Architecture dans le nuage : les bonnes pratiques

Janvier 2010

Dernière mise à jour - Mai 2010

Jinesh Varia

jvaria@amazon.com

Introduction

Cela fait plusieurs années que les architectes logiciels découvrent et mette en œuvre divers concepts et bonnes pratiques visant à développer des applications hautement dimensionnables. Dans « l'ère du Téra » actuelle, ces concepts sont d'autant plus applicables du fait de la croissance continue des ensembles de données, de l'imprévisibilité des schémas de trafic et de la demande en matière de temps de réponse toujours plus rapides. Ce document a pour but de renforcer et de réitérer certains de ces concepts traditionnels tout en envisageant leur évolution potentielle dans le contexte du cloud computing. Il traite également de certains concepts innovants, tels que l'élasticité, qui sont apparus en raison de la nature dynamique du nuage.

Ce document est destiné aux *architectes cloud* qui se préparent à déplacer une application d'entreprise d'un environnement physique fixe vers un environnement virtuel dans le nuage. Le but de ce document consiste à mettre en évidence les concepts, les principes et les bonnes pratiques qui sont nécessaires pour créer de nouvelles *applications dans le nuage* ou pour réaliser la migration d'applications existantes dans le nuage.

Contexte

En tant qu'architecte cloud, il est important de comprendre les avantages proposés par le cloud computing. Vous allez apprendre dans cette section quels sont quelques uns des avantages commerciaux et techniques du cloud computing, tout en vous familiarisant avec les divers services AWS actuellement disponibles.

Les avantages commerciaux ducloud computing

Il existe de clairs avantages commerciaux résultant du développement d'applications dans le nuage. Voici quelques uns de ces avantages :

Un investissement initial en infrastructure pratiquement nul : si vous devez développer un système à grande échelle, l'investissement en espace immobilier, en sécurité physique, en équipements (racks, serveurs, routeurs, alimentations de secours) peut vous coûter une véritable fortune, sans parler de la gestion des équipements (gestion de l'énergie, refroidissement) et du personnel d'exploitation. En raison de ces coûts initiaux élevés, il faudrait généralement que le projet passe par un cycle de plusieurs approbations de la direction avant même de pouvoir être lancé. Mais maintenant, grâce au cloud computing à la demande, il n'y a ni coût fixe ni coût de démarrage.

Une infrastructure juste-à-temps : dans le passé, si votre application devenait populaire et que vos systèmes ou votre infrastructure n'étaient pas extensibles, vous tombiez victimes de votre propre réussite. À l'inverse, si votre investissement était lourd et que vous ne deveniez pas populaire, vous deveniez une victime de votre échec. En déployant des applications dans le nuage en effectuant votre propre approvisionnement juste-à-temps, vous n'avez pas à vous inquiéter de la capacité de pré-approvisionnement des systèmes de grande échelle. Cela augmente la souplesse tout en réduisant les risques et les coûts d'exploitation, car vous dimensionnez uniquement en fonction de votre *croissance* et vous ne payez que pour ce que vous utilisez.

Un usage plus efficace des ressources : les administrateurs systèmes se préoccupent généralement de l'approvisionnement en matériel (quand leur capacité devient insuffisante) et d'un meilleur usage de l'infrastructure (en cas de capacité excessive et de capacité non utilisée). Avec le nuage, ils peuvent gérer les ressources avec plus d'efficacité et d'efficience en prévoyant des applications qui demandent et cèdent les ressources à la demande.

Un coût basé sur le niveau d'utilisation : avec des tarifs à la demande, vous n'êtes facturé que sur l'infrastructure qui a été utilisée. Vous ne payez pas pour des infrastructures attribuées mais non utilisées. Cela ajoute une nouvelle dimension aux économies de coûts. Vous pouvez constater des économies de coût immédiates (parfois dès la

facturation du mois suivant) lorsque vous déployez un patch d'optimisation pour mettre à jour votre applications dans le nuage. Par exemple, une couche de mémoire cache peut réduire les demandes de données de 70 %, augmentant les économies immédiatement, ce qui vous permet d'être récompensé dès la facture suivante. De plus, si vous développez des plateformes sur le nuage, vous pouvez transmettre à vos propres clients la même structure flexible, variable et basée sur l'utilisation.

Un délai de mise sur le marché réduit : la parallélisation est l'une des meilleures façons d'accélérer le traitement. Si une tâche exigeant beaucoup en termes de calcul ou de données et qui peut être exécutée en parallèle prend 500 heures à traiter sur un ordinateur, au moyen de l'architecture dans le nuage [6], il serait possible de démultiplier et lancer 500 instances afin de traiter la même tâche en 1 heure. La disponibilité d'une infrastructure élastique offre à l'application la capacité d'exploiter la parallélisation d'une façon rentable, réduisant ainsi le délai de mise sur le marché.

Les avantages techniques du cloud computing

Voici quelques uns des avantages techniques résultant du cloud computing :

L'automatisation – « Infrastructure contrôlable par script » : vous pouvez créer des systèmes reproductibles de construction et déploiement en tirant partie de l'infrastructure programmable (basée sur l'interface de programmation).

Un dimensionnement automatique : vous pouvez étendre et réduire votre application afin qu'elle s'adapte aux demandes imprévues sans intervention humaine. Le dimensionnement automatique favorise l'automatisation et contribue à l'efficacité.

Un dimensionnement proactif : Étendez et réduisez votre application afin qu'elle s'adapte à vos demandes prévues au moyen d'une compréhension appropriée de vos schémas de trafic, dans le but de maintenir des coûts faibles pendant le dimensionnement.

*Un cycle de vie de développement plus efficace :*les systèmes de production peuvent facilement être clonés en vue d'être utilisés dans les environnements de développement et d'essais. Les environnements de préparation peuvent facilement être activés en production.

*Une meilleure testabilité :*vous ne manquez jamais d'équipement pour les essais. Injectez et automatisez les essais à chaque phase du processus de développement. Vous pouvez lancer un « labo d'essai instantané » avec des environnements préconfigurés seulement pendant la durée de la phase d'essais.

*La reprise sur sinistre et la continuité des opérations :*Le nuage offre une option moins onéreuse pour maintenir une flotte de serveurs DR et d'éléments de stockage des données. Avec le nuage, vous pouvez tirer partie de la géo-distribution et reproduire l'environnement dans d'autres emplacements en quelques minutes seulement.

*« Faites déborder » le trafic vers le nuage :*à l'aide de quelques clics et de tactiques efficaces d'équilibrage, vous pouvez créer une application complète à l'épreuve des dépassements de capacité en acheminant le trafic en excès vers le nuage.

Comprendre le nuage Amazon Web Services

Le nuage Amazon Web Services (AWS) fournit une infrastructure hautement fiable et dimensionnable pour déployer des solutions sur le Web, avec une assistance et des frais d'administration minimes, et davantage de flexibilité par rapport à ce que vous attendez généralement de votre propre infrastructure, que ce soit sur site ou dans un centre de données.

AWS offre actuellement un éventail de services en matière d'infrastructure. Le diagramme ci-dessous va vous présenter la terminologie AWS et vous aider à comprendre la façon dont votre application peut interagir avec les différents services d'Amazon Web Services, ainsi que la façon dont divers services interagissent les uns avec les autres.

Amazon Elastic Compute Cloud (Amazon EC2)[1] est un service web qui fournit une capacité de calcul redimensionnable dans le nuage. Vous pouvez combiner le système d'exploitation, le logiciel d'application et les paramètres de configuration associés dans une *Amazon Machine Image* (AMI). Vous pouvez alors utiliser ces AMI pour obtenir plusieurs *instances* virtuelles et pour les désactiver au moyen de simples appels de service web afin d'étendre ou de réduire la capacité, en fonction des changements de capacité qui sont requis. Vous pouvez acheter des*On-Demand Instances*dans lesquelles vous payez les instances à l'heure, ou des *Reserved Instances* dans lesquelles vous payez un prix unique et réduit et vous recevez un taux d'utilisation plus faible que pour les On-demand Instances ou les *Spot Instances*dans lesquelles vous pouvez faire des offres pour la capacité non utilisée, réduisant ainsi vos coûts. Les instances peuvent être lancées dans une ou plusieurs *régions* géographiques. Chaque région comporte plusieurs *Zones de disponibilité*. Les Zones de disponibilité sont des endroits distincts conçus pour être isolés en cas de panne dans les autres Zones de disponibilité, et elles fournissent une connectivité de réseau peu onéreuse à faible latence pour d'autres Zones de disponibilité dans la même région.

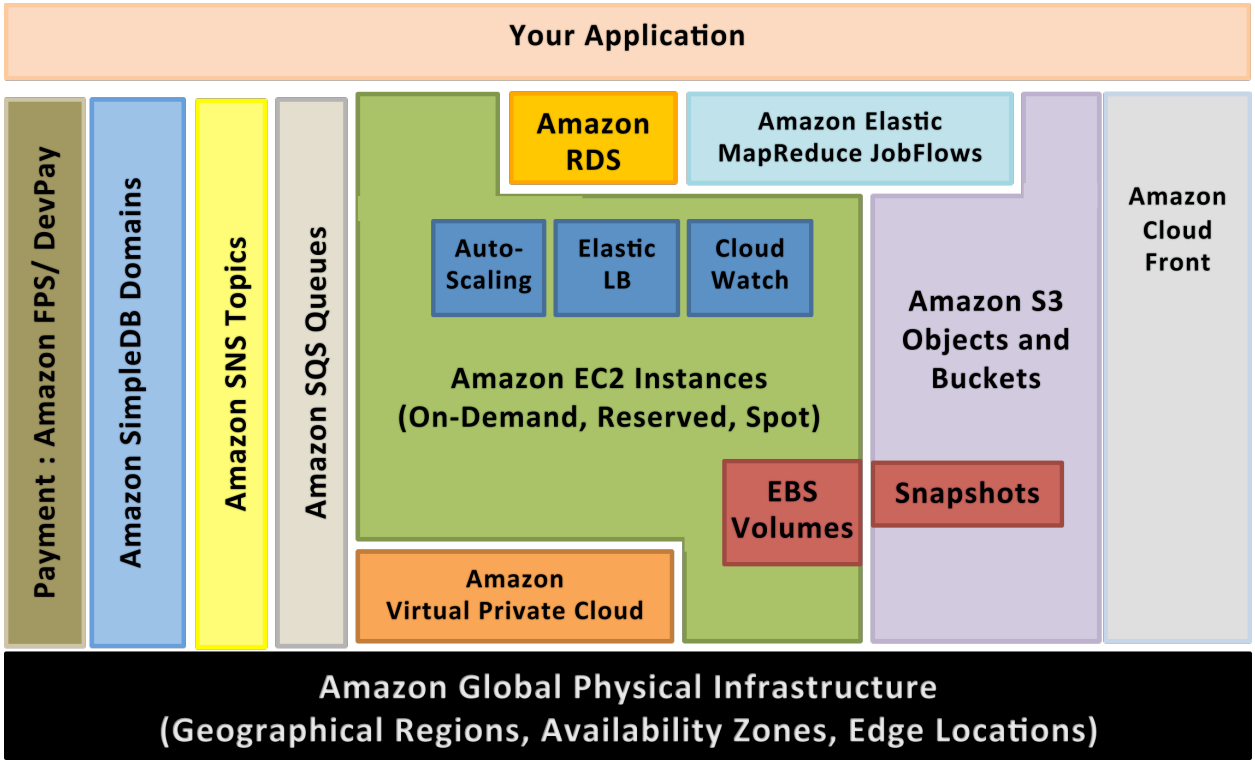

Figure 1: Amazon Web Services

Les adresses *Elastic IP* vous permettent d'affecter une adresse IP statique puis de l'attribuer à une instance de façon programmatique. Vous pouvez activer le monitoring sur une instance Amazon EC2 à l'aide d'Amazon CloudWatch[2] afin d'avoir une plus grande visibilité de l'utilisation des ressources, des performances optimales et des schémas globaux de demande (y compris les métriques telles que l'utilisation du CPU, les lectures et écritures sur disque et le trafic du réseau). Vous pouvez créer un *Auto-scaling Group* à l'aide de la fonction Auto-scaling[3] afin de dimensionner automatiquement votre capacité en fonction de certaines conditions basées sur les métriques recueillies par Amazon CloudWatch. Vous pouvez également répartir le trafic entrant en créant un *elastic load balancer* à l'aide du service

[1] Vous pouvez obtenir davantage d'informations sur Amazon EC2 sur http://aws.amazon.com/ec2
[2] Vous pouvez obtenir davantage d'informations sur Amazon CloudWatch sur http://aws.amazon.com/cloudwatch/
[3] Vous pouvez obtenir davantage d'informations sur la fonctionnalité Auto-scaling sur http://aws.amazon.com/auto-scaling

[4]Elastic Load Balancing. Les volumes Elastic Block Storage (EBS)[5] *d'Amazon* offrent un stockage persistant lié au réseau pour les instances Amazon EC2. Il est possible de créer des *snapshots* systématiques instantanés des volumes EBS et de les stocker sur Amazon Simple Storage Service (Amazon S3)[6].

Amazon S3 est un service de stockage de données réparties qui est très durable. À l'aide d'une simple interface de services web, vous pouvez stocker et récupérer de grandes quantités de données en tant qu'*objet*s dans des *seaux* (conteneurs) à tout moment, à partir de n'importe où sur le Web, au moyen de verbes HTTP standard. Les copies d'objets peuvent être distribuées et placées en mémoire cache sur 14 *emplacements bordures* dans le monde entier en créant une *distribution* au moyen du service[7] Amazon CloudFront – un service web pour la diffusion de contenu (contenu statique ou en streaming). Amazon SimpleDB[8] est un service web proposant la fonctionnalité de base des consultations en temps réel des bases de données et de la création de requêtes sur des données structurées - sans la complexité opérationnelle. Vous pouvez organiser les ensembles de données en *domaines* et vous pouvez exécuter des requêtes sur toutes les données stockées dans un domaine particulier. Les domaines sont des collections d'*articles* qui sont décrits par des *paires attributs-valeurs*. Amazon Relational Database Service[9] (Amazon RDS) offre un moyen facile de créer, opérer et dimensionner une base de données relationnelle dans le nuage. Vous pouvez lancer une *instance DB* et avoir accès à une base de données MySql avec toutes ses fonctionnalités sans vous inquiéter des tâches courantes d'administration de base de données, telles que les sauvegardes de sécurité, la gestion des patchs, etc.

Amazon Simple Queue Service (Amazon SQS)[10] est une file d'attente distribuée fiable, hautement dimensionnable et hébergée pour stocker les *messages* alors qu'ils se déplacent entre les ordinateurs et les composants d'application.

Amazon Simple Notifications Service (Amazon SNS) [11]propose une façon simple d'informer les applications ou les individus à partir du nuage en créant des *Thèmes* et en utilisant un protocole publier-abonner.

Amazon Elastic MapReduce[12] offre un cadre Hadoop hébergé qui fonctionne sur l'infrastructure à l'échelle du web d'Amazon Elastic Compute Cloud (Amazon EC2) et Amazon Simple Storage Service (Amazon S3) pour vous permettre de créer des *JobFlows* personnalisés. JobFlows est une séquence des *étapes* de MapReduce.

Amazon Virtual Private Cloud (Amazon VPC)[13] vous permet d'étendre votre réseau d'entreprise dans un nuage privé contenu dans AWS. Amazon VPC utilise le mode tunnel IPSec qui vous permet de créer une connexion sécurisée entre une passerelle dans votre centre de données et une passerelle dans AWS.

AWS propose également divers services de paiement et de facturation[14] qui tirent partie de l'infrastructure de paiement d'Amazon.

---

[4] Vous pouvez obtenir davantage d'informations sur la fonctionnalité Elastic Load Balancing sur http://aws.amazon.com/elasticloadbalancing
[5] Vous pouvez obtenir davantage d'informations sur Elastic Block Store sur http://aws.amazon.com/ebs
[6] Vous pouvez obtenir davantage d'informations sur Amazon S3 sur http://aws.amazon.com/s3
[7] Vous pouvez obtenir davantage d'informations sur Amazon CloudFront sur http://aws.amazon.com/cloudfront
[8] Vous pouvez obtenir davantage d'informations sur Amazon Simple DB sur http://aws.amazon.com/simpledb
[9] Vous pouvez obtenir davantage d'informations sur Amazon RDS sur http://aws.amazon.com/rds
[10] Vous pouvez obtenir davantage d'informations sur Amazon SQS sur http://aws.amazon.com/sqs
[11] Vous pouvez obtenir davantage d'informations sur Amazon SNS sur http://aws.amazon.com/sns
[12] Vous pouvez obtenir davantage d'informations sur Amazon ElasticMapReduce sur http://aws.amazon.com/elasticmapreduce
[13] Vous pouvez obtenir davantage d'informations sur Amazon Private Cloud sur http://aws.amazon.com/vpc
[14] Vous pouvez obtenir davantage d'informations sur Amazon Flexible Payment Service sur http://aws.amazon.com/fps et Amazon DevPay sur http://aws.amazon.com/devpay

Tous les services d'infrastructure AWS offrent une structure de tarifs à la demande ne demandant aucun engagement ni contrat à long terme. Par exemple, vous payez l'utilisation des instances Amazon EC2 à l'heure, et dans le cas d'Amazon S3, vous payez le stockage et le transfert des données au gigaoctet. Vous pouvez obtenir davantage d'informations sur chacun de ces services et leur tarifs à l'utilisation sur le site web AWS.

Veuillez remarquer que l'utilisation du nuage AWS ne signifie pas devoir sacrifier la flexibilité et le contrôle auxquels vous êtes habitués :

- Vous êtes libre de vous servir du modèle et langage de programmation, ou système d'exploitation (Windows, OpenSolaris ou toute version de Linux) de votre choix.
- Vous êtes libre de choisir tous les produits AWS qui satisfont au mieux à vos exigences - vous pouvez utiliser n'importe lequel de ces services de façon individuelle ou dans n'importe quelle association.
- Étant donné qu'AWS fournit des ressources redimensionnables (stockage, largeur de bande et calcul), vous êtes libre d'utiliser autant ou aussi peu que vous le souhaitez, et de ne payer que pour ce que vous utilisez.
- Vous êtes libre d'utiliser les outils de gestion de système que vous avez employés dans le passé et d'étendre votre centre de données dans le nuage.

Concepts liés au nuage

Le nuage renforce certains anciens concepts concernant la développement d'architectures internet hautement dimensionnables [13] tout en présentant de nouveaux concepts qui transforment complètement la façon dont les applications sont construites et déployées. C'est ainsi qu'en avançant du concept à la mise en œuvre, il se peut que vous ayez le sentiment que « Tout a changé, mais rien n'est différent ». Le nuage modifie certains processus, schémas, pratiques et philosophies tout en renforçant certains principes traditionnels d'architecture axée sur le service que vous avez appris, car ils sont encore plus importants qu'auparavant. Dans cette section, vous allez observer certains de ces nouveaux concepts liés aux nuage, ainsi que quelques concepts SOA réitérés.

Les applications traditionnelles étaient bâties avec des préconceptions mentales qui étaient logiques sur les plans économique et architectural lors de leur développement. Le nuage lance certaines nouvelles philosophies que vous devez comprendre, et qui sont traitées ci-dessous :

Développer des architectures dimensionnables

Il est indispensable de développer une architecture dimensionnable pour pouvoir exploiter une infrastructure dimensionnable.

Le nuage est conçu pour offrir une dimensionnabilité conceptuellement infinie. Cependant, vous ne pouvez pas pleinement exploiter cette dimensionnabilité de l'infrastructure si votre architecture n'est pas dimensionnable. Les deux doivent fonctionner ensemble. Vous devez identifier les composants monolithiques et les goulots d'étranglement de votre architecture, identifier les domaines où vous ne pouvez pas exploiter les capacités d'approvisionnement à la demande de votre architecture, puis œuvrer pour *réusiner* votre application dans le but d'exploiter l'infrastructure dimensionnable et de tirer partie du nuage.

Les caractéristiques d'une application véritablement dimensionnable :

- Une augmentation des ressources engendre une augmentation proportionnelle des performances
- Un service dimensionnable est en mesure de traiter l'hétérogénéité
- Un service dimensionnable est efficace au niveau des opérations
- Un service dimensionnable est résistant
- Un service dimensionnable devrait devenir plus rentable au fur et à mesure qu'il croît (le coût unitaire diminue alors que le nombre d'unités augmente)

Ces éléments doivent devenir une partie intégrale de votre application, et si vous concevez votre architecture en tenant compte des caractéristiques précédentes, votre architecture et votre infrastructure fonctionneront ensemble pour vous fournir la dimensionnabilité que vous recherchez.

Comprendre l'élasticité

Le graphique ci-dessous illustre les différentes démarches qu'un architecte cloud peut adopter pour dimensionner ses applications en fonction de la demande.

La démarche de dimensionnement extensif : ne pas s'inquiéter d'établir une architecture d'application dimensionnable et investir dans des ordinateurs plus gros et plus puissants (dimensionnement vertical) pour répondre à la demande. Cette démarche fonctionne généralement dans une certaine mesure, mais elle coûte une fortune (cf. « énorme dépense en immobilisations » dans le diagramme) et il est également possible que la demande dépasse la capacité avant le déploiement de la « grosse artillerie » (cf. « vous venez de perdre vos clients » dans le diagramme).

La démarche traditionnelle de dimensionnement progressif : créer une architecture dimensionnable horizontalement et investir dans l'infrastructure par étapes. La plupart des entreprises et des applications web à grande échelle suivent ce modèle en répartissant leurs composants d'application, en fédérant leurs ensembles de données et en se servant d'une conception axée sur le service. Cette démarche est souvent plus efficace que la démarche de dimensionnement extensif. Il est cependant nécessaire de prévoir la demande de façon régulière, puis de déployer l'infrastructure de façon progressive pour répondre à la demande. Cela conduit souvent à une capacité excessive (« gaspillage d'argent ») et à un contrôle manuel constant (« gaspillage de cycles humains »). De plus, cette démarche ne fonctionne habituellement pas si l'application est victime d'un feu viral (souvent désigné comme « effet Slashdot »[15]).

Note : les deux démarches comportent des coûts initiaux et sont de nature réactive.

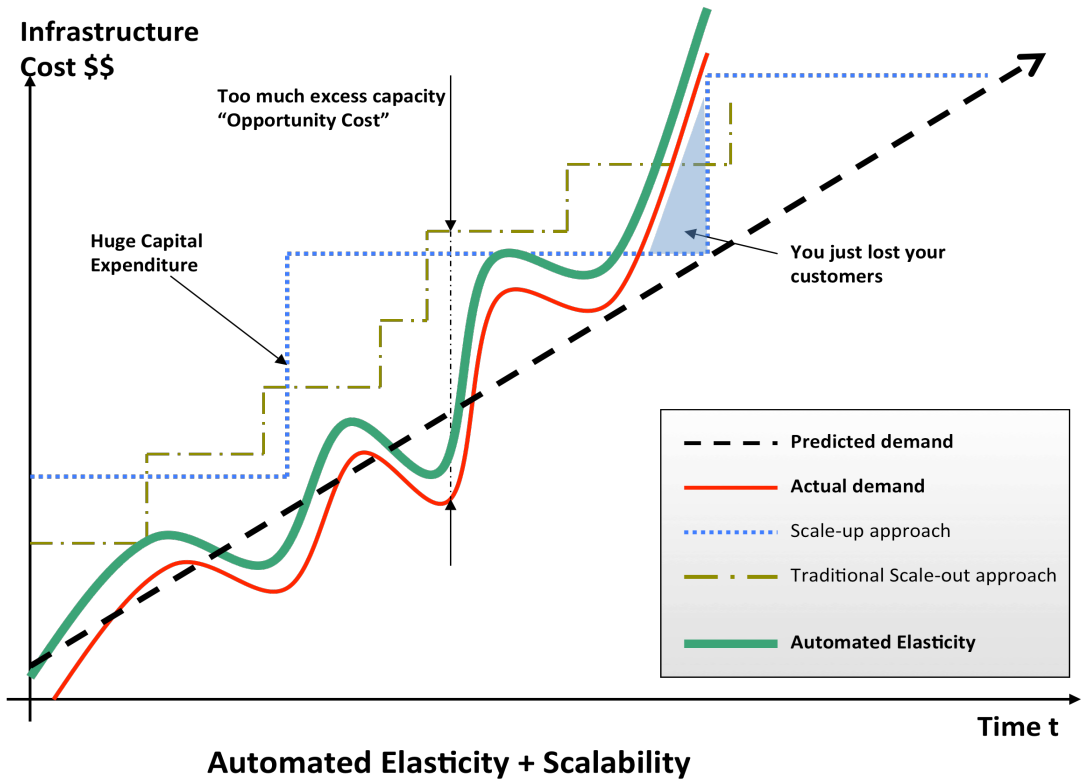

Figure 2: élasticité automatisée

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

Dans ces infrastructure traditionnelles, il est généralement nécessaire de prédire la quantité de ressources de calcul que votre application utilisera sur une période de plusieurs années. Si vous sous-estimez cette quantité, vos applications ne disposeront pas de la puissance nécessaire pour traiter le trafic inattendu, ce qui peut provoquer le mécontentement des clients. Si vous surestimez cette quantité, vous gaspillez de l'argent dans des ressources superflues.

La nature à la demande, élastique de la*démarche du nuage* (élasticité automatisée) permet cependant à l'infrastructure d'être synchronisée de près (lors des expansions et des réductions) avec la demande réelle, ce qui optimise l'utilisation globale tout en réduisant les coûts.

L'élasticité est l'une des propriétés fondamentales du nuage. L'élasticité est le pouvoir de dimensionner les ressources de calcul, qu'il s'agisse d'expansion ou de réduction, avec facilité et un minimum de frictions. Il est important de comprendre que l'élasticité est à la base de la plupart des avantages du nuage. En tant qu'architecte cloud, vous devez assimiler ce concept et l'intégrer dans votre architecture d'application afin d'optimiser les avantages du nuage.

Les applications ont traditionnellement été développées pour des infrastructures fixes, rigides et pré-dimensionnées. Les entreprises n'ont jamais eu le besoin d'obtenir et d'installer des serveurs de façon quotidienne. Par conséquent, la plupart des architectures de logiciel ne prennent pas en charge le déploiement ou la réduction rapides d'équipements. Étant donné que le temps d'approvisionnement et l'investissement initial pour l'obtention de nouvelles ressources étaient trop élevés, les architectes logiciels n'ont jamais investi dans le temps et les ressources nécessaires pour optimiser l'utilisation des équipements. Cela était acceptable si les équipements sur lesquels l'application fonctionnait étaient sous-utilisés. La notion « d'élasticité » dans le cadre de l'architecture a été négligée, car l'idée même de disposer de nouvelles ressources en quelques minutes était impossible.

Avec le nuage, cette mentalité doit changer. Le cloud computing simplifie le processus d'acquisition des ressources nécessaires ; il n'y a plus besoin de passer des commandes à l'avance et de conserver des matériels non utilisés. Les architectes cloud peuvent désormais demander ce dont ils ont besoin à peine quelques minutes avant d'en avoir besoin, voire automatiser le processus d'approvisionnement, tirant ainsi partie de la grande échelle et du temps de réponse rapide du nuage. Le principe est le même pour la libération des ressources inutiles ou sous-utilisées quand vous n'en avez pas besoin.

Si vous n'êtes pas en mesure d'adopter le changement et de mettre en œuvre l'élasticité dans l'architecture de votre application, il se peut que vous ne puissiez pas tirer pleinement partie du nuage. En tant qu'architecte cloud, il vous faut penser créativement et réfléchir aux façons de mettre en œuvre l'élasticité dans votre application. Par exemple, les infrastructures qui devaient appliquer les builds pendant la nuit et réaliser des tests de régression et unitaires chaque nuit à 2:00 du matin pendant deux heures (souvent désignés comme « QA/Build box ») restaient désœuvrées pendant le reste de la journée. Maintenant, avec l'infrastructure élastique, il est possible d'exécuter les builds de nuit dans des boîtiers « vivants » en ne payant que pour 2 heures par nuit. De même, les applications web de dossier d'incident interne qui devaient toujours fonctionner en pleine capacité (5 serveurs 24x7x365) pour répondre à la demande pendant la journée peuvent maintenant être dimensionnées pour fonctionner à la demande (5 serveurs de 9:00 à 17:00 et 2 serveurs de 17:00 à 9:00) en fonction des schémas de trafic.

La conception d'architectures cloud intelligentes et élastiques de façon à ce que l'infrastructure ne fonctionne que lorsque vous en avez besoin est tout un art. L'élasticité devrait être l'une des exigences de conception architecturale, ou une propriété de système. Questions à vous poser : Quels sont les composants ou couches de l'architecture de mon application qui peuvent devenir élastique ? Que faut-il faire pour rendre ce composant *élastique* ? Quel sera l'impact de la mise en œuvre de l'élasticité dans l'architecture globale de mon système ?

Dans la prochaine section, vous allez apprendre des techniques spécifiques pour mettre en œuvre l'élasticité dans vos applications. Pour tirer partie efficacement des avantages du nuage, il est important de développer l'architecture avec cet état d'esprit.

Ne pas craindre les contraintes

Lorsque vous décidez de déplacer vos applications dans le nuage et que vous tentez d'adapter les spécifications de votre système à celles qui sont disponibles dans le nuage, vous constaterez que le nuage ne dispose peut-être pas des spécifications exactes qui correspondent aux ressources que vous avez sur place. Par exemple, « le nuage ne fournit pas un montant X de RAM dans un serveur » ou « ma base de données a besoin de plus d'IOPS que ce que je peux obtenir dans une seule instance ».

Vous devez comprendre que le nuage fournit des *ressources abstraites* et qu'elles deviennent puissantes quand vous les combinez au modèle de dimensionnement à la demande. Il ne faut pas avoir peur ni se sentir limité en utilisant les ressources du nuage, car il est important de comprendre que, même si vous n'obtenez pas une copie exacte de votre matériel dans l'environnement du nuage, vous avez la possibilité d'obtenir davantage de ressources dans le nuage pour équilibrer ce besoin.

Par exemple, si le nuage ne vous fournit pas un montant exact ou supérieur de RAM dans un serveur, essayez d'utiliser une mémoire cache répartie telle que *memcached*[16]ou de segmenter vos données sur plusieurs serveurs. Si vos bases de données nécessitent plus d'IOPS et qu'elle ne correspond pas directement à celle du nuage, vous pouvez choisir parmi plusieurs recommandations en fonction du type de données et du cas d'usage. S'il s'agit d'une application dont la demande de lecture est lourde, vous pouvez répartir la charge de lecture parmi une flotte d'esclaves synchronisés. Vous pouvez également utiliser un algorithme de *partitionnement horizontal* [10] qui achemine les données là où elles doivent se trouver, ou vous pouvez utiliser diverses solutions de regroupement de bases de données.

Rétrospectivement, lorsque vous combinez les capacités de dimensionnement à la demande avec la flexibilité, vous vous rendrez compte que les contraintes apparentes peuvent en fait être brisées de façon à améliorer le dimensionnement et les performances globales du système.

Administration virtuelle

L'arrivée du nuage a changé le rôle de l'administrateur systèmes en « administrateur systèmes virtuels ». Cela signifie simplement que les tâches quotidiennes effectuées par ces administrateurs sont maintenant devenues encore plus intéressantes alors qu'ils continuent à en apprendre davantage sur les applications et à décider ce qui est le mieux pour l'entreprise dans son ensemble. L'administrateur système n'a plus besoin d'obtenir des serveurs, d'installer des logiciels et de connecter des dispositifs de réseau, car toute cette dure besogne est devenue l'affaire de quelques clics et appels de ligne de commande. Le nuage encourage l'automatisation car l'infrastructure est programmable. Les administrateurs systèmes doivent rester à l'avant-garde de la technologie et apprendre comment gérer les ressources abstraites du nuage à l'aide de scripts.

De même, le rôle des administrateurs de bases de données est devenu « administrateur de bases de données virtuelles », dans lequel il ou elle gère les ressources au moyen d'une console web, exécute des scripts augmentant la capacité par la programmation dans les cas où la capacité du matériel de la base de données devient insuffisante, et automatise les processus quotidiens. Le DBA virtuel doit désormais apprendre de nouvelles méthodes de déploiement (images de machine virtuelle), adopter de nouveaux modèles (parallélisation des requêtes, géo-redondance et reproduction asynchrone) [11], repenser la démarche architecturale en matière de données (sharding [9], partitionnement horizontal [13], mise en fédération [14]) et exploiter les différentes options de stockage disponibles dans le nuage pour différents types d'ensembles de données.

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

Dans l'entreprise traditionnelle, les développeurs d'application ne collaborent peut-être pas de près avec les administrateurs réseaux, et les administrateurs réseaux ne savent peut-être rien de l'application. Par conséquent, plusieurs optimisations possibles de la couche réseau et de la couche architecture du réseau sont négligées. Dans le nuage, les deux rôles ont dans une certaine mesure été fusionnés. Lors de la conception de l'architecture de futures applications, les entreprises doivent encourager davantage de pollinisation croisée des connaissances parmi les deux rôles tout en comprenant qu'ils sont fusionnés.

Les bonnes pratiques du nuage

Dans cette section, vous allez apprendre les bonnes pratiques qui vous aideront à développer une application dans le nuage.

Concevez en pensant aux défaillances et rien ne sera défaillant

La règle empirique : soyez pessimiste lorsqu'il s'agit de concevoir des architectures dans le nuage et partez du principe qu'il y aura des problèmes. En d'autres termes, vous devez toujours concevoir, mettre en œuvre et déployer en prévoyant une reprise automatique après une défaillance.

Prévoyez en particulier que votre matériel tombera *à coup sûr* en panne. Prévoyez qu'il y aura *à coup sûr* des coupures de courant. Prévoyez qu'un désastre quelconque frappera *à coup sûr* votre application. Prévoyez que vous serez *à coup sûr* submergé par plus de demandes que prévu un jour ou l'autre. Prévoyez qu'avec le temps, votre logiciel d'application sera lui aussi défaillant. En étant pessimiste, vous pouvez réfléchir à des stratégies de reprise lors de la phase de conception, ce qui vous aidera à concevoir un meilleur système dans son ensemble.

Si vous vous rendez compte qu'il y aura des problèmes avec le temps, que vous intégrez ce facteur dans votre architecture, et que vous construisez des mécanismes pour prendre en charge ces problèmes avant la catastrophe dans le domaine de l'infrastructure dimensionnable, vous serez en mesure de créer une architecture résistant aux défaillances qui est optimisée pour le nuage.

Questions à vous poser : Que se passe-t-il si un nœud est défaillant dans votre système ? Comment détectez-vous cette défaillance ? Comment puis-je remplacer ce nœud ? À quels genres de scénarios dois-je me préparer ? Quels sont mes points de défaillance individuels ? Si un équilibreur de charge se trouve devant un éventail de serveurs d'applications, que se passe-t-il en cas de panne de l'équilibreur de charge ? Si votre architecture comporte des éléments maître-esclave, que se passe-t-il si le nœud maître est défaillant ? Comment le basculement se produit-il et une nouvelle instance esclave est-elle créée et synchronisée avec le maître ?

De même que vous concevez votre équipement en pensant aux pannes, vous devez aussi concevoir vos logiciels en pensant aux pannes. Questions à vous poser : Qu'arrive-t-il à mon application si les services dépendants changent d'interface ? Que se passe-t-il si le service en aval dépasse le temps ou renvoie une exception ? Que se passe-t-il si les clés de mémoire cache s'accroissent au-delà de la limite de mémoire d'un instance ?

Construisez des mécanismes pour prendre en charge ces défaillances. Par exemple, les stratégies suivantes peuvent s'avérer utiles en cas de défaillance :

1. Disposer d'une stratégie de sauvegarde et restauration systématique pour vous données, et l'automatiser
2. Construire des threads de processus qui sont réinitialisés lors du redémarrage
3. Permettre que l'état du système soit resynchronisé en rechargeant les messages en file d'attente
4. Garder les images virtuelles pré-configurées et pré-optimisées pour soutenir (2) et (3) lors du lancement/redémarrage
5. Éviter les sessions en mémoire ou le contexte d'utilisateur à état, les déplacer en stockage de données.

Une bonne architecture dans le nuage devrait fonctionner sans être affectée par les redémarrages et les réinitialisations. Dans GrepTheWeb (traité dans le document Cloud Architectures [6]), en utilisant une combinaison d'Amazon SQS et Amazon SimpleDB, l'architecture globale du contrôleur est très résistante aux divers types de défaillances mentionnés dans cette section. Par exemple, si l'instance sur laquelle le thread de contrôleur fonctionnait se ferme, elle peut être

relancée et récupérer l'état antérieur comme si rien ne s'était passé. Cela est possible en créant une Amazon Machine Image préconfigurée, qui, lors de son lancement, retire tous les messages de la file d'attente Amazon SQS et lit leur état à partir d'un domaine Amazon SimpleDB lors du redémarrage.

En élaborant vos conceptions en partant du principe que les équipements sous-jacents seront défaillants, vous vous préparez aux défaillances qui se produiront effectivement à l'avenir.

Ce principe de conception vous aidera à concevoir des applications conviviales au niveau des opérations, comme le met aussi en évidence le document de Hamilton [11]. Si vous pouvez étendre ce principe pour mesurer et équilibrer proactivement et dynamiquement la charge, vous serez sans doute en mesure de prendre en charge la variance du réseau et des performances de disque qui existent en raison de la nature à utilisateurs partagés du nuage.

Voici des tactiques AWS spécifiques pour mettre en œuvre cette bonne pratique :

1. Basculez sans heurts au moyen des IP élastiques : une IP élastique est une IP statique qui peut subir un remapping dynamique. Vous pouvez effectuer le remapping et le basculement vers un autre ensemble de serveurs afin que votre trafic soit acheminé vers les nouveaux serveurs. Cela fonctionne très bien lorsque vous souhaitez mettre à niveau une ancienne version ou en cas de panne de matériel.
2. Utilisez plusieurs zones de disponibilité : sur le plan conceptuel, les zones de disponibilité sont semblables à des centres de données logiques. En déployant votre architecture dans plusieurs zones de disponibilité, vous pouvez assurer une grande disponibilité. Utilisez la fonctionnalité de déploiement Amazon RDS Multi-AZ [21] afin de reproduire automatiquement les mises à jour de base de données dans plusieurs zones de disponibilité.
3. Maintenez une Amazon Machine Image afin de pouvoir restaurer et cloner les environnements très facilement dans différentes zones de disponibilité ; maintenez plusieurs esclaves de base de données dans les zones de disponibilité et configurez des reproductions à chaud.
4. Utilisez Amazon CloudWatch (ou plusieurs outils de monitoring open source en temps réel) pour obtenir une plus grande visibilité et prendre les mesures nécessaires en cas de panne de matériel ou de dégradation des performances. Configurez un Auto-scaling Group pour maintenir une taille de flotte fixe, pour qu'elle remplace les instances Amazon EC2 défaillantes.
5. Utilisez Amazon EBS et établissez des tâches cron pour que les instantanés incrémentiels soient automatiquement chargés sur Amazon S3 et que les données soient préservées indépendamment de vos instances.
6. Utilisez Amazon RDS et définissez la période de rétention pour les sauvegardes, afin qu'elles puissent se faire de façon automatisée.

Découplez vos composants

Le nuage renforce le principe de conception SOA qui déclare que *plus les composants du système sont découplés, plus le dimensionnement est large et de qualité*.

Le facteur clé consiste à construire des composants qui ne dépendent pas étroitement les uns des autres, de manière à ce que si l'un d'entre eux meurt (défaillance), dort (pas de réponse) ou est occupé (lent à répondre) pour une raison quelconque, les autres composants du système ont construits pour continuer à fonctionner comme s'il n'y avait pas de problème. En fait, le couplage faible isole les différentes couches et composants de votre application pour que chaque composant interagisse de façon asynchrone avec les autres et les traite comme une « boîte noire ». Par exemple, dans le cas de l'architecture d'application web, vous pouvez isoler le serveur app du serveur web et de la base de données. Le

serveur app n'est pas en rapport avec votre serveur web et vice versa, ce qui permet de découpler ces couches et d'éviter les dépendances au niveau du code ou des perspectives fonctionnelles. Dans le cas de l'architecture de traitement par lots, vous pouvez créer des composants *asynchrones* qui sont indépendants les uns des autres.

Questions à vous poser : Quels composants ou fonctionnalités de l'entreprise peuvent être isolés de l'application monolithique actuelle pour fonctionner de façon autonome ? Ensuite, comment puis-je ajouter davantage d'instances de ce composant sans casser mon système actuel tout en servant plus d'utilisateurs ? Quels efforts seront nécessaires pour encapsuler le composant pour qu'il interagisse avec d'autres composants de façon asynchrone ?

Le découplage de vos composants, la construction de systèmes *asynchrones* et le dimensionnement horizontal deviennent très importants dans le contexte du nuage. Non seulement vous permettront-ils de dimensionner en ajoutant davantage d'instances du même composant, mais ils vous permettrons également de concevoir des modèles hybrides innovants dans lesquels quelques composants continuent à fonctionner sur site tandis que d'autres composants peuvent tirer partie du nuage et l'utiliser pour obtenir davantage de puissance de calcul et de largeur de bande. C'est ainsi qu'avec un minimum d'efforts, vous pouvez faire « déborder » le trafic en excès vers le nuage en mettant en œuvre des tactiques intelligentes d'équilibrage de la charge.

Il est possible de développer un système à faible couplage au moyen de *files d'attente de messages*. Si une file d'attente/mémoire tampon est employée pour connecter deux composants, elle peut prendre en charge l'accès simultané, la grande disponibilité et les crêtes de charge. Cela a pour effet de permettre au système global de continuer à être performant, même si des parties de composants sont provisoirement non disponibles. Si un composant meurt ou devient provisoirement non disponible, le système met les messages en mémoire tampon et les traite lorsque le composant redevient disponible.

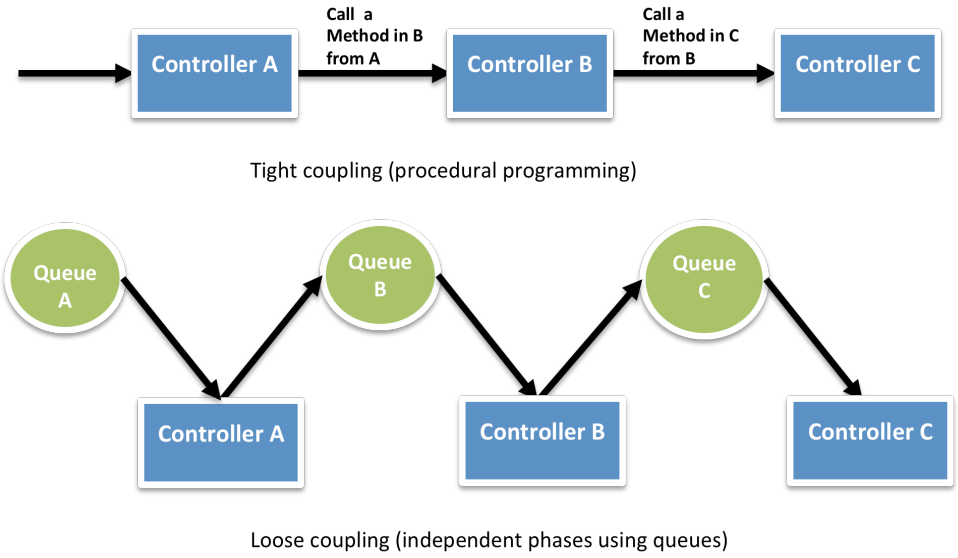

Figure 3: découpler les composants à l'aide des files d'attente

Vous verrez une utilisation intensive des files d'attente dans l'architecture GrepTheWeb représentée dans le document Cloud Architectures [6]. Dans GrepTheWeb, si de nombreuses demandes atteignent le serveur de façon soudaine (une situation de surcharge induite par l'Internet) ou le traitement d'expressions normales prend plus de temps que la moyenne (temps de réponde lent d'un composant), les files d'attente Amazon SQS mettent les demandes en mémoire tampon de façon durable pour que ces délais n'affectent pas les autres composants.

Voici des tactiques AWS spécifiques pour mettre en œuvre cette bonne pratique :

1. Utilisez Amazon SQS pour isoler les composants [18]
2. Utilisez Amazon SQS comme mémoire tampon entre les composants [18]
3. Concevez chaque composant de façon à exposer une interface de service et pour qu'il soit responsable de sa propre dimensionnabilité dans toutes les dimensions concernées et pour qu'il interagisse avec les autres composants de façon asynchrone
4. Intégrez le construct logique d'un composant dans une Amazon Machine Image pour qu'il puisse être déployé plus souvent
5. Rendez votre application sans état dans la mesure du possible. Stockez l'état de la session hors du composant (dans Amazon Simple DB le cas échéant)

Mettez en œuvre l'élasticité

Le nuage introduit un nouveau concept d'élasticité dans votre application. L'élasticité peut être mise en œuvre de trois façons différentes :

1. Dimensionnement cyclique proactif : un dimensionnement périodique se produisant à des intervalles fixes (quotidiennement, hebdomadairement, mensuellement, trimestriellement)
2. Dimensionnement proactif basé sur un évènement : un dimensionnement se produisant lorsque vous prévoyez une grande augmentation de demandes de trafic en raison d'un évènement programmé par l'entreprise (lancement d'un nouveau produit, campagnes de marketing)
3. Dimensionnement automatique basé sur la demande. En utilisant un service de monitoring, votre système peut émettre des éléments déclencheurs pour prendre les mesures nécessaires pour étendre ou réduire en fonction des métriques (l'utilisation des serveurs ou du réseau i/o, par exemple)

Pour mettre en œuvre « l'élasticité », il faut tout d'abord automatiser le processus de déploiement et rationaliser les processus de configuration et de construction. Cela permettra au système d'effectuer le dimensionnement sans aucune intervention humaine.

Cela engendrera des avantages immédiats en matière de coûts dans la mesure où l'utilisation globale est optimisée, en assurant que vos ressources soient étroitement alignées sur la demande au lieu de faire fonctionner des serveurs qui sont potentiellement sous-utilisés.

Automatisez votre infrastructure

L'un des avantages les plus importants de l'environnement du nuage est la possibilité d'utiliser les API du nuage pour automatiser votre processus de déploiement. Il est recommandé que vous preniez le temps de créer un processus de déploiement automatisé pendant la phase initiale du processus de migration au lieu d'attendre jusqu'à la fin. La création d'un processus de déploiement automatisé et reproductible contribuera à réduire les erreurs et favorisera un processus de mise à jour efficace et dimensionnable.

Pour automatiser le processus de déploiement :

- Créez une bibliothèque de « recettes » - de petits scripts fréquemment utilisés (pour l'installation et la configuration)
- Gérez le processus de configuration et déploiement au moyen d'agents intégrés dans un AMI
- Démarrez vos instances

Démarrez vos instances

Permettez que vos instances vous posent une question lors du démarrage « qui suis-je et quel est mon rôle ? » Chaque instance devrait avoir un rôle (« serveur DB », « serveur app », « serveur esclave » dans le cas d'une application web) à jouer dans l'environnement. Ce rôle peut être transmis comme un argument pendant le lancement, informant l'AMI, quand son instance est ouverte, des mesures à prendre après son démarrage. Lors du démarrage, les instances devraient récupérer les ressources nécessaires (code, scripts, configuration) en fonction du rôle et « s'attacher » à une grappe pour servir leur fonction. Voici les avantages de démarrer vos instances :

1. Recréer l'environnement (dév, préparation, production) à l'aide de quelques clics et très peu d'efforts
2. Davantage de contrôle sur vos ressources abstraites dans le nuage
3. Réduire les erreurs humaines de déploiement

4. Créer en environnement se réparant et se découvrant automatiquement qui est plus résistant aux pannes de matériel

Tactiques AWS spécifiques pour automatiser votre infrastructure

1. Définissez vos Auto-scaling Groups pour différentes grappes à l'aide de la fonctionnalité Amazon Auto-scaling
2. Surveillez les métriques de votre système (CPU, mémoire, disque I/O, réseau I/O) à l'aide d'Amazon CloudWatch et prenez les mesures nécessaires (lancement dynamique de nouveaux AMI au moyen du service Auto-scaling) ou envoyez des notifications.
3. Stockez et récupérez les informations de configuration machine de façon dynamique : utilisez Amazon SimpleDB pour obtenir les données de configuration lors du démarrage d'une instance (p.ex. chaînes de connexion de base de données). Il est également possible d'utiliser SimpleDB pour stocker des informations sur une instance, telles que son adresse IP, le nom de l'ordinateur et le rôle.
4. Concevez un processus de construction de sorte qu'il dépose les derniers builds dans un seau dans Amazon S3 ; téléchargez la dernière version d'une application lors du démarrage du système.
5. Investissez dans des outils de gestion de ressources de construction (scripts automatisés, images préconfigurées) ou utilisez des outils intelligents de gestion de configuration open source tels que Chef[17], Puppet[18], CFEngine [19]ou Genome[20].
6. Intégrez un système d'exploitation minimal (JeOS[21]) et vos dépendances logicielles dans une Amazon Machine Image pour faciliter sa gestion et sa maintenance. Passez les fichiers de configuration ou les paramètres lors du lancement et récupérez les données d'utilisateur[22] et les métadonnées d'instance après le lancement.
7. Réduisez le temps d'intégration et de lancement en démarrant à partir des volumes Amazon EBS[23] et en joignant plusieurs volumes Amazon à une instance. Créez des instantanés des volumes communs et partagez les instantanés[24] parmi les comptes lorsque c'est approprié.
8. Les composants d'application de devraient pas se baser sur la santé ou l'emplacement du matériel sur lequel ils fonctionnent. Par exemple, attachez dynamiquement l'adresse IP d'un nouveau nœud à la grappe. Basculez et démarrez automatiquement un nouveau clone en cas de défaillance.

Pensez en parallèle

Le nuage fait de la parallélisation un jeu d'enfant. Qu'il s'agisse de demander des données au nuage, de stocker des données dans le nuage, de traiter des données (ou exécuter des tâches) dans le nuage, en tant qu'architecte cloud, vous devez assimiler le concept de la parallélisation lorsque vous concevez des architectures dans le nuage. Il est conseillé, non seulement de mettre en œuvre la parallélisation autant que possible, mais aussi de l'automatiser, car le nuage vous permet de créer très facilement un processus reproductible.

[17] Vous pouvez obtenir davantage d'informations sur Chef sur http://wiki.opscode.com/display/chef/Home
[18] Vous pouvez obtenir davantage d'informations sur Puppet sur http://reductivelabs.com/trac/puppet/
[19] Vous pouvez obtenir davantage d'informations sur CFEngine sur http://www.cfengine.org/
[20] Vous pouvez obtenir davantage d'informations sur Genome sur http://genome.et.redhat.com/
[21] http://en.wikipedia.org/wiki/Just_enough_operating_system
[22] Vous pouvez consulter des informations sur les métadonnées d'instance et les données d'utilisateur sur http://docs.amazonwebservices.com/AWSEC2/latest/DeveloperGuide/index.html?AESDG-chapter-instancedata.html
[23] Vous pouvez obtenir davantage d'informations sur la fonctionnalité Boot From Amazon EBS sur http://genome.et.redhat.com/ http://developer.amazonwebservices.com/connect/entry.jspa?externalID=3121
[24] Apprenez comment partager un instantané sur http://aws.amazon.com/ebs/

Pour ce qui est d'accéder (récupérer et stocker) aux données, le nuage est conçu pour traiter des opérations massives en parallèle. Afin d'obtenir les meilleures performances et débit, vous devez exploiter *la parallélisation des demandes*. Le multithreading de vos demandes au moyen de plusieurs threads en parallèle pourra stocker ou récupérer les données plus rapidement que si les demandes sont faites par séquence. Ainsi, lorsque c'est possible, les processus d'une application dans le nuage devraient être sécurisés par rapport aux threads au moyen d'une philosophie de non partage et en exploitant le multithreading.

Lorsqu'il s'agit de traiter ou d'exécuter des demandes dans le nuage, la parallélisation devient encore plus importante. Dans le cas d'une application web, une bonne pratique générale consiste à répartir les demandes entrantes entre plusieurs serveurs web asynchrones à l'aide d'un équilibreur de charge. Dans le cas d'une application de traitement par lots, le nœud maître peut engendrer plusieurs nœuds esclaves pour traiter les tâches en parallèle (comme dans les cadres de traitement réparti comme Hadoop[25])

Le nuage prend toute sa valeur quand vous associez élasticité et parallélisation. Votre application dans le nuage peut rassembler une grappe d'instances de calcul qui sont dimensionnées en quelques minutes avec à peine quelques appels API, réaliser un travail en exécutant des tâches en parallèle, stocker les résultats et fermer toutes les instances. L'application GrepTheWeb traitée dans [6] en est un exemple.

Tactiques AWS spécifiques pour la parallélisation :

1. Utilisez le multithreading sur vos demandes Amazon S3 comme indiqué dans le document des bonnes pratiques [2]
2. Utilisez le multithreading sur vos demandes Amazon SimpleDB GET et BATCHPUT [3][4] [5]
3. Créez un JobFlow à l'aide du service Amazon Elastic MapReduce pour chacun de vos traitements par lots quotidiens (indexation, analyse de log, etc.) qui calculera les tâches en parallèle et vous économisera du temps.
4. Utilisez le service Elastic Load Balancing et répartissez votre charge parmi plusieurs serveurs app web *dynamiquement*

Gardez les données dynamiques proches des éléments de calcul et les données statiques proches de l'utilisateur final

En général, une bonne pratique consiste à garder vos données aussi près que possible de vos éléments de calcul ou de traitement afin de réduire la latence. Dans le nuage, cette bonne pratique est encore plus appropriée et importante dans la mesure où vous devez souvent faire face à des latences internet. De plus, dans le nuage, vous payez la largeur de bande entrant et sortant du nuage au gigaoctet de transfert de données, et les coûts peuvent rapidement grimper.

Si une grande quantité de données devant être traitées se trouve en dehors du nuage, il peut s'avérer moins cher et plus rapide « d'expédier » et de transférer les données vers le nuage en premier, puis d'effectuer les calculs. Par exemple, dans le cas d'une application d'entreposage de données, il est conseillé de déplacer l'ensemble des données vers le nuage, puis d'exécuter les requêtes parallèles sur cet ensemble de données. Dans le cas d'une application web qui stocke et récupère des données de bases de données relationnelles, il est conseillé de déplacer la base de données et le serveur app dans le nuage d'un seul coup.

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

Si les données sont générées dans le nuage, les applications qui les utilisent devraient également être déployées dans le nuage pour qu'elles puissent tirer partie du transfert de données gratuit dans le nuage et des faibles latences. Par exemple, dans el cas d'une application web d'e-commerce qui génère des données de logs et de parcours, il est conseillé de faire fonctionner l'analyseur de log et les moteurs de reporting dans le nuage.

À l'inverse, si les données sont statiques et ne changent pas souvent (par exemple fichiers image, vidéo, audio, PDF, JS, CSS), il est conseillé de tirer partie d'un service de diffusion de contenu afin que les données statiques soient placées en mémoire cache dans un emplacement bordure proche de l'utilisateur final (demandeur), ce qui a pour effet de réduire la latence de l'accès. En raison de la mémoire cache, un service de diffusion de contenu fournit un accès plus rapide aux objets populaires.

Voici des tactiques AWS spécifiques pour mettre en œuvre cette bonne pratique :

1. Envoyez vos disques de données à Amazon au moyen du service Import/Export[26]. Il peut être moins cher et plus rapide de déplacer de grandes quantités de données au moyen du sneakernet[27] que de les charger par internet.
2. Utilisez la même zone de disponibilité pour lancer une grappe de machines
3. Créez une distribution de votre seau Amazon S3 et laissez Amazon CloudFront placer le contenu en mémoire cache dans ce seau parmi les 14 emplacements bordures dans le monde entier

Bonnes pratiques de sécurité

Dans un environnement à plusieurs utilisateurs, les architectes cloud expriment souvent leur inquiétude concernant la sécurité. *Il faut mettre en œuvre la sécurité dans chaque couche de l'architecture de l'application.* La sécurité physique est généralement prise en charge par votre prestataire de service (livre blanc sur la sécurité [7]), ce qui constitue un avantage supplémentaire découlant de l'utilisation du nuage. Vous êtes responsable de la sécurité au niveau du réseau et de l'application et vous devriez mettre en œuvre les bonnes pratiques pertinentes pour votre entreprise. Dans cette section, vous allez vous familiariser avec quelques outils, fonctionnalités et directives spécifiques pour sécuriser votre application dans le nuage dans l'environnement AWS. Il est recommandable de tirer partie de ces outils et fonctionnalités afin d'assurer la sécurité de base, puis de mettre en œuvre des bonnes pratiques supplémentaires au moyen de méthodes standard le cas échéant ou en fonction des besoins.

Protégez vos données de transit

Si vous devez échanger des informations sensibles ou confidentielles entre un navigateur et un serveur web, configurez SSL sur votre instance de serveur. Vous aurez besoin d'un certificat de la part d'une autorité de certification externe telle que VeriSign[28] ou Entrust[29]. La clé publique incluse dans le certificat authentifie votre serveur pour le navigateur, et elle sert de base pour la création de la clé de session partagée pour chiffrer les données dans les deux sens.

Créez un nuage privé virtuel au moyen de quelques appels de ligne de commande (à l'aide d'Amazon VPC). Cela vous permettra d'utiliser vos propres ressources logiquement isolées dans le nuage AWS, puis de connecter ces ressources directement à votre propre centre de données au moyen de connexions VPN chiffrées selon la norme industrielle IPSec.

[26] Vous pouvez obtenir davantage d'informations sur Amazon Import Export Services sur http://aws.amazon.com/importexport
[27] http://en.wikipedia.org/wiki/Sneakernet
[28] http://www.verisign.com/ssl/
[29] http://www.entrust.net/ssl-products.htm

Vous pouvez aussi définir [15] un serveur OpenVPN sur une instance Amazon EC2 et installer le client OpenVPN sur les ordinateurs de tous les utilisateurs.

Protégez vos données au repos

Si vous vous inquiétez concernant le stockage d'informations sensibles et confidentielles dans le nuage, vous devriez chiffrer ces données (fichiers individuels) avant de les charger dans le nuage. Par exemple, chiffrez les données au moyen de n'importe quel outil PGP open source[30] ou commercial[31] avant de les stocker en tant qu'objets Amazon S3, puis déchiffrez-les après leur téléchargement. Cela est souvent une bonne pratique lors de la construction d'applications conformes aux normes HIPPA [8] qui doivent stocker des informations médicales protégées (PHI).

Sur Amazon EC2, le chiffrage des fichiers dépend du système d'exploitation. Les instances Amazon EC2 fonctionnant sous Windows peuvent utiliser la fonctionnalité intégrée Encrypting File System (EFS) [16]. Cette fonctionnalité prendra en charge le chiffrage et le déchiffrage des fichiers et des dossiers de façon automatique et elle rendra le processus transparent pour les utilisateurs [19]. Cependant, malgré son nom, EFS ne chiffre pas tout le système de fichiers ; il chiffre des fichiers individuels. Si vous avez besoin d'un volume chiffré complet, envisagez d'utiliser le produit open source TrueCrypt[32] ; il s'intègrera très bien aux volumes EBS formatés avec NTF. Les instances Amazon EC2 fonctionnant sous Linux peuvent traiter les volumes EBS à l'aide de systèmes de fichiers chiffrés utilisant diverses approches (EncFS[33], Loop-AES[34], dm-crypt[35], TrueCrypt[36]). De même, les instances Amazon EC2 fonctionnant sous OpenSolaris peuvent utiliser ZFS[37] Encryption Support [20]. Quelle que soit l'approche sélectionnée, les fichiers et volumes de chiffrage dans Amazon EC2 contribuent à la protection des fichiers et données de log, si bien que seuls les utilisateurs et processus du serveur peuvent voir les données en texte en clair, alors que tout objet ou personne en dehors du serveur ne peut voir que des données chiffrées.

Quel que soit le système d'exploitation ou la technologie que vous choisissez, les données de chiffrage au repos constituent un défi pour ce qui est de gérer les clés utilisées pour chiffrer les données. Si vous perdez les clés, vous perdez vos données à jamais, et si vos clés sont découvertes, les données peuvent être en situation de risque. Assurez-vous donc d'étudier les capacités de gestion de clé de tous les produits que vous choisissez et d'établir une procédure pour minimiser les risques de perte des clés.

En plus de la protection de vos données contre l'indiscrétion, envisagez de les protéger contre les sinistres. Capturez des instantanés périodiques de vos volumes Amazon EBS afin d'assurer leur grande durabilité et disponibilité. Les instantanés sont incrémentiels de par leur nature, ils sont stockés sur Amazon S3 (un géo-emplacement séparé) et ils peuvent être restaurés à l'aide de quelques clics ou appels de ligne de commande.

Protégez vos certificats AWS

AWS fournit deux types de certificats de sécurité : clés d'accès AWS et certificats X.509. Votre clé d'accès AWS se compose de deux parties : votre *ID de clé d'accès* et votre *clé d'accès secrète*. Lorsque vous utilisez REST ou Query API, vous devez employer votre clé d'accès secrète pour calculer une signature pour inclure votre demande d'authentification. Pour prévenir la falsification en vol, toutes les demandes doivent être envoyées par HTTPS.

---

[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/

Si votre Amazon Machine Image (AMI) est en train d'exécuter des processus qui doivent communiquer avec d'autres services web AWS (pour appeler la file d'attente Amazon SQS queue ou pour lire des objets dans Amazon S3, par exemple), une erreur de conception courante est d'intégrer les certificats AWS dans l'AMI. Au lieu d'intégrer les certificats, ils devraient être passés comme arguments pendant le lancement et chiffrés avant d'être transférés [17].

Si votre clé d'accès secrète est découverte, vous devriez en obtenir une nouvelle par rotation[38] sur un nouvel ID d'accès. À titre de bonne pratique, il est recommandé d'incorporer un mécanisme de rotation de clé dans votre architecture d'application afin de l'utiliser régulièrement ou occasionnellement (lorsqu'un employé mécontent quitte l'entreprise) afin de garantir que les clés à risque ne durent pas pour toujours.

Vous pouvez aussi utiliser les certificats X.509 pour authentifier certains services AWS. Le fichier de certificat contient votre clé publique dans un corps de certificat DER à codage base64. Un fichier séparé contient la clé privée PKCS#8 à codage base64.

AWS accepte l'authentification multi-facteurs[39] en tant que protection supplémentaire pour travailler avec vos informations de compte sur aws.amazon.com et AWS Management Console[40].

Sécurisez votre application

Chaque instance Amazon EC2 est protégée par un ou plusieurs *groupes de sécurité*[41], des ensembles de règles nommées qui précisent quel trafic de réseau entrant doit être transmis à votre instance. Vous pouvez indiquer les ports TCP et UDP, les types et codes ICMP et les adresses sources. Les groupes de sécurité vous offrent une protection de base semblable à celle des pare-feux pour le fonctionnement des instances. Par exemple, les instances appartenant à une application web peuvent comporter les paramètres de groupe de sécurité suivants :

[38] http://aws.amazon.com/about-aws/whats-new/2009/08/31/seamlessly-rotate-your-access-credentials/
[39] Vous pouvez obtenir davantage d'informations sur l'authentification multi-facteurs sur http://aws.amazon.com/mfa/
[40] AWS Management Console http://aws.amazon.com/console/
[41] Vous pouvez obtenir davantage d'informations sur les groupes de sécurité sur http://docs.amazonwebservices.com/AWSEC2/2009-07-15/UserGuide/index.html?using-network-security.html

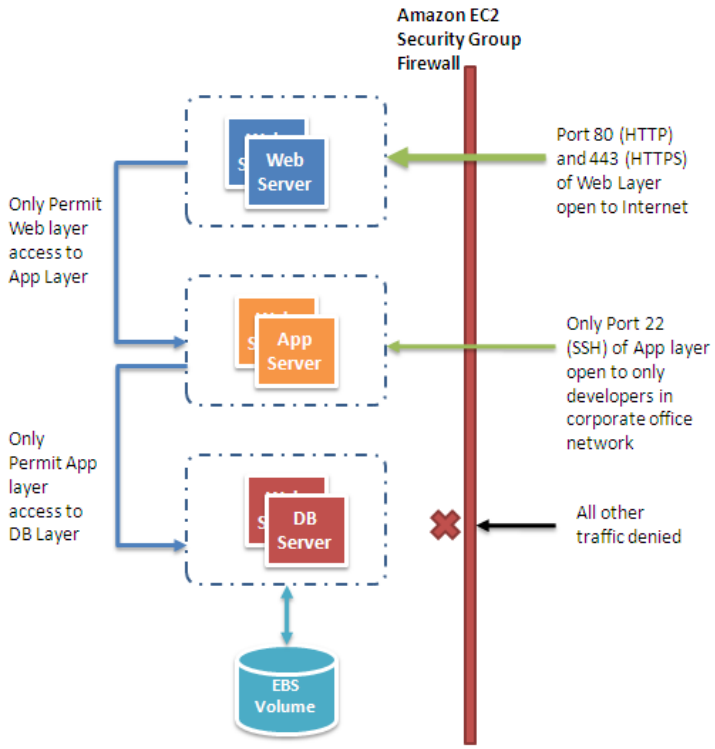

Figure 4: sécurisez votre application web au moyen des groupes de sécurité Amazon EC2

Une autre façon de restreindre le trafic entrant consiste à configurer des pare-feux logiciels sur vos instances. Les instances Windows peuvent se servir du pare-feu intégré[42]. Les instances Linux peuvent se servir du *netfilter*[43] et des *iptables*.

Avec le temps, les erreurs de logiciel sont détectées et réparées à l'aide de patchs. Vous devriez respecter les directives de base suivantes pour maximiser la sécurité de votre application :

- Téléchargez régulièrement les patchs du site web du fournisseur et mettez à jour vos AMI
- Redéployez les instances des nouveaux AMI et testez vos applications pour assurer que les patchs ne causent pas de dégâts. Assurez que les derniers AMI soient déployés sur *toutes* les instances
- Investissez dans des scripts de test pour exécuter des vérifications de sécurité de façon périodique et automatisez le processus
- Assurez que le logiciel tiers est configuré avec les paramètres offrant le plus de sécurité
- N'exécutez jamais vos processus comme *racine* ou *administrateur* à moins que cela ne soit absolument nécessaire

Toutes les pratiques de sécurité standard de l'ère pré-cloud, telles que les bonnes pratiques de codage, l'isolation des données sensibles, sont toujours applicables et devraient être mises en œuvre.

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

Rétrospectivement, le nuage absorbe la complexité de la sécurité physique pour vous laisser aux commandes des outils et fonctionnalités vous permettant de sécuriser votre application.

Futures orientations en matière de recherche

Le moment où les applications ne s'appuieront plus sur un matériel physique n'est plus si éloigné. Tout comme le branchement d'un micro-ondes ne nécessite aucune connaissance électrique, il sera possible de *brancher* une application dans le nuage afin de recevoir la puissance requise pour la faire fonctionner, tout comme tout appareil d'usage général. En tant qu'architecte, vous gérerez des ressources abstraites de calcul, stockage et réseau à la place de serveurs physiques. Les applications continueront à fonctionner, que le matériel physique sous-jacent tombe en panne, qu'il soit retiré ou qu'il soit remplacé. Les applications s'adapteront aux schémas changeants des demandes en déployant les ressources *instantanément* et automatiquement, obtenant ainsi les niveaux d'utilisation les plus élevés en permanence. La dimensionnabilité, la sécurité, la grande disponibilité, la résistance aux pannes, la testabilité et l'élasticité seront des propriétés configurables de l'architecture de l'application, et elles seront une partie automatisée et intégrale de la plateforme sur laquelle elles sont construites.

Cependant, nous n'en sommes pas encore là. Vous pouvez actuellement développer des applications dans le nuage avec quelques unes de ces qualités en mettant en œuvre les bonnes pratiques mises en évidence dans ce document. Les bonnes pratiques en matière d'architectures de cloud computing continueront à évoluer et, en tant que chercheurs, nous devrions tourner notre attention non seulement sur l'amélioration du nuage, mais aussi sur le développement d'outils, technologies et processus qui permettront aux développeurs et architectes de brancher les applications au nuage avec la plus grande facilité.

Conclusion

Ce document a fournit une orientation normative pour que les architectes cloud conçoivent des applications efficaces dans le nuage.

En se concentrant sur les concepts et les bonnes pratiques - telles que la conception en pensant aux défaillances, le découplage des composants d'application, la compréhension et la mise en œuvre de l'élasticité et son association à la parallélisation, et l'intégration de la sécurité dans chaque aspect de l'architecture de l'application - les architectes cloud peuvent comprendre quelles sont les considérations de conception qui sont nécessaires pour construire des applications hautement dimensionnables dans le nuage.

Le nuage AWS offre des services d'infrastructure hautement fiables et payables à l'utilisation. Les tactiques AWS spécifiques mises en évidence dans ce document permettront de concevoir des applications de nuage employant ces services. En tant que chercheur, il vous est conseillé de jouer avec ces services commerciaux, d'apprendre à partir du travail de vos collègues, et construire en haut, d'améliorer et de réinventer le cloud computing.

Remerciements

L'auteur est profondément reconnaissant à Jeff Barr, Steve Riley, Paul Horvath, Prashant Sridharan et Scot Marvin pour leurs observations sur les versions préliminaires de ce document. Un grand merci à Matt Tavis pour les éléments de réflexion qu'il a apporté. Sans ces contributions, ce document n'aurait pas vu le jour.

Une partie du contenu de ce livre blanc est dérivé d'un chapitre écrit par le même auteur et publié dans le livre 'Cloud Computing: Paradigms and Patterns', Copyright © 2010 John Wiley & Sons, Inc.

Références et lectures complémentaires

1. **Amazon S3 Team, Best Practices for using Amazon S3,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1904, 2008-11-26
2. **Amazon S3 Team, Amazon S3 Error Best Practices,** http://docs.amazonwebservices.com/AmazonS3/latest/index.html?ErrorBestPractices.html, 2006-03-01
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
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
5. **Amazon SimpleDB Team, Query 101: Building Amazon SimpleDB Queries,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1231&categoryID=176, 2008-02-07
6. **J. Varia, Cloud Architectures,** http://jineshvaria.s3.amazonaws.com/public/cloudarchitectures-varia.pdf, 2007-07-01
7. **Amazon Security Team, Overview of Security Processes,** http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf, 2009-06-01
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
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
10. D. Pritchett, Shard Lessons, http://www.addsimplicity.com/adding_simplicity_an_engi/2008/08/shard-lessons.html, 2008-08-24
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
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
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
15. E. Hammond, Escaping Restrictive/Untrusted Networks with OpenVPN on EC2, http://alestic.com/2009/05/openvpn-ec2, 2009-05-02
16. R. Bragg, The Encrypting File System, http://technet.microsoft.com/en-us/library/cc700811.aspx, 2009
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
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
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

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

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