Arquitecturas de red para servicios en Cloud computing

Autores: Jorge Lastras Hernansanz, Javier Lázaro Requejo y Jonatan David Mirón García

Directores: Prof. Rafael Moreno Vozmediano y Prof. Rubén Santiago Montero

Universidad: Universidad Complutense de Madrid - Facultad de Informática

Asignatura: Sistemas Informáticos

Curso: 2008/2009

Índice de contenidos

Agradecimientos

Nos gustaría dar las gracias a nuestros directores del proyecto, Rubén Santiago Moreno y Rafael Moreno Vozmediano, por su comprensión a la hora de solventar y adaptarnos a los distintos problemas que nos han ido surgiendo. También querríamos agradecerle a Raúl Sampedro su ayuda a la hora de configurar el cluster y el OpenNEbula.

Resumen

Hoy en día la mayoría de los servidores residen en grandes computadoras en las que solo se utiliza una pequeña parte de la CPU. Este desaprovechamiento ha motivado que se instalen máquinas virtuales en los servidores para usar más el hardware y reducir el número de servidores. Esto tiene consecuencias directas en ahorro de dinero, energía y administración de manera que la virtualización está en auge hoy en día.

Otra ventaja significativa es la facilidad de escalabilidad de los servidores, ya que se puede hacer uso de proveedores como Amazon EC2 para contratar máquinas virtuales y alojar los servicios. Así se pueden cubrir las necesidades de algunas empresas en las que requieren en periodos de tiempo muy breves soportar una gran demanda de sus servicios sin invertir grandes cantidades de dinero en hardware que más adelante no aprovecharán.

El objetivo de este proyecto consistirá en el diseño de un sistema para la gestión y virtualización de la red de servicios en clouds.

Para realizarlo se implementará una infraestructura piloto a partir de dos clusters físicamente separados. Cada uno de ellos dispondrá de un front-end y dos nodos en los que se sostendrá un pool de máquinas virtuales. Cada conjunto de máquinas virtuales estará en una red local y se instalará un túnel entre los dos front-end de los clusters para que ambas redes tengan acceso entre ellas. Este túnel es posible implementarlo gracias a que los front-end tienen conexión a Internet pero esto no significa que su seguridad pueda estar comprometida porque toda la información que viaja a través del túnel estará cifrada.

Como servicio de red optaremos por instalar servidores web en los back-ends, es decir, en todas las máquinas virtuales. En uno de los front-end se instalará otro servidor web que actuará como balanceador de carga para repartir las peticiones web entre todos los back-ends. De esta manera conseguimos virtualizar una aplicación web sencilla.

Trabajar en este proyecto nos ha servido para aprender de tecnologías que desconocíamos por completo como la Virtualización o Cloud Computing los cuales consideramos que tendrán un futuro más que relevante en el mundo de la informática. Gran parte del trabajo ha consistido en documentarnos de estas tecnologías así que el despliegue de la infraestructura no nos llevó demasiado tiempo.

También hemos aprendido que virtualizar requiere un estudio de la utilización de los recursos de las aplicaciones para analizar si conviene o no instalar máquinas virtuales. Esto es importante porque a veces es fácil sobrecargar el hardware y como consecuencia no se obtiene el rendimiento deseado de las aplicaciones virtualizadas.

Abstract

Nowadays most of the servers that reside on the computers systems only use a small rate of the CPU. This advantage has prompted to install virtual machines on servers to use more hardware and reduce the number of servers. This has a direct impact on saving money, energy and administration so that virtualization is being more popular.

Another significant advantage is the ease of scalability of the servers, since you can use providers like Amazon EC2 virtual machines for hire and accommodation services. This can cover the needs of some companies in which they need in very short periods of time support a strong demand for its services without investing large amounts of money on hardware that will not utilize later.

The aim of this project consists of designing a system for management and virtualization of network services in clouds.

To realize an infrastructure pilot will be implemented from two physically separated clusters. Each has a front-end and two nodes, which in turn support a pool of virtual machines. Each set of virtual machines will be on a local network and we will install a tunnel between the two front-end clusters for both networks have access between them. This tunnel can be implemented by having the front-end connected to the Internet but this does not mean that the security might be compromised because all the information that travels through the tunnel will be encrypted.

As a service network we will choose to install web servers on the back-ends, i.e., all virtual machines. In one of the front-end web server will be installed that will act as a load balancer to distribute web requests among all back-ends. In this way we virtualize a simple web application.

Work on this project has helped us to learn unknown technologies like Virtualization and Cloud Computing which will be relevant in future in the world of computing. The documentation of this kind of technologies has taken a long time and the deployment of the infrastructure was done in a relative short time.

We have also learned that virtualising requires a study of resource utilization of applications for analyzing whether or not to install virtual machines. This is important because sometimes it's easy to overload the hardware and as a result is not obtained the desired performance of virtualized applications.

1 Introducción y objetivos

1.1 Justificación del proyecto

- **Necesidad del despliegue de servicios virtualizados**

Hasta ahora la mayoría de los servidores web se trataban de grandes computadoras difícilmente escalables y que se ajustaban muy poco a las necesidades del servicio en cada momento.

Por ejemplo, una página web de deportes que de forma habitual sea visitada por, por ejemplo 10.000 personas, ante un determinado acontecimiento como una final o como las Olimpiadas, va a ver multiplicado ese número de visitas, con lo que ha de dar cabida a dichos picos.

Al tener grandes máquinas hardware o centros de datos capaces de dar soporte a dichos picos de visitas, de cierta forma se produce un claro desaprovechamiento hardware en una situación habitual

Para evitar este desaprovechamiento de los servidores surge la virtualización, con la que podemos hacer un mejor uso y aprovechamiento del hardware del que disponemos, además de implementar una solución fácilmente escalable y con un tiempo de despliegue de nuevos servidores realmente corto (casi instantáneo) frente al uso convencional.

Además, tal y como veremos más adelante en el apartado de "Cloud Computing", empezamos a alejarnos del término Servidor Web como la típica computadora física (o un conjunto de computadoras), situados en un lugar determinado, y empezamos a verlos como un servicio web, en el que pueden estar repartidos sus distintos componentes entre distintos lugares del mundo. En los que podemos incrementar o disminuir su capacidad simplemente contratando un ancho de banda en un servidor (por ejemplo Amazon) y desplegando allí una imagen.

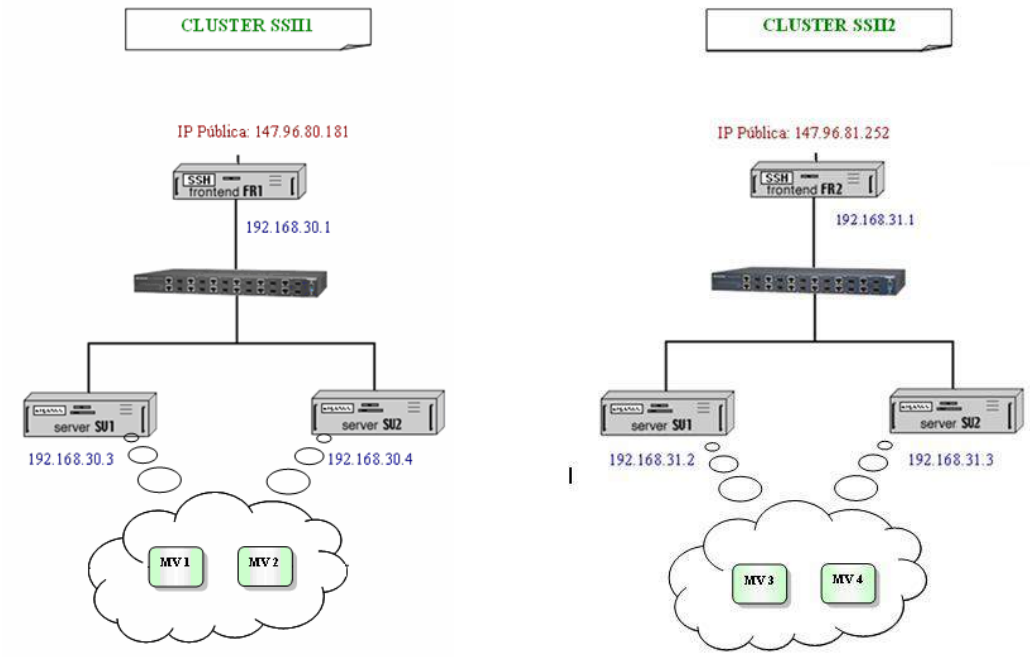
1.2 Objetivos

- **Diseño de un sistema para la gestión y virtualización de la red de servicios en clouds**
- **Implementación de una infraestructura piloto**

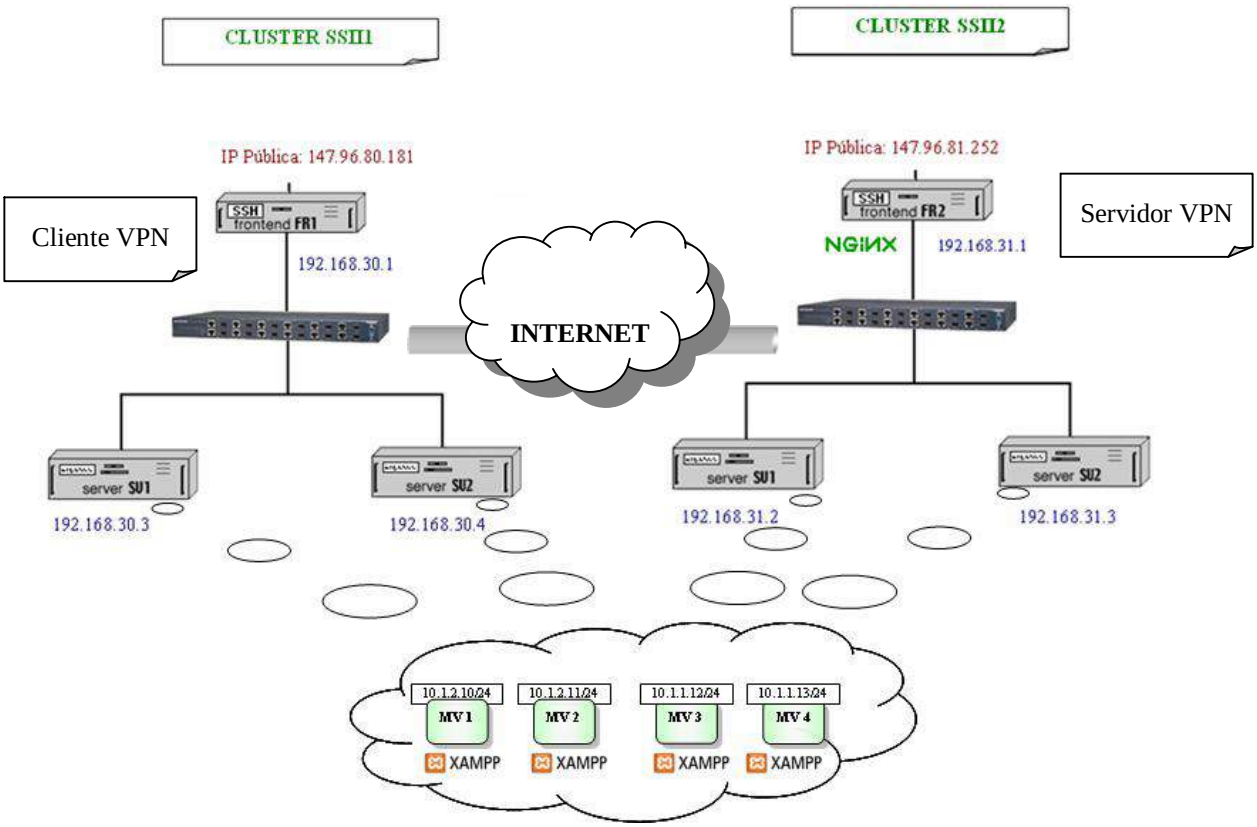
Tal y como veremos más adelante, la realización de nuestro proyecto consistirá en una implementación de un servidor web a partir de la unión de recursos virtualizados de diferentes dominios.

Nuestra implementación básicamente va a estar formada por dos clusters, sin ningún tipo de conexión local, en principio, entre ellos, es decir, separados por Internet. Nuestro objetivo será aunar las posibilidades que ambos clusters nos ofrecen aprovechando los recursos de los dos y formando así un servidor web con el doble de capacidad de procesamiento de peticiones.

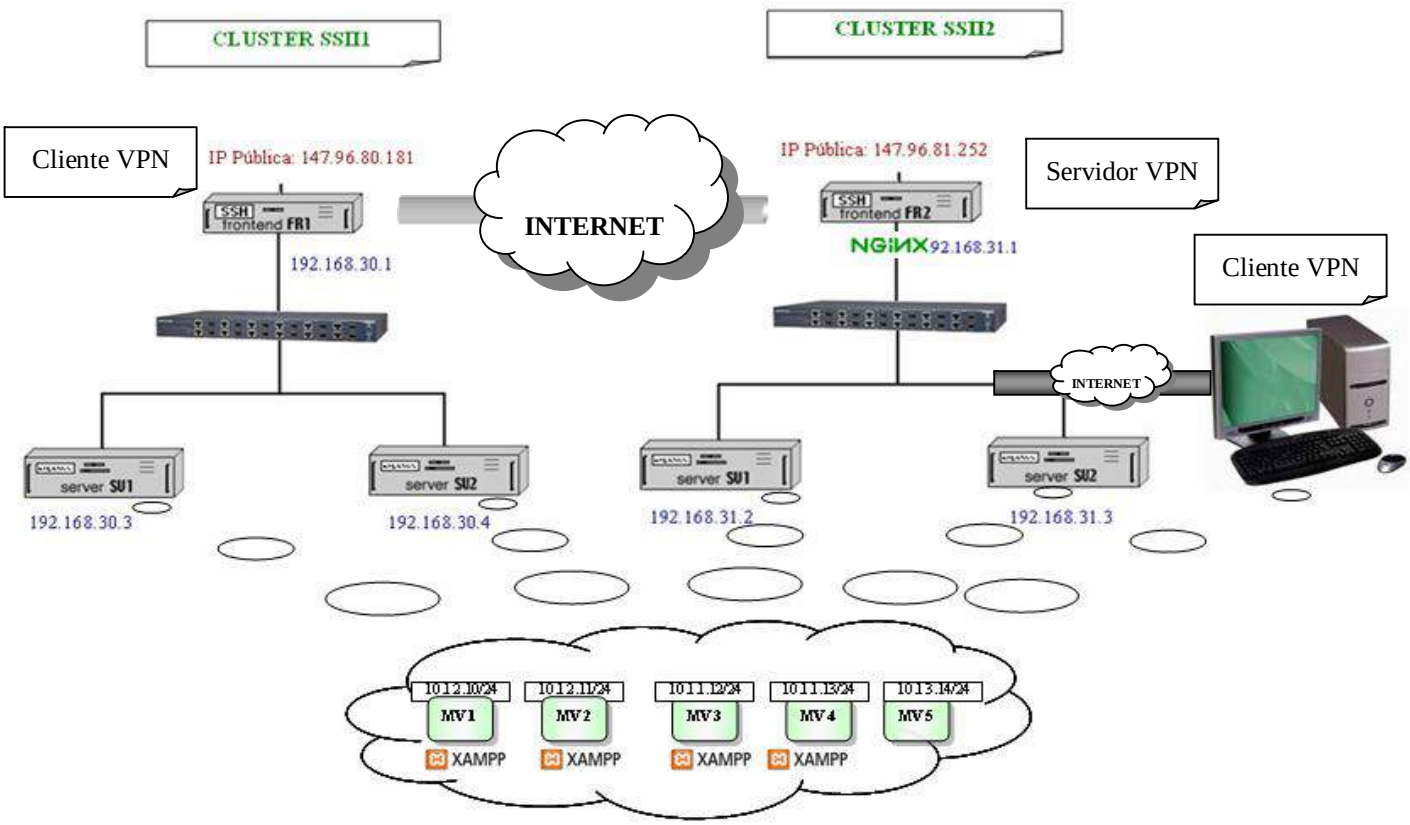
Por tanto, tal y como podemos ver en el esquema, nuestros clusters podrían estar alojando, gracias al software de OpenNEbula, una serie de máquinas virtuales (en este caso son dos por cada cluster, pero hemos hecho pruebas de hasta 4 máquinas virtuales, dos en cada nodo, sin que se vea muy afectado el rendimiento), además, dicho software nos abstrae de la tecnología de virtualización concreta que estemos usando (Xen, KVM...), y nos permite pensar en el conjunto de máquinas virtuales sin ligarlos necesariamente a un nodo del cluster en concreto (aunque obviamente cada máquina virtual ha de residir en un nodo en concreto, pudiéndolas migrar de uno a otro de forma sencillísima).



El objetivo de nuestra arquitectura de red será, por tanto, unir los recursos (los pool de máquinas virtuales) en uno solo de forma completamente segura gracias al uso de VPN (uso de la infraestructura de Internet para crear una "red local"), de tal forma que hayamos podido incrementar nuestra capacidad de procesamiento de forma segura y siendo capaces de abstraer de cualquier tipo de arquitectura física, pudiendo añadir nuevas máquinas virtuales (servidores web) a nuestro pool de máquinas virtuales desde otros computadores de cualquier parte del mundo, todo ello de forma completamente transparente para el usuario final.



También contemplaremos, al tratarse de una arquitectura de Cloud Computing, la posibilidad de en un momento dado y ante un incremento de la demanda de nuestro servicio, unir nuevas máquinas virtuales a nuestra pool de servidores web, contratándolas desde servicios como Amazon EC2 o utilizando otros recursos hardware que tengamos disponibles (por ejemplo el ordenador de nuestra casa, que es el ejemplo que pudimos realizar).


CLUSTER SSII1
CLUSTER SSII2
Cliente VPN
IP Pública: 147.96.80.181
INTERNET
IP Pública: 147.96.81.252
Servidor VPN
SSH frontend FR1
192.168.30.1
SSH frontend FR2
NGINX 92.168.31.1
Cliente VPN
INTERNET
server SU1
server SU2
192.168.30.3
192.168.30.4
server SU1
server SU2
192.168.31.2
192.168.31.3
10.12.10/24
10.12.11/24
10.11.12/24
10.11.13/24
10.13.14/24
MV1
MV2
MV3
MV4
MV5
XAMPP
XAMPP
XAMPP
XAMPP

2 Virtualización

En informática, virtualización es un término amplio que se refiere a la abstracción de los recursos de una computadora. Este término es bastante antiguo: viene siendo usado desde antes de 1960, y ha sido aplicado a diferentes aspectos y ámbitos de la informática, desde sistemas computacionales completos hasta capacidades o componentes individuales.

El tema en común de todas las tecnologías de virtualización es la de ocultar los detalles técnicos a través de la encapsulación. La virtualización crea un interfaz externo que esconde una implementación subyacente mediante la combinación de recursos en locaciones físicas diferentes, o mediante la simplificación del sistema de control. Un reciente desarrollo de nuevas plataformas y tecnologías de virtualización han hecho que se vuelva a prestar atención a este maduro concepto.

De modo similar al uso de términos como "abstracción" y "orientación a objetos", virtualización es usado en muchos contextos diferentes.

Asimismo, el término virtualización es un concepto importante en contextos no computacionales. Muchos sistemas de control implementan interfaces virtuales en un mecanismo complejo; de esta manera el pedal del acelerador de un automóvil moderno hace más que solo aumentar el flujo del combustible hacia el motor; y el sistema de vuelos por cables (fly by wire) presenta un avión virtual simplificado que tiene muy poco que ver con la implementación física.

2.1 Virtualización de plataforma

El sentido original del término *virtualización*, nacido en 1960, es el de la creación de una máquina virtual utilizando una combinación de hardware y software. Para nuestra conveniencia vamos a llamar a esto virtualización de plataforma. El término *máquina virtual* aparentemente tiene su origen en el experimento del sistema de paginación (paging system) de IBM M44/44X. La creación y administración de las máquinas virtuales también se refiere a la creación de *seudo máquinas*, en los primeros días de la CP-40, y de *virtualización de servidores* más recientemente. Los términos *virtualización* y *máquina virtual* han adquirido, a través de los años, significados adicionales.

La virtualización de plataforma es llevada a cabo en una plataforma de hardware mediante un software "host" ("anfitrión", un *programa de control*) que simula un entorno computacional (*máquina virtual*) para su software "guest". Este software "guest", que generalmente es un sistema operativo completo, corre como si estuviera instalado en una plataforma de hardware autónoma. Típicamente muchas máquinas virtuales son simuladas en una máquina física dada. Para que el sistema operativo "guest" funcione, la simulación debe ser lo suficientemente robusta como para soportar todas las interfaces externas de los sistemas guest, las cuales pueden incluir (dependiendo del tipo de virtualización) los drivers de hardware.

Existen muchos enfoques a la virtualización de plataformas, aquí se listan con base en cuan completamente es implementada una simulación de hardware.

- Emulación o simulación: la máquina virtual simula un hardware completo, admitiendo un sistema operativo "guest" sin modificar para una CPU completamente diferente. Este enfoque fue muy utilizado para permitir la creación de software para

nuevos procesadores antes de que estuvieran físicamente disponibles. Por ejemplo Bochs, PearPC, Qemu sin aceleración, y el emulador Hercules. La emulación es puesta en práctica utilizando una variedad de técnicas, desde state machines hasta el uso de la recopilación dinámica en una completa plataforma virtual.

- Virtualización nativa y virtualización completa: la máquina virtual simula un hardware suficiente para permitir un sistema operativo "guest" sin modificar (uno diseñado para la misma CPU) para correr de forma aislada. Típicamente, muchas instancias pueden correr al mismo tiempo. Este enfoque fue el pionero en 1966 con CP-40 y CP[-67]/CMS, predecesores de la familia de máquinas virtuales de IBM. Algunos ejemplos: VMware Workstation, VMware Server, Parallels Desktop, Virtual Iron, Adeos, Mac-on-Linux, Win4BSD, Win4Lin Pro y z/VM.
- Virtualización parcial (y aquí incluimos el llamado "address space virtualization"): la máquina virtual simula múltiples instancias de mucho (pero no de todo) del entorno subyacente del hardware, particularmente address spaces. Este entorno admite compartir recursos y aislar procesos, pero no permite instancias separadas de sistemas operativos "guest". Aunque no es vista como dentro de la categoría de máquina virtual, históricamente éste fue un importante acercamiento, y fue usado en sistemas como CTSS, el experimental IBM M44/44X, y podría decirse que en sistemas como OS/VS1, OS/VS2 y MVS.
- Paravirtualización: la máquina virtual no necesariamente simula un hardware, en cambio ofrece un API especial que solo puede usarse mediante la modificación del sistema operativo "guest". La llamada del sistema al hypervisor tiene el nombre de "hypercall" en Xen y Parallels Workstation; está implementada vía el hardware instruction DIAG ("diagnose") en el CMS de VM en el caso de IBM (este fue el origen del término hypervisor). Ejemplo: VMware ESX Server, Win4Lin 9x y z/VM.
- Virtualización a nivel del sistema operativo: virtualizar un servidor físico a nivel del sistema operativo permitiendo múltiples servidores virtuales aislados y seguros correr en un solo servidor físico. El entorno del sistema operativo "guest" comparte el mismo sistema operativo que el del sistema "host" (el mismo kernel del sistema operativo es usado para implementar el entorno del "guest"). Las aplicaciones que corren en un entorno "guest" dado lo ven como un sistema autónomo. Ejemplos: Linux-VServer, Virtuozzo, OpenVZ, Solaris Containers y FreeBSD Jails.
- Virtualización de aplicaciones: consiste en el hecho de correr una desktop o una aplicación de server localmente, usando los recursos locales, en una máquina virtual apropiada. Esto contrasta con correr la aplicación como un software local convencional (software que fueron "instalados" en el sistema). Semejantes aplicaciones virtuales corren en un pequeño entorno virtual que contienen los componentes necesarios para ejecutar, como entradas de registros, archivos, entornos variables, elementos de uso de interfaces y objetos globales. Este entorno virtual actúa como una capa entre la aplicación y el sistema operativo, y elimina los conflictos entre aplicaciones y entre las aplicaciones y el sistema operativo. Los ejemplos incluyen el Java Virtual Machine de Sun, Softricity, Thinstall, Altiris y Trigence (esta metodología de virtualización es claramente diferente a las anteriores; solo una pequeña línea divisoria los separa de entornos de máquinas virtuales como Smalltalk, FORTH, Tel, P-code).

2.2 Ventajas y desventajas de la virtualización

La virtualización en la empresa

La virtualización en la empresa tiene una clara aplicación práctica: la consolidación de servidores. La consolidación de servidores consiste simplemente en la reducción del número de servidores.

Existen distintas maneras de consolidar, y una de ellas es la virtualización. Frente a otras vías para la consolidación, la virtualización permite reducir el número de servidores y optimizar al mismo tiempo su utilización. Es decir, que si antes de consolidar teníamos 100 servidores con una utilización media de CPU del 30%, después de consolidar con virtualización tendremos 50 servidores con una utilización media de CPU del 60%. Si consolidamos sin virtualización, podríamos tener 70 servidores con una utilización media del 40% (los números son meramente ilustrativos).

Muchas compañías se encuentran actualmente inmersas en proyectos de consolidación de servidores, pero ¿por qué consolidar, y no seguir con el modelo de servidores independientes?

Si preguntásemos a un empleado del departamento de informática de cualquier compañía que nos describiera el CPD, seguramente lo haría basándose en los servidores existentes. Nos mencionaría, por ejemplo, el servidor de base de datos, el servidor del correo electrónico, el servidor de CRM… Y también nos comentaría que cada servidor es de un fabricante diferente y cuenta con sistemas operativos diferentes. Por tanto, también se necesitan administradores formados en las diversas tecnologías existentes, y herramientas de gestión específicas, porque (digamos) lo que vale para monitorizar los servidores con Windows, no vale para los servidores con Unix.

Esta morfología se ha originado porque los CPDs actuales han ido creciendo basándose en silos aislados, en función de las necesidades del negocio. Si había que montar una nueva base de datos X, se examinaban las distintas comparativas (benchmarks) y se compraba el servidor más potente para esa base de datos X, del fabricante y sistema operativos indicados. Si seguidamente había que montar un servicio de atención al cliente, se compraría el servidor mejor para la aplicación de CRM elegida, muy posiblemente de otro fabricante y otros sistema operativo. Y así sucesivamente.

Después de décadas de un crecimiento de este tipo, los CPDs han llegado a un punto en que se han vuelto inmanejables, debido al alto número de servidores, la dificultad para coordinar cambios en esos CPDs, las múltiples consolas que es necesario visualizar para conocer exactamente cómo se está comportando el CPD, etc.

Para empeorar más la situación, estos CPDs generan unos costes altísimos. Cada máquina ocupa un espacio y consume electricidad para su alimentación y refrigeración. Los productos están licenciados para todo el hardware (CPUs) presente en los servidores, cuando los estudios demuestran que la utilización media de cada servidor está en torno al 30%. Esto quiere decir que sen está pagando por un 70% de máquina que no usamos normalmente, con sus costes asociados de licencias, mantenimiento, soporte, etc.

Por último, diversos estudios muestran que aproximadamente el 75% del presupuesto de TI de una compañía se gasta en mantenimiento de la estructura existente, mientras que tan sólo el 25% se dedica a innovación.

Afortunadamente, la virtualización nos puede ayudar a cambiar todo esto. La virtualización no es una moda, sino la respuesta a una necesidad que tienen los CPDs actuales.

La virtualización permite a las empresas evolucionar desde el CPD tradicional, basado “en hierro”, a un CPD de nueva generación, basado “en software”, en el que un pool de recursos compartidos se asigna dinámicamente a las aplicaciones que lo necesiten. Este nuevo CPD permitirá a los administradores centrarse en el servicio y no en la operación, mediante la abstracción del hardware y eliminación la gestión física de dispositivos.

Curiosamente, lo contrario a la virtualización, llamada agregación de servidores o grid computing, es otra forma de virtualización. Consiste en que diversos servidores funcionen como uno solo, y también nos puede ayudar a construir el nuevo CPD.

Por tanto, como resumen, cabría destacar entre las principales ventajas de la virtualización:

- **Consolidación de servidores.**
- **Aumento de la disponibilidad**, reducción de tiempos de parada.
- **Reducción de los costes de administración.**
- **Mejora de las políticas de backup**, recuperación ágil mediante puntos de control de las máquinas virtuales.
- **Aprovechamiento óptimo de los recursos disponibles.** Respuesta rápida ante cambios bajo demanda.
- **Continuidad de negocio y recuperación ante desastres**. En caso de fallo de un sistema físico, los sistemas lógicos allí contenidos pueden distribuirse dinámicamente a otros sistemas.
- **Escalabilidad**. Crecimiento ágil con contención de costes.
- **Virtual appliance**: máquinas virtuales preconfiguradas, cargar y funcionar. Máquinas paquetizadas y preconfiguradas para desempeñar una función determinada (servidores de correo, bases de datos, centralitas VoIP, aplicaciones cerradas).
- **Mantenimiento de aplicaciones heredadas.** Aplicaciones propietarias que no han sido adaptadas a las nuevas versiones de sistema operativo.
- **Eficiencia energética.**

2.3 Virtualización del front-end o del back-end

La virtualización puede realizarse a todos los niveles del CPD: desde los servidores de entrada o de aplicaciones (front-end), a los servidores que contienen las aplicaciones críticas para el negocio o las bases de datos (back-end), pasando por servidores de desarrollo o pruebas. De la gran variedad de soluciones comerciales de virtualización, algunas están más orientadas a virtualizar los servidores de negocio, mientras que otras son más adecuadas para servidores no críticos.

Por ejemplo, las soluciones de VMWare, pioneras en virtualización y muy bien valoradas por los usuarios, son más adecuadas para el front-end por diversos motivos. Por mencionar algunos, el hardware sobre el que se ejecutan (arquitectura x86) no es capaz de direccionar tantos datos de una sola vez como otras arquitecturas (porque no es una arquitectura nativa 64 bits). Esto se traduce en un menor rendimiento para cargas de trabajo pesadas (p.e., consulta a una base de datos grande). Y este hardware tiene características de fiabilidad y disponibilidad medias (RAS – Reliability Availability Serviceability), es decir, se estropea con más frecuencia y su reparación es más complicada que otras arquitecturas.

En el plano de software, las máquinas virtuales VMWare no admiten sistemas operativos Unix estables tradicionales de las aplicaciones críticas, como AIX o HP-UX. Por todos estos motivos, VMWare no es la solución óptima para virtualizar las aplicaciones críticas, que correrán por ejemplo en HP-UX, necesitarán rendimientos muy buenos y minimizar la paradas debido a averías hardware.

Las soluciones óptimas para virtualizar los servidores críticos son las que se ejecutan en servidores de alta gama, por ejemplo HP Integrity Virtual Machines. Esta solución permite crear máquinas virtuales sobre servidores con arquitectura Itanium. Itanium proporciona estupendas características de rendimiento (procesador 64 bits puro, con enormes cachés, etc.), fiabilidad y disponibilidad a la altura de los grandes ordenadores o mainframes. En HP Integrity Virtual Machines es posible ejecutar sistemas operativos estables como HP-UX.

2.4 La importancia de la gestión de la virtualización

En el apartado anterior analizábamos la importancia de elegir de la tecnología más adecuada para virtualizar correctamente el servidor que deseemos; tanto o más importante es disponer de una buena herramienta de gestión.

La virtualización presenta múltiples ventajas para los CPDs, principalmente desde el punto de vista de simplificación de la infraestructura física y conexiones. Pero al realizarse por software lo que antes se hacía por hardware, se introducen una nueva problemática que no existían en los entornos físicos.

Por ejemplo, si una máquina virtual puede ejecutarse en distintos servidores físicos, debemos saber en qué servidor físico se está ejecutado en cada momento. O si un servidor físico cuenta con varias máquinas virtuales que pueden estar arrancadas o no, debemos conocer en todo momento el estado de esas máquinas virtuales.

Normalmente, en un CPD convivirán servidores físicos y virtuales, por lo que la herramienta de gestión deberá permitir la gestión de los dos tipos de plataformas, idealmente en una única consola. Además, si se utilizan diversas tecnologías de virtualización, desde esta consola deberán poderse invocar de manera transparente todas las herramientas de gestión específicas de cada plataforma.

E idealmente, debería tratarse de una herramienta de gestión de la infraestructura integrable con herramientas de gestión empresarial, que nos avisasen de la repercusión que un problema en una máquina virtual puede tener en el negocio. Un ejemplo, si se para la máquina virtual que contiene la base de datos de clientes, se generaría una alerta en la herramienta de gestión de la infraestructura virtual, alerta que se redirigirá a la herramienta de gestión empresarial informando de que el servicio “Atención al cliente” está indisponible.

3 Introducción a Xen

Xen es una máquina virtual de código abierto desarrollada por la Universidad de Cambridge.

La meta del diseño es poder ejecutar instancias de sistemas operativos con todas sus características, de forma completamente funcional en un equipo sencillo. Xen proporciona aislamiento seguro, control de recursos, garantías de calidad de servicio y migración de máquinas virtuales en caliente. Los sistemas operativos deben ser modificados explícitamente para correr Xen (aunque manteniendo la compatibilidad con aplicaciones de usuario). Esto permite a Xen alcanzar virtualización de alto rendimiento sin un soporte especial de hardware.

Uso

Las máquinas virtuales son usadas a menudo por IBM y otras compañías en sus servidores y ordenadores centrales para abstraer la mayor cantidad de aplicaciones posibles y proteger las aplicaciones poniéndolas en máquinas virtuales diferentes (semejante a una jaula chroot). Puede también ser utilizada, no sólo por razones de seguridad o funcionamiento, sino también para poder tener arrancados diferentes sistemas operativos en el mismo ordenador. Con la migración de máquinas virtuales en caliente de Xen se puede conseguir hacer balance de cargas sin tiempos muertos.

Comparación con otras máquinas virtuales

- Denali utiliza la virtualización para proporcionar máquinas virtuales de alto rendimiento en ordenadores x86. La máquina virtual Denali da soporte a Sistemas Operativos mínimamente especializados hacia servicios de Internet. El sistema puede escalar a millares de máquinas virtuales. A diferencia de Xen, Denali no preserva el interfaz binario (ABI), y algunas aplicaciones deben ser recompiladas para que funcionen con las librerías del sistema operativo; en este sentido es similar a Exonúcleo.
- Virtuozzo sustituye la capa de la abstracción del hardware por una versión modificada permitiéndole funcionar con un mejor rendimiento de los sistemas operativos, pero fuerza a que todas las Máquinas Virtuales se ejecuten en un equipo y con el mismo sistema operativo. Actualmente existe una versión para Windows 2003 y para Fedora Core Linux.

Paravirtualización con Xen

Xen utiliza una técnica llamada paravirtualización para alcanzar alto rendimiento (es decir, bajas penalizaciones del rendimiento, típicamente alrededor del 2%, con los peores casos de rendimiento rondando el 8%; esto contrasta con las soluciones de emulación que habitualmente sufren penalizaciones de un 20%).

Con la paravirtualización, se puede alcanzar alto rendimiento incluso en arquitecturas (x86) que no suelen conseguirse con técnicas tradicionales de virtualización. A diferencia de las máquinas virtuales tradicionales, que proporcionan entornos basados en software para simular hardware, Xen requiere portar los sistemas operativos para adaptarse al API de Xen. Hasta el momento hay ports para NetBSD, Linux, FreeBSD y Plan 9.

En 2005, Novell muestra un port de NetWare para Xen. Un port de Windows XP fue creado durante el desarrollo inicial de Xen, pero las licencias de Microsoft prohíben su lanzamiento público.

Virtualización completa con Xen

Intel ha realizado modificaciones a Xen para soportar su arquitectura de extensiones Vanderpool. Esta tecnología permitirá que sistemas operativos sin modificaciones se ejecuten en máquinas virtuales Xen, si el sistema soporta las extensiones Vanderpool o Pacífica (de Intel y AMD respectivamente, extensiones para soportar virtualización de forma nativa). Prácticamente, esto significará que habrá una mejora de rendimiento, y que será posible virtualizar Windows sin tener que modificarlo.

Migración de máquinas virtuales

Las máquinas virtuales Xen pueden ser migradas en caliente entre equipos físicos sin pararlos. Durante este proceso, la memoria de la máquina virtual es copiada iterativamente al destino sin detener su ejecución. Una parada muy breve de alrededor de 60 a 300 ms es necesaria para realizar la sincronización final antes de que la máquina virtual comience a ejecutarse en su destino final. Una tecnología similar es utilizada para suspender las máquinas virtuales a disco y cambiar a otra máquina virtual.

Xen en la actualidad

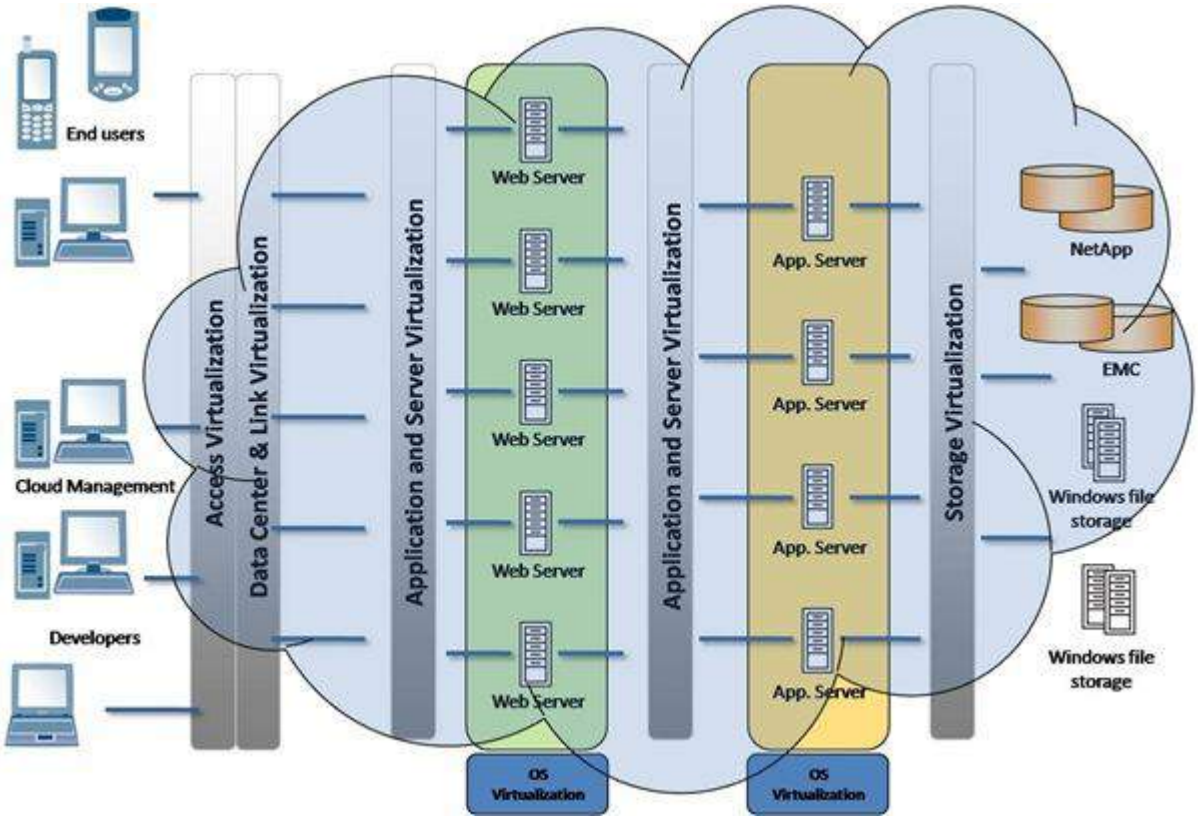
El 15-08-2007 Citrix adquiere XenSource, por un valor de 500 millones de dólares estadounidenses. Esta empresa ha lanzado recientemente XenServer 4.1, habiendo un producto gratuito, el XenServer Express Edition, aunque solo puede soportar cuatro máquinas virtuales.

Plataformas soportadas

Xen funciona actualmente en sistemas basados en x86. Actualmente se están portando las plataformas AMD64, IA64 y PPC. Los ports de otras plataformas son técnicamente posibles y podrán estar disponibles en el futuro.

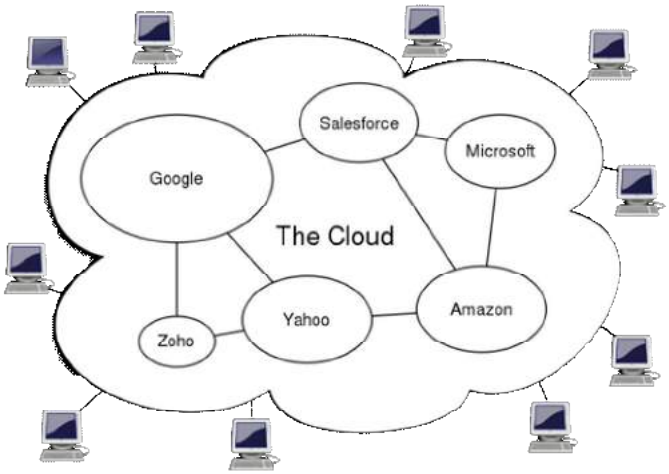
4 Cloud Computing

La **computación en nube**, es una tecnología que permite ofrecer servicios de computación a través de Internet. En este tipo de computación todo lo que puede ofrecer un sistema informático se ofrece como servicio, de modo que los usuarios puedan acceder a los servicios disponibles "en la nube de Internet" sin conocimientos (o, al menos sin ser expertos) en la gestión de los recursos que usan. Según el IEEE Computer Society es un paradigma en el que la información se almacena de manera permanente en servidores en Internet y se envía a cachés temporales de cliente, lo que incluye equipos de sobremesa, centros de ocio, portátiles, etc. Esto se debe a que, pese las capacidades de las PC ha mejorado sustancialmente, gran parte de su potencia es desaprovechada, al ser maquinas de propósito general.



La computación en nube es un concepto general que incorpora el software como servicio, tal como la Web 2.0 y otros recientes, también conocidos como tendencias tecnológicas, donde el tema en común es la confianza en Internet para satisfacer las necesidades de cómputo de los usuarios.

Como ejemplos de computación en nube destacan Amazon EC2, Google Apps, eyeOS y Microsoft Azure que proveen aplicaciones comunes de negocios en línea accesibles desde un navegador web, mientras el software y los datos se almacenan en los servidores.



Este modelo está inspirado en la idea de disponer de infraestructuras tecnológicas de modo que los recursos informáticos sean compartidos dinámicamente, se encuentren virtualizados y resulten accesibles como un servicio. Aúna de esta manera gran parte de las nuevas tendencias de software como servicio, virtualización de recursos, redes grids e informática bajo demanda. En el modelo cloud computing, los grandes clusters de sistemas se enlazan entre sí para proporcionar servicios tecnológicos como si se tratase de un único superordenador global.

Cloud Computing está influido por las tendencias hacia la virtualización, la automatización, el procesamiento masivamente paralelo y la orientación al servicio, cloud computing surge como consecuencia de las expectativas creadas por la Web 2.0 entre los usuarios. Incluso hay voces que sostienen que la idea de que una aplicación no tenga que existir en un lugar concreto, sino que pueda estar compuesta de múltiples piezas procedentes de múltiples sitios se la debemos a la Web 2.0.

4.1 Principales Características

- ***Agilidad***. La disposición de los recursos de la infraestructura desde el punto de vista del usuario es rápida y poco costosa.
- ***Coste***. Es muy reducido y los gastos de capital se reducen a los gastos operativos.
- ***Dispositivo de localización y la independencia***. Permiten a los usuarios el acceso a los sistemas mediante un navegador web, independientemente de su ubicación o del dispositivo que estén usando, por ejemplo, PC, móviles. Se accede a través de Internet, por lo que los usuarios pueden hacerlo desde diversos lugares.
- ***Multi-tenencia***. Permite compartir los recursos y los costes entre un gran número de usuarios, lo que permite:
    - Centralización de la infraestructura en zonas con costes más bajos (como el inmobiliario, la electricidad, etc.).
    - Aumenta la capacidad de carga (no es necesario que los usuarios de máquinas de más alto nivel posible de carga).

    - Utilización y mejora de la eficiencia de los sistemas que a menudo se utilizan sólo 10-20%.

- ***Fiabilidad.*** Mejora mediante la utilización redundante de múltiples sitios, para poder realizar recuperaciones en caso de fallo. Sin embargo, la mayoría de los principales servicios de cloud computing han sufrido cortes, y los administradores de la infraestructura son capaces de hacer muy poco cuando se ven afectados.
- ***Escalabilidad*** del sistema.
- ***Seguridad*** debido a la mejora general, la centralización de datos, el aumento de la seguridad centrada en los recursos, etc. Aunque plantea dudas acerca de la pérdida de control sobre determinados datos sensibles. La seguridad es a menudo tan buena o mejor que los sistemas tradicionales, en parte porque los proveedores están en condiciones de dedicar recursos a la solución de los problemas de seguridad que muchos clientes no pueden pagar.
- ***Sostenibilidad*** se logra por medio de la mejora de la utilización de los recursos y sistemas más eficientes. Sin embargo, las computadoras y las infraestructuras asociadas son grandes consumidores de energía.

4.2 Cloud computing como modelo alternativo de adquisición y entregas

Como todas las corrientes tecnológicas transformadoras, su adopción implica cambios significativos en todos los niveles de la cadena de suministro que afectan al propio modelo de negocio, tanto de los proveedores de este tipo de ofertas como de las empresas clientes que basen su actividad total o parcialmente en la entrega de servicios TI.

Es más, ya se ha llegado a definir como: “un modelo alternativo de adquisición y entrega de servicios que cambiará las relaciones entre compradores de productos y servicios TI, así como la forma en que estos suministradores entregan sus ofertas”.

Aunque cloud computing se ha convertido en un término de moda que está siendo utilizado en formas incluso contradictorias, podemos destacar su carácter transformador, en cualquiera de sus variedades, lo que está produciendo un cambio en el entorno TI. La virtualización, la orientación al servicio e Internet han convergido para fomentar un fenómeno que permite a consumidores y negocios elegir cómo adquirir o suministrar servicios TI (...). Los servicios suministrados a través de la ‘nube’ se piensa que pueden robustecer una economía basada en la entrega y consumo de cualquier cosa, desde el almacenamiento a la computación, el vídeo o la gestión de las finanzas.

Se considera realmente innovador el cambio de mentalidad que se está produciendo para que la tecnología sea usada por la gente preocupándose sólo de lo que puede hacer con ella, no por cómo implementarla. Porque cloud computing no es una arquitectura ni una infraestructura, sino una abstracción de las relaciones entre consumidores y proveedores de servicios.

Pero al moverse todavía en sus inicios, cloud computing también tiene sus riesgos, como la falta de acuerdos de nivel de servicio (SLA) y de una base probada de casos de éxito de clientes, además de los nuevos retos de seguridad que añade, de los que los relacionados con el cumplimiento normativo podrían ser de peso en determinados sectores de actividad. Uno de ellos tiene su raíz en algo que suele suceder con los nuevos modelos: la falta de acuerdo en su propia definición. La mezcla de intereses con que las compañías del sector suelen acercarse a las tendencias que más prometen acaban por añadir confusión donde

debería primar el conocimiento. Una confusión que acaba por llevar a las empresas usuarias (carentes de las herramientas necesarias para realizar sus propios análisis de necesidades y sacar las mejores conclusiones para su negocio) a quedar en manos de los mensajes comerciales.

4.3 Ventajas

Las ventajas de cloud computing parecen evidentes, al permitir a las empresas escalar rápidamente, en función de sus necesidades, sin tener que añadir equipamiento, software ni personal. A través de la "nube" (una red pública, generalmente Internet), los clientes pueden acceder bajo demanda (siguiendo el modelo de pago por uso) a un gran número de recursos informáticos asignados dinámicamente, dotándose así de una enorme capacidad de procesamiento y almacenamiento sin necesidad de instalar máquinas localmente, lo que se traduce en considerables ahorros de todo tipo, incluso de consumo energético.

En definitiva, el modelo cloud facilita a las empresas de todo tamaño y sector focalizar sus recursos en optimizar sus procesos, liberándolas del mantenimiento, actualización y amortización de grades inversiones tecnológicas en sistemas, que con frecuencia son menos eficientes y están infrautilizados dentro de cada organización.

Si bien, de entrada, parece que cloud computing está más indicado para pequeñas y medianas empresas sin grandes entornos TI y con poca capacidad de inversión, el modelo también se está abriendo a las grandes organizaciones, ya sea para soportar determinadas aplicaciones o para apoyar proyectos concretos de duración limitada. Por su naturaleza, las compañías con entornos informáticos distribuidos serán las que más aprovechen las posibilidades de cloud computing, pero también puede aportar grandes beneficios para todo tipo de organizaciones y sectores. Administración Pública, compañías privadas, Universidades y otras instituciones ya están utilizando esta tecnología, que permite sustituir las granjas de servidores por una infraestructura distribuida y basada en estándares abiertos.

Un ejemplo de esto, son conocidos casos de éxito, como los de BitCurrent y The New York Times, que recientemente alquiló el servicio de Amazon para pasar los artículos aparecidos en el diario durante las últimas décadas a PDF susceptibles de búsquedas. Un proyecto que, de haberlo emprendido internamente, hubiera tardado 14 años en completarse y que, gracias a Amazon, pudo culminar en un solo día por nada más que 240 dólares.

Otro ejemplo que podemos destacar es el de Salesforce.com. A través de la herramienta Salesforce para Google Apps se ofrece una potente combinación de aplicaciones básicas para ampliar la productividad empresarial (correo electrónico, calendario, documentos, hojas de cálculo o mensajería instantánea) y CRM (ventas, marketing, servicios y apoyo, partners) que permite a los profesionales comunicarse y colaborar en tiempo real a través de la web. Esa colaboración puede materializarse en compartir no sólo información a través del correo electrónico, sino también en gestionar y compartir documentos online, en su comunicación instantánea o en la exposición de tareas de ventas y campañas de marketing, entre otros.

4.4 Riesgos

Las grandes expectativas con que está siendo recibido por buena parte de la industria y de los expertos se enfrentan todavía a una serie de obstáculos que habrá que ir eliminando si el objetivo es que cloud computing se convierta en una opción mayoritariamente aceptada. En estos primeros inicios de su evolución, permanecen abiertos algunos riesgos, de especial gravedad cuando se trata de confiar a la "nube" las aplicaciones críticas de negocio.

Podemos enumerar algunos de ellos:

- ***Privacidad de los datos***. Si los datos no están del todo seguros ni siquiera en los propios centros de datos corporativos, el peligro aumenta cuando se dejan en manos de la "nube". Además, hay que tener en cuenta que muchos países obligan a que los datos de sus ciudadanos sean guardados dentro de sus territorios nacionales. Sin embargo, en el modelo cloud computing los datos pueden residir en cualquier lugar, sin que el cliente sea consciente de su ubicación geográfica.
- ***Seguridad***. Como entornos heterogéneos y abiertos, se han de reforzar las medidas de seguridad contra amenazas externas y la corrupción de los datos. Es necesario que los proveedores sigan las mejores prácticas en encriptación y en seguridad física y lógica, garantizando también altos niveles de disponibilidad, accesibilidad y escalabilidad.
- ***Licencias de software***. El modelo típico de licenciamiento del software corporativo no siempre se adapta bien al mundo del cloud computing, donde una aplicación podría estar corriendo sobre un elevado número de servidores. No obstante, se piensa que esto se solucionará con un proceso de maduración que llegará por sí mismo, según evolucionen los modelos de programación actuales hacia formas más 'cooperativas'.
- ***Aplicaciones***. Para que las aplicaciones funcionen en el nuevo modelo, han de ser escritas de modo que puedan ser divididas entre múltiples servidores. Y como no todas están preparadas para ello, las empresas se ven obligadas a reescribirlas.
- ***Interoperabilidad***. Es necesario crear estándares universales que garanticen la interoperatividad entre servicios, algo que ahora no ocurre. Es importante que se creen estándares universalmente aceptados, que no se reproduzcan las fracturas que el mundo de la informática ha ido viendo generación a generación entre estándares incompatibles.
- ***Cumplimiento normativo***. ¿Qué sucede cuando los auditores quieren certificar que la empresa esta cumpliendo las regulaciones normativas y la aplicación en cuestión está corriendo en la nube? Este es un problema todavía sin solución.
- ***SLA***. Hoy no tiene mucho sentido confiar a un tercero las aplicaciones de la empresa si no se regula el cumplimiento de acuerdos de nivel de servicio (SLA) que garanticen un determinado rendimiento.
- ***Monitorización de red***. Otra cuestión que permanece sin respuesta es cómo hacer un instrumento de la empresa la red y las aplicaciones en escenarios cloud. ¿Que tipos de herramientas de monitorización de redes y aplicaciones se requerirán?

4.5 Mitigación de estos riesgos

¿Que pedir a un proveedor de storage - cloud?

- ***Medidas de seguridad*.** Los usuarios deben cifrar sus propios ficheros. No permitir que sea el proveedor quien lo haga.
- ***Acceso y auditoría de los datos*.** El acceso a los datos permite averiguar si alguien los ha modificado. Algunos proveedores como Amazon.com y Nirvanix ofrecen esta posibilidad sin coste adicional.
- ***Localización de los datos almacenados*.** En algunos casos los clientes necesitan saber la localización exacta de los datos por motivos de cumplimiento normativo. Nirvanix ofrece la opción de bloquear los datos en localizaciones físicas específicas pero Amazon no.
- ***Tiempos de caída del servicio***. Los clientes del Simple Storage Service de Amazon sufrieron durante horas una caída del servicio el pasado febrero. Aún así, los cloud services pueden ser más fiables que el propio centro de datos del cliente. Amazon y Nirvanix dan créditos a los clientes si el tiempo de actividad mensual cae por debajo del 99,9%.
- ***Recuperación de desastres*.** No hay modo de evitarlo: de vez en cuando se producen catástrofes. Por ello hay que asegurarse de que el proveedor cuenta con un plan de recuperación de desastres que permita acceder offline a los principales centros de datos.

4.6 Oportunidades de mercado

El flamante mercado de cloud computing ofrece oportunidades para un amplio grupo de proveedores y suministradores de tecnología de múltiples perfiles. Grandes consolidados de las TI convencionales y del mundo Web, así como un número creciente de startups, empiezan a disputarse (compitiendo y aliándose) un segmento de la demanda que ve en la “nube” una opción atractiva.

El mercado de cloud computing acoge fundamentalmente dos tipos de agentes:

- ***Enablers***. Aportan las infraestructuras subyacentes del modelo, focalizándose por lo general en áreas tecnológicas como virtualización y automatización del centro de datos. Es el caso de firmas como IBM, VMware/EMC, Red Hat, Intel, Sun, Citrix o BladeLogic, entre otras.
- ***Proveedores de servicios***. Las compañías que como Amazon, Salesforce.com, Rackspace, Google o Microsoft (también decidida a explotar las posibilidades de cloud computing) ponen a disposición de los clientes sus grandes entornos de computación, creados con las plataformas e infraestructuras de los “enablers” y por lo general siguiendo el modelo SaaS.

Los pioneros en el desarrollo del cloud computing son principalmente los llamados "**hijos de la web**":

- ***Amazon***. Está añadiendo en la actualidad nuevas características a su plataforma de servicios Elastic Compute Cloud para mejorar su disponibilidad. EC2 proporciona capacidad de computación adaptable a las necesidades de los clientes, generalmente desarrolladores. Con la nueva característica Availability Zones, las aplicaciones pueden ser asignadas a múltiples localizaciones con propósitos de failover. Con Elastic IP Addresses se simplifica la gestión de direcciones, al permitir, por ejemplo, activar y desactivar servidores sin tener que cambiar los establecimientos DNS. EC2 opera en conjunción con otros servicios de Amazon, como Simple Storage Service y SimpleDB.
- ***Google***. Estrena ahora un servicio de hospedaje de aplicaciones Web de empresa en su propia infraestructura con una nueva herramienta para desarrolladores, App Engine. El objetivo de Google es simplificar el despliegue de una nueva aplicación y facilitar su escalado cuando sea preciso.
- ***App Engine***. Está basado en tecnología ya utilizada por Google, como Bigtable, un sistema de almacenamiento distribuido que forma la base de Google Earth, y su propio sistema de ficheros GFS. En la versión lanzada en abril (que como release previa no contiene todas las características del futuro servicio) sólo trabajan 10.000 desarrolladores, pero está previsto ir aumentando ese número. También hasta su puesta en marcha operativa, la capacidad estará limitada a 500 MB de almacenamiento y una transmisión de hasta 10 Gbps por día y aplicación.
- ***Salesforce.com***. Confía en replicar el éxito de su oferta SaaS en el modelo cloud computing ofreciendo una plataforma de desarrollo en la "nube". Se trata de Force.com. dirigida a empresas que no quieran o no puedan invertir tiempo y dinero en infraestructuras de software internas. La compañía ha anunciado además la integración de las aplicaciones online de productividad de Google en su oferta de soluciones de gestión con los clientes (CRM) bajo demanda.

4.7 Desarrollo de Cloud Computing en Europa y en España

Liderado por IBM, a través de su laboratorio de investigación de Haifa, en Reservoir participan un total de 13 socios europeos, entre ellos Telefónica Investigación y Desarrollo y el Grupo de Arquitectura de Sistemas Distribuidos de la Universidad Complutense de Madrid, junto a otras entidades como SAP Research, Sun Microsystems, Thales y universidades de Umea, Londres, Lugano y Messina.

El objetivo de Reservoir durante los próximos 3 años es desarrollar la tecnología necesaria para desplegar soluciones de infraestructura que proporcionen servicios bajo demanda, a un precio competitivo y asegurando calidad de servicio. Esta tecnología será decisiva para los centros de datos del futuro, virtualizando la infraestructura que soporta los servicios y permitiendo que varios centros de datos compartan su infraestructura con el fin de satisfacer demandas puntuales que excedan la capacidad local. En otras palabras, una empresa podrá acceder a servicios o infraestructura adicional por medio de la tecnología Reservoir en alguno de los sitios del cloud. Si existe una demanda puntual para un servicio hospedado en un centro de datos, éste podrá alquilar dinámicamente servicios adicionales. El proyecto incluye la validación de la nueva tecnología en escenarios de la administración pública, procesos de negocio, computación bajo demanda y telecomunicaciones. El proyecto

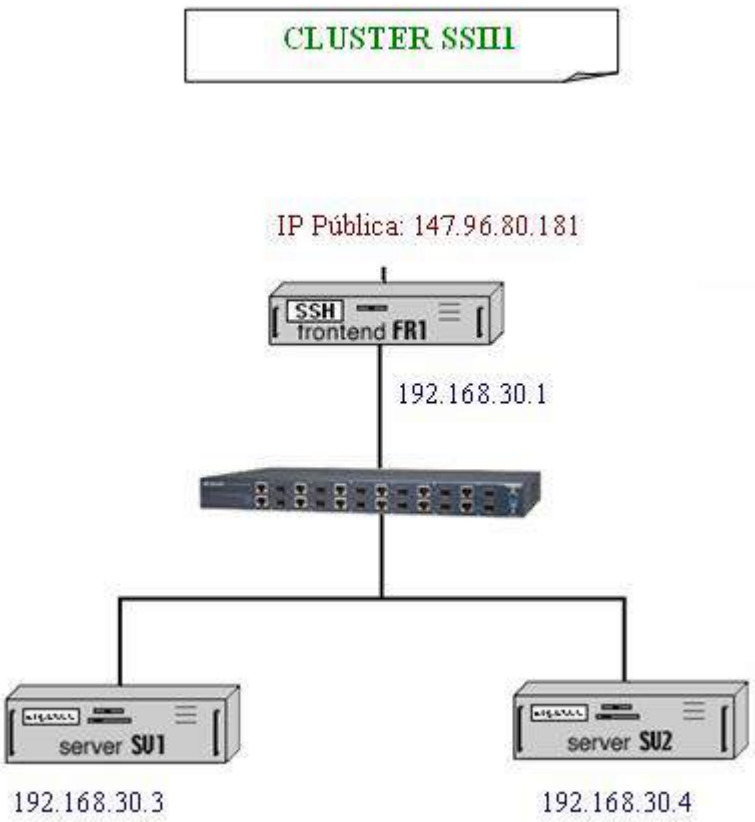
está organizado en tres actividades tecnológicas principales: gestión de servicios, gestión de entornos de ejecución virtuales e infraestructura de virtualización.

En el modelo cloud de Reservoir se emplea el motor de gestión de virtualización OpenNEbula, una tecnología de código abierto desarrollada por el Grupo de Arquitectura de Sistemas Distribuidos de la Universidad Complutense de Madrid, que está siendo adoptada en numerosas soluciones cloud y datacenter, y con el que trabajaremos para implementar el proyecto.

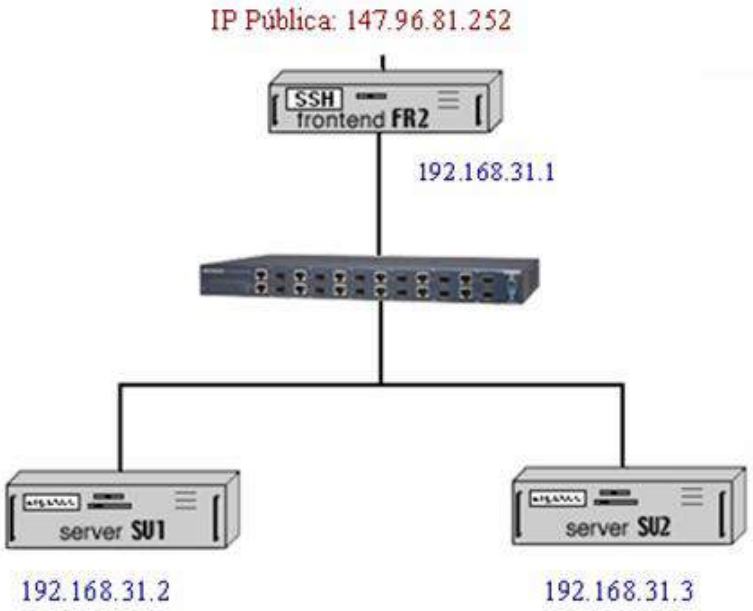
5 Implementación del proyecto

Arquitectura de red física:

Para la implantación del proyecto, usaremos dos clusters, el SSII1 y el SSII2, con acceso a Internet por medio de las IP 147.96.80.181 y 147.96.81.251 respectivamente. Los clusters estarán compuestos por el nodo principal (front-end), el cual es el único que tiene acceso directo a Internet, y por dos nodos secundarios (back-end). Todas las comunicaciones de datos las haremos a través del protocolo SSH.



CLUSTER SSH2



Como plataforma para la virtualización de máquinas virtuales, usaremos **Xen**. A la hora de instalar y configurar la plataforma Xen, hemos de diferenciar entre el Sistema Operativo del host (dom0), que será aquel que use el hypervisor, y el de los Sistemas Operativos que utilicemos en las máquinas virtuales (domU).

En nuestro proyecto, principalmente lanzamos imágenes para nuestras máquinas virtuales (domU) de Ubuntu 8.04 y de ttylinux (una imagen de un Linux muy ligero, apenas 5 MB, pero sin apenas funcionalidad).

Instalar Xen

Para instalar Xen y todas las dependencias, todo lo que tienes que hacer es ejecutar el siguiente comando:

```
ubuntu@cluster01:~$ apt-get install ubuntu-xen-server
```

Si la instalación la estuviésemos haciendo sobre un ordenador de sobremesa en lugar de un servidor (el cual no necesita tantos componentes gráficos), ejecutaríamos el comando (el resto de la instalación es más o menos igual):

```
ubuntu@cluster01:~$ install ubuntu-xen-desktop
```

Todo esto nos instalará todo el paquete de xen-tools con todas las herramientas necesarias para lanzar nuestras máquinas virtuales. (En el caso de la versión desktop, si lo deseamos podemos instalar también el paquete Xenman que nos ayudará a manejar más facilmente dichas MV)

Lo siguiente será efectuar:

```
ubuntu@cluster01:~$ mv /lib/tls /lib/tls.disabled
```

También necesitaremos añadir el modulo *loop* a el kernel para cada vez que lancemos nuestro sistema. Con lo que abrimos para editar el archivo /etc/modules y añadimos la siguiente línea (en caso de que no exista ya):

```
ubuntu@cluster01:~$ vim /etc/modules
```

```
[...]
loop max_loop = 64
[...]
```

Ahora puedes ver en el directorio /boot qué kernels y ramdisks están instalados en el sistema:

```
ubuntu@cluster01:~$ ls –l /boot/
```

Ahí han de aparecer tanto el kernel como el ramdisk que hemos modificados, los podremos observar al tener al final de su nombre la extensión "xen", por ejemplo:

`-/boot/vmlinuz-2.6.X-Y-xen` sería el nuevo kernel modificado, y `/boot/initrd.img-2.6.X-Y-xen` su ramdisk.

Para ver qué distribuciones de Linux podremos instalar con xen-tools (el método más fácil de crear y lanzar máquinas virtuales, el paquete viene por defecto al instalar el ubuntu-xen-server), ejecutamos el siguiente comando:

```
ubuntu@cluster01:~$ ls –l /usr/lib/xen-tools
```

Lo siguiente será crear el directorio donde almacenaremos nuestras MV, por ejemplo en /home/xen (más tarde, al instalar el Open Nebula, ya no usaremos ninguno de estos directorios...):

```
ubuntu@cluster01:~$ mkdir /home/xen
```

A continuación usaremos xen-tools para crear las máquinas virtuales (es la forma más sencilla de hacerlo y comprobar que a funcionado todo, pero existen muchas otras opciones).

Para ello, usaremos el comando xen-create-image, el cual nos creará en solo unos instantes un disco de imagen para nuestra máquina virtual, no obstante, primero hemos de configurar el archivo `/etc/xen-tools/xen-tools.conf` con los parámetros que queremos que nos lance por defecto dicho comando (con la configuración inicial de las máquinas virtuales).

Todo lo que no aparezca en el comando xen-create-image se sacará de dicho fichero.

```
ubuntu@cluster01:~$ vim /etc/xen-tools/xen-tools.conf
```

```
[...]
dir = /home/xen
[...]
dist = hardy     # default distribution to
install
[...]
gateway = 192.168.0.1
netmask = 255.255.255.0
broadcast = 192.168.0.255
[...]
passw = 1
[...]
mirror = http://archive.ubuntu.com/ubuntu
[...]
```

La línea `dist` nos indica la distribución que queremos instalar en nuestra máquina virtual. La línea `kernel` debe contener el núcleo de Xen, y el initrd su ramdisk. Por defecto, dicho archivo contiene los valores `kernel = /boot/vmlinuz-`uname -r`` y `initrd = /boot/initrd.img-`uname -r`` que automáticamente se traduce a lo que tenemos en nuestro sistema, con lo que no es necesario modificar dicha línea.

La línea `passwd = 1` te permite especificar una root password, por último, además de los típicos parámetros de red (sin los cuales nuestras MV no tendrían conexión), la línea de `mirror` indica la URL de la que nos bajaremos las distintas imágenes.

Tras esto, reiniciamos:

```
ubuntu@cluster01:~$ reboot
```

Podemos comprobar (tras seleccionar correctamente en el GRUB la versión de xen), que el kernel es correcto:

```
ubuntu@cluster01:~$ uname -r
```

Ahora, por tanto, podemos crear nuestro primer dominio, xen1.example.com. Podremos indicar todos los parámetros que queramos, utilizando los de `/etc/xen-tools/xen-tools.conf` en los casos en los que no estén especificados:

```
ubuntu@cluster01:~$ xen-create-image –-hostname=xen1.example.com –-
size=2Gb –-swap=256Mb --ide \ --ip=192.168.0.101 --
netmask=255.255.255.0 --gateway=192.168.0.1 --force \ --dir=/home/ --
install-method=debootstrap --dist=hardy \ --
mirror=http://archive.ubuntu.com/ubuntu/ --passwd
```

Tras un rato (se ha de bajar y configurar la imagen), ya tendremos el archivo de configuración en `/etc/xen/xen1.example.com.cfg`. (Con los datos especificados en las dos operaciones anteriores).

NOTA: Cuando ejecutamos estos pasos en la versión de escritorio, tuvimos que reemplazar los parámetros`" file: " por "tap:aio:"`, de otra forma no nos funcionaba.

Por último, para crear la máquina virtual, simplemente hemos de ejecutar:

```
ubuntu@cluster01:~$ xm /create /etc/xen/xen1.example.com.cfg
```

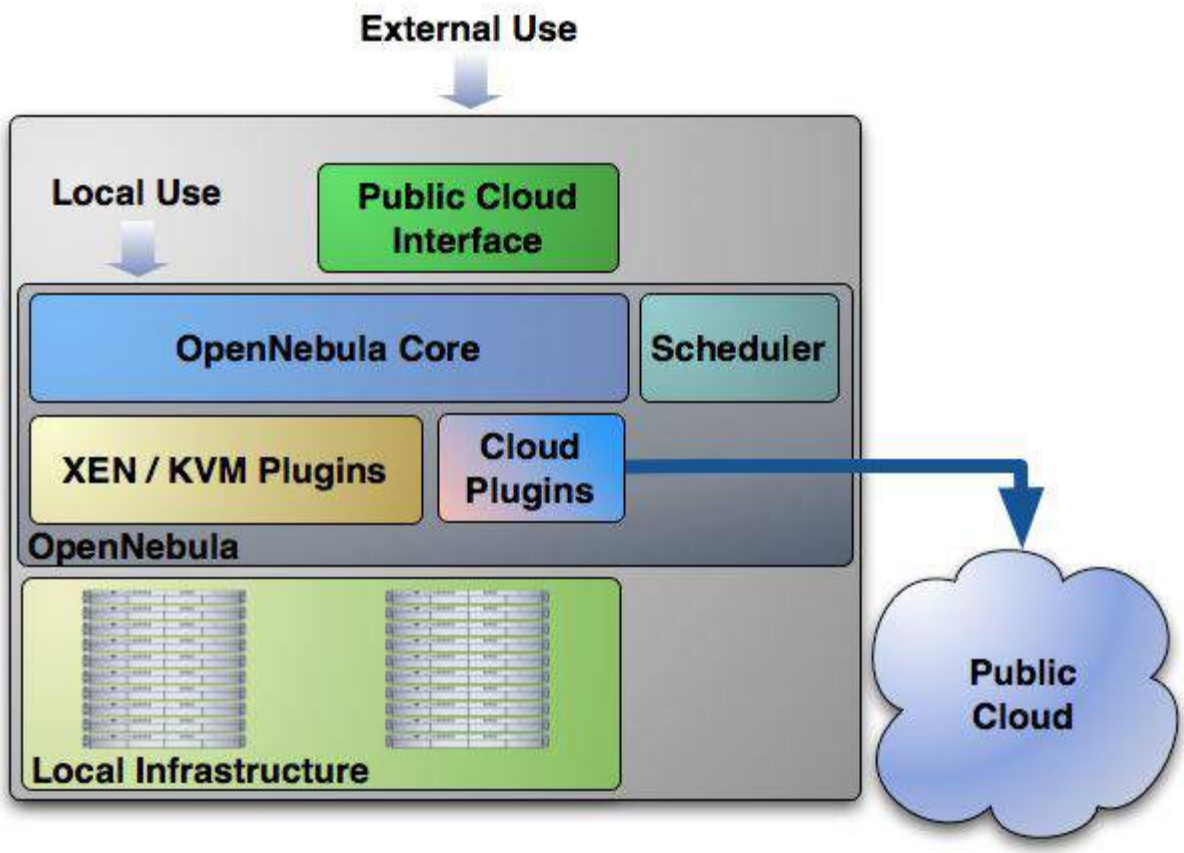
Y para acceder a ella:

```
ubuntu@cluster01:~$ xm console xen1.example.com
```

7 OpenNEbula

OpenNEbula (ONE), de código abierto, es un motor de infraestructura virtual que permite el despliegue y la recolocación dinámica de los servicios virtualizados (grupos de máquinas virtuales) en el mismo o en sitios diferentes. OpenNEbula extiende los beneficios de la virtualización de plataformas de un sólo recurso físico a un conjunto de recursos. Esta disociación no afecta sólo al servidor de la infraestructura física, sino también a la propia ubicación física.

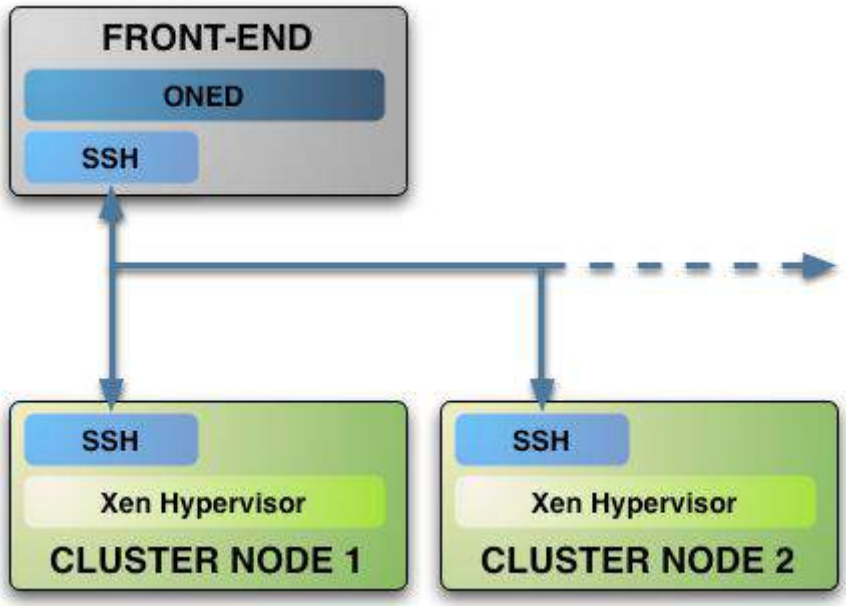
OpenNEbula puede ser utilizado principalmente como una herramienta de virtualización para gestionar la infraestructura virtual del centro de datos o de grupo. Esta aplicación se suele denominar **nube privada**. Además OpenNEbula es escalable dinámicamente, se puede multiplicar el número de "clouds" externos, conformando un cloud **híbrido**. Cuando se combina con una interfaz de "clouds", OpenNEbula puede ser utilizado como motor para el manejo de los mismos, con una gestión dinámica y escalable de la infraestructura de back-end.



7.1 Características principales de OpenNEbula

En esta sección nos vamos a centrar principalmente, en los pasos a seguir para lograr una correcta utilización de ONE en el cluster SSII1 (la configuración del cluster SSII2 será análoga). Consiste en un front-end y dos nodos en los que instalaremos OpenNEbula y lo configuraremos para gestionar las máquinas virtuales que colgarán de los nodos del mencionado grupo.

A continuación vemos una imagen en la que el front-end es la máquina donde está instalado OpenNEbula. ONE soporta hipervisores Xen y KVM para la gestión del ciclo de vida de las máquinas virtuales a través de ellos. Aunque en esta parte nos vamos a centrar en el hipervisor de Xen que es el que hemos utilizado para nuestro proyecto.



Requisitos

En cuanto al HARDWARE vamos a utilizar un grupo formado por 3 equipos:

- **Front-end de Cluster:** Este va a ser el servidor de OpenNEbula. Podemos llamarlo *cluster01* y asumir que tiene una IP de 192.168.30.1.
- **Los nodos del clúster:** De los otros dos equipos colgará la ejecución de máquinas virtuales. Los llamaremos *cluster02* y *cluster03* a lo largo de esta explicación y tendrán direcciones IP, 192.168.30.3 y 192.168.30.4, respectivamente.

En lo referente al SOFTWARE deberíamos configurar tanto XEN como NIS correctamente, además de la instalación de "OpenNEbula tarball".

7.2 Configuración del sistema

Configuración de NIS

Vamos a crear un usuario común y un grupo para las tres máquinas, una de las maneras más fáciles es utilizar el NIS. Suponiendo *cluster01* es el servidor NIS, inicie sesión en ella y escriba:

```
ubuntu@cluster01:~$ groupadd xen
ubuntu@cluster01:~$ useradd -G xen oneadmin
ubuntu@cluster01:~$ cd /var/yp
ubuntu@cluster01:~$ make
```

Tenemos que saber el ID del grupo al que pertenece. Y después el GIDs del <oneadmin>:

```
ubuntu@cluster01:~$ id oneadmin
```

Ahora tenemos que crear los grupos locales (vamos a llamarlos rootxen grupo) en *cluster02* y *cluster03* que incluye sus raíces y compartir con el GID xen grupo, por lo que el grupo local comparte la xen grupo de privilegios.

```
ubuntu@cluster02:~$ echo "rootxen:x:<xen_gid>:root >> /etc/group
```

Cambiamos en el comando anterior <xen_gid> para el número correspondiente. Repetimos este proceso para *cluster03*.

Configuración de SSH

La cuenta <oneadmin> tiene que ser de confianza en los nodos del servidor de OpenNEbula, pudiendo acceder a ellos con una contraseña:

```
ubuntu@cluster01:~$  ssh-keygen
```

Ahora tenemos que decirle a todos los nodos que tienen que confiar en esta clave. Así que tenemos que copiar el creado id_rsa.pub en todos los nodos y, a continuación, añadir una confianza como clave pública. Repetiremos esto para todos los nodos del clúster:

```
ubuntu@cluster01:~$  scp id_rsa.pub cluster02:
```

Y, a continuación, iniciar sesión en el nodo de clúster

```
ubuntu@cluster02:~$  cd ~/.ssh
ubuntu@cluster02:~$  cat id_rsa.pub >> authorized_keys
```

Ahora se puede utilizar ssh con la cuenta <oneadmin> de *cluster01* a uno de los nodos del clúster, se podrá acceder a una sesión sin tener que escribir una contraseña.

Imágenes para las máquinas virtuales

Asumimos que el grupo de front-end es también el repositorio de imágenes. Vamos a crear una carpeta para las imágenes:

```
ubuntu@cluster01:~$  mkdir /opt/nebula/images
```

Tanto las imágenes como las carpetas tienen que ser leídas desde la cuenta <oneadmin>. OpenNEbula utilizará la imagen que corresponda para el nodo elegido y su ejecución segura mediante copia scp.

NOTA: Nosotros realmente usamos el directorio /images/xen para almacenar las imágenes y compartirlas con el resto de nodos del cluster.

Configuración de la red

Asumimos que los nodos del clúster están configurados para permitir la creación de redes para reducir el número de máquinas virtuales. Esto significa tener un puente software en los nodos, ligado a su interfaz de red física. Vamos a suponer que este puente se llama eth0.

7.3 Instalación de OpenNEbula

Utilizando la cuenta **oneadmin** descargamos el tarball OpenNEbula y lo descomprimimos en la carpeta de inicio. Cambiamos a la carpeta recién creada y escribimos:

```
ubuntu@cluster01:~$  scons
```

Vamos a instalar en OpenNEbula en modo autónomo. Una vez que la compilación termina con éxito, permitirá instalarlo en la carpeta de destino:

```
ubuntu@cluster01:~$ ./install.sh -d /opt/nebula/ONE
```

Ahora vamos a configurar el entorno:

```
ubuntu@cluster01:~$ export ONE_LOCATION=/opt/nebula/ONE/
ubuntu@cluster01:~$export ONE_XMLRPC=http://localhost:2633/RPC2
ubuntu@cluster01:~$  export PATH=$ONE_LOCATION/bin:$PATH
```

Ahora es el momento de iniciar el demonio OpenNEbula y el planificador. Escribimos lo siguiente en *cluster01*:

```
ubuntu@cluster01:~$ $ONE_LOCATION/bin/one start
```

Si se recibe un mensaje de que el planificador ha arrancado, entonces en que OpenNEbula está instalado y corriendo.

7.4 Configuración del grupo en OpenNEbula

OpenNEbula necesita saber cómo acceder y usar los nodos del clúster, por lo que permite establecer el grupo en OpenNEbula. Lo primero que hay que hacer es añadir los hosts. Esto puede hacerse por medio de la comando onehost. Por lo tanto, permite añadir tanto *cluster02* y *cluster03*:

```
ubuntu@cluster01:~$ onehost create cluster02 im_xen vmm_xen tm_ssh
ubuntu@cluster01:~$ onehost create cluster03 im_xen vmm_xen tm_ssh
```

Los 3 parámetros que se le pasa a la llamada: im_xen, vmm_xen y tm_ssh; son la información del controlador (control de los nodos del clúster físico), el conductor de virtualización (interfaz de los nodos del clúster) y la transferencia del conductor (para mover las imágenes entre los nodos del clúster), respectivamente. Estos controladores se han configurado en el $ONE_LOCATION/etc/oned.conf.

El controlador de la información puede extraer información recogida en XEN, un conductor de virtualización entiende cómo implementar el uso de máquinas virtuales XEN y por último, la transferencia del controlador que es capaz de transferir imágenes mediante una copia segura scp.

7.5 Uso de Open Nebula

Para probar si todo está funcionando podemos seguir los siguientes pasos. Lo primero que hay que hacer, es verificar que se han añadido los hosts sin problemas. Ejecutamos el siguiente comando como <oneadmin> y comprobamos los resultados:

```
ubuntu@cluster01:~$  onehost list


 HID NAME     RVM TCPU FCPU ACPU   TMEM    FMEM   STAT
 0 cluster02  0   800  800  800  8194468 7867604  on
 1 cluster03  0   800  797  800  8387584 1438720  on
```

Vamos a construir la plantilla para poder lanzar la máquina virtual a partir de una imagen de ubuntu 8.04 (en la que instalaremos los servidores web), para ello usaremos el siguiente archivo de configuración:

NOTA: Siguiendo la instalación habitual, el directorio donde se guardarían las imágenes sería en /opt/nebula en lugar de /images/xen./images

```
### VM definition template ###

NAME=ubuntu.8-04

CPU=0.5
MEMORY=128

OS = [
        kernel="/images/xen/vmlinuz-2.6.18-5-xen-686",
        initrd="/images/xen/initrd.img-2.6.18-5-xen-686",
        root="sda1",
        boot="sda1"
]

DISK = [
        source="/images/xen/imagenUbuntu/ubuntu.8-04.img",
        target="/dev/sda1",
        readonly="no"
]

RAW = [
    type="xen",
    data='on_poweroff="destroy"'
]

# --- 1 NIC ---

NIC = [ mac="00:ff:72:17:20:27"]

# --- VNC server ---

#GRAPHICS = [
#  type    = "vnc",
#  listen  = "127.0.0.1",
#  port    = "5"]
```

Lo guardaremos en el archivo XenGuest.one.

Podemos añadir más parámetros, pero estos son los más importantes. De esta plantilla, cabe señalar que no es necesario tener una imagen de intercambio en el repositorio de imágenes, pero no se puede crear ninguna máquina virtual si OpenNEbula no se ha arrancado previamente.

Una vez que tengamos los requisitos adaptados a nuestras necesidades (en especial, los campos de CPU y memoria), asegurándose de que *encajan* en la máquina virtual, permite presentar la VM (suponiendo que se encuentran actualmente en la carpeta de inicio):

```
ubuntu@cluster01:~$ onevm submit XenGuest.one
```

Esto debería volver con un ID, que podemos usar para identificar la máquina virtual, una vez más a través del uso de la orden onevm:

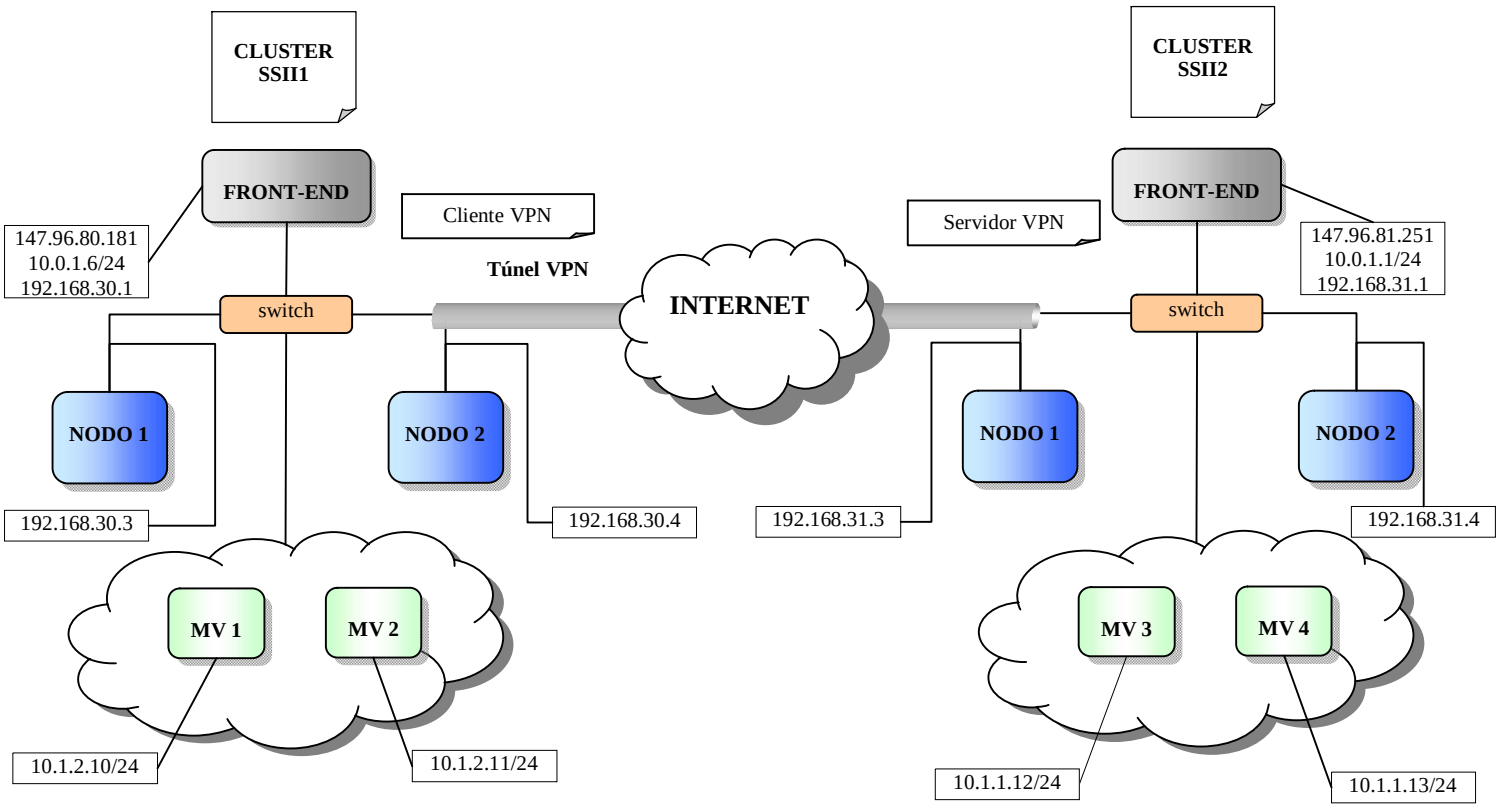
```
ubuntu@cluster01:~$ onevm list
```

La salida debería ser parecida a:

```
ID NAME STAT CPU MEM HOSTNAME  TIME
0 one-0 runn 0 65536 cluster02 00 0:00:02
```

El campo **STAT** indica el estado de la máquina virtual. Si hay el estado es **runn**, la máquina virtual está en marcha y funcionando. Dependiendo de cómo se configure su imagen, puede mostrar la dirección IP. Si ese es el caso se puede intentar acceder a la máquina virtual. Podemos mantener la conexión en otro terminal para poder comprobar en directo la migración. Esta migración debe producirse sin problemas.

8 Configuración de la red



8.1 CLUSTER SSII1

FRONT END

Interfaces de red

- **eth0**: Este interfaz contiene una dirección pública y acceso a Internet.

```
Link encap:Ethernet  HWaddr 00:02:93:60:50:b5
inet addr:147.96.80.181  Bcast:147.96.80.255  Mask:255.255.255.0
inet6 addr: fe80::202:93ff:fe60:50b5/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

- **eth1:** Interfaz de red que accede a la red local del cluster. En esta red se encuentran los nodos que contienen las máquinas virtuales.

```
Link encap:Ethernet  HWaddr 00:02:c0:a8:1e:01
inet addr:192.168.30.1  Bcast:192.168.30.255  Mask:255.255.255.0
inet6 addr: fe80::202:c0ff:fea8:1e01/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

- **tun01**: Interfaz que accede al túnel de OpenVPN. En este cluster actuará como cliente.

```
Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-
00-00
inet addr:10.1.1.6  P-t-P:10.1.1.5  Mask:255.255.255.255
UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
```

- **lo**:

```
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

Tabla de rutas

La tabla de rutas del front end se detalla a continuación:

```
ubuntu@cluster01:~$ route -n
Kernel IP routing table
Destination  Gateway     Genmask         Flags Metric Ref Use Iface
10.1.1.5     0.0.0.0   255.255.255.255  UH     0     0   0   tun0
147.96.80.0  0.0.0.0   255.255.255.0    U      0     0   0   eth0
192.168.30.0 0.0.0.0   255.255.255.0    U      0     0   0   eth1
10.1.1.0     10.1.1.5  255.255.255.0    UG     0     0   0   tun0
10.1.2.0     0.0.0.0   255.255.255.0    U      0     0   0   eth1
0.0.0.0    147.96.80.1    0.0.0.0       UG    100    0   0   eth0
```

NODO 1

Interfaces de red

- **eth1:** Interfaz de red que accede a la red local del cluster.

```
Link encap:Ethernet  HWaddr 00:02:c0:a8:1e:03
inet addr:192.168.30.3  Bcast:192.168.30.255  Mask:255.255.255.0
inet6 addr: fe80::202:c0ff:fea8:1e03/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

- **peth0**:

```
Link encap:Ethernet  HWaddr 00:02:c0:a8:1e:03
inet6 addr: fe80::202:c0ff:fea8:1e03/64 Scope:Link
UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
```

- **vif12.0**:

```
Link encap:Ethernet  HWaddr fe:ff:ff:ff:ff:ff
inet6 addr: fe80::fcff:ffff:feff:ffff/64 Scope:Link
UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
```

- **vif14.0**:

```
Link encap:Ethernet  HWaddr fe:ff:ff:ff:ff:ff
inet6 addr: fe80::fcff:ffff:feff:ffff/64 Scope:Link
UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
```

- **lo**:

```
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

Tabla de rutas

La tabla de rutas del nodo 1 se detalla a continuación:

```
ubuntu@cluster02:~$ route -n
Kernel IP routing table
Destination    Gateway      Genmask      Flags Metric Ref Use Iface
192.168.30.0   0.0.0.0    255.255.255.0    U     0      0   0   eth0
0.0.0.0      192.168.30.1   0.0.0.0        UG    100    0   0   eth0
```

NODO 2

El Nodo 2 del cluster SSII1 es análogo al Nodo 1 salvo que la ip del eth1 es 192.168.30.4, así que omitiré los detalles.

MV 1

Interfaces de red

- **eth0**: Interfaz de red que accede a la red 10.1.2.0

```
Link encap:Ethernet  HWaddr 00:ff:72:17:20:27
inet addr:10.1.2.10  Bcast:10.1.2.255  Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

- **lo**:

```
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

Tabla de rutas

La tabla de rutas del nodo 1 se detalla a continuación:

```
ubuntu@mv01:~$ route -n
Kernel IP routing table
Destination  Gateway     Genmask         Flags Metric Ref Use Iface
10.1.2.0     0.0.0.0    255.255.255.0     U      0    0   0   eth0
0.0.0.0      10.1.2.1   0.0.0.0           UG     0    0   0   eth0
```

MV 2

La configuración de la máquina virtual 2 es similar a la 1 salvo que su ip es 10.1.2.11

8.2 CLUSTER SSII2

FRONT END

Interfaces De red

- **eth0**: Este interfaz contiene una dirección pública y acceso a Internet.

```
Link encap:Ethernet  HWaddr 00:02:93:60:51:fb
inet addr:147.96.81.251  Bcast:147.96.81.255  Mask:255.255.255.0
inet6 addr: fe80::202:93ff:fe60:51fb/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

- **eth1:** Interfaz de red que accede a la red local del cluster. En esta red se encuentran los nodos que contienen las máquinas virtuales.

```
Link encap:Ethernet  HWaddr 00:02:c0:a8:1f:01
inet addr:192.168.31.1  Bcast:192.168.31.255  Mask:255.255.255.0
inet6 addr: fe80::202:c0ff:fea8:1f01/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

- **tun0**: Interfaz que accede al túnel que OpenVPN. En este cluster actuará como servidor.

```
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

- **lo**:

```
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

Tabla de rutas

La tabla de rutas del front end se detalla a continuación:

```
ubuntu@cluster01:~$ route -n
Kernel IP routing table
Destination  Gateway      Genmask      Flags Metric Ref Use  Iface
10.1.1.6     10.1.1.2  255.255.255.255  UGH    0     0   0    tun0
10.1.1.2     0.0.0.0   255.255.255.255  UH     0     0   0    tun0
147.96.81.0  0.0.0.0   255.255.255.0    U      0     0   0    eth0
192.168.31.0 0.0.0.0   255.255.255.0    U      0     0   0    eth1
10.1.1.0     0.0.0.0   255.255.255.0    U      0     0   0    eth1
10.1.1.0     0.0.0.0   255.255.255.0    U      0     0   0    eth1
10.1.2.0     10.1.1.2  255.255.255.0    UG     0     0   0    tun0
0.0.0.0    147.96.81.1    0.0.0.0       UG     100   0   0    eth0
```

Los Nodos 1 y 2 son análogos al Nodo 1 del cluster SSII1 salvo que la ip de eth1 es 192.168.31.3 y 192.168.31.4 respectivamente.

La configuración de las máquinas virtuales 1 y 2 son similares a las máquinas virtuales del cluster SSII1 salvo que la red a la que pertenecen es la 10.1.1.0

9 OpenVPN

OpenVPN es una solución de conectividad basada en software: SSL (Secure Sockets Layer) VPN Virtual Private Network (red virtual privada), OpenVPN ofrece conectividad punto-a-punto con validación jerárquica de usuarios y host conectados remotamente, resulta una muy buena opción en tecnologías Wi-Fi (redes inalámbricas EEI 802.11) y soporta una amplia configuración, entre ellas balanceo de cargas entre otras. Está publicado bajo la licencia GPL, de software libre.

9.1 Introducción

OpenVPN, es un excelente producto de software creado por James Yonan en el año 2001 y que ha estado siendo mejorado desde entonces.

Ninguna otra solución ofrece una mezcla semejante de seguridad a nivel empresarial, seguridad, facilidad de uso y riqueza de características.

Es una solución multiplataforma que ha simplificado mucho la configuración de VPN's dejando atrás los tiempos de otras soluciones difíciles de configurar como IPsec y haciéndola más accesible para gente inexperta en este tipo de tecnología.

Supongamos que necesitamos comunicar diferentes sucursales de una organización. A continuación veremos algunas soluciones que se han ofrecido como respuesta a este tipo de necesidades.

En el pasado las comunicaciones se realizaban por correo, teléfono o fax. Hoy en día hay factores que hacen necesaria la implementación de soluciones más sofisticadas de conectividad entre las oficinas de las organizaciones a lo largo del mundo.

Dichos factores son:

La aceleración de los procesos de negocios y su consecuente aumento en la necesidad de intercambio flexible y rápido de información.

Muchas organizaciones tienen varias sucursales en diferentes ubicaciones así como también tele trabajadores remotos desde sus casas, quienes necesitan intercambiar información sin ninguna demora, como si estuvieran físicamente juntos.

La necesidad de las redes de computación de cumplir altos estándares de seguridad que aseguren la autenticidad, integridad y disponibilidad.

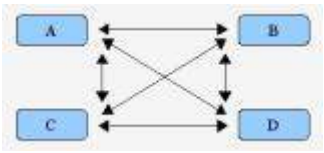
Líneas dedicadas

Las necesidades antes mencionadas se satisfacían en principio colocando líneas dedicadas entre las diferentes ubicaciones remotas a un costo mucho mayor que el de simple acceso a Internet. Se necesitaban conexiones físicas reales necesitando de un proveedor en cada sitio resultando en una solo línea de comunicación entre dos partes.

Por ejemplo, para una red de 4 nodos en la cual se buscase comunicación de todos con todos, habría que tender 6 líneas de comunicación.

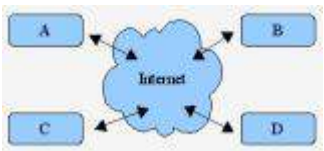
Además, para dar conectividad a trabajadores domésticos o viajeros se implementaban servicios RAS1 para aquellos que necesitaban conectarse

temporalmente mediante conexiones de módem o líneas ISDN2 donde la organización se comportaba como un proveedor de Internet (ISP).



Acceso mediante Internet y VPNs

Con la llegada de Internet y la baja de costos en conectividad se desarrollaron nuevas tecnologías. Surgió entonces la idea de utilizar a Internet como medio de comunicación entre los diferentes sitios de la organización. Surge así la idea de las VPN's que son "Virtuales" y "Privadas". Virtuales porque no son redes directas reales entre partes, sino solo conexiones virtuales provistas mediante software sobre la red Internet. Además son privadas porque solo la gente debidamente autorizada puede leer los datos transferidos por este tipo de red logrando la seguridad mediante la utilización de modernos mecanismos de criptografía. Retomando el ejemplo anterior de una organización con cuatro sitios, ahora solo necesitamos cuatro conexiones a Internet en lugar de las seis dedicadas de antes. Además los que se conectan temporalmente, también lo hacen mediante una conexión a Internet, mucho más barata y accesible desde muchos lugares, como por ejemplo de cybers cafés.



Usos de las VPN's

Las VPN's se usan generalmente para:

- Conexión entre diversos puntos de una organización a través de Internet
- Conexiones de trabajadores domésticos o de campo con IP's dinámicas
- Soluciones extranet para clientes u organizaciones asociadas con los cuales se necesita intercambiar cierta información en forma privada pero no se les debe dar acceso al resto de la red interna.

Además brinda una excelente fiabilidad en la comunicación de usuarios móviles así como también al unir dos puntos distantes como agencias de una empresa dentro de una sola red unificada.

9.2 Implementación de VPN

Supongamos que se tienen dos sitios de una organización conectados a Internet. En ambos se contará con un equipo de conexión a la red de redes que cumplirá la función de ruteo hacia y desde Internet así como firewall para protegerse de accesos no autorizados. El software VPN debe estar instalado en ese firewall o algún dispositivo

protegido por él. Uno de los sitios será el "servidor" y será el sitio que contiene la información y sistemas que queremos compartir, mientras que al otro lo llamaremos "cliente". El servidor será entonces configurado para aceptar conexiones desde el cliente (y viceversa). Llegado este punto habremos logrado tener dos sitios comunicados como en una red directa real pero aún no es una VPN dado que falta implementar la "privacidad", pues cualquier nodo intermedio de Internet puede leer la información que viaja sin protección. Lo que se debe hacer seguidamente es establecer mecanismos de cifrado que mediante uso de claves aseguren que solo equipos o personas dueños de esas claves puedan acceder a los datos enviados por la VPN. Todos los datos enviados del punto A al B deberán ser cifrados antes de ser enviados y descifrados en el otro extremo para posteriormente ser entregados normalmente a su destinatario final. Uno de los factores que diferencian a una implementación de VPN de otra, son los mecanismos que utilicen para cifrar y distribuir claves a todos los integrantes de dicha red.

9.3 Protocolos

Las soluciones de VPN pueden ser implementadas a diferentes niveles del modelo OSI de red.

Implementaciones de capa 2 – Enlace

El encapsulamiento a este nivel ofrece ciertas ventajas ya que permite transferencias sobre protocolos no-IP, como por ejemplo IPX4 de Netware Systems. Teóricamente, las tecnologías implementadas en capa 2 pueden tunelizar cualquier tipo de paquetes y en la mayoría de los casos lo que se hace es establecer un dispositivo virtual PPP5 con el cual se establece la conexión con el otro lado del túnel.

Algunos ejemplos de estas tecnologías:

- *PPTP:* Point to Point Tunneling Protocol. Desarrollado por Microsoft, es una extensión de PPP.

Su principal desventaja es que solo puede establecer un túnel por vez entre pares.

- *L2F:* Layer 2 Forwarding. Desarrollado por la empresa Cisco principalmente, ofrece mejores posibilidades que PPTP principalmente en el uso de conexiones simultáneas.
- *L2TP:* Layer 2 Tunneling Protocol. Usado por Cisco y otras fabricantes, se ha convertido en estándar de la industria y combina las ventajas de PPTP y L2F y además eliminando las desventajas. Dado que esta solución no ofrece mecanismos de seguridad, para su uso deberá ser combinada con otros mecanismos generalmente implementados en capa 3 del modelo OSI.
- *L2Sec:* Layer 2 Security Protocol. Desarrollado para proveer una solución con seguridad, utiliza para ellos SSL/TLS aunque impone una sobrecarga bastante grande en la comunicación para lograrlo.

Implementaciones de capa 3 – Red

IPsec es la tecnología más aceptada en este punto y fue desarrollada como un estándar de seguridad de Internet en capa 3. IPsec se pude utilizar para encapsular cualquier

tráfico de capa 3 pero no el tráfico de capas inferiores, por lo que no se podrá utilizar para protocolos no-IP como IPX o mensajes de broadcast. Su principal ventaja es que puede ser usado prácticamente en cualquier plataforma existiendo una gran variedad de soluciones tanto de software como de hardware.

Existen dos métodos principales usados por IPsec:

- *Modo Tunnel*. Todos los paquetes IP son encapsulados en un nuevo paquete y enviados a través del túnel siendo desempaquetados en el otro extremo y posteriormente dirigidos a su destinatario final. En este modo, se protegen las direcciones IP de emisor y receptor así como el resto de los metadatos de los paquetes.
- *Modo Transporte*. Solo la carga útil (payload) de la sección de datos es cifrada y encapsulada. La sobrecarga entonces, es sensiblemente menor que en el caso anterior, pero se exponen los metadatos a posibles atacantes que podrán ver quien se está comunicando con quien.

Implementaciones de capa 7 – Aplicación

También es posible establecer túneles en la capa de aplicación y de hecho son ampliamente utilizados hoy en día siendo algunas aproximaciones soluciones como SSL6 y TLS7. El usuario accede a la VPN de la organización a través de un browser iniciando la conexión en un sitio web seguro (HTTPS-Secured website).

Además, existen otros productos como SSL-Explorer y otros que ofrecen una combinación de gran flexibilidad, seguridad fuerte y facilidad de configuración. La seguridad es lograda mediante cifrado del tráfico usando mecanismos SSL/TLS, los cuales han probado ser muy seguros y están siendo constantemente sometidos a mejoras y pruebas.

Implementación OpenVPN

OpenVPN es una excelente nueva solución para VPN que implementa conexiones de capa 2 o 3, usa los estándares de la industria SSL/TLS para cifrar y combina todos las características mencionadas anteriormente en las otras soluciones VPN. Su principal desventaja por el momento es que hay muy pocos fabricantes de hardware que lo integren en sus soluciones. De todos modos no hay que preocuparse siempre que contemos con un Linux en el cual podremos implementarlo sin ningún problema mediante software.

9.4 Seguridad en VPN

Para cifrar datos se usan Passwords o claves de cifrado.

OpenVPN tiene dos modos considerados seguros, uno basado en claves estáticas pre-compartidas y otro en SSL/TLS usando certificados y claves RSA.

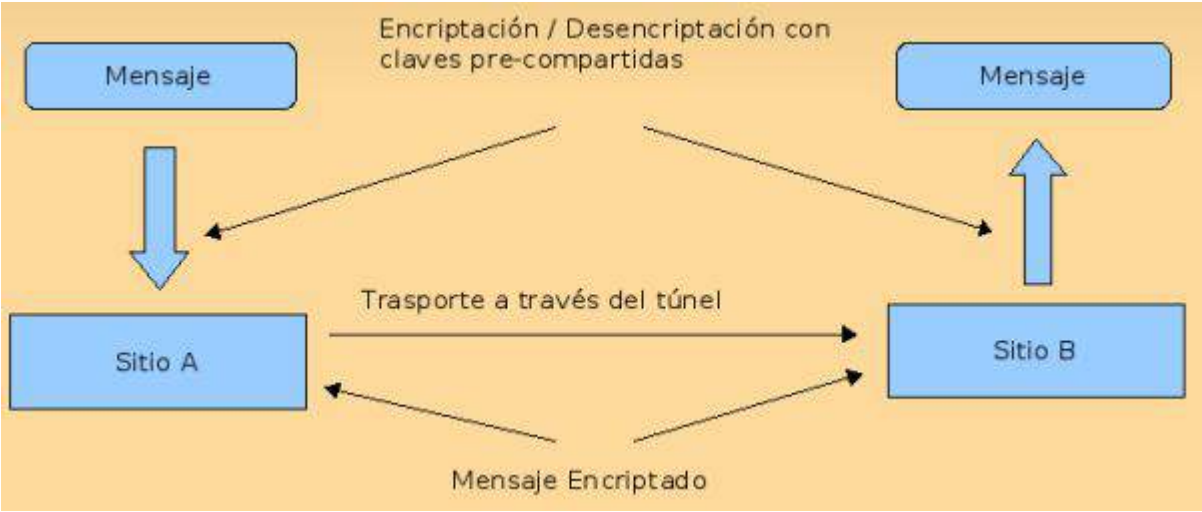
Cuando ambos lados usan la misma clave para cifrar y descifrar los datos, estamos usando el mecanismo conocido como “clave simétrica” y dicha clave debe ser instalada en todas las máquinas que tomarán parte en la conexión VPN.

Si bien SSL/TLS + claves RSA es por lejos la opción más segura, las claves estáticas cuentan con la ventaja de la simplicidad.

Veremos a continuación ese método y otros que aportan mayor seguridad y facilidad de distribución.

Cifrado simétrico y claves pre-compartidas

Cualquiera que posea la clave podrá descifrar el tráfico, por lo que si un atacante la obtuviese comprometería el tráfico completo de la organización ya que tomaría parte como un integrante más de la VPN.

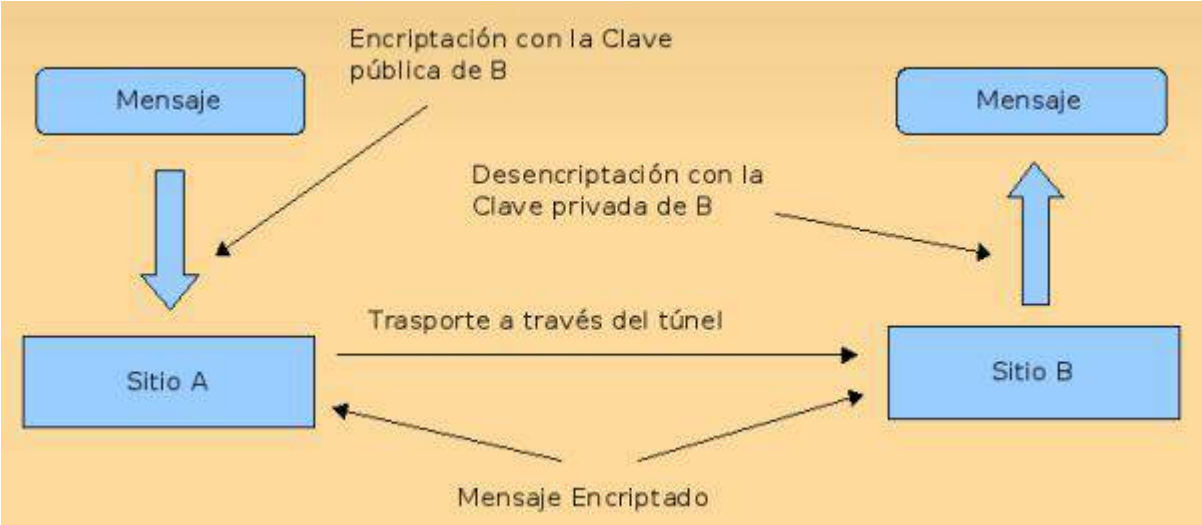


Es por ello que mecanismos como IPsec cambian las claves cada cierto período de tiempo, asociando a las mismas ciertos períodos de tiempo de validez, llamados “tiempo de vida” o “lifetime”. Una buena combinación de tiempo de vida y largo de la clave asegurarán que un atacante no pueda descifrar la clave a tiempo, haciendo que cuando finalmente la obtenga (porque lo hará), ya no le sirva por estar fuera de vigencia. IPSec utiliza su propio protocolo para intercambiar claves llamado IKE9 que ha sido desarrollado desde mediados de los noventa y aún no ha sido terminado.

Cifrado asimétrico con SSL/TLS

SSL/TLS usa una de las mejores tecnologías de cifrado para asegurar la identidad de los integrantes de la VPN. Cada integrante tiene dos claves, una pública y otra privada.

La pública es distribuida y usada por cualquiera para cifrar los datos que serán enviados a la contraparte quien conoce la clave privada que es la única que sirve para descifrar los datos. El par de clave pública/privada es generado a partir de algoritmos matemáticos que aseguran que solo con la clave privada es posible leer los datos originales. El día que alguien encuentre algún defecto a ese algoritmo, todos aquellos conectados a Internet estarán comprometidos en forma instantánea.



Es de destacar que la clave privada debe permanecer secreta mientras que la clave pública debe ser intercambiada para que nos puedan enviar mensajes.

Seguridad SSL/TLS

Las bibliotecas SSL/TLS son parte del sofware OpenSSL que vienen instaladas en cualquier sistema moderno e implementan mecanismos de cifrado y autenticación basadas en certificados. Los certificados generalmente son emitidos por entidades de reconocida confiabilidad aunque también podemos emitirlos nosotros mismos y usarlos en nuestra propia VPN. Con un certificado firmado, el dueño del mismo es capaz de demostrar su identidad a todos aquellos que confíen en la autoridad certificadora que lo emitió.

9.5 Ventajas y Desventajas de OpenVPN

Ventajas

OpenVPN provee seguridad, estabilidad y comprobados mecanismos de cifrado sin sufrir la complejidad de otras soluciones VPN como las de IPsec.

Además ofrece ventajas que van más allá que cualquier otra solución como ser:

- Posibilidad de implementar dos modos básicos, en capa 2 o capa 3, con lo que se logran túneles capaces de enviar información en otros protocolos no-IP como IPX o broadcast (NETBIOS).
- Protección de los usuarios remotos. Una vez que OpenVPN ha establecido un túnel el firewall de la organización protegerá el laptop remoto aun cuando no es un equipo de la red local. Por otra parte, solo un puerto de red podrá ser abierto hacia la red local por el remoto asegurando protección en ambos sentidos.
- Conexiones OpenVPN pueden ser realizadas a través de casi cualquier firewall. Si se posee acceso a Internet y se puede acceder a sitios HTTPS, entonces un túnel OpenVPN debería funcionar sin ningún problema.

- Soporte para proxy. Funciona a través de proxy y puede ser configurado para ejecutar como un servicio TCP o UDP y además como servidor (simplemente esperando conexiones entrantes) o como cliente (iniciando conexiones).
- Solo un puerto en el firewall debe ser abierto para permitir conexiones, dado que desde OpenVPN 2.0 se permiten múltiples conexiones en el mismo puerto TCP o UDP.
- Las interfaces virtuales (tun0, tun1, etc.) permiten la implementación de reglas de firewall muy específicas.
- Todos los conceptos de reglas, restricciones, reenvío y NAT10 pueden ser usados en túneles OpenVPN.
- Alta flexibilidad y posibilidades de extensión mediante scripting. OpenVPN ofrece numerosos puntos para ejecutar scripts individuales durante su arranque.
- Soporte transparente para IPs dinámicas. Se elimina la necesidad de usar direcciones IP estáticas en ambos lados del túnel.
- Ningún problema con NAT. Tanto los clientes como el servidor pueden estar en la red usando solamente IPs privadas.
- Instalación sencilla en cualquier plataforma. Tanto la instalación como su uso son increíblemente simples.
- Diseño modular. Se basa en un excelente diseño modular con un alto grado de simplicidad tanto en seguridad como red.

Desventajas

- No tiene compatibilidad con IPsec que justamente es el estándar actual para soluciones VPN.
- Falta de masa crítica.
- Todavía existe poca gente que conoce como usar OpenVPN.
- Al día de hoy sólo se puede conectar a otras computadoras. Pero esto está cambiando, dado que ya existen compañías desarrollando dispositivos con clientes OpenVPN integrados.

9.6 Instalación en el cluster

A continuación se detallan los pasos seguidos para la instalación del túnel entre los dos clusters SSII1 y SSII2.

SSII2

El cluster SSII2 (147.96.81.251) se eligió como servidor OpenVPN, por lo tanto el cluster SSII1 actuará como cliente.

En el servidor OpenVPN necesitamos crear una serie de claves y certificados iniciales, para poder autenticar y cifrar la información que transitará desde/hacia el servidor/clientes.

Contamos con una serie de scripts en el directorio /usr/share/doc/openvpn/examples/easy-rsa los cuales nos ayudarán mucho a ejecutar ésta tarea inicial.

El primer paso consistirá en copiar este directorio (easy-rsa) hacia /etc/openvpn.

```
ubuntu@cluster01:~$ cp -a /usr/share/doc/openvpn/examples/easy-rsa
/etc/openvpn
ubuntu@cluster01:~$ cd /etc/openvpn/easy-rsa
```

Una vez dentro de ese directorio procedemos a ejecutar los siguientes pasos:

```
ubuntu@cluster01:~$ vars
ubuntu@cluster01:~$ sh clean-all
ubuntu@cluster01:~$ sh build-ca
```

Con ellos lo que haremos es:

- Inicializar variables de ambiente para poder trabajar con los siguientes scripts de shell para generar las variables
- Inicializamos el directorio de las claves (borrando potenciales archivos viejos)
- build-ca: procedemos a generar el certificado CA

Siguiente a la generación del Certificado de autoridad, procedemos a crear el certificado del servidor y de su clave de encriptación:

```
ubuntu@cluster01:~$ sh build-key-server server
```

En éste paso, también se nos pedirá nuevamente información sobre el certificado propio del servidor.

Este paso nos generará dos archivos en el directorio */etc/openvpn/easy-rsa/keys/* que se copiarán dentro del mismo servidor hacia /etc/openvpn, ellos son:

- *server.crt*
- *server.key*

El cliente debe tener su propio certificado y clave de seguridad.

Para generar el certificado y claves privadas ejecutamos en nuestro servidor, dentro del directorio */etc/openvpn/easy-rsa/2.0*

```
ubuntu@cluster01:~$ sh build-key client1
```

El parámetro Diffie-Hellman debemos generarlo así:

```
ubuntu@cluster01:~$ sh build-dh
```

Hacia el directorio /etc/openvpn del servidor copiamos los siguientes archivos:

- ca.crt
- ca.key
- server.key
- server.crt
- dh1024.pem

Para que el servidor pueda acceder a la red 10.1.2.0 del SSII1 hay que proceder de la siguiente manera:

```
ubuntu@cluster01:~$ mkdir /etc/openvpn/easy-rsa/2.0/ccd
```

Dentro de éste directorio creamos un archivo con el nombre del cliente (client1) que contiene la siguiente línea:

```
iroute 10.1.2.0 255.255.255.0
```

Aquí lo que le indicamos al servidor es que al conectarse client1 (este cliente remoto que es un Linux con una red detrás) que por favor cree una ruta hacia la red 10.1.2.0/24 que vaya hacia ese cliente remoto. Con eso logramos que las máquinas de la red del servidor puedan ver a las del cliente.

El archivo server.conf quedará así:

```
port 1194
proto udp
dev tun
ca ca.crt
cert server.crt
```

```
key server.key
dh dh1024.pem
#Direcciones que se asignaran a los
#clientes, el server es .1
server 10.1.1.0 255.255.255.0

ifconfig-pool-persist ipp.txt

#Ruta para que los clientes alcancen la red local del server (56.0/24)
push "route 10.1.1.0 255.255.255.0"

client-config-dir ccd
route 10.1.2.0 255.255.255.0
client-to-client
push "route 10.1.2.0 255.255.255.0"

keepalive 10 120
comp-lzo
persist-key
persist-tun
status openvpn-status.log
verb 4
```

SSII1

Hacia el directorio /etc/openvpn de cada cliente copiamos los siguientes archivos:

- ca.crt
- client1.crt
- client1.key

El archivo client.conf quedará así:

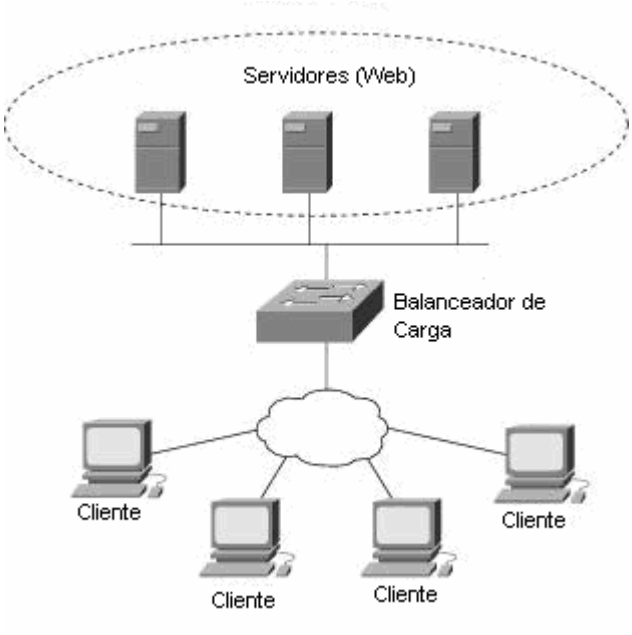
```
client
dev tun
proto udp
remote 147.96.81.251 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca ca.crt
```

```
cert client1.crt
key client1.key
comp-lzo
verb 4
```

10 Balanceador de carga:

Uno de los principales problemas de los mayores sitios web en Internet es cómo gestionar las solicitudes de un gran número de usuarios. Se trata de un problema de escalabilidad que surge con el continuo crecimiento del número de usuarios.

Un **balanceador de carga** fundamentalmente es un dispositivo de hardware o software que se pone al frente de un conjunto de servidores que atienden una aplicación (en nuestro caso será en el front-end) y, tal como su nombre lo indica, asigna o balancea las solicitudes que llegan de los clientes a los servidores usando algún algoritmo (desde un simple Round Robin o Asignación por pesos, hasta algoritmos más sofisticados).



Como balanceadores de carga, nosotros hemos probado 3 distintos: Nginx, HAProxy y Pound, pasamos a describirles cada uno de ellos.

Nginx (engine x) es un servidor HTTP, reverse proxy y servidor proxy IMAP/POP3. Nginx fue desarrollado por Igor Sysoev para Rambler.ru, el segundo sitio web más visitado de Rusia, donde ha estado funcionando en producción más de dos años y medio. Igor ha lanzado el código fuente bajo una licencia estilo BSD. Aunque aún se encuentra en una etapa beta, Nginx es conocido por su estabilidad, gran conjunto de características, configuración simple, y bajo consumo de recursos.

Entre las características HTTP nos encontramos: manejo de ficheros estáticos, índices y autoíndices, reverse proxying acelerado sin caché y con balanceo de carga y tolerancia a fallos, FastCGI, arquitectura modular y soporte SSL.

En Apache-ES han medido el rendimiento y en el caso que se trataba, servicio de imágenes y contenido estático, gana Nginx debido a unas modificaciones que trae de base.

Instalación:

```
ubuntu@mv01:~$ wget http://sysoev.ru/nginx/nginx-0.7.59.tar.gz
ubuntu@mv01:~$ tar xvfz nginx-0.7.59.tar.gz
ubuntu@mv01:~$./configure
ubuntu@mv01:~$ make
ubuntu@mv01:~$ sudo make install
```

También lo podríamos hacer de forma automática con:

```
ubuntu@mv01:~$ apt-get install nginx
```

En su modalidad de balanceador de carga, que es la que hemos usado en este caso, su configuración es bastante sencilla, quedando el nginx.conf de esta forma:

```
user  www-data www-data;
worker_processes  2;

error_log  logs/error.log debug;

pid        logs/nginx.pid;

events {
    worker_connections  10240;
```

```
}


http {
   # include       conf/mime.types;
    default_type  application/octet-stream;

    log_format    main    '$remote_addr  -  $remote_user  [$time_local]
$status '
                      '"$request" $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "http_x_forwarded_for"';

    access_log  logs/access.log  main;

    sendfile        on;
    tcp_nopush     on;
    tcp_nodelay    on;

      upstream backends{
            server 10.1.2.10:80;
            server 10.1.2.11:80;
            server 10.1.1.12:80;
            server 10.1.1.13:80;
      }
    server {
        listen       80;

        access_log  logs/host.access.log  main;

        location / {
            proxy_pass         http://backends;
            #proxy_pass http://10.1.1.69:80;
            proxy_redirect     off;

            proxy_set_header   Host             $host;
            proxy_set_header   X-Real-IP        $remote_addr;
            proxy_set_header   X-Forwarded-For
      $proxy_add_x_forwarded_for;

            client_max_body_size       10m;
            client_body_buffer_size    128k;
```

```
            proxy_connect_timeout      90;
            proxy_send_timeout         90;
            proxy_read_timeout         90;

            proxy_buffer_size          4k;
            proxy_buffers              4 32k;
            proxy_busy_buffers_size    64k;
            proxy_temp_file_write_size 64k;
        }
    }
}
```

Por último, para iniciar el Nginx, hemos de ejecutar:

```
ubuntu@mv01:~$ /etc/init.d/nginx start
```

10.2 POUND

El programa **POUND** es un reverse proxy, balanceador de carga y un servidor Web front-end HTTPS. Pound fue desarrollado exclusivamente para permitir la distribución de carga entre varios servidores Web y permitir usar el protocolo de seguridad SSL para aquellos servidores Web que no lo ofrecen de forma nativa. Pound está distribuido bajo licencia GPL.

Dado que su uso es bastante concreto, su instalación y configuración es realmente sencilla, permitiéndonos tener un balanceador de carga operativo en un instante:

Instalación:

```
ubuntu@mv01:~$ wget http://www.apsis.ch/pound/Pound-2.4.4.tgz
ubuntu@mv01:~$ tar xvfz http:// Pound-2.4.4.tgz
ubuntu@mv01:~$./configure
ubuntu@mv01:~$ make
ubuntu@mv01:~$ sudo make install
```

Archivo de configuración:

```
User        "www-data"
Group       "www-data"
#RootJail   "/chroot/pound"

LogLevel    1

## check backend every X secs:
Alive       30



# poundctl control socket
Control "/var/run/pound/poundctl.socket"



######################################################################
## listen, redirect and ... to:

## redirect all requests on port 8080 ("ListenHTTP") to the local
webserver (see
 "Service" below):
```

```
ListenHTTP
      Address 147.96.81.251
      Port  80

      ## allow PUT and DELETE also (by default only GET, POST and
HEAD)?:
      xHTTP       0

      Service
            BackEnd
                  Address     10.1.2.10
                  Port  80
            End
            BackEnd
                  Address 10.1.2.11
                  Port  80
            End
            BackEnd
                  Address 10.1.1.12
                  Port  80
            End
            BackEnd
                  Address 10.1.1.13
                  Port  80
            End
      En
End
```

Por último, para iniciarlo, hemos de modificar el archivo:/etc/default/pound, poniendo el startup=1;

```
ubuntu@mv01:~$ vim /etc/default/pound

# Defaults for pound initscript
# sourced by /etc/init.d/pound
# installed at /etc/default/pound by the maintainer scripts

# prevent startup with default configuration
# set the below varible to 1 in order to allow pound to start
startup=1
```

Finalmente, iniciamos el proceso pound:

```
ubuntu@mv01:~$ /etc/init.d/pound start
```

El **HAProxy** es una solución muy rápida que ofrece alta disponibilidad, balanceo de carga y proxying para aplicaciones TCP y HTTP. Es particularmente útil en web sites con una alta demanda y un gran número de conexiones simultaneas. Al igual que el Nginx o el Pound, es realmente fácil de configurar e instalar:

Los tres balanceadores de carga que hemos usado tienen un rendimiento similar, sin embargo, si es verdad que los dos últimos (HAProxy y Pound) son con los que menos problemas hemos tenido al configurarlos, ya que tienen muchas menos opciones que el Nginx (principalmente el Pound) y su utilidad prácticamente está limitada a la de balanceador de carga. (Esto es lo que nos ayudó a decidirnos a no optar por Apache como balanceador de carga, ya que, pese a que es posible usarlo como tal, su configuración es algo más laboriosa).

La forma más sencilla de instalar el HAProxy es directamente desde los repositorios de Ubuntu, en este caso, valdría con ejecutar:

```
ubuntu@mv01:~$ apt-get install haproxy
```

El archivo de configuración quedaría de la siguiente forma:

```
#this config needs haproxy-1.1.28 or haproxy-1.2.1

global
      log 127.0.0.1     local0
      log 127.0.0.1     local1 notice
      #log loghost      local0 info
      maxconn 4096
      #chroot /usr/share/haproxy
      user haproxy
      group haproxy
      daemon
      #debug
      #quiet

defaults
      log   global
      mode  http
```

```
      option      httplog
      option      dontlognull
      retries     3
      option redispatch
      maxconn     2000
      contimeout  5000
      clitimeout  50000
      srvtimeout  50000


listen      appli1-rewirite 147.96.81.251:80
      cookie      SERVERID rewrite
      balance     roundrobin
server app1_1 10.1.2.10:80 cookie app1inst1 check inter 2000 rise 2
fall 5
server app1_2 10.1.2.11:80 cookie app1inst2 check inter 2000 rise 2
fall 5
server app1_3 10.1.1.13:80 cookie app1inst3 check inter 2000 rise 2
fall 5
server app1_4 10.1.1.14:80 cookie app1inst4 check inter 2000 rise 2
fall 5


      errorfile   400   /etc/haproxy/errors/400.http
      errorfile   403   /etc/haproxy/errors/403.http
      errorfile   408   /etc/haproxy/errors/408.http
      errorfile   500   /etc/haproxy/errors/500.http
      errorfile   502   /etc/haproxy/errors/502.http
      errorfile   503   /etc/haproxy/errors/503.http
      errorfile   504   /etc/haproxy/errors/504.http
```

Por último, para iniciarlo, hemos de modificar el archivo:/etc/default/haproxy, poniendo el ENABLED=1;

```
# Set ENABLED to 1 if you want the init script to start haproxy
ENABLED=1
# Add extra flags here.
```

Finalmente, iniciamos el proceso haproxy:

```
ubuntu@mv01:~$ /etc/init.d/haproxy start
```

11 Servidores Web

En las Máquinas Virtuales (back-end), hemos de ofrecer al cliente la aplicación web que deseemos, para ello utilizaremos los servidores web.

Un servidor web es un programa que implementa el protocolo HTTP (HyperText Transfer Protocol). Este protocolo pertenece a la capa de aplicación del modelo OSI y está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML (HyperText Markup Language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música.

Es un programa que se ejecuta continuamente en un ordenador, manteniéndose a la espera de peticiones por parte de un cliente (un navegador web) y que responde a estas peticiones adecuadamente, mediante una página web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error.

Por tanto, un servidor web se mantiene a la espera de peticiones HTTP por parte de un cliente HTTP que solemos conocer como navegador. El cliente realiza una petición al servidor y éste le responde con el contenido que el cliente solicita (es el navegador el encargado de interpretar de forma visual el código html, css, javascript...).

En nuestro caso, la mayoría de las pruebas las realizamos usando como servidores web en nuestras máquinas virtuales, tanto Nginx (cuya configuración es la misma que en el modo balanceador de carga, salvo que no hemos de modificar el archivo de configuración nginx.conf), y Xampp, que pasamos a describir a continuación.

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor Web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl. El programa está liberado bajo la licencia GNU y actúa como un servidor Web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP esta disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X.

La filosofía detrás de XAMPP es la construcción de una versión fácil de instalar para los desarrolladores que entran al mundo de Apache. Para hacerlo más conveniente para los desarrolladores, XAMPP está configurado con todas las funciones activadas.

En este caso, y dado que el Sistema Operativo de nuestro cluster es GNU/Linux, instalaremos una versión de XAMPP llamada LAMPP. Su instalación y puesta en marcha es sencillísima.

Instalación:

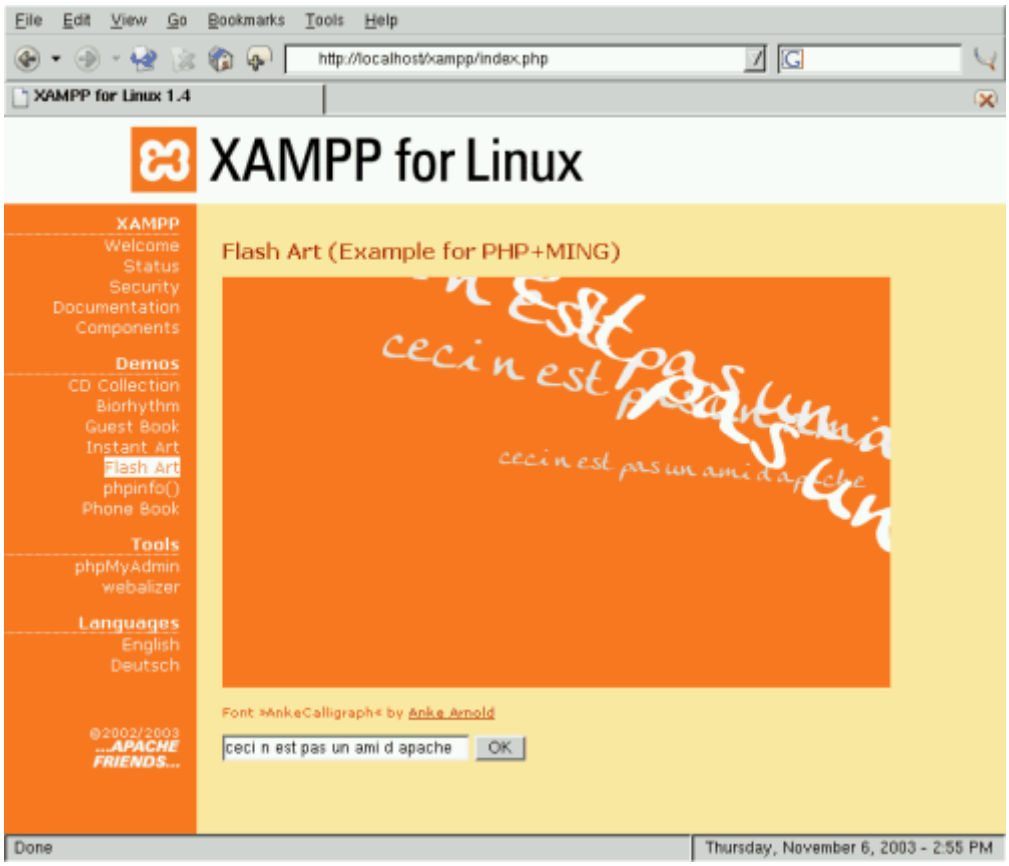
```
ubuntu@cluster01:~$ wget
http://www.apachefriends.org/download.php?xampp-linux-1.7.1.tar.gz

ubuntu@cluster01:~$ su

ubuntu@cluster01:~$ tar xvfz xampp-linux-1.7.1.tar.gz -C /opt
```

Tras esto, tendremos la instalación de LAMPP hecha en el directorio /opt/lampp (por defecto en el archivo de configuración tenemos establecido que se ejecuta en el puerto 80, obviamente esto lo podemos cambiar)

Para iniciar LAMPP, bastará con ejecutar:

```
ubuntu@cluster01:~$ /opt/lampp/lampp start
```

Ya está, ahora, accediendo a la dirección http://localhost (si accedemos desde la máquina donde hemos instalado el LAMPP), o http://ipPublica-MaquinaDondeInstalamosLAMPP (si accedemos desde cualquier otra máquina, y la instalación la hemso hecho desde una máquina con IP pública), nos aparecerá la siguiente web:


File Edit View Go Bookmarks Tools Help
http://localhost/xampp/index.php
XAMPP for Linux 1.4
XAMPP for Linux
XAMPP
Welcome
Status
Security
Documentation
Components
Demos
CD Collection
Biorhythm
Guest Book
Instant Art
Flash Art
phpinfo()
Phone Book
Tools
phpMyAdmin
webalizer
Languages
English
Deutsch
©2002/2003
...APACHE FRIENDS...
Flash Art (Example for PHP+MING)
Font »AnkeCalligraph« by Anke Arnold
ceci n est pas un ami d apache
OK
Done
Thursday, November 6, 2003 - 2:55 PM

Conclusiones

Antes de nada, queremos valorar el acierto de haber podido trabajar en este proyecto. Sin lugar a dudas realizarlo nos ha servido para aprender sobre tecnologías que desconocíamos por completo, y que durante la Carrera apenas se tocan, tal y como es el tema de Virtualización o el de Cloud Computing, los cuales consideramos que tendrán un futuro más que relevante en el mundo de la informática.

También nos ha permitido afianzar y ampliar los conocimientos que teníamos sobre el Sistema Operativo Linux, las redes VPN, trabajar con clusters (hasta entonces tampoco lo habíamos hecho), configuración IP, configuración de los distintos balanceadores de carga con los que hemos trabajado, etc.

Por otra parte, consideramos que la mayor parte del tiempo que le hemos tenido que dedicar al proyecto ha sido más para aprender que en lo que es la implementación en sí.

Respecto a nuestra opinión sobre Cloud Computing y el mundo de la Virtualización, sin lugar a dudas creemos que facilita muchísimo el despliegue de servidores web. La posibilidad de hacer sistemas fácilmente escalables, en los que podamos aprovechar recursos de cualquier parte del mundo, que podamos añadir nuevas máquinas virtuales (servidores web) a nuestro sistema contratándolas de servicios como Amazon EC2, hace que podamos usar Internet como infraestructura para implementar nuestro sitio web sin la necesidad de disponer de un centro de datos propio (que en muchos casos estará desaprovechado al no adecuarse exactamente a las necesidades de cada momento), o, por el contrario, en caso de disponer ya de un centro de datos propio donde alojar nuestro servicio web, poder disponer de una infraestructura adicional para situaciones en las que sea necesaria satisfacer una mayor demanda del servicio.

Estamos convencidos de que el futuro irá orientado hacia el llamado "Cloud Computing", tal y como se puede observar, por eso consideramos que ha sido realmente gratificante poder aprender y entender algunas de las distintas tecnologías que lo forman. De hecho, una gran parte del tiempo que hemos dedicado al proyecto ha sido de documentación.

Palabras clave:

Cloud computing, máquina virtual, virtualización, servidor web, túnel, OpenNEbula, Xen, OpenVPN, EC2, Nginx.

Bibliografía

Información General:

- Sitio web de Ubuntu. http://www.ubuntu-es.org/node/5290
- "Instala, administra, securiza y virtualiza Entornos Linux" de Antonio Ángel Ramos, Jean Paul García-Morán, Fernando Picouto, Jacinto Grijalba, Maikel Mayan, Ángel García, Eduardo Inza y Carlos Alberto Barbero. Editorial RA-MA
- Página web de Docencia del profesor Juan Carlos Fabero Jiménez: http://www.fdi.ucm.es/profesor/jcfabero/

Virtualización (Xen):

- http://www.techweek.es/virtualizacion/tech-labs/1003109005901/ventajas-desventajas-virtualizacion.1.html
- http://es.wikipedia.org/wiki/Virtualización
- http://es.wikipedia.org/wiki/Xen
- http://www.xen.org/
- http://www.howtoforge.com/ubuntu-8.04-server-install-xen-from-ubuntu-repositories
- Professional Xen Virtualization – William von Hagen – ISBN 978-0-470-13811-3

Cloud Computing (OpenNEbula):

- http://www.idg.es/comunicaciones/articulo.asp?id=191003
- http://en.wikipedia.org/wiki/Cloud_computing
- http://www.OpenNEbula.org/

VPN:

- OpenVPN: Building and Integrating Virtual Private Networks – Markus Feilner – ISBN 1-904811-85
- Sito web de OpenVPN. http://openvpn.net/howto.html
- http://laurel.datsi.fi.upm.es/~rpons/openvpn_como/

- http://www.linuxhomenetworking.com/

Balanceadores de Carga:

- http://www.linuxhomenetworking.com/
- http://www.apsis.ch/pound
- http://nginx.net
- http://wiki.nginx.org
- http://en.wikipedia.org/wiki/Nginx
- http://cesar.la/servidores-baratos-seguros-y-eficientes.html

Servidores Web (XAMPP):

- http://es.wikipedia.org/wiki/XAMPP
- http://www.apachefriends.org/es/xampp.html
- http://www.apsis.ch/pound

Derechos sobre el proyecto: