Arquitectura para la nube: Prácticas recomendadas

Enero de 2010

Última actualización: mayo de 2010

Jinesh Varia

jvaria@amazon.com

Introducción

Desde hace mucho tiempo, los arquitectos de software han descubierto e implementado varios conceptos y prácticas recomendadas para crear aplicaciones con un elevado nivel de escalabilidad. En la era del tera actual, estos conceptos son todavía más aplicables debido al aumento del crecimiento de los conjuntos de datos, la impredecibilidad de los patrones de tráfico y la necesidad de tiempos de respuesta más rápidos. Este documento reforzará y reiterará algunos de estos conceptos tradicionales, y tratará la forma en la que podrían evolucionar en el contexto de la informática de nube. Tratará, además, algunos conceptos sin precedentes como la elasticidad que ha surgido debido a la naturaleza dinámica de la nube.

Este documento está orientado a los *arquitectos de nube* que están preparándose para trasladar una aplicación de nivel empresarial de un entorno físico a un entorno de nube virtualizado. El enfoque que se adopta en este documento es el de resaltar conceptos, principios y prácticas recomendadas para la creación de nuevas *aplicaciones de nube* o migrar aplicaciones existentes a la nube.

Contexto

Como arquitecto de nube, es importante comprender cuáles son los beneficios que tiene la informática de nube. En este apartado aprenderá algunas de las ventajas tanto empresariales como técnicas que tiene la informática de nube, así como los diferentes servicios que AWS pone a su disposición.

Ventajas empresariales de la informática de nube

La creación de aplicaciones en la nube presenta ciertas ventajas empresariales obvias. A continuación mencionamos algunas de ellas:

Inversión inicial en infraestructura casi nula: si tiene que crear un sistema a gran escala podría costarle una fortuna invertir en un edificio, en seguridad física, en hardware (bastidores, servidores, enrutadores, fuentes de alimentación de respaldo), gestión del hardware (gestión energética, refrigeración) y en personal de operaciones. Por los elevados costes iniciales, este proyecto necesitaría varias rondas de aprobación antes de que el proyecto pudiera ponerse en marcha. Ahora, con la informática de nube, al estilo servicio, no existe ningún tipo de coste fijo ni de coste de puesta en marcha.

Infraestructura justo a tiempo: anteriormente, si su aplicación adquiría cierta popularidad y sus sistemas o su infraestructura no se ampliaba, podría convertirse en víctima de su propio éxito. Por otra parte, si invertía mucho y no conseguía popularidad, se convertía en víctima de su fracaso. Mediante la implementación de aplicaciones en la nube con aprovisionamiento automático justo a tiempo no tendrá que preocuparse de obtener con antelación capacidad para sistemas a gran escala. Este aspecto aumenta la agilidad, reduce los riesgos y disminuye los costes operativos, ya que tendrá que ampliar únicamente cuando *crezca* y solo pagará por lo que use.

Utilización de recursos más eficiente: los administradores del sistema suelen preocuparse a la hora de tener que conseguir hardware (cuando se quedan sin capacidad) y cuando existen mayores niveles de utilización de la infraestructura (cuando tienen demasiada capacidad y capacidad inactiva). Con la nube pueden gestionar los recursos con mayor eficacia y eficiencia en un entorno en el que las aplicaciones solicitan y ceden los recursos bajo demanda.

Costes según el uso: con el sistema de fijación de precios que imita a los servicios públicos, únicamente se le cobrará la infraestructura que haya utilizada. No tendrá que pagar por infraestructura asignada pero no utilizada. Esto añade una nueva dimensión a los ahorros de costes. Podrá percibir ahorros inmediatos (a veces de una forma tan inmediata como en su próxima factura mensual) cuando implemente una revisión de optimización que tiene como objetivo actualizar su aplicación de nube. Por ejemplo, si una capa de almacenamiento en caché puede reducir sus solicitudes de datos en un

70%, los ahorros empezarán a acumularse inmediatamente y verá la recompensa directamente en la próxima factura. Además, si está creando plataformas sobre la nube, podrá trasladar la misma estructura de coste basada en el uso, flexible y variable, a sus propios clientes.

Reducción del tiempo de comercialización: la paralelización es una de las mejores formas de acelerar el procesamiento. Si un trabajo que realiza un uso intensivo de la capacidad informática o de los datos que puede ejecutarse en paralelo tarda 500 horas en procesarse en una máquina, con las arquitecturas de nube [6] sería posible generar y ejecutar 500 instancias y procesar el mismo trabajo en una hora. Tener disponible una infraestructura elástica ofrece a la aplicación la posibilidad de aprovechar la paralelización de una forma rentable, reduciendo el tiempo de comercialización.

Ventajas técnicas de la informática de nube

Entre algunas de las ventajas técnicas de la informática de nube se incluyen las siguientes:

Automatización - "Infraestructura para la que pueden crearse secuencias de comandos": podrá crear sistemas de creación e implementación que podrán repetirse aprovechando la infraestructura programable (gestionada mediante API).

Auto-scaling (Escalado automático): podrá ampliar y reducir sus aplicaciones para adaptarlas a demandas inesperadas sin ningún tipo de intervención humana. La función Auto-scaling promueve la utilización y hace que el sistema funcione de una forma más eficiente.

Escalado proactivo: amplíe y reduzca su aplicación para cumplir con demandas previstas, con una planificación y conocimiento correcto de sus patrones de tráfico para que los costes no sean elevados a la hora de escalar.

Ciclo de vida del desarrollo más eficiente: los sistemas de producción podrán clonarse con facilidad para utilizarlos como entornos de desarrollo y prueba. Los entornos de ensayo podrán trasladarse fácilmente a entornos de producción.

Funciones de prueba mejoradas: no se quede nunca sin hardware para la realización de pruebas. Inyecte y automatice pruebas en todas las etapas del proceso de desarrollo. Podrá instaurar un laboratorio de pruebas instantáneo con entornos preconfigurados que estará vigente únicamente durante el periodo de la fase de pruebas.

Recuperación de desastres y continuidad empresarial: la nube proporciona una opción de menor coste para mantener una flota de servidores y almacenamiento de recuperación de desastres. Con la nube, puede sacar partido a la distribución geográfica y replicar el entorno en otros lugares en tan solo unos minutos.

"Desbordar" el tráfico hacia la nube: con solo unos clics y tácticas de equilibrado de carga eficaces podrá crear una aplicación totalmente a prueba de desbordamiento enrutando el exceso de tráfico hacia la nube.

Descripción de la nube de Amazon Web Services

La nube de Amazon Web Services (AWS) proporciona una infraestructura muy fiable y escalable para la implementación de soluciones a nivel de web, con costes de soporte y administración mínimos, y con más flexibilidad de la que cabría esperar de su propia infraestructura, ya sea en sus instalaciones o en un centro de datos.

AWS ofrece actualmente diversos servicios de infraestructura. El diagrama que aparece a continuación le presentará la terminología de AWS, y le ayudará a comprender la forma en la que su aplicación puede interactuar con los diferentes Amazon Web Services, además de la forma en la que los diferentes servicios interactúan entre sí.

Amazon Elastic Compute Cloud (Amazon EC2)[1] es un servicio web que proporciona capacidad informática con tamaño modificable en la nube. Podrá agrupar el sistema operativo, las aplicaciones de software y la configuración asociada en una *Amazon Machine Image* (AMI). Estas AMI podrán utilizarse para aprovisionar varias *instancias* virtualizadas, así como desactivarlas utilizando sencillas llamadas a servicios web para ampliar o reducir la capacidad rápidamente, para adaptarla a sus cambios de necesidades. Podrá comprar *Instancias bajo demanda* en las que paga por instancias por hora o *Instancias reservadas*, en las que mediante un único pago de bajo importe recibirá un índice de uso más bajo para ejecutar la instancia que con una Instancia bajo demanda o *Instancias puntuales*, en las que podrá ofertar por capacidad no utilizada y reducir su coste aún más. Las instancias pueden ejecutarse en una o más *regiones* geográficas. Cada región cuenta con varias *Zonas de disponibilidad*. Las Zonas de disponibilidad son regiones diferentes que están diseñadas para estar aisladas de fallos que se produzcan en otras Zonas de disponibilidad, y que proporcionan conectividad de red de baja latencia a otras Zonas de disponibilidad de la misma Región.

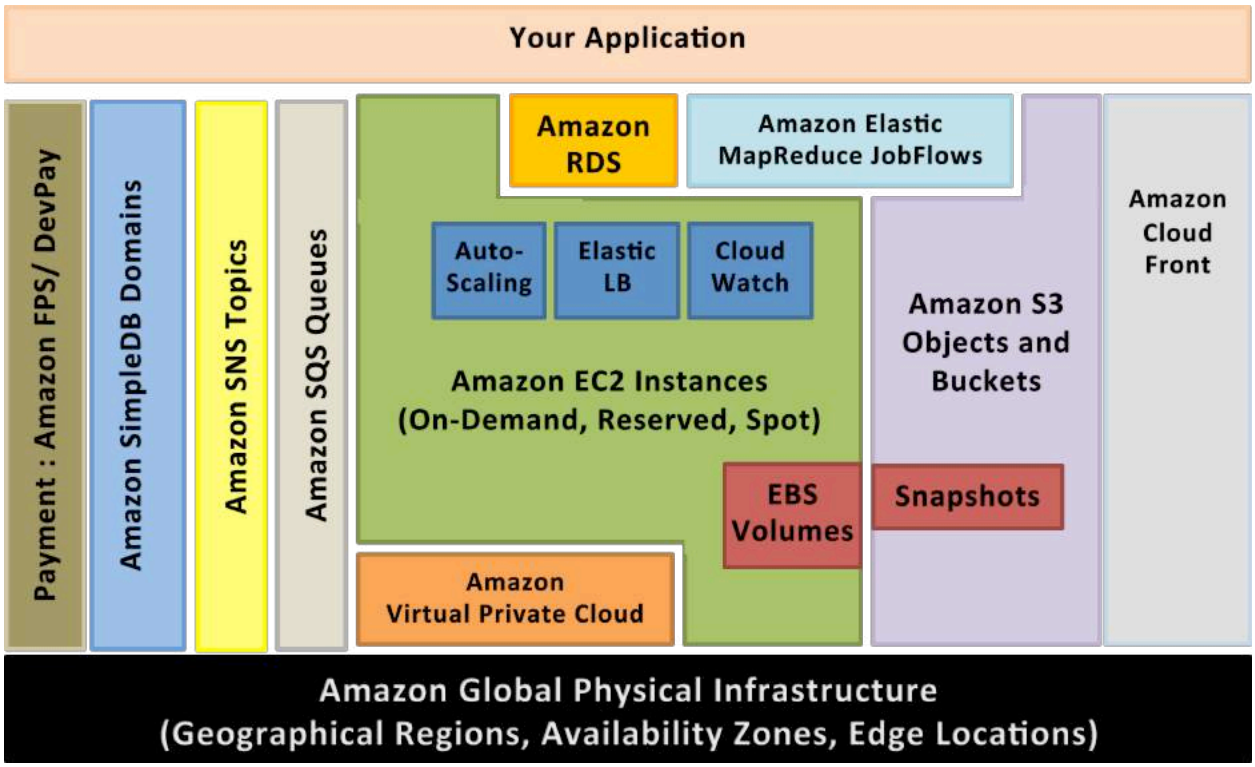

Ilustración 1: Amazon Web Services

Las direcciones *IP elásticas* le permiten designar una dirección IP estática y asignarla mediante programación a una instancia. Podrá habilitar funciones de supervisión en una instancia Amazon EC2 mediante Amazon CloudWatch[2] para conseguir visibilidad de la utilización de recursos, rendimiento operativo y patrones de demanda globales (incluyendo métricas tales como utilización de CPU, lecturas y escrituras en disco y tráfico de la red). Podrá crear un *Grupo de escalado automático* utilizando la función Auto-scaling[3] para ampliar o reducir su capacidad en determinadas condiciones basándose en la métrica que Amazon CloudWatch recopila. También podrá distribuir el tráfico entrante mediante la creación de un *equilibrador de carga elástico* utilizando el servicio Elastic Load Balancing[4]. Los volúmenes

[1] Tiene a su disposición más información sobre Amazon EC2 en http://aws.amazon.com/ec2
[2] Tiene a su disposición más información sobre Amazon CloudWatch en http://aws.amazon.com/cloudwatch/
[3] Tiene a su disposición más información sobre la función Auto-scaling en http://aws.amazon.com/auto-scaling
[4] Tiene a su disposición más información sobre la función Elastic Load Balancing en http://aws.amazon.com/elasticloadbalancing

Elastic Block Storage (EBS)[5] *de Amazon* proporcionan almacenamiento persistente conectado a la red a las instancias de Amazon EC2. Es posible crear *instantáneas* en un punto del tiempo coherentes de volúmenes EBS y almacenarlas en Amazon Simple Storage Service (Amazon S3)[6].

Amazon S3 es un almacén de datos distribuido y de alta resistencia. A través de una sencilla interfaz de servicios web podrá almacenar y recuperar grandes cantidades de datos como *objeto*s en *depósitos* (contenedores) en cualquier momento, y desde cualquier parte de la web, utilizando verbos HTTP estándar. Es posible distribuir copias de objetos y almacenarlas en caché en 14 *ubicaciones de borde* situadas en diversas partes del mundo mediante la creación de una *distribución* utilizando el servicio Amazon CloudFront[7], un servicio web para la distribución de contenido (tanto estático como en transmisión). Amazon SimpleDB[8] es un servicio web que ofrece la funcionalidad básica de una base de datos: búsquedas en tiempo real y consultas sencillas de datos estructurados pero sin complejidad operativa. Podrá organizar el conjunto de datos en *dominios* y ejecutar consultas en todos los datos almacenados en un dominio determinado. Los dominios son colecciones de *artículos* descritos mediante *pares atributo-valor*. Amazon Relational Database Service[9] (Amazon RDS) proporciona una forma sencilla de configurar, utilizar y escalar una base de datos relacional en la nube. Podrá ejecutar una *Instancia de base de datos* y obtener acceso a una base de datos MySQL con todas las funciones sin tener que preocuparse de tareas comunes de bases de datos tales como copias de seguridad, gestión de revisiones, etc.

Amazon Simple Queue Service (Amazon SQS)[10] es una cola distribuida alojada fiable, altamente escalable, diseñada para almacenar *mensajes* mientras viajan entre los equipos y los componentes de la aplicación.

Amazon Simple Notifications Service (Amazon SNS) [11]proporciona una forma sencilla de avisar a aplicaciones o a personas desde la nube mediante la creación de *Temas* y la utilización de un protocolo de publicación-suscripción.

Amazon Elastic MapReduce[12] ofrece un marco de trabajo Hadoop alojado que se encuentra en ejecución en la infraestructura web de Amazon Elastic Compute Cloud (Amazon EC2) y Amazon Simple Storage Service (Amazon S3), y le permite crear *JobFlows* (Flujos de trabajo) personalizados. JobFlow es una secuencia de *pasos* de MapReduce.

Amazon Virtual Private Cloud (Amazon VPC)[13] le permite ampliar su red corporativa a una nube privada que se encuentra dentro de AWS. Amazon VPC utiliza el modo de túnel IPSec que le permite crear una conexión segura entre una puerta de enlace de su centro de datos y una puerta de enlace de AWS.

AWS ofrece además diversos servicios de pago y de facturación[14] que hacen uso de la infraestructura de pago de Amazon.

Todos los servicios de infraestructura de AWS ofrecen un marco de fijación de precios similar al de servicios públicos como la luz o el agua, y que no exige compromisos a largo plazo ni contratos. Podrá, por ejemplo, pagar por la hora de uso de una instancia Amazon EC2, y por el gigabyte de transferencia de datos y almacenamiento en el caso de Amazon

[5] Tiene a su disposición más información sobre la función Elastic Block Store en http://aws.amazon.com/ebs
[6] Tiene a su disposición más información sobre la función Amazon S3 en http://aws.amazon.com/s3
[7] Tiene a su disposición más información sobre la función Amazon CloudFront en http://aws.amazon.com/cloudfront
[8] Tiene a su disposición más información sobre la función Amazon SimpleDB en http://aws.amazon.com/simpledb
[9] Tiene a su disposición más información sobre la función Amazon RDS en http://aws.amazon.com/rds
[10] Tiene a su disposición más información sobre la función Amazon SQS en http://aws.amazon.com/sqs
[11] Tiene a su disposición más información sobre la función Amazon SNS en http://aws.amazon.com/sns
[12] Tiene a su disposición más información sobre la función Amazon ElasticMapReduce en http://aws.amazon.com/elasticmapreduce
[13] Tiene a su disposición más información sobre la función Amazon Virtual Private Cloud en http://aws.amazon.com/vpc
[14] Tiene más información sobre Amazon Flexible Payments Service (Servicio de pagos flexibles de Amazon) en http://aws.amazon.com/fps y sobre Amazon DevPay en http://aws.amazon.com/devpay

S3. En el sitio web de AWS tiene más información sobre cada uno de estos servicios y sus estructuras de precios según el uso.

Tenga en cuenta que la utilización de la nube de AWS no le exige tener que sacrificar la flexibilidad y el control al que está acostumbrado:

- Podrá utilizar el modelo de programación, lenguaje o sistema operativo (Windows, OpenSolaris o cualquier distribución de Linux) que elija.
- Tendrá la libertad de elegir los productos AWS que mejor se adapten a sus necesidades; podrá utilizar los servicios tanto de forma independiente como combinando los servicios que desee.
- Dado que AWS ofrece recursos (almacenamiento, ancho de banda y estructuras informáticas) de tamaño modificable, podrá consumir muchos o pocos servicios y pagar únicamente por lo que utilice.
- Podrá utilizar las herramientas de gestión de sistemas que haya utilizado anteriormente y ampliar su centro de datos a la nube.

Conceptos sobre la nube

La nube refuerza algunos conceptos antiguos sobre la creación de arquitecturas de Internet altamente escalables [13], e introduce algunos conceptos nuevos que modifican por completo la forma en la que las aplicaciones se crean e implementan. Por lo tanto, cuando avance desde el concepto a la implementación, es posible que tenga la sensación de que "Todo ha cambiado, pero nada es diferente". La nube cambia diversos procesos, patrones, prácticas y filosofías, y refuerza algunos principios de arquitectura orientada a servicios tradicionales que ya ha aprendido, ya que ahora son más importantes que nunca. En este apartado verá algunos de estos nuevos conceptos de nube y conceptos de SOA reiterados.

Las aplicaciones tradicionales se desarrollaban con ciertas ideas preconcebidas que tenían sentido desde un punto de vista económico y arquitectónico en el momento en el que se desarrollaban. La nube aporta nuevas filosofías que debe comprender, y que se tratan a continuación:

Creación de arquitecturas escalables

Para poder sacar partido a una infraestructura escalable resulta de vital importancia crear una arquitectura escalable.

La nube está diseñada para proporcionar escalabilidad, desde un punto de vista conceptual, infinita. Sin embargo, no podrá aprovechar toda la escalabilidad de la infraestructura si su arquitectura no es escalable. Ambas deben trabajar mano a mano. Tendrá que identificar los componentes monolíticos y los cuellos de botella de su arquitectura, identificar aquellas áreas en las que no puede sacar partido al aprovisionamiento bajo demanda en su infraestructura, y trabajar para *refactorizar* su aplicación con el objetivo de aprovechar la infraestructura escalable y sacar partido a la nube.

Características de una aplicación realmente escalable:

- El aumento de recursos deriva en un aumento de rendimiento proporcional
- Un servicio escalable es capaz de gestionar la heterogeneidad
- Un servicio escalable es eficiente desde un punto de vista operativo
- Un servicio escalable es sólido
- Un servicio escalable debe ser más rentable cuando crece (el coste por unidad disminuye al aumentar el número de unidades)

Estas son algunas de las características que deben convertirse en parte inherente de su aplicación, y si diseña su arquitectura teniendo las características anteriores en mente, su arquitectura y su infraestructura trabajarán de forma conjunta para ofrecerle la escalabilidad que está buscando.

Descripción de la elasticidad

El gráfico que aparece a continuación muestra los diferentes enfoques que un arquitecto de nube puede adoptar para ampliar sus aplicaciones con el fin de cumplir la demanda.

Enfoque de ampliación: no preocuparse sobre la arquitectura de aplicación escalable y realizar importantes inversiones en equipos más grandes y más potentes (escalado vertical) para dar cabida a la demanda. Este enfoque suele funcionar hasta un punto, pero puede costarle una fortuna (vea "Enorme gasto económico" en el diagrama), o la demanda podría ser superior a la capacidad antes de que se implemente el nuevo sistema (vea "Acaba de perder sus clientes" en el diagrama).

El enfoque de escalabilidad horizontal: consiste en la creación de una arquitectura que se escala de forma horizontal, invirtiendo en infraestructura en pequeños fragmentos. La mayoría de aplicaciones empresariales y webs de gran escala siguen este patrón distribuyendo los componentes de sus aplicaciones, federando sus conjuntos de datos y empleando un diseño orientado al servicio. Este enfoque suele ser más eficaz que un enfoque de escalabilidad vertical. Sin embargo, sigue siendo necesario predecir la demanda en intervalos regulares para, posteriormente, implementar la infraestructura en fragmentos para dar cabida a la demanda. Es por esto que a veces deriva en un exceso de capacidad ("quema de efectivo") y en una supervisión manual constante ("quema de ciclos humanos"). Además, no suele funcionar si la aplicación es víctima de fuego viral (a menudo conocido como el efecto barra punto[15]).

Nota: ambos enfoques tienen costes de puesta en marcha, y ambos enfoques presentan una naturaleza reactiva.

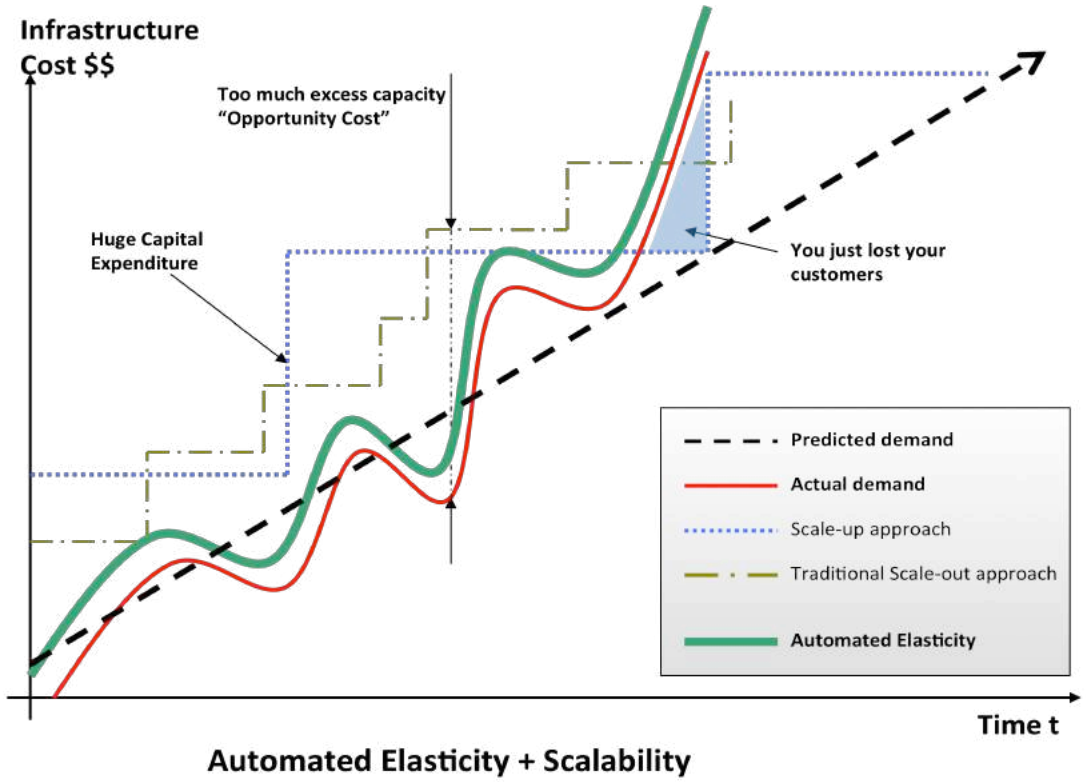

Ilustración 2: Elasticidad automatizada

[15] http://en.wikipedia.org/wiki/Slashdot_effect (en inglés)

La infraestructura tradicional necesita la predicción de la cantidad de recursos informáticos que su aplicación va a utilizar durante un periodo de varios años. Si sus previsiones son inferiores a la realidad, sus aplicaciones no contarán con la potencia suficiente para gestionar el tráfico inesperado, lo que podría derivar en falta de satisfacción por parte del cliente. Si sus previsiones superan a la realidad, estará desperdiciando dinero con recursos superfluos.

La naturaleza bajo demanda y elástica del *enfoque de nube* (Elasticidad automatizada), sin embargo, permite que la infraestructura esté estrechamente adaptada (a medida que aumenta y disminuye) con la demanda real, aumentando de este modo la utilización global y reduciendo los costes.

La elasticidad es una de las propiedades fundamentales de la nube. La elasticidad consiste en la potencia de escalar los recursos informáticos ampliándolos y reduciéndolos con una fricción mínima. Es importante comprender que la elasticidad impulsará en último término la mayoría de ventajas de la nube. Como arquitecto de la nube, deberá interiorizar este concepto y ponerlo en funcionamiento en su arquitectura de aplicaciones para sacar el mayor beneficio posible de la nube.

Tradicionalmente, las aplicaciones se han creado para infraestructuras fijas, rígidas y preaprovisionadas. Las empresas nunca han tenido la necesidad de aprovisionar e instalar servidores todos los días. Es por ello que la mayoría de infraestructuras de software no afrontan la rápida implementación o reducción de hardware. Dado que el tiempo de aprovisionamiento y la inversión inicial para adquirir nuevos recursos eran demasiado altos, los arquitectos de software nunca invertían tiempo ni recursos en optimizar la utilización del hardware. Si el hardware en el que la aplicación se estaba ejecutando estaba infrautilizado, no había ningún problema. La noción de "elasticidad" dentro de una arquitectura se ignoraba, ya que la idea de poder contar con nuevos recursos en cuestión de minutos era imposible.

Esta concepción debe cambiar con la nube. La informática de nube optimiza el proceso de adquisición de los recursos necesarios. Ya no hay que realizar pedidos con antelación ni conservar hardware no utilizado. Ahora, los arquitectos de la nube pueden solicitar los recursos que necesitan minutos antes de necesitarlos o automatizar el proceso de obtención, aprovechando la enorme escalabilidad y el rápido tiempo de respuesta de la nube. La misma situación se aplica a la hora de liberar los recursos innecesarios o infrautilizados cuando ya no los necesite.

Si no puede adoptar el cambio e implementar la elasticidad en la arquitectura de su aplicación puede que no tenga posibilidad de sacar el máximo partido a la nube. Como arquitecto de nube, debe adoptar una forma de pensar creativa, y pensar en formas en las que puede implementar la elasticidad en su aplicación. Por ejemplo, la infraestructura que solía ejecutar versiones nocturnas diarias y realizar pruebas de regresión y de unidad todas las noches a las 2 de la mañana (a menudo denominada como "Caja de control de calidad/versión") permanecía inactiva durante el resto del día. Ahora, con la infraestructura elástica, pueden ejecutarse versiones nocturnas en cajas que son "vivas" y pagar únicamente las dos horas de la noche durante las que están en funcionamiento. Del mismo modo, una aplicación web de tickets de resolución de problemas internos que solía funcionar a la máxima capacidad (5 servidores 24x7x365) para cumplir la demanda durante el día ahora puede aprovisionarse para que se ejecute bajo demanda (5 servidores desde las 9:00 hasta las 17:00 y 2 entre las 17:00 y las 9:00) en base al patrón de tráfico.

El diseño de arquitecturas de nube elásticas inteligentes, para que la infraestructura funcione únicamente cuando la necesita, es todo un arte. La elasticidad debe ser uno de los requisitos de diseño arquitectónicos de un sistema. La pregunta que debe hacerse es "¿qué componentes o capas de mi arquitectura de aplicaciones pueden ser elásticos? ¿Qué se necesitará para hacer que ese componente sea *elástico*? ¿Qué impacto tendrá implementar la elasticidad en toda mi arquitectura de sistema?

En el próximo apartado podrá ver técnicas específicas para implementar la elasticidad en sus aplicaciones. Para poder sacar el máximo partido posible a las ventajas de la nube es importante crear la arquitectura con esta perspectiva.

Sin tener miedo a las limitaciones

Cuando decida migrar sus aplicaciones a la nube e intente corresponder las especificaciones de su sistema con las que tiene a su disposición en la nube, podrá observar que la nube no tiene las especificaciones exactas que tenía el recurso con el que contaba en sus instalaciones. Por ejemplo, "La nube no proporciona X GB de RAM en un servidor" o "Mi base de datos necesita más IOPS de las que puedo tener en una sola instancia".

Debe comprender que la nube proporciona *recursos abstractos*, y que son potentes cuando los combina con el modelo de aprovisionamiento bajo demanda. No debe sentir miedo ni limitaciones al utilizar recursos de nube, ya que es importante comprender que, incluso si no puede conseguir una réplica exacta de su hardware en el entorno de nube, tiene la posibilidad de conseguir más recursos a través de la nube para compensar esa necesidad.

Por ejemplo, si la nube no le proporciona una cantidad exacta o mayor de RAM en un servidor, pruebe a utilizar una memoria caché distribuida como, por ejemplo*memcached*[16], o a particionar sus datos entre varios servidores. Si sus bases de datos necesitan más IOPS y no se corresponde directamente con la que le ofrece la nube, existen numerosas recomendaciones entre las que puede elegir en función de los tipos de datos y del caso de uso. Si se trata de una aplicación que realiza un número importante de lecturas, podrá distribuir la carga de lecturas entre un número de esclavos sincronizados. Asimismo, también puede utilizar un algoritmo de *sharding* [10] que redirija los datos hacia donde deben estar, o utilizar varias soluciones de agrupamiento en clúster de base de datos.

Mirando hacia atrás, cuando combine las posibilidades de aprovisionamiento bajo demanda con la flexibilidad, podrá comprobar que las aparentes limitaciones pueden romperse de formas que mejorarán la escalabilidad y el rendimiento global del sistema.

Administración virtual

La llegada de la nube ha cambiado el papel del Administrador del sistema para convertirlo en "Administrador del sistema virtual". Esto sencillamente significa que las tareas diarias realizadas por estos administradores ahora son incluso más interesantes, ya que aprenden más sobre las aplicaciones y deciden qué es lo mejor para la empresa. El Administrador del sistema ya no tendrá que aprovisionar servidores, instalar software y cablear dispositivos de red, ya que todo ese trabajo sucio se realiza con unos clics y llamadas a línea de comandos. La nube promueve la automatización, ya que la infraestructura es programable. Los administradores del sistema deberán ascender por la pila tecnológica y aprender a gestionar recursos de nube abstractos utilizando secuencias de comandos.

Del mismo modo, el papel de Administrador de la base de datos pasa a convertirse en "Administrador de la base de datos virtual", en el que gestiona los recursos mediante una consola web, ejecuta secuencias de comandos que añaden nueva capacidad mediante programación en caso de que el hardware de la base de datos se quede sin capacidad y que automatiza los procesos cotidianos. El DBA virtual debe ahora aprender nuevos métodos de implementación (imágenes de máquinas virtuales), adoptar nuevos modelos (paralelización de consultas, redundancia geográfica y replicación asíncrona [11]), volver a rediseñar el enfoque arquitectónico de los datos (sharding [9], particionamiento horizontal [13], federación[14]) y aprovechar las diferentes opciones de almacenamiento disponibles en la nube para diferentes tipos de conjuntos de datos.

En la empresa tradicional, los desarrolladores de aplicaciones podrían no trabajar codo con codo con los administradores de red, y los administradores de red podrían no tener conocimientos sobre la aplicación. Es por ello que

[16] http://www.danga.com/memcached/

diversas posibles optimizaciones de la capa de red y de la capa de la arquitectura de la aplicación se ignoran. Con la nube, los dos roles se han convertido en uno. A la hora de crear la arquitectura de las aplicaciones del futuro, las empresas deben promover la dispersión del conocimiento entre los dos roles, y entender que se están fusionando.

Prácticas recomendadas sobre la nube

En este apartado aprenderá algunas prácticas recomendadas que le ayudarán a crear una aplicación en la nube.

Diseñe teniendo el fallo en mente y nada fallará

Regla básica: sea pesimista al diseñar arquitecturas en la nube. Asuma que las cosas van a fallar. En otras palabras, diseñe, implemente y despliegue en todo momento para la recuperación automatizada desde un fallo.

En concreto, debe asumir que su hardware *va a* fallar. Asuma que *tendrán lugar* cortes de suministro eléctrico. Asuma que algún desastre *va a* atacar su aplicación. Asuma que *va a* verse sobrepasado algún por un número de solicitudes por segundo superior al esperado. Asuma que, con el paso del tiempo, su aplicación de software también va a fallar. Siendo pesimista pensará en estrategias de recuperación durante el diseño, lo que le ayuda a mejorar el diseño del sistema en términos globales.

Si adquiere conciencia de que las cosas van a fallar e incorpora esa forma de pensar en su infraestructura, y crea mecanismos que gestionan los fallos antes de que se produzca el desastre para afrontar una infraestructura escalable, terminará creando una arquitectura tolerante a fallos que está optimizada para la nube.

Preguntas que debe hacerse: ¿Qué pasa si falla uno de los nodos del sistema? ¿Cómo podría reconocer ese fallo? ¿Cómo puedo sustituir ese nodo? ¿Para qué tipo de situaciones tengo que planificar? ¿Cuáles son mis puntos únicos de fallo? Si hay un equilibrador de carga frente a una matriz de servidores de aplicaciones, ¿qué pasa si el equilibrador de carga falla? Si su arquitectura contiene maestros y esclavos, ¿qué pasa si el nodo maestro falla? ¿Cómo se produce la conmutación por error y de qué forma se instancia un nuevo esclavo y se sincroniza con el maestro?

Al igual que diseña teniendo en mente los fallos del hardware, debe diseñar también considerando los fallos del software. Preguntas que debe hacerse: ¿Qué pasa con mi aplicación si los servicios dependientes cambian su interfaz? ¿Qué pasa si un servicio descendente agota su tiempo de espera o devuelve una excepción? ¿Qué pasa si las claves de la caché superan el límite de memoria de una instancia?

Cree mecanismos para gestionar ese fallo. Algunas de las estrategias que pueden ayudarle en caso de producirse un fallo son las siguientes:

1. Cuente con una estrategia de copia de seguridad y restauración coherente para sus datos y automatícela
2. Cree hilos de proceso que se reanuden al reiniciar
3. Permita que el estado del sistema pueda volver a sincronizarse volviendo a cargar mensajes de las colas
4. Conserve imágenes virtuales preconfiguradas y preoptimizadas para admitir (2) y (3) en la ejecución/arranque
5. Evite sesiones en memoria o contexto de usuario con estado, desplace estos elementos a almacenes de datos.

Una buena arquitectura de nube debe ser impermeable a reinicios y reejecuciones. En GrepTheWeb (tratado en el documento Cloud Architectures (Arquitecturas de nube) [6]), mediante la utilización de una combinación de Amazon SQS y Amazon SimpleDB, la arquitectura general del controlador es muy resistente a los fallos que se mencionan en este apartado. Por ejemplo, si la instancia en la que el hilo del controlador se estaba ejecutando muere, podrá activarse y reanudar el estado anterior como si nada hubiera pasado. Esto se consiguió creando una Amazon Machine Image (Imagen de máquina de Amazon), que cuando se ejecuta retira de cola todos los mensajes de la cola de Amazon SQS y lee sus estados desde un dominio Amazon SimpleDB durante el reinicio.

Diseñar asumiendo que el hardware subyacente va a fallar le preparará para el futuro cuando se produzca el fallo.

Este principio de diseño le ayudará a diseñar aplicaciones compatibles con operaciones, como también se resalta en el documento de Hamilton [11]. Si puede ampliar este principio para medir de forma proactiva y equilibrar la carga dinámicamente, es posible que pueda asumir las variaciones de rendimiento de la red y del disco, producidas debido a la naturaleza con varios inquilinos de la nube.

Tácticas específicas de AWS para implementar esta práctica recomendada:

1. Conmutación por error que utiliza correctamente las IP elásticas: una IP elástica es una IP estática que puede volver a asignarse de forma dinámica. Podrá reasignar y conmutar por error rápidamente a otro conjunto de servidores, de forma que el tráfico se redirija a los nuevos servidores. Funciona a la perfección cuando se desea actualizar de versiones antiguas a versiones nuevas, o en caso de producirse fallos en la red
2. Uso de varias Zonas de disponibilidad: las Zonas de disponibilidad son, desde un punto de vista conceptual, como centros de datos lógicos. Mediante la implementación de su arquitectura en varias zonas de disponibilidad podrá asegurarse elevados niveles de disponibilidad. Utilice la funcionalidad de implementación Amazon RDS Multi-AZ [21] para replicar de forma automática actualizaciones de la base de datos en varias Zonas de disponibilidad.
3. Mantenga una Amazon Machine Image de forma que pueda restaurar y clonar entornos con gran facilidad en una Zona de disponibilidad diferente; mantenga varios esclavos de base de datos entre Zonas de disponibilidad y configure replicación activa.
4. Utilice Amazon CloudWatch (o diversas herramientas de supervisión en tiempo real de código abierto) para conseguir mayor visibilidad y poder emprender las acciones adecuadas en caso de producirse un fallo de hardware o disminución del rendimiento. Configure un grupo de Auto scaling para mantener un tamaño de flota fijo, de forma que sustituya instancias de Amazon EC2 en mal estado por instancias nuevas.
5. Utilice Amazon EBS y configure trabajos cronológicos para que se carguen automáticamente a Amazon S3 instantáneas incrementales, y para que los datos sean persistentes con independencia de sus instancias.
6. Utilice Amazon RDS y defina el periodo de retención de copias de seguridad, para que pueda realizar copias de seguridad automáticas.

Desacople sus componentes

La nube refuerza el principio de diseño de SOA que dicta que *cuanto menos vinculados estén los componentes del sistema, más y mejor se escala*.

La clave es crear componentes que no tengan estrechas dependencias entre sí, para que si un componente fuera a morir (fallar), entrar en inactividad (no responder) o permanecer ocupado (lenta respuesta) por alguna razón, el resto de componentes del sistema estén creados para que puedan seguir funcionando como si no estuviera produciéndose ningún tipo de fallo. En esencia, la ausencia de estrechas dependencias aísla las diversas capas y componentes de su aplicación, de forma que cada componente interactúe de forma asíncrona con el resto, y los trata como una "caja negra". Por ejemplo, en el caso de la arquitectura de una aplicación web, puede aislar el servidor de aplicaciones del servidor web y de la base de datos. El servidor de aplicaciones no conoce al servidor web y viceversa, esto concede desacoplamiento entre estas dos capas, por lo que no existen dependencias desde la perspectiva del código o la funcionalidad. En el caso de arquitecturas de procesamiento por lotes, puede crear componentes *asíncronos* que son independientes del resto.

Preguntas que debe hacerse: ¿Qué componente o función empresarial podría aislarse de la aplicación monolítica actual y podría ejecutarse de forma independiente? Y, a continuación, ¿cómo puedo añadir más instancias de dicho componente sin acabar con mi sistema actual y, al mismo tiempo, ofrecer servicio a más usuarios? ¿Qué esfuerzos habrá que poner en práctica para encapsular el componente, de forma que pueda interactuar con el resto de los componentes de forma asíncrona?

Desacoplar sus componentes, crear sistemas *asíncronos* y escalar de forma horizontal son aspectos muy importantes en el contexto de la nube. No solo va a permitirle escalar añadiendo más instancias del mismo componente, sino que además va a permitirle diseñar innovadores modelos híbridos en los que ciertos componentes siguen ejecutándose en sus instalaciones, mientras que otros componentes pueden sacar partido al escalado de la nube y utilizar la nube para contar con potencia informática y ancho de banda adicionales. De esta forma, y con un esfuerzo mínimo, podrá "desbordar" el exceso de tráfico a la nube mediante la implementación de tácticas de equilibrado de carga inteligente.

Mediante las *colas de mensajes* puede construirse un sistema poco acoplado. Si se utiliza una cola o búfer para conectar dos componentes cualquiera, podrá admitir la concurrencia, la alta disponibilidad y los picos de carga. De esta forma, el sistema entero sigue funcionando incluso si determinadas piezas de componentes no están disponibles de forma temporal. Si uno de los componentes muere o deja de estar disponible de forma disponible, el sistema almacenará los mensajes en el búfer y se procesarán cuando el componente vuelva a estar activo.

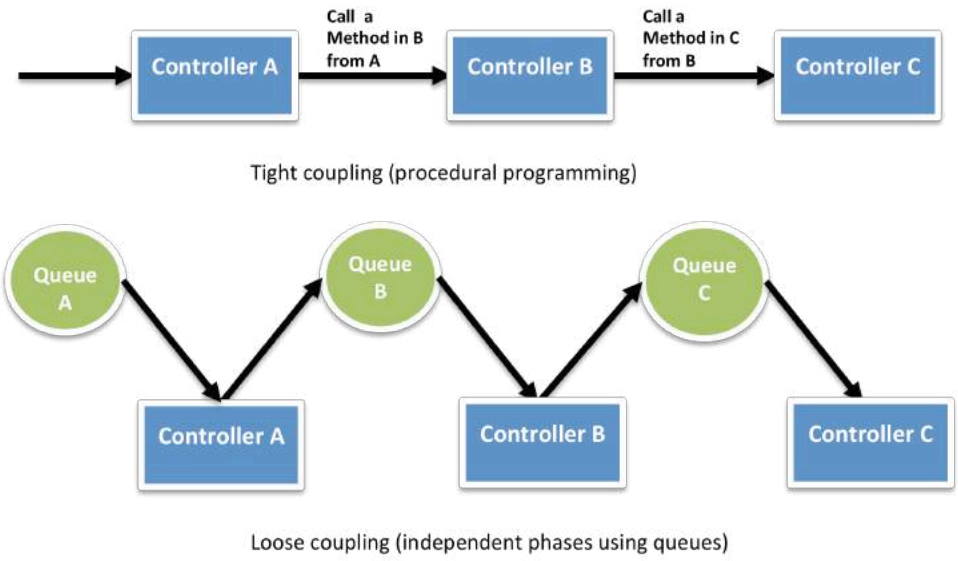

Ilustración 3: desacople de componentes mediante la utilización de Colas

En el documento Cloud Architectures (Arquitecturas de nube) verá un uso intensivo de las colas en la Arquitectura GrepTheWeb [6]. En GrepTheWeb, si llegan al servidor de forma repentina lotes de solicitudes (una situación de sobrecarga inducida por Internet) o el procesamiento de expresiones regulares lleva una cantidad de tiempo superior a la media (baja velocidad de respuesta de un componente), las colas de Amazon SQS almacenan en el búfer las solicitudes durante cierto tiempo, para que estos retrasos no afecten a otros componentes.

Tácticas específicas de AWS para implementar esta práctica recomendada:

1. Use Amazon SQS para aislar los componentes [18]
2. Use Amazon SQS como búferes entre componentes [18]
3. Diseñe cada componente como si expusiera una interfaz de servicio y fuera responsable de su propia escalabilidad en todas las dimensiones adecuadas e interactuara con otros componentes de

forma asíncrona

4. Agrupe la construcción lógica de un componente en una Amazon Machine Image, para que pueda implementarse con mayor frecuencia
5. Haga que sus aplicaciones sean lo más "sin estado" posible. Almacene el estado de la sesión fuera del componente (en Amazon SimpleDB, en caso de ser posible)

Implemente la elasticidad

La nube aporta un nuevo concepto de elasticidad en sus aplicaciones. La elasticidad puede implementarse de tres formas:

1. Proactive Cyclic Scaling (Escalado cíclico proactivo): escalado periódico que tiene lugar en intervalos fijos (todos los días, todas las semanas, todos los meses, todos los trimestres)
2. Proactive Event-based Scaling (Escalado basado en eventos proactivos): escalado que tiene lugar únicamente cuando espera un aumento de solicitudes de tráfico repentino debido a un evento empresarial planificado (lanzamiento de un nuevo producto, campañas de marketing)
3. Escalado automático basado en la demanda. Mediante la utilización de un servicio de supervisión su sistema puede enviar desencadenadores para que se realicen las acciones adecuadas de forma que escale ampliando o reduciendo en base a métricas (utilización de los servidores o de la e/s de la red, por ejemplo)

Para poder implementar la elasticidad primero debe automatizarse el proceso de implementación y optimizar el proceso de configuración y creación. Esto le garantizará que el sistema pueda ampliarse sin ningún tipo de intervención por parte del hombre.

De esta forma conseguirá beneficios de coste inmediatos, ya que la utilización global aumenta al asegurarse de que sus recursos están estrechamente vinculados con la demanda, en lugar de ejecutar potencialmente servidores que están infrautilizados.

Automatice su infraestructura

Una de las ventajas más importantes que tiene el uso de un entorno de nube es la posibilidad de utilizar las API de la nube para automatizar su proceso de implementación. Se recomienda que se tome el tiempo necesario para crear un proceso de implementación automatizado en las fases iniciales del proceso de migración, y que no espere hasta el final. La creación de un proceso de implementación automatizado y que puede repetirse le ayudará a reducir los errores, y facilitará un proceso de actualización eficiente y escalable.

Para automatizar el proceso de implementación:

- Cree una biblioteca de "recetas:" secuencias de comando de instalación y configuración utilizadas con frecuencia
- Gestione el proceso de configuración e implementación utilizando agentes agrupados en el interior de una AMI
- Incluya sus instancias en el cargador de arranque

Incluya sus instancias en el cargador de arranque

Permita que sus instancias le realicen al arrancar la siguiente pregunta: "¿quién soy y cuál es mi función?" Toda instancia debe tener una función ("servidor de base de datos", "servidor de aplicaciones", "servidor esclavo", en el caso de una aplicación web) que desempeñar en el entorno. Esta función debe trasladarse como argumento durante la ejecución, lo que indica a la AMI cuando se instancia los pasos que se deben tomar después de que esté arrancada. Al arrancar, las instancias deben obtener los recursos necesarios (código, secuencias de comandos, configuración) en base a su papel y "adjuntarse" a ellas mismas a un clúster para realizar su función. Ventajas de incluir sus instancias en el cargador de arranque:

1. Recrear el entorno (desarrollo, ensayo, producción) con solo unos clics y con esfuerzo mínimo
2. Más control de los recursos abstractos basados en la nube
3. Reducción de los errores de implementación inducidos por el hombre
4. Crear un entorno con recuperación y detección automáticas, que es más resistente a los fallos de hardware

Tácticas específicas de AWS para la automatización de su infraestructura

1. Defina grupos de escalado automático para diferentes clústeres utilizando la función Amazon Auto-scaling de Amazon EC2.
2. Supervise las métricas de su sistema (CPU, memoria, E/S de disco, E/S de red) utilizando Amazon CloudWatch y tome las medidas adecuadas (ejecución de nuevas AMI de forma dinámica utilizando el servicio Auto-scaling) o envíe notificaciones.
3. Guarde y recupere información de configuración de la máquina de forma dinámica: utilice Amazon SimpleDB para recuperar los datos de configuración durante el arranque de una instancia (p. ej. cadenas de conexión a la base de datos). También puede utilizarse SimpleDB para almacenar información sobre una instancia, como por ejemplo su dirección IP, el nombre de la máquina y su función.
4. Diseñe un proceso de creación que envíe las versiones más recientes a un depósito de Amazon S3, descargue la versión más reciente de una aplicación durante el arranque del sistema.
5. Invierta en la creación de herramientas de gestión de recursos (secuencias de comandos automatizadas, imágenes preconfiguradas) o utilice herramientas de gestión de configuración de código abierto inteligentes como, por ejemplo, Chef[17], Puppet[18], CFEngine [19]o Genome[20].
6. Agrupe Just Enough Operating System (JeOS[21]) y sus dependencias de software en una Amazon Machine Image para que sea más fácil de gestionar y mantener. Transmita los archivos o parámetros de configuración en el momento de ejecución y recupere los datos sobre usuarios[22] y los metadatos de la instancia tras la ejecución.
7. Reduzca el agrupamiento y el tiempo de ejecución arrancando desde volúmenes Amazon EBS [23] y conectando varios volúmenes Amazon EBS a una instancia. Cree instantáneas de volúmenes comunes y comparta instantáneas[24] entre cuentas siempre que proceda.
8. Los componentes de la aplicación no deben asumir el estado ni la ubicación del hardware en el que se está ejecutando. Por ejemplo, vincule de forma dinámica la dirección IP de un nuevo nodo al clúster. Conmute por error automáticamente e inicie un nuevo clon en caso de producirse un fallo.

Piense en paralelo

La nube crea paralelización sin esfuerzo. No importa si está solicitando datos a la nube, almacenando datos en ella, procesando datos (o ejecutando trabajos) en la nube, como arquitecto de la nube debe internalizar el concepto de la paralelización a la hora de diseñar arquitecturas en la nube. Se recomienda no solo implementar la paralelización allí donde sea posible, sino que además se insta a que se automatice, ya que la nube le permite crear un proceso que puede repetirse siempre que quiera con mucha facilidad.

[17] Puede encontrar más información sobre Chef en http://wiki.opscode.com/display/chef/Home (en inglés)
[18] Puede encontrar más información sobre Puppet en http://reductivelabs.com/trac/puppet/ (en inglés)
[19] Puede encontrar más información sobre CFEngine en http://www.cfengine.org/ (en inglés)
[20] Puede encontrar más información sobre Genome en http://genome.et.redhat.com/ (en inglés)
[21] http://es.wikipedia.org/wiki/JeOS
[22] Puede encontrar más información sobre los metadatos de instancia y los datos de usuario en http://docs.amazonwebservices.com/AWSEC2/latest/DeveloperGuide/index.html?AESDG-chapter-instancedata.html (en inglés)
[23] Puede encontrar más información sobre la función Boot From Amazon EBS en http://developer.amazonwebservices.com/connect/entry.jspa?externalID=3121 (en inglés)
[24] Vea cómo compartir una instantánea en http://aws.amazon.com/ebs/

En lo que respecta a acceder (recuperar y almacenar) datos, la nube está diseñada para gestionar un enorme número de operaciones en paralelo. Para conseguir el máximo rendimiento y posibilidades de procesamiento, debe sacar partido a la *paralelización de solicitudes*. Incluir sus solicitudes en varios subprocesos mediante la utilización de varios subprocesos simultáneos recuperará los datos a una velocidad mayor que si se solicitan de forma secuencial. Por lo tanto y siempre que sea posible, los procesos de una aplicación de nube deben realizarse a prueba de subprocesos mediante una filosofía de "no compartir nada" y aprovechar el subprocesamiento múltiple.

En lo que respecta a procesar o ejecutar solicitudes en la nube, sacar partido a la paralelización es, todavía, más importante. Una práctica recomendada general, en el caso de una aplicación web, es distribuir las solicitudes entrantes entre diversos servidores web asíncronos utilizando un equilibrador de carga. En el caso de una aplicación de procesamiento por lotes, su nodo maestro puede generar varios nodos trabajadores esclavos que procesan las tareas en paralelo (de la misma forma que en marcos de trabajo de procesamiento distribuido como, por ejemplo, Hadoop[25])

La belleza de la nube brilla cuando combina elasticidad y paralelización. Su aplicación de nube puede poner en marcha un clúster de instancias informáticas que se aprovisionan en tan solo unos minutos con unas llamadas a API, realizar un trabajo ejecutando tareas en paralelo, almacenar los resultados y finalizar todas las instancias. La aplicación GrepTheWeb, de la que se habla en [6], es un ejemplo de ello.

Tácticas específicas de AWS para la paralelización:

1. Incluya en varios subprocesos sus solicitudes de Amazon S3, tal y como se especifica en el documento Prácticas recomendadas 2]
2. Incluya en varios subprocesos sus solicitudes GET y BATCHPUT de Amazon SimpleDB [3][4] [5]
3. Cree un JobFlow utilizando el servicio Amazon Elastic MapReduce Service para cada uno de sus procesos por lotes diarios (indexación, análisis de registros, etc.) lo que computará el trabajo en paralelo y le permitirá ahorrar tiempo.
4. Utilice el servicio Elastic Load Balancing y divida su carga entre varios servidores de aplicaciones web *de forma dinámica*

Mantenga los datos dinámicos más cerca de la estructura informática y los datos estáticos más cerca del usuario final

En términos generales, se recomienda conservar sus datos lo más cerca posible de sus elementos informáticos o de procesamiento, con el objetivo de reducir la latencia. En la nube, esta práctica recomendada adquiere especial relevancia e importancia, ya que a menudo debe afrontar las latencias de Internet. Además, en la nube, está pagando el ancho de banda que entra y sale de la nube por gigabyte de transferencia de datos, y el coste puede ascender con mucha rapidez.

Si es necesario procesar una gran cantidad de datos que reside en el exterior de la nube, es posible que resulte más barato y rápido "enviar" y transferir los datos primero a la nube y, posteriormente, realizar el cómputo. Por ejemplo, en el caso de una aplicación de almacenamiento de datos, es aconsejable trasladar el conjunto de datos a la nube y, a continuación, efectuar consultas en paralelo en el conjunto de datos. En el caso de las aplicaciones web que almacenan y recuperan datos de bases de datos relacionales, se recomienda trasladar a la nube tanto el servidor de aplicaciones como la base de datos y el servidor de aplicaciones.

[25] http://hadoop.apache.org/

Si los datos se generan en la nube, las aplicaciones que consuman los datos deberán implementarse también en la nube, para que puedan sacar el máximo partido posible a las transferencias de datos libres dentro de la nube y a cifras de latencia más bajas. Por ejemplo, en el caso de una aplicación web de comercio electrónico que genera registros y datos de secuencia de clic, se recomienda ejecutar el analizador de registros y los motores de generación de informes en la nube.

Por el contrario, si los datos son estáticos y no van a cambiar con frecuencia (como, por ejemplo, en el caso de imágenes, vídeo, audio, PDF, JS, archivos CSS), se recomienda utilizar un servicio de distribución de contenido para que los datos estáticos se almacenen en la caché en una ubicación de borde que se encuentre más cerca del usuario final (el solicitante), para así reducir la latencia de acceso. Gracias al almacenamiento en caché, los servicios de distribución de contenido ofrecen un acceso más rápido a objetos populares.

Tácticas específicas de AWS para implementar esta práctica recomendada:

1. Envíe sus unidades de datos a Amazon utilizando el servicio Import/Export (Importar/Exportar)[26]. Es posible que sea más rápido trasladar grandes cantidades de datos utilizando sneakernet[27] que cargando los archivos a Internet.
2. Utilice la misma Zona de disponibilidad para ejecutar un clúster de máquinas
3. Cree una distribución de su depósito Amazon S3 y deje que Amazon CloudFront almacene en caché el contenido de dicho depósito en sus 14 ubicaciones de borde de todo el mundo

Prácticas recomendadas de seguridad

En un entorno con varios inquilinos, los arquitectos de la nube suelen expresar sus preocupaciones en lo que respecta a la seguridad. *La seguridad debe implementarse en todas las capas de la arquitectura de la aplicación de nube.* La seguridad física suele gestionarla su proveedor de servicios (Documento técnico sobre seguridad [7]), lo que representa una ventaja adicional a la hora de utilizar la nube. La seguridad a nivel de red y de aplicaciones es responsabilidad suya, y debe implementar las prácticas recomendadas que correspondan a su negocio. En este apartado podrá acceder a información sobre herramientas, funciones y directrices específicas sobre cómo proteger su aplicación de nube en el entorno de AWS. Le recomendamos que haga uso de las herramientas y funciones mencionadas para implementar seguridad básica y, a continuación, implementar prácticas recomendadas de seguridad utilizando métodos estándar según proceda o según considere adecuado.

Proteja los datos en tránsito

Si necesita intercambiar información confidencial o delicada entre un navegador y un servidor web, configure SSL en su instancia de servidor. Necesitará un certificado de una autoridad de certificación externa como, por ejemplo, VeriSign[28] o Entrust[29]. La clave pública que incluye el certificado autentica su servidor en el navegador, y actúa como base para crear la clave de sesión compartida que se utiliza para cifrar los datos en ambas direcciones.

Cree una Nube virtual privada realizando algunas llamadas a línea de comandos (mediante Amazon VPC). Esto le permitirá utilizar sus propios recursos aislados de forma lógica dentro de la nube de AWS, para posteriormente conectar estos recursos directamente a su propio centro de datos utilizando las conexiones IPSec VPN estándar del sector.

[26] Tiene a su disposición más información sobre Amazon Import Export Services en http://aws.amazon.com/importexport (en inglés)
[27] http://es.wikipedia.org/wiki/Sneakernet
[28] http://www.verisign.com/ssl/
[29] http://www.entrust.net/ssl-products.htm

También podrá instalar [15] un servidor OpenVPN en una instancia Amazon EC2, e instalar el cliente OpenVPN en los PC de todos los usuarios.

Proteja los datos que se encuentran en reposo

Si le preocupa almacenar datos confidenciales y delicados en la nube, deberá cifrar los datos (archivos independientes) antes de cargarlos a la nube. Podrá cifrar los datos utilizando cualquier herramienta de código abierto[30] o comercial[31] basada en PGP antes de almacenarlos como objetos de Amazon S3 y descifrarlos tras su descarga. Esta suele ser una práctica recomendada a la hora de crear aplicaciones conformes a HIPPA [8] que deben almacenar Información Protegida sobre Salud (PHI).

En Amazon EC2, el cifrado de archivos depende del sistema operativo. Las instancias de Amazon EC2 que ejecuten Windows podrán utilizar la función Encrypting File System (EFS) (Sistema de archivos con cifrado) integrada [16]. Esta función gestionará el cifrado y descifrado de archivos y carpetas de forma automática, y hará que el proceso sea transparente para el usuario [19]. Sin embargo, y a pesar de su nombre, EFS no cifra todo el sistema de archivos, sino que cifra archivos independientes. Si necesita contar con un volumen totalmente cifrado, baraje la posibilidad de utilizar el producto de código abierto TrueCrypt[32]; se integrará a la perfección con volúmenes EBS con formato NBS. Las instancias de Amazon EC2 que ejecuten Linux pueden montar volúmenes EBS utilizando diversos enfoques (EncFS[33], Loop-AES[34], dm-crypt[35], TrueCrypt[36]). De la misma forma, las instancias de Amazon EC2 que ejecuten OpenSolaris podrán sacar partido al Soporte de cifrado de ZFS[37] [20]. Independientemente del enfoque que elija, el cifrado de archivos y volúmenes en Amazon EC2 le ayuda a proteger archivos y a registrar datos de forma que únicamente los usuarios y procesos del servidor puedan ver los datos en texto plano, y que toda persona o aplicación que se encuentre fuera del servidor vea únicamente datos cifrados.

Independientemente del sistema operativo o tecnología que escoja, el cifrado de los datos que se encuentran en reposo presenta un reto: la gestión de las claves utilizadas para cifrar los datos. Si pierde las claves, perderá los datos para siempre, y si las claves se ponen en riesgo, los datos podrían estar en una situación comprometida. Por lo tanto, asegúrese de estudiar las capacidades de gestión de claves de los productos que elija, y establezca un procedimiento que minimice el riesgo que supone perder las claves.

Además de proteger sus datos de accesos no autorizados, piense cómo puede protegerlos de los desastres. Realice instantáneas periódicas de los volúmenes Amazon EBS para asegurarse de que presentan elevados niveles de resistencia y disponibilidad. Las instantáneas son de naturaleza incremental, y se almacenan en Amazon S3 (en ubicaciones geográficas independientes), y pueden restaurarse con tan solo unos clics o llamadas de línea de comandos.

Proteja sus credenciales de AWS

AWS proporciona dos tipos de credenciales de seguridad: claves de acceso AWS y certificados X.509. Su clave de acceso AWS está compuesta por dos elementos: su *Id. de clave de acceso* y su *clave de acceso secreto*. Cuando utilice la API REST o Query, tendrá que utilizar su clave de acceso secreta para calcular la firma que desea incluir en su solicitud de

[30] http://www.gnupg.org
[31] http://www.pgp.com/
[32] http://www.truecrypt.org/
[33] http://www.arg0.net/encfs
[34] http://loop-aes.sourceforge.net/loop-AES.README
[35] http://www.saout.de/misc/dm-crypt/
[36] http://www.truecrypt.org/
[37] http://www.opensolaris.org/os/community/zfs/

autenticación. Para evitar la modificación de los datos mientras están en tránsito, todas las solicitudes deben enviarse a través de HTTPS.

Si su Amazon Machine Image (AMI) está ejecutando procesos que tienen que establecer comunicación con otros servicios web de AWS (para sondear la cola Amazon SQS o para leer objetos desde Amazon S3, por ejemplo), uno de los errores de diseño que suelen cometerse es incrustar las credenciales de AWS en la AMI. En lugar de incrustar las credenciales, deben transmitirse como argumentos durante la ejecución, y cifrarse antes de enviarlos [17].

Si se pone en riesgo la seguridad de su clave de acceso secreta, debe obtener una nueva rotando[38] a una nueva Id. de clave de acceso. Se recomienda que, como práctica recomendada, incorpore un mecanismo de rotación de clave en la arquitectura de su aplicación, para que pueda utilizarla de forma regular u ocasional (por ejemplo, cuando un empleado descontento se marcha de la empresa) para asegurarse de que las claves que se han puesto en peligro no duran eternamente.

Asimismo, también puede utilizar certificados X.509 para la autenticación en determinados servicios AWS. El archivo de certificado contiene su clave pública en un certificado DER codificado en base64. Un archivo independiente contiene la correspondiente clave privada PKCS#8 codificada en base64.

AWS admite la autenticación mediante varios factores[39] como elemento de protección adicional para trabajar con la información de su cuenta en aws.amazon.com y en AWS Management Console[40].

Proteja su aplicación

Cada instancia Amazon EC2 está protegida por uno o más *grupos de seguridad*[41], conjuntos de reglas con nombre que especifican qué tráfico de red de entrada debe trasladarse a su instancia. Podrá especificar puertos TCP y UDP, tipos y códigos ICMP y direcciones de origen. Los grupos de seguridad le conceden protección similar a la que ofrece un firewall para las instancias que se encuentran en ejecución. Por ejemplo, las instancias que pertenecen a una aplicación web pueden tener la siguiente configuración de grupo de seguridad:

[38] http://aws.amazon.com/about-aws/whats-new/2009/08/31/seamlessly-rotate-your-access-credentials/ (en inglés)
[39] Tiene a su disposición más información sobre la Autenticación mediante varios factores en http://aws.amazon.com/mfa/ (en inglés)
[40] AWS Management Console http://aws.amazon.com/console/ (en inglés)
[41] Tiene a su disposición más información sobre Security Group (Grupo de seguridad) en http://docs.amazonwebservices.com/AWSEC2/2009-07-15/UserGuide/index.html?using-network-security.html (en inglés)

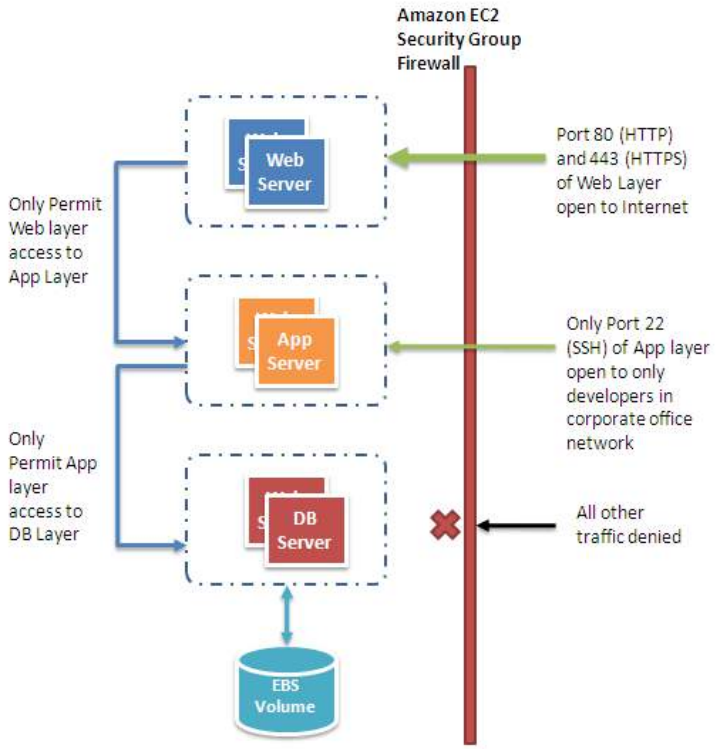

Ilustración 4: Proteger su aplicación web mediante grupos de seguridad de Amazon EC2

Otra forma que tiene a su disposición de restringir el tráfico entrante es configurar firewalls de software en sus instancias. Las instancias de Windows pueden utilizar el firewall integrado[42]. Las instancias de Linux pueden utilizar *netfilter*[43] e *iptables*.

Con el paso del tiempo se descubren errores en el software, y para resolverlos son necesarias revisiones. Para poder maximizar la seguridad de su aplicación debe garantizar los siguientes principios básicos:

- Descargue de forma regular revisiones del sitio web del proveedor y actualice sus AMI
- Vuelva a implementar las instancias desde las nuevas AMI y pruebe sus aplicaciones para asegurarse de que sus revisiones no rompen nada. Asegúrese de que la AMI más reciente esté implementada en *todas* las instancias
- Invierta en secuencias de comando de prueba, para poder ejecutar comprobaciones de seguridad de forma periódica y automatizar el proceso
- Asegúrese de que el software de otros fabricantes esté configurado con la configuración más segura
- No ejecute nunca sus procesos con el inicio de sesión *root* o *Administrador*, a menos que sea estrictamente necesario

Siguen siendo de aplicación todas las prácticas de seguridad estándar anteriores a la era de la nube, como son la adopción de buenas prácticas de generación de código y el aislamiento de datos confidenciales. Estas prácticas deben seguir implementándose.

[42] http://technet.microsoft.com/en-us/library/cc779199(WS.10).aspx, marzo de 2003 (en inglés)
[43] http://www.netfilter.org/

A modo de análisis, la nube le libera de la complejidad que supone la seguridad física y le otorga el control mediante herramientas y funciones diseñadas para que pueda proteger su aplicación.

Futuras vías de investigación

El día en el que las aplicaciones dejarán de tener conciencia del hardware físico no está demasiado lejano. De la misma forma que para enchufar un microondas y encenderlo no hace falta ningún tipo de conocimiento sobre la electricidad, debe poder *conectarse* una aplicación a la nube para que reciba la potencia que necesita para ejecutarse, como si fuera un electrodoméstico. En su papel de arquitecto, gestionará recursos informáticos, de almacenamiento y de red abstractos, en lugar de servidores físicos. Las aplicaciones seguirán funcionando incluso si el hardware físico subyacente falla, se retira o se sustituye. Las aplicaciones se adaptarán por sí mismas a las fluctuaciones de los patrones de demanda mediante la implementación de recursos *instantánea* y automática, consiguiendo así los más altos niveles de utilización en todo momento. Escalabilidad, Seguridad, Alta disponibilidad, Tolerancia a fallos, Posibilidad de realizar pruebas y Elasticidad serán propiedades configurables de la arquitectura de la aplicación, y formarán parte automatizada e intrínseca de la plataforma en la que están creadas.

Sin embargo aún no estamos en este punto. En la actualidad puede crear aplicaciones en la nube con alguna de estas calidades mediante la implementación de las prácticas recomendadas que se resaltan en el documento técnico. Las prácticas recomendadas para las arquitecturas informáticas de nube seguirán evolucionando, y como investigadores no solo nos debemos centrar en mejorar la nube, sino también en crear herramientas, tecnologías y procesos que faciliten a los desarrolladores y a los arquitectos conectar con facilidad las aplicaciones.

Conclusión

Este documento ha proporcionado pautas prescriptivas a los arquitectos de nube para el diseño de aplicaciones de nube eficientes.

Centrándose en los conceptos y las prácticas recomendadas (como, por ejemplo, diseñando para el fallo, desacoplando los componentes de la aplicación, comprendiendo e implementando la elasticidad, combinándola con la paralelización, e integrando la seguridad en todos y cada uno de los aspectos de la arquitectura de la aplicación), los arquitectos de nube pueden comprender las consideraciones de diseño necesarias para crear aplicaciones de nube altamente escalables.

La nube de AWS ofrece servicios de infraestructura de pago según el uso que presentan excelentes niveles de fiabilidad. Las tácticas específicas de AWS resaltadas en este documento ayudarán a diseñar aplicaciones de nube utilizando estos servicios. Como investigador, le recomendamos que juegue con estos servicios comerciales, aprenda del trabajo de otras personas, mejore, potencie e invente más informática de nube.

Agradecimientos

El autor está profundamente agradecido a Jeff Barr, Steve Riley, Paul Horvath, Prashant Sridharan y Scot Marvin por sus comentarios en los primeros borradores de este documento. Agradecimientos especiales a Matt Tavis por facilitar datos de gran valor. Sin sus aportaciones este documento no habría sido posible.

Parte del contenido de este documento técnico tiene como referencia un capítulo escrito por el mismo autor, que aparece en el libro "Cloud Computing: Paradigms and Patterns",

Referencias y más documentación

1. **Amazon S3 Team, Best Practices for using Amazon S3,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1904, 2008-11-26 (en inglés)
2. **Amazon S3 Team, Amazon S3 Error Best Practices,** http://docs.amazonwebservices.com/AmazonS3/latest/index.html?ErrorBestPractices.html, 2006-03-01 **(en inglés)**
3. **Amazon SimpleDB Team, Query 201: Tips and Tricks for Amazon SimpleDB Query,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1232&categoryID=176, 2008-02-07 **(en inglés)**
4. **Amazon SimpleDB Team, Building for Performance and Reliability with Amazon SimpleDB,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1394&categoryID=176, 2008-04-11 (en inglés)
5. **Amazon SimpleDB Team, Query 101: Building Amazon SimpleDB Queries,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1231&categoryID=176, 2008-02-07 **(en inglés)**
6. **J. Varia, Cloud Architectures,** http://jineshvaria.s3.amazonaws.com/public/cloudarchitectures-varia.pdf, 2007-07-01 (en inglés)
7. **Amazon Security Team, Overview of Security Processes,** http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf, 2009-06-01 (en inglés)
8. **Amazon Web Services Team, Creating HIPPA-Compliant Medical Data Applications With AWS,** http://awsmedia.s3.amazonaws.com/AWS_HIPAA_Whitepaper_Final.pdf, 2009-04-01 **(en inglés)**
9. D. Obasanjo, Building Scalable Databases: Pros and Cons of Various Database Sharding Schemes, http://www.25hoursaday.com/weblog/2009/01/16/BuildingScalableDatabasesProsAndConsOfVariousDatabaseShardingSchemes.aspx, 2009-01-16 (en inglés)
10. D. Pritchett, Shard Lessons, http://www.addsimplicity.com/adding_simplicity_an_engi/2008/08/shard-lessons.html, 2008-08-24 (en inglés)
11. J. Hamilton, On Designing and Deploying Internet-Scale Services, 2007, *21st Large Installation System Administration conference (LISA 'O7)*, http://mvdirona.com/jrh/talksAndPapers/JamesRH_Lisa.pdf (en inglés)
12. J. Dean and S. Ghemawat, MapReduce: Simplified data processing on large clusters2004-12-01, In Proc. of the 6th OSDI, http://labs.google.com/papers/mapreduce-osdi04.pdf (en inglés)
13. T. Schlossnagle, *Scalable Internet Architectures*, Sams Publishing , 2006-07-31,
14. M. Lurie, The Federation: Database Interoperability, http://www.ibm.com/developerworks/data/library/techarticle/0304lurie/0304lurie.html, 2003-04-23 (en inglés)
15. E. Hammond, Escaping Restrictive/Untrusted Networks with OpenVPN on EC2, http://alestic.com/2009/05/openvpn-ec2, 2009-05-02 (en inglés)
16. R. Bragg, The Encrypting File System, http://technet.microsoft.com/en-us/library/cc700811.aspx, 2009 (en inglés)
17. S. Swidler, How to keep your AWS credentials on an EC2 instance securely, http://clouddevelopertips.blogspot.com/2009/08/how-to-keep-your-aws-credentials-on-ec2.html, 2009-08-31 (en inglés)
18. **Amazon SQS Team, Building Scalable, Reliable Amazon EC2 Applications with Amazon SQS,** http://sqs-public-images.s3.amazonaws.com/Building_Scalabale_EC2_applications_with_SQS2.pdf, 2008 (en inglés)
19. Microsoft Support Team, Best Practices For Encrypting File System (Windows), http://support.microsoft.com/kb/223316, 2009)
20. Solaris Security Team, ZFS Encryption Project (OpenSolaris), http://www.opensolaris.org/os/project/zfs-crypto/, 2009-05-01 (en inglés)

21. **Amazon RDS Team, Amazon RDS Multi-AZ Deployments,**
http://docs.amazonwebservices.com/AmazonRDS/latest/DeveloperGuide/Concepts.DBInstance.html#Concepts.MultiAZ
(en inglés) 2010-05-15

22. **Amazon SimpleDB Team, Amazon SimpleDB Consistency Enhancements**
http://developer.amazonwebservices.com/connect/entry.jspa?externalID=3572 2010-02-24 **(en inglés)**