keystone_admin)]# glance image-create
--name="CirroS 0.3.3" --disk-format=qcow2 --container-format=bare --is-
public=true --file cirros-0.3.3-x86_64-disk.img
+-----+-----+
| Property          | Value                                     |
+-----+-----+
```

```
+-----+-----+
| checksum      | 133eae9fb1c98f45894a4e60d8736619 |
| container_format | bare                               |
| created_at    | 2014-12-11T17:40:32               |
| deleted       | False                             |
| deleted_at    | None                              |
| disk_format   | qcow2                             |
| id            | 71fb6cbf-1f0e-4567-bf63-962e02145fbe |
| is_public     | True                              |
| min_disk      | 0                                  |
| min_ram       | 0                                  |
| name          | CirrOS 0.3.3                      |
| owner         | 4e6c1002c09d41c595a55fc2a040681f |
| protected     | False                             |
| size          | 13200896                          |
| status        | active                            |
| updated_at    | 2014-12-11T17:40:32               |
| virtual_size  | None                              |
+-----+-----+
[root@lenuage images(keystone_admin)]#
```

Para verificar que la imagen ha sido correctamente añadida al repositorio es glance proporciona listar todas aquellas que se encuentran disponibles gracias al subcomando image-list:

```
[root@lenuage images(keystone_admin)]# glance image-list --human-readable
+-----+-----+-----+-----+-----+-----+
| ID          | Name          | Disk Format | Format | Size   | Status |
+-----+-----+-----+-----+-----+-----+
| e02145fbe   | CirrOS 0.3.3 | qcow2       | bare   | 12.6MB | active |
| 45d522b25   | Ubuntu       | qcow2       | bar    | 244.3MB | active |
+-----+-----+-----+-----+-----+-----+
[root@lenuage images(keystone_admin)]#
```

(ids de imágenes editados para mantener el formato de la documentación)

Relacionado con esto último, haciendo uso de 'nova image-list' en nova, conseguiremos listar aquellas imágenes disponibles con la que a partir de ellas poder crear instancias. Además la columna 'status' nos proporcionará información de la disponibilidad de cada imagen. Para la creación de cada nueva instancia utilizaremos bien el ID de la imagen o su nombre completo.

```
[root@lenuage images(keystone_admin)]# nova image-list
+-----+-----+-----+-----+
| ID          | Name          | Status | Server |
+-----+-----+-----+-----+
| 71fb6cbf    | CirrOS 0.3.3 | ACTIVE |         |
| 8dcc802f    | Ubuntu       | ACTIVE |         |
+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
[root@lenuage images(keystone_admin)]#
```

(ids de imágenes editados para mantener el formato de la documentación)

Antes de proceder con la creación de la imagen propiamente dicha listaremos los tamaños o 'flavors' que tenemos disponibles para su creación.

```
[root@lenuage ~(keystone_admin)]# nova flavor-list
+-----+-----+-----+-----+-----+-----+-----+
| ID | Name       | Memory_MB | Disk | Swap | VCPUs | Is_Public |
+-----+-----+-----+-----+-----+-----+-----+
| 1  | m1.tiny    | 512       | 1    |      | 1      | True      |
| 2  | m1.small   | 2048      | 20   |      | 1      | True      |
| 3  | m1.medium  | 4096      | 40   |      | 2      | True      |
| 4  | m1.large   | 8192      | 80   |      | 4      | True      |
| 5  | m1.xlarge  | 16384     | 160  |      | 8      | True      |
+-----+-----+-----+-----+-----+-----+-----+
[root@lenuage ~(keystone_admin)]#
```

En para este caso utilizaremos el más pequeño, denominado m1.tiny. Esta configuración proporciona un único núcleo virtual, 512Mb de RAM y un 1Gb de disco. A pesar de esto se trata de unos números más que suficientes para el sistema de la imagen.

Utilizaremos el comando 'nova boot' con sus diferentes opciones para crear la imagen:

nova boot --image ID\_IMAGEN --flavor NOMBRE\_FLAVOR NOMBRE\_INSTANCIA

```
[root@lenuage ~(keystone_admin)]# nova boot --image 71fb6cbf-1f0e-4567-
bf63-962e02145fbe --flavor m1.tiny cirros-test
+-----+-----+-----+-----+-----+-----+-----+
| Property                               | Value                               |
+-----+-----+-----+-----+-----+-----+-----+
| OS-DCF:diskConfig                      | MANUAL                             |
| OS-EXT-AZ:availability_zone            | nova                               |
| OS-EXT-SRV-ATTR:host                   | -                                  |
| OS-EXT-SRV-ATTR:hypervisor_hostname   | -                                  |
| OS-EXT-SRV-ATTR:instance_name          | instance-0000000c                 |
| OS-EXT-STS:power_state                 | 0                                  |
| OS-EXT-STS:task_state                  | scheduling                         |
| OS-EXT-STS:vm_state                   | building                          |
| OS-SRV-USG:launched_at                 | -                                  |
| OS-SRV-USG:terminated_at              | -                                  |
| accessIPv4                             |                                    |
| accessIPv6                             |                                    |
| adminPass                              | 8N9JLhRq9SzV                     |
| config_drive                          |                                    |
| created                                | 2015-05-17T17:21:31Z              |
| flavor                                 | m1.tiny (1)                       |
| hostId                                |                                    |
```

```
| id | a4f08405-e87b-4b7f-8345-45 |
| image | CirrOS 0.3.3 (71fb6cbf-1f0e1) |
| key_name | - |
| metadata | {} |
| name | cirros-test |
| os-extended-volumes:volumes_attached | [] |
| progress | 0 |
| security_groups | default |
| status | BUILD |
| tenant_id | 4e6c1002c09d41c595a52a040681f |
| updated | 2015-05-17T17:21:31Z |
| user_id | admin |
+-----+-----+
[root@lenuage ~(keystone_admin)]#
```

Una vez creada, podemos usar el comando 'nova list' para listarlas.

```
[root@lenuage ~(keystone_admin)]# nova list
+-----+-----+-----+-----+-----+
| ID | Name | Status | State | Networks |
+-----+-----+-----+-----+-----+
| ba0b0a6d | cirros | ACTIVE | Running | network=192.168.1.99 |
| a4f08405 | ubuntu12.04 | ACTIVE | Running | network=192.168.1.99 |
| 9ea50875 | cirros-test | ACTIVE | Running | network=192.168.1.99 |
+-----+-----+-----+-----+-----+

[root@lenuage ~(keystone_admin)]#
```

Openstack utiliza el estado denominado como 'ACTIVO' para indicar que la máquina ha sido conectada correctamente. A continuación realizaremos una sencilla prueba de conexión.

```
[root@lenuage ~(keystone_admin)]# ssh cirros@192.168.1.98
The authenticity of host '192.168.1.98 (192.168.1.98)' can't be
established.
RSA key fingerprint is 65:99:8c:1d:8b:41:a1:c7:4d:25:7e:2b:a2:5b:a5:69.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.98' (RSA) to the list of known
hosts.
cirros@192.168.1.98's password:
$ hostname
cirros
$ date
Sun May 17 11:41:40 MDT 2015
$
```

## Capítulo 9. Problemas encontrados

Esta documentación no podría estar completa sin hacer referencia a aquellos problemas, situaciones inesperadas o mal funcionamientos encontrados durante la etapa de desarrollo de este proyecto. La realización del mismo no ha estado en absoluto exenta de dificultades y durante muchas fases se han tenido que afrontar problemas con lo que inicialmente no se contaba y de los que apenas se tenía información o medios, sobre como solucionarlos.

Por ello, se ha considerado oportuno añadir algunos de estas situaciones a esta documentación para que, a parte de dejar constancia de ellos, el futuro lector pueda utilizar este apartado para su provecho y evitar así tropezar con algunas de las piedras que se han encontrado en el camino de este desarrollo.

### Falta de espacio en disco en el servidor OpenStack

Una vez finalizada exitosamente la instalación y posterior configuración el servidor de OpenStack, uno de los problemas encontrados fue la falta de espacio en disco tras la creación de las primeras instancias de máquinas virtuales. A pesar de haber asignado más de 500Gb de disco para tal fin, todo apuntaba a que OpenStack sólo estaba detectando menos de 20Gb.

Este problema, se manifestaba en la aplicación impidiendo la creación de nuevas instancias, con un genérico mensaje de error 500.

```
[root@lenuage ~(keystone_admin)]# nova boot --image 8dcc802f-b892-40b2-8589-08c45d522b25 --flavor m1.small myubuntu --availability_zone=nova
ERROR: The server has either erred or is incapable of performing the requested operation. (HTTP 500) (Request-ID: req-f5eeb722-61a6-46ad-862f-d867862fb80c)
```

Tras analizar qué podría estar causando el problema, conseguí darme cuenta de que por defecto OpenStack sólo trabaja en la partición raíz del sistema, por lo que si quería aprovechar todo el espacio disponible debía extender el tamaño de dicha partición.

### Extendiendo el tamaño de raíz '/'

```
[root@lenuage ~(keystone_admin)]# df -H
Filesystem              Size  Used Avail Use% Mounted on
```



```
/dev/mapper/VolGroup-lv_root 19G 6.3G 12G 36% /
tmpfs 34G 0 34G 0% /dev/shm
/dev/xvda1 508M 81M 402M 17% /boot
```

Como se puede apreciar, de los más de 500Gb destinados al disco sólo 19 pertenecen a la partición raíz, que es la que utiliza la configuración por defecto de OpenStack para distribuir las máquinas.

```
[root@lenuage ~(keystone_admin)]# fdisk -l
Disk /dev/xvda: 536.9 GB, 536870912000 bytes
255 heads, 63 sectors/track, 65270 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0007bc25

    Device Boot      Start         End      Blocks   Id  System
/dev/xvda1    *           1           64       512000    83  Linux
Partition 1 does not end on cylinder boundary.
/dev/xvda2           64        2611     20458496    8e  Linux LVM

Disk /dev/xvdd: 119 MB, 119197696 bytes
255 heads, 63 sectors/track, 14 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/mapper/VolGroup-lv_root: 18.8 GB, 18798870528 bytes
255 heads, 63 sectors/track, 2285 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/mapper/VolGroup-lv_swap: 2147 MB, 2147483648 bytes
255 heads, 63 sectors/track, 261 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

Antes de comenzar con el proceso de modificación de particiones, será necesario añadir el repositorio de EPEL (Extra Packages for Enterprise Linux). Desde este repositorio serán instalados los paquetes necesarios para poder realizar los cambios en el tamaño de las particiones de una forma fiable y segura.

```
[root@lenuage ~]# rpm -Uvh  
http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-  
8.noarch.rpm
```

Para realizarlo utilizaremos la herramienta 'growpart'. Como dicha herramienta no está instalada y dependiendo de cada distribución la podremos encontrar integrada en un paquete u otro, usaremos la la opción *whatprovides* del gestor de paquetes 'yum' para saber que paquete nos proporciona dicha funcionalidad.

```
yum whatprovides */nombredelpaquete
```

```
[root@lenuage ~(keystone_admin)]# yum whatprovides */growpart  
Loaded plugins: priorities, security  
272 packages excluded due to repository priority protections  
epel/filelists_db | 8.4 MB  
00:01  
openstack-havana/filelists_db | 1.5 MB  
00:02  
puppetlabs-deps/filelists_db | 182 kB  
00:00  
puppetlabs-products/filelists_db | 917 kB  
00:00  
sl-security/filelists_db | 1.4 MB  
00:08  
sl6x-security/filelists_db | 1.4 MB  
00:08  
cloud-utils-growpart-0.27-10.el6.x86_64 : Script for growing a  
partition  
Repo : epel  
Matched from:  
Filename : /usr/bin/growpart  
[root@lenuage ~(keystone_admin)]#
```

Tal y como se puede observar en la información anterior, 'cloud-utils-growpart' es el nombre paquete que contiene dicha funcionalidad para Scientific Linux.

Conocida esta información procedemos con la instalación del paquete.

```
[root@lenuage ~(keystone_admin)]# yum install cloud-utils-growpart -y
```

Una vez terminada la instalación, tecleamos el siguiente comando, especificando es el disco virtual que queremos modificar junto con el número de partición a modificar. En este caso '/dev/xvda' y la número dos respectivamente.

```
[root@lenuage ~(keystone_admin)]# growpart /dev/xvda 2  
CHANGED: partition=2 start=1026048 old: size=40916992 end=41943040 new:  
size=1047536502,end=1048562550
```

Después de ejecutar el anterior comando, es necesario reiniciar la máquina. Para ello utilizaremos el comando reboot.

```
[root@lenuage ~(keystone_admin)]# reboot
```

Una vez reiniciada, accederemos de nuevo y ejecutamos los siguientes comandos.

```
[root@lenuage ~(keystone_admin)]# pvresize /dev/xvda2
Physical volume "/dev/xvda2" changed
1 physical volume(s) resized / 0 physical volume(s) not resized
```

```
[root@lenuage ~(keystone_admin)]# lvextend -l +100%FREE
/dev/mapper/VolGroup-lv_root
Extending logical volume lv_root to 497.50 GiB
Logical volume lv_root successfully resized
```

Dependiendo del tamaño elegido la nueva partición, esta última opción podría tardar varios minutos en completarse. Una vez finalizada avisará al usuario, devolviéndole el control del prompt del sistema.

```
[root@lenuage ~(keystone_admin)]# resize2fs /dev/mapper/VolGroup-
lv_root
resize2fs 1.41.12 (17-May-2010)
Filesystem at /dev/mapper/VolGroup-lv_root is mounted on /; on-line
resizing required
old desc_blocks = 2, new_desc_blocks = 32
Performing an on-line resize of /dev/mapper/VolGroup-lv_root to
130416640 (4k) blocks.
The filesystem on /dev/mapper/VolGroup-lv_root is now 130416640 blocks
long.
[root@lenuage ~(keystone_admin)]#
```

Para comprobar si efectivamente ha funcionado podemos utilizar de nuevo la orden 'df'.

```
[root@lenuage ~]# df -H
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/VolGroup-lv_root	526G	6.4G	493G	2%	/
tmpfs	34G	0	34G	0%	/dev/shm
/dev/xvda1	508M	81M	402M	17%	/boot

Una vez finalizado, OpenStack identificará el nuevo tamaño de la partición y el problema de creación de nuevas máquinas debería quedar solucionado.

## Error al borrar OU desde Active Directory User and Computers

<http://technet.microsoft.com/en-us/library/cc736842%28v=ws.10%29.aspx>

### Problemas con xapi

Tras varias semanas con todo el despliegue de máquinas virtuales funcionando correctamente, se comenzaron a percibir algunos problemas de forma puntual que propiciaban cortes de conexión intermitentes con las máquinas virtuales.

Inicialmente, ésto resultó muy extraño dado que las máquinas afectadas llevaban algunas semanas trabajando ya de una forma más o menos estable y no se habían producido cambios en sus configuraciones que hubieran podido propiciar tales problemas.

Bien es cierto, que inicialmente no se le dio demasiada importancia a este asunto y se achacó el mal comportamiento a un problema relacionado con las comunicaciones. Con el paso de los días estos problemas se hicieron más y más frecuentes, hasta llegado a un punto en el que se convirtieron bastante molestos dejando en muchos casos las máquinas colgadas completamente.

Desde el más absoluto desconocimiento lo primero que se pensó fue en reiniciar las máquinas afectadas, para de esta forma intentar identificar si se podría deber a algún problema con algún servicio. De ser así, es posiblme un reinicio y posterior arranque en fresco devolviera al sistema a su estado habitual.

Desgraciadamente, estos reinicios, sólo hicieron más que agravar el problema derivándolo a un problema mucho mayor con algunas máquinas mostrando errores incluso al inicio. Como medida desesperada se optó por reiniciar la máquina física, pero tras el reinicio de ésta, todas las máquinas virtuales basadas en Linux dejaron de poder ser iniciadas.

El error mostrado desde la interfaz gráfica Xen Center no resultaba demasiado descriptivo en sí mismo y simplemente mostraba un error al inicio del proceso de carga del que le era imposible recuperarse. A partir de ahí y debido a la gravedad aparente del asunto, se decidió revisar más a fondo los registros de errores, accediendo al sistema hipervisor desde su propia consola ordenes.

Una vez conectados por SSH al servido físico xenserver-05, se utilizó el comando para abrir la consola de administración de xen, pero la única información que proporcionaba el servicio era que el servicio de xapi no estaba siendo ejecutado y ofrecía como solución el intentar reiniciar dicho servicio.

Después de esto, todo parecía indicar que al menos se había descubierto algo más sobre el origen del problema y para intentar solucionarlo se debería, al menos, devolver a un estado consistente el servicio xapi.

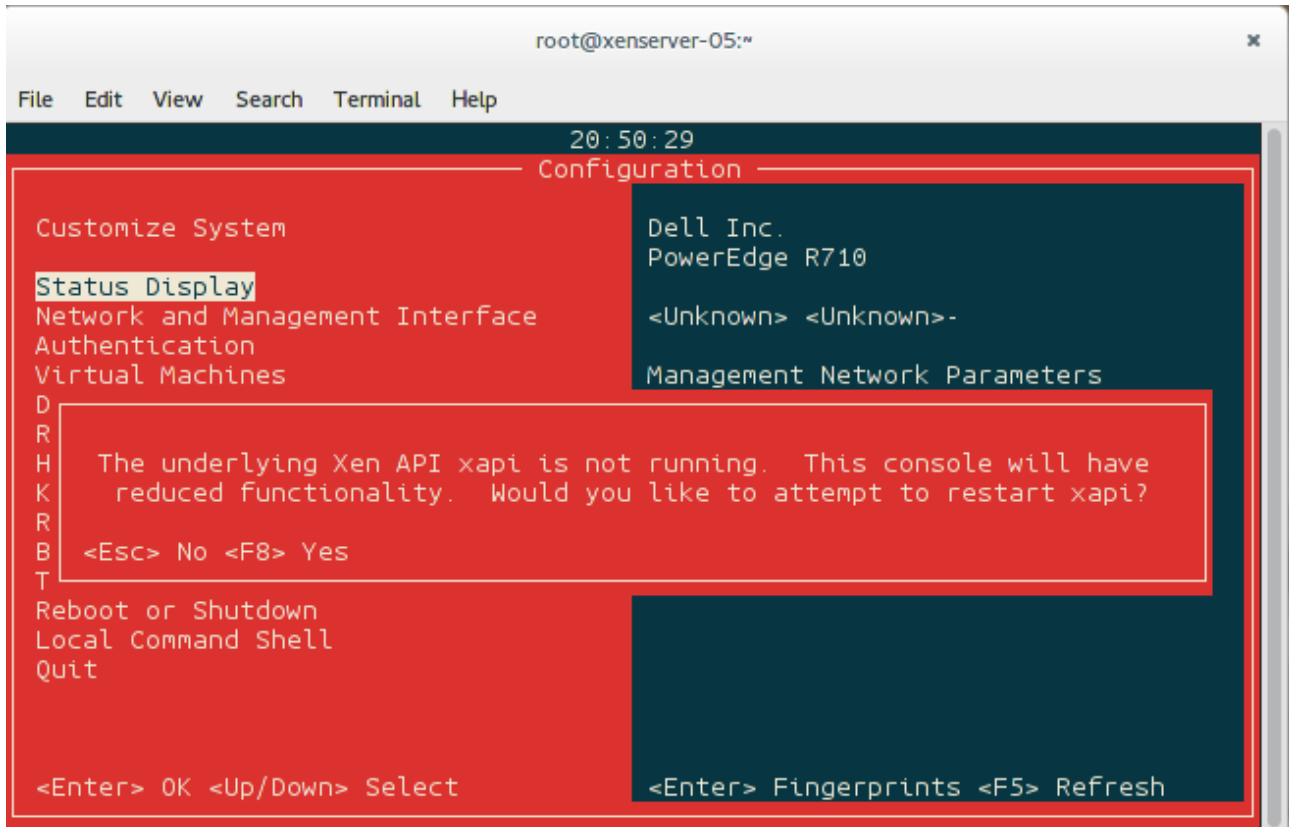


Imagen 28: Error de Xen API desde la interfaz de consola de xenserver

Para ello se intentó reiniciarlo en varias ocasiones, aunque sin éxito. Había algo más que impedía solucionar el problema por los cauces normales.

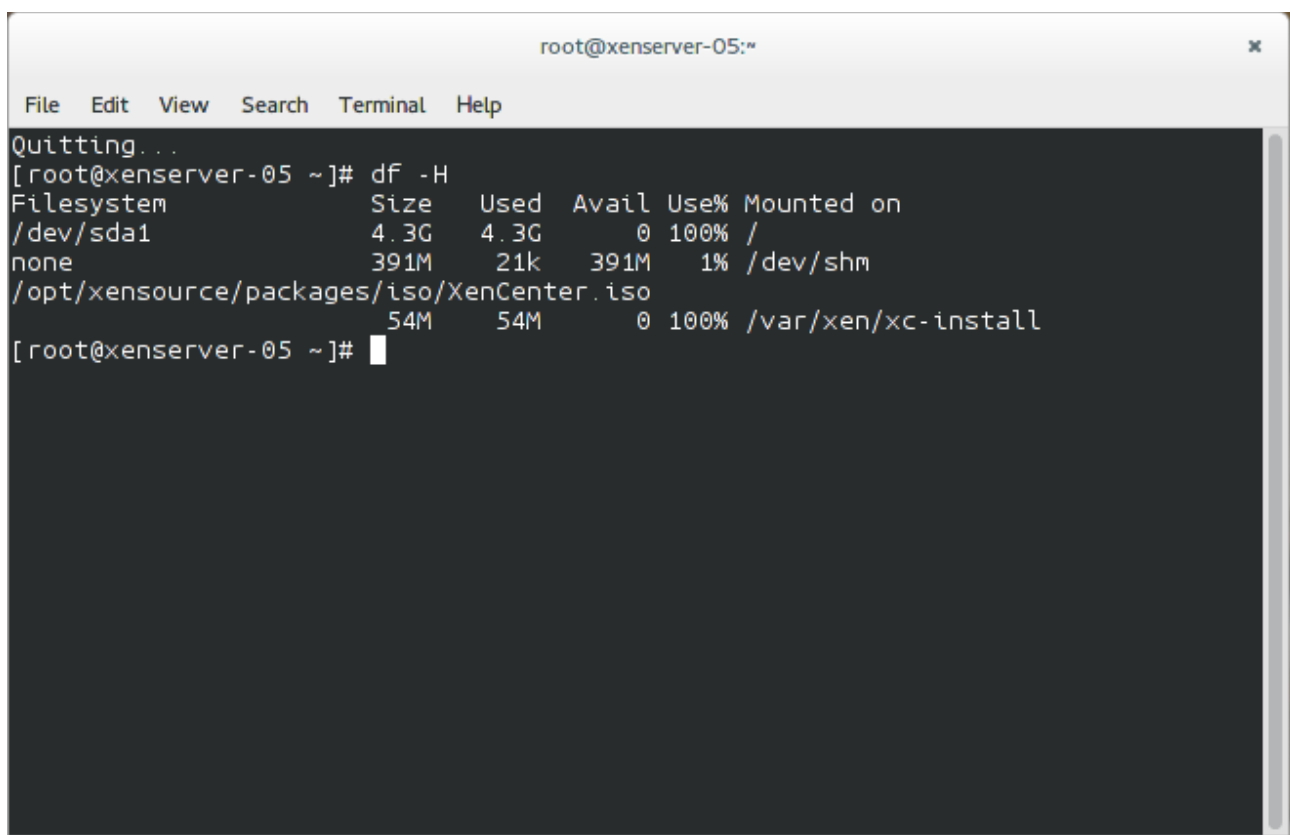
Una vez descartado el inicio manual del servicio se decidió entrar de lleno en los registros de errores almacenados en /var/log/. Revisando éstos y aún sin demasiadas pistas, la única referencia que encontraba era un error relacionado con xenstorage service que no podría arrancar. Poca información y además bastante inconexa teniendo en cuenta los escasos datos con los que se contaban.

Después de varios días intentado identificar cual podría ser la causa del problema y apoyándome en la comunidad de usuarios de Xen, se consiguió identificar el problema raíz que estaba desencadenando todos estos comportamientos tan extraños.

Aparentemente el problema residía en que el directorio raíz '/' de xenserver-05 estaba completamente lleno y eso impedía crear una serie de archivos de log al iniciar las máquinas lo que hacía abortar inmediatamente el arranque de éstas.

Parecía que el problema estaba identificado, pero ahora había que liberar algo de espacio. Para ello decidí borrar todos los archivos localizados el directorio '/tmp' así como la gran mayoría de los logs relacionados con Xen Server (miles de ellos en este punto). También se borraron las imágenes de máquinas no usadas y algunos de los archivos utilizados para actualizar el sistema que se encontraban en desuso.

Una vez liberado el espacio reinicie los servicios relacionados con xen y parece que todo volvió a la calma. El error de xapi dejó de aparecer y fue posible iniciar de nuevo todas las máquinas virtuales.

A screenshot of a terminal window titled 'root@xenserver-05:~'. The terminal shows the output of the 'df -H' command, which displays disk space usage for various filesystems. The output is as follows:

```
Quitting...
[root@xenserver-05 ~]# df -H
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        4.3G   4.3G    0 100% /
none            391M   21k  391M   1% /dev/shm
/opt/xensource/packages/iso/XenCenter.iso
54M    54M    0 100% /var/xen/xc-install
```

*Imagen 29: Sin espacio en disco en el servidor físico*

A pesar de que la solución es sencilla ha sido un problema difícil de diagnosticar y con unas consecuencias demoledoras para este entorno, dado que impedía iniciar las máquinas virtuales.

## Heartbleed

HeartBleed el nombre que se le ha dado al bug considerado por muchos como la mayor amenaza de seguridad de la era de Internet. En abril de 2014, algunos empleados de Google Security encontraron un importante fallo de seguridad en una de las librerías OpenSSL relacionadas con el cifrado de archivos.

Debido a su gran importancia, vamos a dedicarle algunas líneas para explicar que ocurrió y sus consecuencias para las todas las máquinas basadas en sistemas GNU/Linux de este proyecto y los millones de dispositivos que diariamente utilizan OpenSSL.

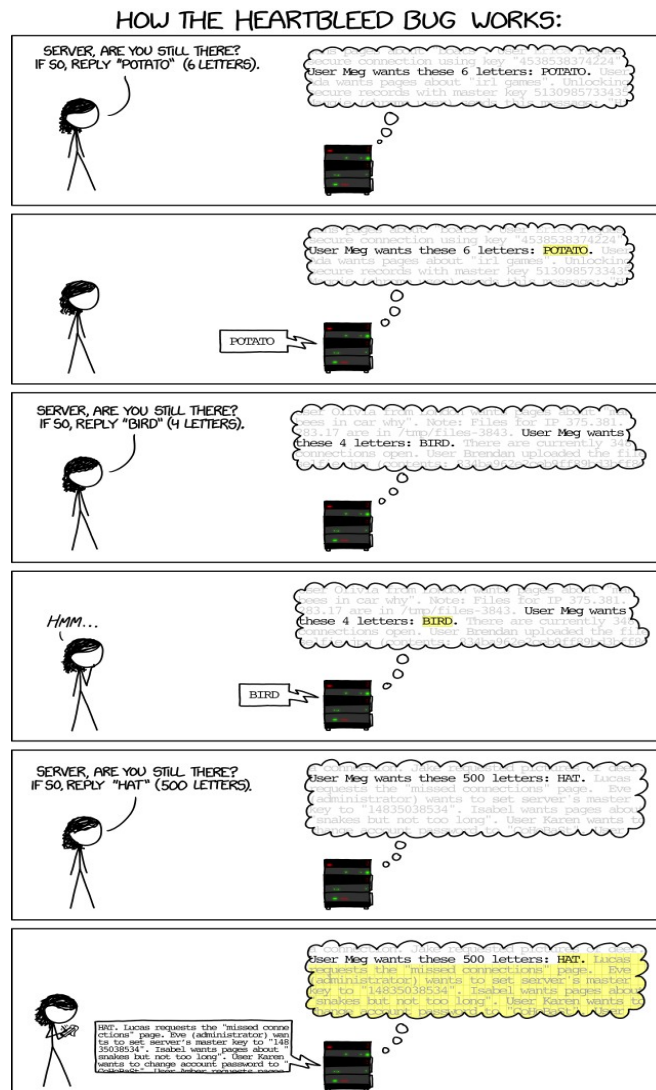
## Explicación

El 17 de Abril de 2014 se hizo de dominio publico el bug CVE-2014-0160 que afectaba a todos los dispositivos que utilizasen una versión no parcheada de ciertas versiones de la librería OpenSSL. Las consecuencias de este problema son terriblemente graves pudiendo dejar en entre dicho la configuración de seguridad más robusta, fiable y probada del planeta. Un atacante malicioso podría obtener datos privados y sensibles de nuestro sistema simplemente utilizando la función heartbleed de la librería OpenSSL que verifica si la conexión con un servidor sigue activa. A diferencia de otras vulnerabilidades encontradas hasta la fecha, para explotar esta falla no es necesario prácticamente nada y por desgracia no deja ningún tipo de registro en los ficheros de registros del sistema.

La función heartbleed se encarga de mantener abierta una conexión entre un cliente y un servidor. De esta forma, cuando el cliente quiere verificar si el servidor sigue ahí escuchando, lo único que debe hacer una llamada a la función hearthbleed del servidor e indicarle una palabra y su longitud. Si la conexión se mantiene activa el servidor contestará con esa misma palabra. El problema surge cuando la palabra enviada y su longitud no coinciden. Pongamos un ejemplo práctico para entender bien el problema.

Para verificar si un servidor mantiene la conexión, desde el cliente le enviamos la palabra “vivo” y le decimos que su longitud es 4. De momento nada extraño, el servidor responde y verificamos que la conexión sigue activa. Hagamos otra prueba con la misma cadena de texto pero esta vez vamos a indicarle que la longitud de dicha cadena sera de 512. Al estar programado en C devuelve contenido en memoria que no te corresponde.

Daniel Fernandez Rodriguez  
"Integración de OpenStack con un Directorio Activo"



*Imagen 30: Explicación gráfica de HeartBleed*

En la mayoría de las ocasiones devolverá contenido basura, pero un atacante podría recopilar suficiente información hasta adquirir algo relevante, como ficheros de configuración o incluso claves privadas del servidor.

```
OpenSSL 1.0.1 through 1.0.1f (inclusive) are vulnerable
OpenSSL 1.0.1g is NOT vulnerable
OpenSSL 1.0.0 branch is NOT vulnerable
OpenSSL 0.9.8 branch is NOT vulnerable
```

Gracias a la excelente trazabilidad de los proyectos de software libre, es sabido que el bug fue introducido en diciembre de 2011 y ha estado presente en la rama oficial 1.0.1 de la librería desde marzo de 2012 hasta que casi dos años mas tarde en abril de 201 fue corregida la vulnerabilidad.



## Solución

El 7 de abril de 2014, mismo día de hacerse publico el agujero de seguridad, apareció una nueva actualización de la librería que solucionaba la vulnerabilidad.

Desde en ese mismo momento en el que fue hecho público, se recomienda encarecidamente cambiar las contraseñas de aquellos servicios vulnerables que hayan podido estar expuestos, así como generar nuevos certificados para cada uno de los servidores que utilizasen las versiones vulnerables de esta librería de cifrado.

En este caso concreto, una vez publicada la vulnerabilidad se procedió a la actualización de openssl en cada una de las máquinas Linux que forma parte de este proyecto. Así mismo y a pesar del bajo índice de exposición de las máquinas de este proyecto, por precaución se cambio la contraseña de acceso de todos los usuarios expuestos.

## SeLinux activado el máquinas RH

SeLinux es el sistema de control de acceso dinámico que viene implementado en RedHat Enterprise Linux (RHEL) y derivadas (CentOS, Fedora, Scientific Linux...). Originalmente desarrollado por la Agencia de Seguridad Nacional de los Estados Unidos de Americana (NSA) es un sistema de control de acceso muy seguro y robusto. Su configuración está basada en etiquetas, lo que supone un paso más allá del antiguo -y a día de hoy poco fiable- sistema de permisos de archivo. De esta manera independientemente de los permisos, si el fichero, archivo o configuración no está etiquetado correctamente su funcionamiento podría no ser el esperado. De este etiquetado se encarga automáticamente SeLinux aunque es responsabilidad del administrador de la máquina modificar ciertos comportamientos o configuraciones del mismo para adaptarlo al uso que queremos hacer del sistema.

Este proyecto se ha visto afectado por algunas configuraciones no se estaban aplicando correctamente debido a SeLinux. OpenStack proporciona un paquete denominado openstack-selinux que nos ayuda con algunas de las configuraciones más típicas que pueden encontrarse, así como las excepciones necesarias para que el sistema funcione.

## Reparar la conexión a la biblioteca NFS

Con cada reinicio del servidor xenserver-05 y consecuente reinicio de las máquinas virtuales alojadas en él se producía la pérdida de acceso a la biblioteca NFS. A pesar de que los servicios de la máquina virtual estaban corriendo sin problema era necesario forzar el refresco de la reconexión desde la máquina host.

Daniel Fernandez Rodriguez  
“Integración de OpenStack con un Directorio Activo”

Para ello simplemente bastaba con seguir un pequeño asistente desde el programa Xen Center.

## Capítulo 10. Ampliaciones

### Actualización a la última versión de OpenStack

Como se ha comentado anteriormente en este documento el ritmo de actualización de OpenStack es bastante rápido, con una nueva versión disponible al público general cada 6 meses. La versión utilizada en este proyecto se corresponde con la versión lanzada en mayo 2014 de nombre clave IceHouse. Durante el desarrollo de este proyecto fue lanzada la décima versión de OpenStack “Juno”. Como en otras actualizaciones esta versión incorpora diversas mejoras en el código y los clientes, mayor funcionalidad y algún que otro componente nuevo.

Una posible ampliación de este proyecto podría ser la de actualizar OpenStack a su versión más reciente, incorporar así las nuevas mejoras y características que nos brinda el nuevo desarrollo y solucionar los posibles problemas de actualizar de versión un sistema en producción.

Como consideración adicional importante, la nueva versión de OpenStack Juno, ya no proporciona sus paquetes compilados para RHEL6 y derivadas (Centos6, ScientificLinux 6, etc) por lo que para instalarlo será necesario realizar previamente una actualización del sistema operativo.

### Diseño de un entorno multinodo

En este desarrollo y en respuesta a la prueba de concepto que pretende ser, todos los componentes de Openstack fueron instalados en el mismo servidor. En un entorno de producción real, donde rendimiento, escalabilidad y tolerancia a desastres deberían ser tomados como requisitos primordiales, sería conveniente separar al menos algunos de los los componentes clave, con el fin de mejorar la arquitectura convirtiéndola de paso en más robusta, escalable y resistente a fallos.

En realidad, el grado de dificultad de implementar este nuevo enfoque no resultaría más complicado que el mononodo aquí implementado. Simplemente se tratará de tener claras las condiciones hardware de las máquinas que en la que se van a instalar cada nodo para así poder repartir mejor la carga de trabajo. OpenStack es un proyecto diseñado para ser ampliado y su arquitectura modular simplifica mucho esta tarea.

Cómo ya se ha comentado, debido a las restricciones hardware con las que se cuentan para

este proyecto, para este proyecto se ha optado por mantener una única instalación en la que se ejecutarán los servicios necesarios para obtener un Openstack funcional.

## Añadir más nodos de computación

Otra ampliación posible, relacionada con la anteriormente comentada, sería la de añadir un mayor número de nodos de computación a la configuración aquí documentada. Los nodos de computación, o *computes nodes*, son aquellas máquinas destinadas a hospedar las instancias de las máquinas virtuales por lo que dependiendo de las características hardware de éste y del tamaño de las máquinas a ser hospedadas, se podrán albergar un mayor o menor número de ellas. Como es lógico, un mayor número de nodos, implica más hardware.

Agregar más nodos de computación al proyecto no tiene demasiado misterio en sí mismo y para ello simplemente bastaría con añadir una nueva máquina con Nova como único componente instalado. Aunque modificación deseable, para este caso no se ha podido realizar por la falta de servidores físicos un mayor número de servidores físicos disponibles

# Referencias bibliográficas

## Libros

Jackson Kevin, Bunch Cody. "OpenStack Cloud Computing Cookbook" PACKT Publishing

## Referencias en Internet

Para las siguientes referencias, la última fecha de consulta fue de 2015, más concretamente entre los meses de febrero y junio de ese mismo año.

## Programas

Marc Andre Lureau "Planner" <http://live.gnome.org/Planner>

Varios autores (The GNOME Project) "Gedit" <https://wiki.gnome.org/Apps/Gedit>

Varios autores (Vim community) "About Vim" <http://www.vim.org/about.php>

Firefox Foundation "Mozilla Firefox" <https://www.mozilla.org/es-ES/firefox/desktop/>

Oracle "Oracle VM VirtualBox" <https://www.virtualbox.org/manual/ch01.html#virtintro>

LibreOffice "The Document Foundation" <https://www.libreoffice.org/>

Varios autores (The GNOME Foundation) "Gnome" <https://www.gnome.org/>

Fermilab y CERN "Scientific Linux" <http://scientificlinux.org/about/>

Vates para Xen-Orchestra "https://xen-orchestra.com/#!/features" <https://xen-orchestra.com>

Linux Foundation "Xen Project" <http://www.xenproject.org>

Oracle "Oracle and Sun Microsystems" <http://www.oracle.com/us/sun/index.html>

Citrix Systems, Inc "XenCenter Development: XenCenter" <http://www.xenserver.org/partners/developing-products-for-xenserver/21-xencenter-development/88-xc-dev-home.html>

## Instalaciones

OpenStack Fundation "OpenStack Official Documentation" <http://docs.openstack.org/>

RedHat "RDO Quickstart" <https://www.rdoproject.org>

Varios autores (documentación Oficial OpenStack) "HowtoIntegrateKeystonewithAD"

<https://wiki.openstack.org/wiki/HowtoIntegrateKeystonewithAD>

Seltzer Brian "OpenStack Icehouse – Active Directory Integration"

<http://behindtheracks.com/2013/08/openstack-active-directory-integration/>

Fontanet Julien, Lambert Olivier "Instalación Orchestra: Manual Installation"

[https://github.com/vatesfr/xo/blob/master/doc/installation/manual\\_installation.md](https://github.com/vatesfr/xo/blob/master/doc/installation/manual_installation.md)

Varios autores, Microsoft "Instalar Windows Server 2012: Requisitos del sistema"

(<http://technet.microsoft.com/es-es/library/jj134246.aspx>)