Virtualització i Cloud

Edgard de Haro Andrés

Resum— Des de fa aproximadament deu anys, l'ús i la dependència d'Internet s'ha tornat quelcom que forma part de les nostres vides, tant a nivell professional com a nivell personal. Per altra banda, també està havent una tendència a que les empreses emmagatzemin grans quantitats de dades per a ser processades posteriorment i poder fer estudis sobre elles. Tant en un cas com l'altre, estan necessitats de capacitat de còmput i d'emmagatzemament. Per obtenir aquesta capacitat, es podria solucionar afegint nous ordinadors, però això augmentaria el cost de la infraestructura alhora que estaríem desaprofitant molts recursos. Per solucionar aquest problema, s'utilitzen màquines virtuals que podrem crear-les quan necessitem recursos o esborrar-les quan ja no siguin necessàries. En aquest article es farà una breu introducció sobre la virtualització i els dos principals hipervisors per després poder-los aplicar sobre una plataforma Cloud creada amb OpenStack. No s'entrarà en detalls d'instal·lació sinó que s'explicarà més des del punt de vista de l'arquitectura i els elements d'aquesta plataforma.

Paraules clau—Virtualització, KVM, XenServer, Cloud, IaaS, OpenStack, CloudStack

Abstract— Since ten years ago, use of Internet has become something that is part of our lives, both professional and personal live. Moreover, exists a tendency of companies to storing large quantity of data in order to process it and do studies with this data. In two cases, they need capacity to compute this and large space to store this data. So, to obtain this capacity you could solve by adding new computers, but this solution would increase the final cost of the infrastructure and would be wasting computational resources. To solve this problem, we use virtual machines that we can create when more computational resources are necessary and destroy when there aren't necessary. This article is an introduction to virtualization and the two principal types of hypervisors. Then we apply one of these hypervisors on a Cloud platform created with OpenStack. We don't will see all the installation process since this is not an installation manual but will explain the architecture of Cloud and its elements.

Index Terms—Virtualization, KVM, XenServer, Cloud IaaS, OpenStack, CloudStack

1 Introducció

TAL com s'ha vist en el resum, l'objectiu final d'aquest article és entendre com funciona de manera general una arquitectura Cloud i de manera més concreta veurem aquesta arquitectura aplicada a OpenStack.

Primer de tot explicarem què és la virtualització, com funcionen els dos casos que s'han escollit com a exemples i farem un anàlisi sobre el rendiment que ofereix cada un dels hipervisors. A continuació s'explicarà què és un Cloud, quins tipus existeixen i entrarem en detall en el tipus de Cloud IaaS. Un cop explicada l'arquitectura d'un Cloud IaaS, es veurà un cas pràctic amb OpenStack. Sobre aquesta plataforma es veuran quins són els diferents elements que forma l'arquitectura i s'explicarà que es pot arribar a fer amb aquesta implementació.

2 Objectius

L'objectiu principal d'aquest treball és entendre com funciona una plataforma Cloud i ser capaç de muntar-la. Per arribar a aquest punt primer cal entendre altres conceptes importants:

• Virtualització: és important entendre com funciona la virtualització i entendre l'arquitectura d'alguns hipervisors degut a que no tots poden executar els mateixos sistemes operatius, necessiten uns requisits concrets i inclús hi ha petites diferencies de rendiment. Per això, primerament analitzarem aquest punt ja que serà l'element més important de la plataforma Cloud.

• Arquitectura Cloud: s'estudiarà l'arquitectura Cloud d'una manera general, tenint en compte els elements que intervenen. Un cop entès com funciona, es veurà la seva correspondència amb OpenStack on s'analitzarà cada un dels serveis dels que disposa.

• OpenStack: en aquest punt es pretén veure les opcions de les que disposa OpenStack per poder construir la nostra pròpia plataforma a partir de l'arquitectura d'OpenStack. No es pretén crear una nova interfície de gestió, sinó veure quines opcions ens ofereix i que ens permet fer.

• *E-mail de contacte: edgarde.haro@e-campus.uab.cat*
• *Menció realitzada: Tecnologies de la Informació.*
• *Treball tutoritzat per: Josep M. Basart i Muñoz (dEIC)*
• *Curs 2013/14*

3 ESTAT DE L'ART

Des de fa uns anys enrere, molts serveis i/o aplicacions que els usuaris utilitzen en el seu dia a dia, funcionen sobre una xarxa, ja sigui interna (una LAN) o externa (amb connectivitat a Internet). Això significa que com a mínim hauran dos ordinadors que s'hauran de comunicar: el client que està utilitzant l'aplicació o servei i el servidor que ofereix aquesta aplicació o servei. Com bé sabem, la capacitat de còmput de qualsevol ordinador és limitada, és a dir, no podem afegir-li treballs de manera que no disminueixi el seu rendiment, sinó que arribarà un punt en el que l'ordinador pràcticament no seria capaç de respondre. Això, aplicat al serveis o aplicacions que utilitzen la xarxa, significa que no podran dependre d'un sol servidor sinó que necessitaran més o menys segons el nombre d'usuaris que vulguin utilitzar aquest servei.

Una possible solució seria afegir més servidors (en aquest cas, s'entén com a servidor el conjunt de hardware i sistema operatiu junt amb els serveis necessaris), aconseguint d'aquesta manera més capacitat de còmput: per exemple, si un servidor podia suportar 500 usuaris amb una càrrega mitjana, amb dos servidors podrem suportar 1.000 usuaris. Tot i ser una solució efectiva en el sentit de que hem pogut augmentar el nombre d'usuaris, no deixa de ser una solució cara degut a que s'ha hagut de comprar un nou servidor i un augment en el consum d'energia.

En aquest punt és on apareix la virtualització. Aquesta solució permet executar molts ordinadors virtuals dins d'un sol ordinador físic. D'aquesta manera reduirem els costos tant en hardware (no cal comprar tants servidors físics) com en consum d'energia. Tot i haver solucionat el problema de reducció de costos, la virtualització no aporta cap solució sobre com augmentar a gran escala el nombre de màquines virtuals: és a dir, ara som capaços d'iniciar noves màquines dins d'un mateix host, però com hem dit anteriorment, tots els ordinadors tenen un límit de còmput, per tant caldrà una manera d'escalar aquest sistema per quan els recursos disponibles no siguin suficients (per exemple quan un host físic no suporta més màquines virtuals per falta de memòria principal). Llavors és quan s'afegeixen nous servidores físics per a virtualitzar altres servidors. Però quan el nombre de servidors físics augmenta molt, la gestió de la infraestructura es complica més. És en aquest punt quan apareix el Cloud per solucionar-ho.

Avui dia, la paraula Cloud és molt utilitzada en diferents situacions: per exemple, Dropbox és considerat com un servei Cloud, la plataforma de Cloud d'Amazon (AWS), els VPS de les empreses de hosting, una aplicació web com Office 365, Google App Engine... A tots ells ens referim com a serveis del Cloud o núvol però no ofereixen el mateix servei. En el cas de Dropbox ofereix un servei d'emmagatzemament de dades a Internet on l'usuari no cal que es preocupi d'on ni com estan guardats els seus fitxers sinó que utilitza el software o aplicació web que ofereixen. Per altra banda, Amazon dona tota una infraestructura on nosaltres podrem utilitzar els recursos que necessitem. Per últim, Google App Engine ens proporciona una sèrie de llibreries (també anomenat framework) amb les quals nosaltres crearem una aplicació que s'executarà fora del nostre ordinador, sense haver de tenir cap control de com s'executa ni on s'executa. Veiem doncs, que no existeix un sol tipus de Cloud sinó que depenent del que es vulgui fer, necessitarem un tipus de plataforma Cloud o un altre. Per tant, unint virtualització i el concepte de Cloud, és possible tenir una plataforma on els recursos es creïn o s'eliminin segons les necessitats en cada moment i podent adaptar les capacitats de còmput a les necessitats de cada usuari.

És per això que les infraestructures Cloud no són tan senzilles com un sol panell de gestió on nosaltres indiquem la creació o eliminació de màquines virtuals, sinó que es necessiten més elements per poder coordinar tots els nodes. Alguns dels elements necessaris, seria per exemple un servidor de missatges el qual s'encarrega de rebre l'estat de tots els nodes de la plataforma (principalment recursos disponibles en cada un d'ells). També serà necessari un servei que s'encarregui d'assignar les instàncies a cada node de còmput. Aquest servei haurà de tenir en compte la càrrega de cada node per no sobrecarregar cap.

4 METODOLOGIA

Com he vist anteriorment, la virtualització és una tècnica que s'utilitza per executar màquines virtuals dintre d'un mateix host físic. Per fer-ho, existeixen diferents productes, alguns d'ells gratuïts, altres de codi obert i altres de pagament. Tots aquests productes fan el mateix: executar màquines virtuals, però ho fan de manera diferent i per tant el rendiment i/o les característiques de tots ells són diferents. Per això, el primer que es farà serà escollir dos d'aquests productes (que siguin compatibles amb la plataforma Cloud d'OpenStack) i analitzar la tècnica que utilitzen per a la virtualització de màquines.

OpenStack pot utilitzar molts tipus d'hipervisors però els més utilitzats són KVM/QEMU, XenServer, VMWare i Hyper-V. VMWare i Hyper-V són dos productes de pagament a part de que VMWare necessita uns requisits de hardware molt específics dels que no es disposa. Per això, s'analitzarà el funcionament de KVM i XenServer, dos plataformes gratuïtes i de codi obert.

Un cop escollit l'hipervisor, haurem de construir la plataforma Cloud. Per entendre de manera teòrica un Cloud, primer veurem una arquitectura general, analitzant els components més importants que composa un Cloud. A continuació veurem un cas pràctic d'una plataforma Cloud: OpenStack. Es tractarà d'analitzar cada un dels seus serveis veient què ens aporta cada un a la plataforma. Aquest article no pretén ser una guia d'instal·lació, per això, ni el procés d'instal·lació ni els paràmetres de configuració de cada un dels serveis no es mostrarà aquí. Tota la documentació referent a aquest procés es pot trobar a [5] i [6].

5 VIRTUALITZACIÓ

Abans d'entrar en detall en el funcionament i arquitectura d'un Cloud cal entendre la base del seu funcionament: la virtualització. A continuació veurem dos mètodes de virtualització que treballen sobre Linux: la virtualització complerta assistida per hardware (KVM) i la paravirtualització (XenServer) [2]. Són dues tècniques completament oposades: la primera virtualitza cada un dels possibles elements hardware que pugui tenir la màquina virtual (és a dir, el sistema operatiu virtualitzat no sap que s'executa de manera virtualitzada) i l'altre ofereix una interfície per treballar sobre el hardware del host físic sense haver de fer una virtualització de tots els elements.

5.1 KVM

KVM és un sistema de virtualització gratuït i de codi obert que funciona sobre Linux. És un sistema de virtualització completa, és a dir, tots els recursos que utilitza la màquina virtual són recursos virtualitzats. Aquest recursos que es posen a disposició de les màquines virtuals solen ser 4: processador (CPU), memòria principal, perifèrics i xarxa. Aquest tipus de virtualització també s'anomena "full virtualization". L'avantatge que té aquest sistema és que, a diferencia d'altres mètodes (com per exemple la paravirtualització), permet executar qualsevol sistema operatiu compatible amb el processador sense haver de fer-li cap modificació ja que el sistema operatiu virtualitzat no sap que s'està executant en un entorn de virtualització.

KVM forma part del nucli de Linux, concretament és un mòdul anomenat kvm.ko. L'únic requisit de KVM és que el processador ha de tenir el conjunt d'instruccions de virtualització. En els processadors Intel, aquest conjunt és l'anomenat VT-x i en els processadors AMD s'anomena AMD-v. Aquest conjunt d'instruccions permet que KVM pugui fer una eficient virtualització amb una menor intervenció de software (en aquest cas, del traductor d'instruccions binàries, el qual és el que més penalització sol introduir a la virtualització). Per tant, quan també intervé una part del hardware en la virtualització, s'anomena "virtualització assistida per hardware" [3]. En cas de no utilitzar aquest conjunt d'instruccions, utilitzar KVM no tindria cap sentit i la virtualització es faria mitjançant QEMU (molt més lent que KVM) on moltes instruccions han de ser traduïdes pel traductor d'instruccions binaries sobre el sistema operatiu del host, afegint un temps de penalització.

Els processadors d'arquitectura x86 i x64 tenen 4 nivells de privilegis: el nivell 0 és el que gestiona les instruccions a executar en el processador. Aquest és el nivell on s'executa el kernel ja que es té els màxims privilegis. En el nivell 1 és on es té el permís per gestionar la memòria i al nivell 2 és on es gestiona les entrades i sortides dels dispositius. El nivell 3 és l'espai d'usuari on s'executen els

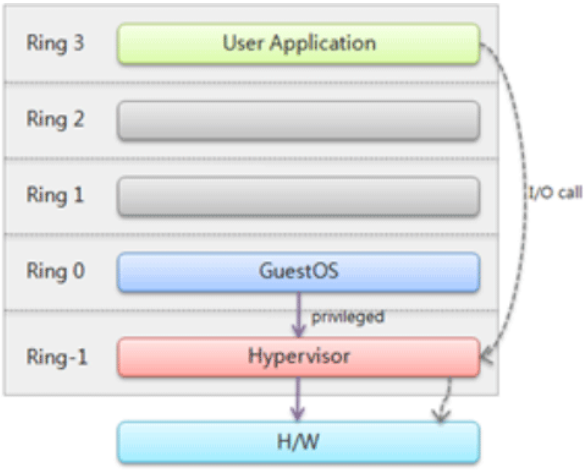

Figura 1. Esquema de la virtualització assistida per hardware. Totes les instruccions ha executar de la màquina virtual passen per l'hipervisor i és aquest el que s'encarrega de decidir si cal executar alguna instrucció del conjunt de virtualització (VT-x o AMD-v).

programes i no hi ha cap permís directe sobre el processador, memòria ni dispositius d'entrada i sortida. Aquest és un model simplificat, ja que a vegades també se sol parlar d'un nivell de sis capes. El que fan VT-x i AMD-v és afegir un nou nivell anomenat -1 el qual té un conjunt d'instruccions que permeten executar aquelles instruccions més complexes de les màquines virtuals (com interrupcions, l'accés a memòria o dispositius...) directament sobre el processador enlloc de que hagin de passar pel traductor binari d'instruccions. A la figura 1 podem veure com totes les instruccions, tant del host físic com de les màquines virtuals, passen per l'hipervisor i és aquest el que decideix si s'executarà la mateixa instrucció sobre el hardware o caldrà canviar l'instrucció per una del conjunt d'instruccions de virtualització.

5.2 XenServer

XenServer és una plataforma de virtualització basada en el projecte Xen que es distribueix de manera gratuïta (amb certes limitacions) on s'inclou un sistema operatiu ja configurat amb l'extensió Xen, un client per a Windows per l'administració a distància del servidor i un conjunt d'eines executables des del mateix hipervisor. XenServer incorpora dos mètodes de virtualització: la paravirtualització i HVM (virtualització assistida per hardware). El mode HVM funciona de manera molt semblant a KVM. Aquest mode serveix per quan es vol instal·lar un sistema operatiu en el qual no es possible modificar el kernel. Seria el cas de Windows, on no es disposa del codi font del seu kernel per introduir els canvis necessaris per a paravirtualitzar el sistema operatiu.

La paravirtualització, a diferencia de la virtualització assistida per hardware, requereix afegir uns canvis al kernel del sistema operatiu convidat i no necessita de les instruccions de virtualització VT-x o AMD-v. Consisteix en afegir, en el kernel del sistema convidat, uns drivers (també anomenat interfície) que es comunicaran a través d'una API amb el sistema operatiu del host. L'objectiu

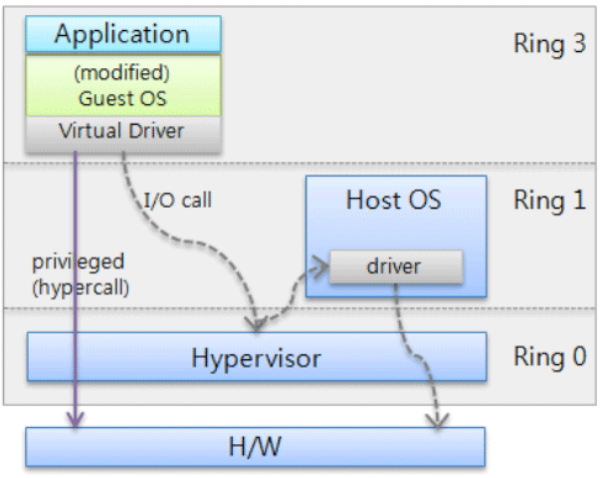

Figura 2. Esquema de la paravirtualització. Veiem com certes crides poden ser executades directament sobre el hardware però altres han de passar per l'hipervisor i és aquest el que s'encarrega d'executar-ho sobre el hardware.

d'aquesta nova capa entre el sistema convidat i el sistema host és poder executar algunes de les instruccions més complexes (com interrupcions, accés a memòria) sense haver de passar pel traductor d'instruccions binaries, augmentant significativament el rendiment de la màquina virtualitzada. Veiem doncs, que les màquines virtuals paravirtualitzades, tenen pràcticament un control complert dels recursos hardware, ja que a través d'aquesta API poden executar les instruccions més privilegiades sobre el host físic. A la figura 2 es pot veure de manera més esquemàtica.

5.3 Comparació de rendiment

Com hem vist, la virtualització assistida per hardware i la paravirtualització són mètodes diferents, pel que és normal esperar rendiments diferents. Per això cal executar una sèrie de benchkmarks sobre màquines virtuals virtualitzades amb els dos mètodes. El sistema operatiu convidat serà exactament el mateix en els dos casos així com els benchmarks executats. Els benchmarks executats seran tres: un d'us intensiu de CPU, un altre d'accés a memòria principal i un altra de lectura i escriptura al disc dur. En el test de CPU, el resultat són MIPS o milions d'instruccions per segon. Quantes més instruccions s'hagin executat per segon significarà que té més rendiment ja que el nombre d'instruccions a executar es fixe, per tant el que tingui més MIPS haurà trigat menys temps en executar-les totes. Per fer les proves de rendiment, s'ha utilitzat una utilitat anomenada "Phoronix Test Suite" [10]. S'ha escollit aquest software degut a la gran quantitat de proves diferents que incorpora i la possibilitat d'afegir-ne de noves que es puguin trobar a Internet.

En el cas del benchmark de memòria principal, les unitats seran de MegaBytes/segon, per tant, interessa que aquest nombre sigui el més alt possible. Aquest test està dividit en dos: per a nombre enters i per a nombre decimals. L'últim benchmark consisteix en desempaquetar un fitxer gran (en aquest test es fa amb el fitxer del codi font del kernel de Linux) i guardar-ho en el disc dur. El que mesura aquest test és el temps que triga en realitzar aquesta feina. Per tant, quan menys temps trigui significarà que tindrà més rendiment. A la taula 1 es mostren els resultats obtinguts.

	KVM	Xen
CPU	1439 MIPS	1518 MIPS
Memòria (enters)	1988,52 MB/s	2156,45 MB/s
Memòria (decimals)	1979,03 MB/s	2159,57 MB/s
Disc dur	37,31 s	34,36 s

Taula 1. Resultats dels benchmarks. La primera columna fa referència als resultats executats sobre KVM i la segona columna sobre XenServer. S'han utilitzat els mateixos test i el mateix sistema operatiu virtualitzat.

Veiem doncs, que Xen dona un rendiment una mica més alt que KVM. Tot i els resultats, en les següents fases s'utilitzarà KVM per simplicitat, ja que l'objectiu del treball no és obtenir una plataforma Cloud d'alt rendiment sinó aprendre com funciona, instal·lar-la i aprofundir en els serveis que ofereix.

6 CLOUD

Un cop sabem què és la virtualització i com funciona, passarem a entendre què és el Cloud, els seus tipus i un model d'arquitectura general amb els elements necessaris per construir-lo. Després analitzarem un cas en concret de Cloud: OpenStack

6.1 Tipus de Clouds

Com hem vist anteriorment, el Cloud és qualsevol tipus de servei ofert per un conjunt de màquines on podem escalar la plataforma de manera fàcil i transparent. També hem dit que segons les necessitats en cada cas, s'utilitzarà un tipus de Cloud o un altre. Veiem doncs, quins tres models de Cloud existeixen:

- **IaaS:** o infraestructura com a servei. Aquest és el cas sobre el qual analitzarem la seva arquitectura. Posa a disposició de l'usuari uns recursos de computació i unes eines amb les quals pot afegir o treure recursos. És el nivell més baix d'un Cloud on podem controlar de manera precisa el nombre d'instàncies que utilitzarem, l'emmagatzemament de les dades... Un exemple d'aquest model és Amazon Web Service (AWS).
- **PaaS:** o plataforma com a servei. Pretén oferir, a través d'unes llibreries (API) una plataforma on executar aplicacions, de manera que l'usuari no cal que es preocupi de temes com la xarxa, l'emmagatzemament... Un exemple de PaaS és l'API de Google de App Engine.
- **SaaS:** o software com a servei. Aquest model pretén que l'usuari no hagi de dependre d'un software que s'hagi d'instal·lar en un ordinador, sinó que pugui utilitzar-lo a través d'un navegador web. L'usuari no té que preocupar-se pels recursos que l'aplicació necessita. Un exemple de

SaaS és l'Office 365.

Veiem doncs, que les propietats d'usabilitat i la de gestió són inverses. En el cas d'IaaS, disposem d'una plataforma que podem gestionar nosaltres mateixos, ajustant el propi sistema operatiu, ja que és la base d'IaaS. En canvi, si volem executar una aplicació nostre sobre una plataforma IaaS haurem d'adaptar-la i configurar les instàncies per aquesta situació. En el cas de PaaS no podem personalitzar el sistema operatiu (aquí ja no veiem les instancies, ja que no las gestionem nosaltres). En canvi, ja tenim una API amb la qual no haurem de preocupar-nos sobre l'emmagatzemament o els recursos de còmput. Per últim, veiem que a SaaS no tenim la possibilitat de configurar res, sinó que directament amb el software podem treballar amb les nostres dades.

Nosaltres ens centrarem en el Cloud IaaS. Primer veurem quina és una arquitectura general d'aquest model de Cloud junt amb els elements necessaris i seguidament ho veurem aplicat sobre OpenStack, un software Open Source per fer Clouds IaaS.

6.2 Arquitectura d'un Cloud

Abans de veure l'arquitectura d'un cas en concret, veurem de manera general quina és l'arquitectura que ha de tenir una plataforma Cloud, quins són els elements mínims i quins són opcionals segons el que es vulgui arribar a fer amb la plataforma.

Com ja sabem, una infraestructura Cloud del tipus IaaS necessita un element bàsic: l'element de còmput. Però com hem explicat anteriorment, un Cloud consta de molts nodes de còmput, ja que si només necessitéssim tres o quatre nodes, es podrien gestionar de manera individual sense la necessitat de desplegar tota aquesta infraestructura. Per tant, sabem que en el Cloud intervindran molts nodes de còmput, que serà on es crearan les instancies (les màquines virtuals).

Quan vulguem crear una instancia nova, d'alguna manera s'haurà d'escollir un dels nodes. Aquesta selecció es podria fer de manera aleatòria, però correm el risc de que se'ns assigni un node que estigui molt carregat, pel que tindríem una instancia amb un rendiment molt baix o inclús no es podria arribar a crear. Per solucionar aquest problema se sol utilitzar un middleware de missatgeria, a través del qual tots els nodes notifiquen de manera regular els recursos disponibles. A través d'aquest middleware, també s'indica als nodes sobre la creació de noves instancies, l'eliminació, la migració... Així doncs, veiem que aquest middleware és un element bàsic per a la infraestructura.

Ara que ja tenim un lloc on crear noves instàncies i un mecanisme per rebre i enviar informació als nodes, cal un següent element molt important: un planificador o scheduler. Aquest element s'encarregarà de, segons els recursos disponibles en cada node, assignar les noves instancies segons la càrrega de cada node. Així aconseguim aprofitar tots els nodes disponibles de manera equitativa, és a dir, intentant mantenir la càrrega dels nodes per igual enlloc de sobrecarregar uns i tenir altres nodes sense utilitzar.

Amb aquests tres elements, tindríem una plataforma Cloud, però només haurien recursos de còmput disponibles de manera escalable, ja que l'emmagatzemament dependrà en tot moment, de l'espai del que disposa l'instancia creada. Per tant, seria interessant tenir recursos d'emmagatzemament que puguin créixer i que l'usuari de la plataforma no hagi de controlar. Seria doncs, un element més a afegir a la plataforma, independent del node de còmput (això no significa que no puguin estar en el mateix host, sinó que seran serveis diferents).

Tots els elements anomenats fins ara (middleware, planificador, recursos de còmput i recursos d'emmagatzemament) són els mínims elements necessaris per a tenir una plataforma Cloud. Tot i que seria una plataforma amb certes limitacions (no hi ha cap sistema que monitoritzi les instancies ni l'autoescalat d'instàncies), la podríem utilitzar perfectament. Però si el que volguéssim és crear una plataforma més complerta, seria interessant afegir els elements que hem dit anteriorment.

Suposem que volem fer un servei web en el qual de les 11h a les 17h té una càrrega molt elevada, però la resta d'hores pràcticament ningú utilitza aquest servei web. No tindria sentit tenir aquest sistema sobre molts servidors físics ja que estaríem desaprofitant molts més recursos dels necessaris. Llavors decidim portar aquesta plataforma a un Cloud, on podem crear i esborrar servidors virtuals segons les necessitats en cada franja horària. Si nosaltres sabem amb exactitud les franges horàries on hi ha més càrrega, podríem fer que uns minuts abans es creessin automàticament més instàncies i al acabar la franja horària d'alta càrrega s'esborressin automàticament. Però si la càrrega és completament variable (no sabem quan poden fer falta més instancies o quan hi ha menys activitat), podríem pensar en afegir un servei que monitoritzi l'estat de cada una. A través d'aquest servei podríem indicar uns certs llindars d'utilització de recursos a partir dels quals podríem utilitzar per crear noves instàncies, d'aquesta manera alliberaríem els recursos altres instàncies fent que millor el rendiment del servei web. És doncs, un element opcional a afegir a la plataforma però com hem vist, és de molta utilitat.

7 Cas pràctic: OpenStack

Al llarg dels apartats anteriors, hem vist què és i com funciona alguns mètodes de virtualització (una de les parts més importants d'un Cloud) i també hem vist, de manera general, quins elements ha de tenir una infraestructura Cloud per aconseguir el seu objectiu: crear instàncies de manera ràpida i escalable sense que l'usuari hagi de preocupar-se dels nodes físics. En aquest apartat s'explicarà un cas concret de plataforma Cloud: OpenS-

tack. El que es pretén és crear una infraestructura com la que té Amazon Web Service (AWS) on tinguem, de manera il·limitada (és a dir, escalant de manera fàcil la plataforma de manera transparent a l'usuari) recursos de còmput i emmagatzemament.

Per aconseguir-ho, OpenStack està construït diferents serveis independents els quals es comuniquen a través d'un middleware de missatges. El que veurem en aquest apartat seran els diferents serveis dels que consta OpenStack. A l'annex 1 podrem veure un diagrama de l'arquitectura d'OpenStack, de tots els serveis que pot utilitzar i de la dependència entre ells.

OpenStack, a diferencia d'altres plataformes Cloud com CloudStack, incorpora pràcticament tots els serveis necessaris per construir una plataforma IaaS com AWS. Però això té també implicacions negatives: el procés d'instal·lació és molt més complex i elaborat que el de CloudStack.

CloudStack consta d'un instal·lador pràcticament automàtic, on gairebé no cal modificar fitxers de configuració. En canvi, a OpenStack s'han de modificar tots els fitxers de cada un dels serveis, afegir permisos a cada servei, crear les bases de dades a ma... Podem dir doncs, que si el que necessitem és una plataforma senzilla, on el més important són els recursos de còmput, CloudStack seria la nostra elecció. Però si volem una plataforma que ofereixi més serveis i més configurable, la millor opció serà OpenStack.

Els serveis dels que consta OpenStack són: d'autenticació (anomenat Keystone), de gestió d'imatges (Glance), de còmput (Nova), de xarxa (Neutron), d'emmagatzemament per blocs i objectes (Cinder i Swift), de monitorització (Ceilometer) i d'automatització (Heat). A continuació veurem les característiques principals de cada servei i el que ens permet fer cada un. Per a la instal·lació dels serveis referir-se al manual [6] i a la documentació d'OpenStack[5].

7.1 Servei d'autenticació (Keystone)

Aquest serveis permet gestionar, de manera centralitzada, les credencials tant dels usuaris com de cada un dels serveis que estan en funcionament dins de la plataforma. És obvi que un usuari hagi d'autenticar-se per utilitzar la plataforma, ja que sinó qualsevol persona podria entrar. Però de cara als serveis també és necessari. Imaginem que s'afegeix un node de còmput que no ha sigut autenticat. Llavors, un usuari vol crear una nova instància on tractarà dades delicades, com per exemple números de comptes bancaris. Si el planificador assignés aquesta nova instancia al node no identificat, la persona que hagi posat aquest nou node tindria accés a la nova instància, podent agafar aquestes dades susceptibles. Per tant, és important que tot servei afegit estigui identificat i autenticat dins de la plataforma. Aquesta autenticació funciona de la mateixa manera pels punts d'entrada de l'API.

Aquest servei introdueix vuit conceptes que és important tenir-los clars:

- **Usuari:** persona, sistema o servei que utilitzarà la plataforma.
- **Credencial:** dada que coneix un usuari per demostrar que és qui diu ser.
- **Autenticació:** Un cop l'usuari a proporcionat les dades necessàries per identificar-se i el sistema ha comprovat que siguin correctes, l'usuari es considera autenticat. En aquest cas se li assigna un *token* que haurà de proporcionar en les successives comunicacions.
- **Token:** conjunt de bits que permeten l'accés als recursos un cop l'usuari està autenticat. Per a futures versions, s'utilitzarà per a nous protocols que el sistema pugui necessitar.
- **Tenant:** "contenidor" que serveix per agrupar i controlar els recursos que poden ser utilitzats per cada usuari o projecte.
- **Service:** des del punt de vista de l'autenticació, els serveis són els punts de sortida de l'API a través de la qual els usuaris poden interaccionar amb el sistema. Alguns dels serveis que tenen aquest punt de sortida són els de còmput (Nova) o d'imatges (Glance).
- **Endpoints:** punts de sortida a través dels quals els usuaris interaccionen amb el sistema. Sol ser en format d'URL ja que l'API que incorpora OpenStack es del tipus REST.
- **Role:** permisos i drets que té un usuari dins la plataforma. Per exemple, que només pugui utilitzar els sistema per crear instancies (permisos restringits) o poder afegir nous serveis, crear usuaris... (alts privilegis).

Veiem doncs, que Keystone no és un servei simple d'autenticació d'usari-contrassenya sinó que és molt més complex amb moltes opcions. Per interaccionar amb el servei, incorpora una utilitat de línia de comandes amb la qual podem crear usuaris, serveis, afegir punts de sortida, rols... També disposa d'un client que es pot utilitzar des de fora del node on s'executa el servei, d'aquesta manera permet una administració a distància del sistema. És important afegir els usuaris i serveis necessaris de manera correcta, ja que sinó els altres serveis poden no funcionar correctament.

7.2 Servei d'imatges (Glance)

Aquest servei guarda de manera centralitzada, les imatges per crear les noves instàncies. També es guarden les plantilles que els usuaris hagin creat a partir de les seves instàncies. D'aquesta manera deslliguem els nodes de còmput de les imatges amb la qual cosa desapareix qualsevol duplicitat d'imatges (això no vol dir que en el servei d'imatges no puguin haver dos plantilles iguals, ja que això ja depèn de com s'estigui utilitzant el servei, però si que evitem que hagi duplicitat d'imatges en nodes diferents).

Glance no té pràcticament opcions, ja que només es dedica a guardar imatges i a oferir-les quan són necessàries. Disposa d'un client de comandes tant per al node on està instal·lat com per a l'administració remota. Les opcions principals que té aquest servei és la d'afegir noves imatges, veure una llista d'imatges i veure les propietats de cada imatge.

7.3 Servei de còmput (Nova)

El servei de còmput és el servei principal de la infraestructura. És aquí on es produeix la virtualització. Aquest servei s'executa tant en el node controlador com en el node de còmput.

En el node controlador, aquest servei és el que fa la planificació, és a dir, busca, a partir dels recursos disponibles, quin és el millor node on crear la nova instància. També incorpora un gestor de xarxes que permet crear xarxes als nodes de còmput.

Per un altra part tenim el node de còmput. En aquest node s'instal·la l'hipervisor escollit (KVM, XenServer, VMWare...). A través d'aquest servei, el node espera peticions d'iniciar noves instàncies, apagar-les, esborrar-les...

Junt amb el servei de còmput (Nova), s'incorpora un servidor proxy de consola VNC o SPICE. Tot i que la instància creada té una adreça IP assignada en un rang determinat, pot ser interessant accedir a través d'un visor VNC en una pàgina web o d'un client d'SPICE per als casos en que ha hagut problemes de xarxa i no puguem accedir directament a través de l'adreça IP. Tot i ser un component opcional de Nova, és molt recomanable instal·lar-ho.

7.4 Servei de xarxa (Neutron)

Tal com hem dit, la configuració de la xarxa es pot fer de dos maneres: la primera és que cada node assigni les adreces a cada instància i utilitzi els recursos de xarxa del mateix node (com són les taules de ruta o el Firewall) per gestionar les instàncies. Aquest mode s'anomena *Legacy Network*. La segona manera és gestionar tot el que estigui relacionat amb la xarxa a través d'un nou servei anomenat Neutron.

Aquest servei actua com a router general, firewall global, pot fer de balancejador de càrrega... Amb Neutron doncs, veiem que podem configurar de manera més global tota la xarxa enlloc de ser una configuració a nivell de nodes.

A través del servei de xarxa, tant si és el que incorpora Nova com si és el servei Neutron, és important remarcar el tema de les adreces. Per defecte, quan es crea una nova instància, l'adreça IP assignada a cada instància és sempre d'una xarxa interna. Per fer accessible aquestes instàncies des d'una xarxa externa (com podria ser internet), s'han d'assignar adreces IP flotants. Aquestes són adreces reservades que seran enrutades cap a altres xarxes i que seran accessibles des d'altres xarxes. Correspon a les adreces IP públiques que tot servidor d'Internet té. Per tant, en un escenari on les instàncies han de ser accessibles des d'una xarxa externa, cada instància tindrà com a mínim un parell d'adreces IP: la privada i la pública.

7.5 Servei d'emmagatzemament de blocs (Cinder)

És important que una infraestructura Cloud tingui un emmagatzemament il·limitat (que sempre puguem afegir nodes per augmentar l'espai). Seria una mala pràctica utilitzar l'espai dels nodes de còmput per l'emmagatzemament de les dades d'usuari que contenen les instàncies ja que estaríem lligant el mètode de virtualització amb l'emmagatzemament, fent que sigui menys escalable. Per això hi ha un servei dedicat exclusivament a l'emmagatzemament d'aquestes dades, d'aquesta manera aquest espai deixa de dependre del node de còmput i passa a ser un node independent. Aquest emmagatzemament es coneix com emmagatzemament per blocs.

Com el servei de còmput, Cinder està compost per la part que s'instal·la en el controlador i la part que s'instal·la en el node que ofereix el servei.

En el node controlador, cal instal·lar el servei planificador (igual que també hi ha el planificador dels recursos de còmput). En canvi, en el node d'emmagatzemament, cal instal·lar un servei que farà que les unitats seleccionades formin part d'un espai d'emmagatzemament comú per a tots els nodes.

7.6 Servei d'emmagatzemament d'objectes (Swift)

L'emmagatzemament d'o objectes, a diferencia del de blocs, s'encarrega de guardar de manera individual cada un dels fitxers que vulguem afegir-hi, sense seguir cap estructura de carpetes. És un servei orientat a web, ja que la seva forma d'accés és a través d'una URL la qual ens retorna el fitxer demanat. Aquest servei és una copia del servei *Simple Storage Service (S3)* d'Amazon.

Aquest servei consta de tres mòduls: la part del controlador, un o més nodes de còmput i un servei proxy. El mòdul del controlador s'utilitzarà bàsicament per l'autenticació del servei. Per totes les altres operacions només s'utilitzarà el node d'emmagatzemament i el node proxy.

El node d'emmagatzemament serà el que té el servei i els discs durs a utilitzar per al servei. En canvi, el node proxy és el que sol fer tota la feina de localitzar els objectes en els nodes corresponents.

7.7 Servei d'automatització (Heat)

Hem vist que les infraestructures Clouds ens són útils per als casos que cal escalar la infraestructura de manera fàcil i ràpida. Hem vist com, a partir de l'arquitectura que tenen (principalment la separació de serveis per a funcionalitats diferents) es crear una infraestructura fàcilment escalable. El que ara ens seria útil és poder escalar de manera automàtica les nostres instàncies o l'espai d'emmagatzemament.

Heat és un sistema que funciona amb plantilles de text pla. Aquestes plantilles permeten escalar servidors (instàncies), les adreces IPs flotants (les públiques), els volums d'emmagatzemament... Aquest servei és el mateix servei que ofereix Amazon, anomenat CloudFormation. Al ser el mateix servei, permet utilitzar plantilles d'Amazon sense haver de modificar-les.

Algunes de les condicions d'escalat són la quantitat de memòria RAM utilitzada, l'ús de CPU, l'ús de la xarxa... Tot de manera individual per a cada instància. Per fer-ho possible, Heat utilitza a Ceilometer per mesurar els paràmetres necessaris de les instàncies per poder fer l'autoescalat. Veiem doncs, que la idea de Heat és l'automatització de certes tasques que tot i que es fan manualment, hi ha vegades que convé automatitzar-les per a no haver d'estar constantment supervisant el sistema.

7.8 Servei de mesures (Ceilometer)

Ceilometer és el servei encarregat de monitoritzar i recollir dades sobre els diferents elements de la infraestructura. Aquestes dades ens poden ser útils per a varies coses: per exemple, ens pot interessar fixar un preu per nombre d'instàncies utilitzades, o per l'espai d'emmagatzemament utilitzat per un usuari, pel nombre d'operacions d'entrada i sortida... Aquestes dades ens poden ser útils per a fixar uns preus segons el que s'hagi utilitzat i cobrar per aquest us.

També és capaç de monitoritzar internament les instàncies: us del processador, memòria RAM lliure i utilitzada, us de la xarxa... Aquest segon cas ens pot servir per autoescalar el sistema (amb l'ajut de Heat). Hem dit anteriorment que amb Heat podem automatitzar certes tasques. Per fer-ho, necessitàvem saber una sèrie de dades a partir de les quals decidiríem si escalar o no. Aquestes dades seran les proporcionarà Ceilometer a partir de la monitorització feta sobre la plataforma.

7.9 Altres components d'OpenStack

Ja hem vist tots els serveis dels que disposa OpenStack per a crear Clouds del tipus IaaS. Però hi ha altres elements que no hem anomenat.

OpenStack també té un gestor que permet fer certes tasques des d'un navegador web. Aquest gestor no ens permet configurar la plataforma (la configuració s'ha de fer tota a ma, a través dels fitxers de configuració di dels scripts de cada servei). El gestor ens permet bàsicament utilitzar la plataforma en mode usuari: podem crear i eliminar instàncies, afegir noves plantilles o crear-les a partir de les nostres instàncies, assignar adreces IPs flotants, afegir grups de seguretat, un visor VNC per a cada instància... Aquest gestor s'anomena Dashboard i no deixa de ser una implementació en mode gràfic de l'API de cada servei.

Un dels altres elements del que no s'ha parlat ha sigut el middleware utilitzat per passar missatges. Hi ha altres plataformes Cloud com CloudStack que incorporen un middleware propi. Aquest no és el cas d'OpenStack, ja que pot utilitzar diferents middlewares. El més recomanat és RabbitMQ, tot i que segons la distribució de Linux utilitzada, també es pot instal·lar Qpid. La idea del middleware és informar, a través d'un bus sobre la xarxa, de l'estat de tots els nodes i des les instruccions a executar. Aquest middleware és del tipus orientat a missatges, ja que la seva funció és enviar missatges informatius o ordres a executar.

Un aspecte important en aquestes plataformes és la disponibilitat del servei. És important que el servei estigui en funcionament el màxim temps possible. Degut a que a vegades, per falles del hardware o altres vegades per software, pot donar-se el cas de que algun servei no funcioni durant un cert temps. Per això, OpenStack s'aprofita de certes aplicacions ja existents per a Linux les quals permeten tenir la plataforma el màxim temps possible gràcies a la redundància dels serveis. No tots els serveis han d'estar redundats. Per exemple, el servei de computació no és necessari ja que el planificador assignarà les noves instàncies als nodes que estiguin funcionant correctament. Però altres serveis dels quals només hi ha un en execució i en depenen més d'ell, seria bo tenir-los redundats. Per exemple, la base de dades MySQL on es guarden les dades dels usuaris, les xarxes creades, les imatges afegides... També seria convenient tenir redundat el servei de middleware (RabbitMQ), ja que sense ell, ningun node sap res sobre els recursos de còmput ni d'emmagatzemament, per lo que no seria possible crear nous recursos. La majoria d'aquest serveis es poden en una configuració d'actiu/passiu. Això significa que sempre hi ha només un servei de cada en execució, i a través d'una aplicació que monitoritza els serveis, es possible detectar quan un servei ha deixat de funcionar per executar-lo en un altre node. Tot i això, és possible fer una configuració actiu/actiu. Aquesta configuració és més complexa i no està suportada per la majoria dels serveis d'OpenStack. Només és possible configurar-ho sobre la base de dades i el middleware.

Com hem dit, tant en la configuració actiu/passiu com actiu/actiu es fa mitjançant aplicacions externes d'OpenStack. Algunes de les aplicacions que s'utilitzen per a poder tenir una alta disponibilitat són per exemple, PaceMaker, un software per crear clústers d'alta disponibilitat, o HAProxy per al balanceig de càrrega i per crear adreces IP virtuals.

8 Conclusions i futurs treballs

Aquest treball s'ha desenvolupat partint de la base de que no tenia cap coneixement sobre el Cloud i per això s'ha hagut d'investigar tant la part teòrica (la virtualització, tipus de Cloud, la seva arquitectura i elements necessaris) com la part pràctica (instal·lació de la plataforma). Tot i haver altres hipervisors per a la virtualització, la idea és la mateixa per a tots, pel que pot considerar-se com un tema tancat en quan a l'anàlisi.

Però per altra part, el cas pràctic podria allargar-se molt més. El que s'ha vist ha sigut els elements principals d'una infraestructura Cloud: el node de còmput, de xarxa (configuració bàsica) i el d'emmagatzemament. Altres serveis han sigut instal·lats però no s'han utilitzat. Degut a que OpenStack és un software disponible per a pràcticament totes les distribucions Linux, no sempre funciona correctament en totes les distribucions: cada una té les seves propietats i a vegades cal seguir procediments diferents al procés d'instal·lació. Per això, aquest procés és quelcom llarg i complex, pel que seria bo resoldre-ho.

Per tant, algunes de les línies futures per a pròxims treballs que personalment crec que serien interessants són:

- **Ràpid desplegament de la plataforma:** existeixen algunes utilitats per Ubuntu (como MAAS i Juju) i per CentOS (Puppet) que permeten el ràpid desplegament de certs serveis, entre ells els d'OpenStack. Com no formava part de la meva planificació, no vaig endinsar-me gaire en aquest tema, però pel que vaig poder veure, tant per a Ubuntu com per a CentOS hi ha poca documentació pel que seria un bon tema a investigar (a part de que agilitza molt el procés d'afegir nous nodes).
- **Provar i instal·lar nous serveis:** han hagut serveis més complexes com el d'emmagatzemament d'objectes (Swift) o el de la xarxa complexa (Neutron) que no han sigut instal·lats ja que no eren imprescindibles. Altres serveis com el d'automatitzacions (Heat) o de monitorització (Ceilometer) han sigut instal·lats però no s'ha aprofundit en el seu funcionament. Considero que són dos elements indispensables per a plataformes mitjanes i grans ja que permeten evitar i simplificar certes tasques al usuari de la plataforma.
- **Plataforma d'alta disponibilitat:** segons la importància de la plataforma, l'alta disponibilitat pot convertir-se en un element indispensable. Per això crec que seria un altra tema a tractar de manera individual degut a la complexitat dels elements que intervenen.
- **Analitzar altres plataformes:** en aquest treball, el cas pràctic ha sigut realitzat amb OpenStack. Hi ha altres plataformes de virtualització com CloudStack o Eucalyptus que seria interessant analitzar el seu funcionament i els serveis que incorpora per a trobar el més adient a cada situació.
- **Altres modalitats de Cloud:** tal com hem dit, existeixen tres tipus de Cloud: IaaS (que ha sigut el que s'ha analitzat), PaaS i SaaS. Un possible treball seria muntar una plataforma PaaS com Cloud Foundry o OpenShift i analitzar el framework que incorpora.

Veiem doncs, que la feina realitzada en aquest treball es pot considerar la base per a altres possibles treballs els quals d'una manera o altra dependran d'OpenStack.

Considero que el Cloud és un servei molt important avui dia degut a que moltes empreses estan virtualitzant els seus servidors amb l'objectiu de reduir costos i millora la gestió d'aquests. Però segons el nombre de màquines virtuals necessàries, la infraestructura utilitzada d'un o dos host físics pot considerar-se inadequada degut a la poca escalabilitat. Per això, cada cop més les empreses estan migrant els seus serveis a alguna plataforma Cloud, ja sigui proporcionada per un proveïdor extern o de forma interna de l'empresa. És per això que considero molt important els coneixements que he adquirit al llarg d'aquest treball, tant a nivell teòric com a nivell pràctic.

Per últim, mencionar que els coneixements que he adquirit gràcies a la realització d'aquest treball, no serà quelcom que acabi aquí, sinó que com he dit anteriorment, cada cop hi ha més empreses que necessiten desplegar aquestes infraestructures, pel que tard o d'hora, és probable ho que tingui que fer al llarg de la meva vida. Amb el material i el temps que disposava, no he pogut arribar a veure amb detall tots els serveis, però si que he pogut veure i analitzar els més utilitzats i això m'ha motivat per seguir treballant en aquest tema en quan em sigui possible (ja sigui a nivell personal com a nivell professional).

AGRAÏMENTS

Per últim, m'agradaria agrair de manera individual la col·laboració i implicació de les persones que han fet possible aquest treball.

En primer lloc al meu tutor de treball, en Josep M. Basart, qui en tot moment ha estat disponible per a qualsevol dubte que he tingut i pels consells que m'ha anat donant al llarg del desenvolupament d'aquest treball. Gràcies pels teus comentaris; m'han ajudat a millorar la qualitat d'aquest treball.

També m'agradaria agrair a la meva parella, l'Ariadna, la seva implicació en haver-se llegit tot l'article i intentar entendre'l sense saber d'informàtica. La seva opinió des del punt de vist de fora de la informàtica em va servir per canviar algunes explicacions de manera que fossin més entenedores.

Finalment m'agradaria agrair a tots els professors que he tingut en totes les assignatures, en especial las del dEIC, ja que han sigut ells qui han fet que tingués aquest interès per a les xarxes i més en concret el que es refereix a les infraestructures tant de xarxa com de qualsevol altra cosa relacionada amb les xarxes (com els Clouds).

BIBLIOGRAFIA

[1] Hertzog, Raphaël; Mas, Roland. *The Debian administrator's handbook*. Llibre electrònic.

[2] "Understandig Full Virtualization, Paravirtualization, and Hardware Assist". http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf [Última consultat: 20/06/2014]

[3] "KVM – Kernel Based Virtual Machine". http://www.redhat.com/rhecm/rest-rhecm/jcr/repository/collaboration/jcr:system/jcr:versionStorage/5e7884ed7f00000102c317385572f1b1/1/jcr:frozenNode/rh:pdfFile.pdf [Última consulta: 20/06/2014]

[4] "Understanding The Cloud Computing Stack: SaaS, PaaS, IaaS". http://www.rackspace.com/knowledge_center/sites/default/files/whitepaper_pdf/Understanding-the-Cloud-Computing-Stack.pdf [Última consulta: 20/06/2014]

[5] "OpenStack Installation Guide for Ubuntu 12.04/14.04 (LTS)". http://docs.openstack.org/icehouse/install-guide/install/apt/content/ [Última consulta: 20/06/2014]

[6] De Haro Andrés, Edgard. *Manual d'OpenStack*. (Manual escrit per mi. Simplificació del manual oficial).

[7] "Api Complete Reference". 18 de Juny de 2014. http://developer.openstack.org/api-ref-guides/bk-api-ref.pdf [Última consulta: 20/06/2014]

[8] "OpenStack Command-Line interface Reference". 19 de Juny del 2014. http://docs.openstack.org/cli-reference/content/ [Última consulta: 20/06/2014]

[9] Jackson, Kevin; Bunch, Cody. *OpenStack Cloud Computing Cookbook*. 2ª Edició. Birmingham: Packt Publishing, Octubre 2013

[10] "Phoronix Text Suite" (software de *benchmarking*). http://www.phoronix-test-suite.com/

APÈNDIX
A1. Diagrama de l'arquitectura d'OpenStack