SYSTEME ET INFRASTRUCTURE

Cloud Computing avec Amazon AWS

Ce qu'il faut savoir

Patrice **BERTRAND**
Directeur général

Smile
OPEN SOURCE SOLUTIONS

www.smile.fr • +33 (0)1 41 40 11 00 • contact@smile.fr
www.smile-oss.com • blog.smile.fr • twitter: @GroupeSmile

PRÉAMBULE

Smile

Smile est une société d'ingénieurs experts dans la mise en œuvre de solutions open source et l'intégration de systèmes appuyés sur l'open source. Smile est membre de l'APRIL, l'association pour la promotion et la défense du logiciel libre, de Alliance Libre, PLOSS, et PLOSS RA, des associations clusters régionaux d'entreprises du logiciel libre.

Smile compte 480 collaborateurs en France, 600 dans le monde, ce qui en fait la première société en France spécialisée dans l'open source.

Depuis 2000, environ, Smile mène une action active de veille technologique qui lui permet de découvrir les produits les plus prometteurs de l'open source, de les qualifier et de les évaluer, de manière à proposer à ses clients les produits les plus aboutis, les plus robustes et les plus pérennes.

Cette démarche a donné lieu à toute une gamme de livres blancs couvrant différents domaines d'application. La gestion de contenus (2004), les portails (2005), la business intelligence (2006), les frameworks PHP (2007), la virtualisation (2007), et la gestion électronique de documents (2008), ainsi que les PGIs/ERPs (2008). Parmi les ouvrages publiés en 2009, citons également « Les VPN open source », et « Firewall est Contrôle de flux open source », et « Middleware », dans le cadre de la collection « Système et Infrastructure ».

Chacun de ces ouvrages présente une sélection des meilleures solutions open source dans le domaine considéré, leurs qualités respectives, ainsi que des retours d'expérience opérationnels.

Au fur et à mesure que des solutions open source solides gagnent de nouveaux domaines, Smile sera présent pour proposer à ses clients d'en bénéficier sans risque. Smile apparaît dans le paysage informatique français comme le prestataire intégrateur de choix pour accompagner les plus grandes entreprises dans l'adoption des meilleures solutions open source.

Ces dernières années, Smile a également étendu la gamme des services proposés. Depuis 2005, un département consulting accompagne nos clients, tant dans les phases d'avantprojet, en recherche de solutions, qu'en accompagnement de projet. Depuis 2000, Smile dispose d'un studio graphique, devenu en 2007 Smile Digital – agence interactive, proposant outre la création graphique, une expertise e marketing, éditoriale et interfaces riches. Smile dispose aussi d'une agence spécialisée dans la TMA (support et l'exploitation des applications) et d'un centre de formation complet, Smile Training. Enfin, Smile est implanté à Paris, Lille, Lyon, Grenoble, Nantes, Bordeaux, Poitiers, Aix-en-Provence et Montpellier. Et présent également en Espagne, en Suisse, au Benelux, en Ukraine et au Maroc.

Quelques références

Intranets et Extranets

Société Générale - Caisse d'Épargne - Bureau Veritas - Commissariat à l'Energie Atomique - Visual - CIRAD - Camif - Lynxial - RATP - Sonacotra - Faceo - CNRS - AmecSpie - INRA - CTIFL - Château de Versailles - Banque PSA Finance - Groupe Moniteur - Vega Finance - Ministère de l'Environnement - Arjowiggins - JCDecaux - Ministère du Tourisme - DIREN PACA - SAS - CIDJ - Institut National de l'Audiovisuel - Cogedim - Diagnostica Stago Ecureuil Gestion - Prolea - IRP-Auto - Conseil Régional Ile de France - Verspieren - Conseil Général de la Côte d'Or - Ipsos - Bouygues Telecom - Prisma Presse - Zodiac - SANEF - ETS Europe - Conseil Régional d'Ile de France - AON Assurances & Courtage - IONIS - Structis (Bouygues Construction) - Degrémont Suez - GS1-France - DxO - Conseil Régional du Centre - Beauté Prestige International - HEC - Veolia

Internet, Portails et e-Commerce

Cadremploi.fr - chocolat.nestle.fr - creditlyonnais.fr - explorimmo.com - meilleurtaux.com - cogedim.fr - capem.fr - Editions-cigale.com - hotels-exclusive.com - souriau.com - pci.fr - odit-france.fr - dsv-cea.fr - egide.asso.fr - Osmoz.com - spie.fr - nec.fr - vizzavi.fr - sogeposte.fr - ecofi.fr - idtgv.com - metro.fr - stein-heurtey-services.fr - bipm.org - buitoni.fr - aviation-register.com - cci.fr - eaufrance.fr - schneider-electric.com - calypso.tm.fr - inra.fr - cnil.fr - longchamp.com - aesn.fr - bloom.com - Dassault Systemes 3ds.com - croix-rouge.fr - worldwatercouncil.org - Projectif - credit-cooperatif.fr - editionsbussiere.com - glamour.com - nmmedical.fr - medistore.fr - fratel.org - tiru.fr - faurecia.com - cidil.fr - prolea.fr - bsv-tourisme.fr - yves.rocher.fr - jcdecaux.com - cg21.fr - veristar.com - Voyages-sncf.com - prismapub.com - eurostar.com - nationalgeographic.fr - eau-seine-normandie.fr - ETS Europe - LPG Systèmes - cnous.fr - meddispar.com - Amnesty International - pompiers.fr - Femme Actuelle - Stanhome-Kiotis - Gîtes de France Bouygues Immobilier - GPdis - DeDietrich - OSEO - AEP - Lagardère Active Média - Comexpo - Reed Midem - UCCIFE - Pagesjaunes Annonces - 1001 listes - UDF - Air Pays de Loire - Jaccede.com - ECE Zodiac - Polytech Savoie - Institut Français du Pétrole - Jeulin - Atoobi.com - Notaires de France - Conseil Régional d'Ile-de-France - AMUE

Applications métier

Renault - Le Figaro - Sucden - Capri - Libération - Société Générale - Ministère de l'Emploi - CNOUS - Neopost - Industries - ARC - Laboratoires Merck - Egide - ATEL-Hotels - Exclusive Hotels - CFRT - Ministère du Tourisme - Groupe Moniteur - Verspieren - Caisse d'Epargne - AFNOR - Souriau - MTV - Capem - Institut Mutualiste Montsouris - Dassault Systèmes - Gaz de France - CAPRI Immobilier - Croix-Rouge Française - Groupama - Crédit Agricole - Groupe Accueil - Eurordis - CDC Arkhineo

Applications décisionnelles

IEDOM – Yves Rocher - Bureau Veritas - Mindscape – Horus Finance – Lafarge – Optimus – CecimObs – ETS Europe – Auchan Ukraine – CDiscount – Maison de la France – Skyrock – Institut National de l'Audiovisuel – Pierre Audouin Consultant – Armée de l'air – Jardiland – Saint-Gobain Recherche – Xinek – Projectif – Companeo – MeilleurMobile.com – CG72 – CoachClub

Ce livre blanc

Ce livre blanc est une présentation synthétique de l'offre de Amazon en matière de Cloud Computing, intitulée *Amazon Web Services*, incluant en particulier S3 (Stockage sécurisé), EC2 (serveurs on-demand), EBS (partition disque pérenne), SQS (middleware orienté messages) et CloudFront (content delivery network).

L'offre Amazon est bien documentée, et il existe une littérature abondante sur le sujet. La valeur ajoutée de ce livre blanc est en premier lieu dans son caractère synthétique: il dresse en une trentaine de page, un panorama des principaux services offerts. Elle est également dans l'analyse que nous nous efforçons d'en faire: ses atouts, faiblesses, risques, et domaines d'application privilégiés.

Comme on le verra, l'offre Amazon entre dans la catégorie dénommée IaaS, « Infrastructure as a Service », que nous positionnerons aux côtés des offres de type « Software as a Service » ou « Platform as a Service », qui ensemble composent le Cloud Computing.

Smile possède une petite antériorité en matière de Cloud Computing, ayant commencé à y construire une expertise depuis plus de deux ans, alors même que les offres étaient encore balbutiantes.

Aujourd'hui, Smile peut aider ses clients à bien analyser les bénéfices qu'ils peuvent tirer de ce type d'offre, et à en faire eux-mêmes le meilleur usage.

Table des matières

Le Cloud Computing

Introduction

Quelle différence entre l'hébergement externalisé et le cloud computing ?

Depuis les débuts du web, de nombreux sites sont hébergés auprès de prestataires spécialisés. Certains d'entre eux, généralement plutôt dans le low-cost, ont industrialisé leurs processus de commande. Le client s'identifie, donne quelques options de configuration de son serveur, entre un numéro de carte bancaire, valide sa commande, et quelques minutes ou heures plus tard son serveur est disponible.

En évitant l'étape d'échange avec un commercial chargé d'élaborer une offre sur mesure, et d'en négocier le prix, on a déjà une économie significative.

Ces services d'hébergement accédés en ligne ne sont pas très différent de ce que propose Amazon. Les différences sont minimes, mais elles constituent un saut quantique malgré tout.

Elles concernent:

- Les services environnants
- La location à l'heure, et donc le véritable *on demand.*
- Le pilotage complet de l'infrastructure par *webservices*.

Commander son serveur en self-service avec un navigateur et une carte bancaire, c'est bien, mais le commander par web-services, cela ouvre des perspectives fondamentalement différentes.

Une des forces de l'offre AWS, c'est que la totalité de l'infrastructure peut être pilotée par *webservices*, c'est-à-dire donc par des applications. Si l'on veut 100 serveurs, il suffit d'une boucle de 1 à 100 sur l'appel du *webservice* correspondant (en réalité, une commande unique suffira), et les voilà alloués et opérationnels.

Une autre différence importante est que Amazon se positionne dans le haut de gamme, avec à certains égards les prix du bas de gamme.

Dans l'hébergement traditionnel, il fallait choisir entre:

- Le bas de gamme, commandé et configuré en ligne, aux prix serrés, mais aux services limités.
- Le haut de gamme commandé en spécifique, avec des prix beaucoup plus élevés.

Non seulement AWS réunit les deux, mais en y ajoutant une panoplie de services que même les hébergeurs haut de gamme traditionnels n'offrent pas, du moins pas packagés de cette manière.

Quel degré de captivité ?

Il est important pour le client de bien mesurer le degré de verrouillage qu'il peut subir en faisant appel à des services web.

Et il est essentiel de se poser la question : puis-je me dégager de Amazon un jour, et à quel prix ?

Certains pourront accepter la dépendance, considérer que Amazon est un acteur solide, qui saura maintenir une qualité de service satisfaisante sur le long terme, et qui maintiendra aussi des prix compétitifs.

Mais imaginons que ce ne soit pas le cas. Imaginons que dans deux ans, la qualité se soit dégradée, ou bien que d'autres acteurs aient une offre plus compétitive. Sera-t-il encore possible d'extraire sa plateforme de chez Amazon, et de la mettre ailleurs ?

On ne peut pas occulter cette question.

C'est pourquoi notre recommandation serait d'utiliser le cloud Amazon comme un hébergement, accompagné de services précieux, mais dont on pourrait se détacher un jour. Concrètement, cela signifie de ne pas construire des applications représentant d'importants investissements et qui soient indissociables des services AWS. Typiquement, on préfèrera mettre en œuvre un excellent outil de Message Queueing open source plutôt que de faire appel à SQS de manière structurante.

Le pricing

Pour chacun des services évoqués, nous analyserons la politique de prix, principe et unités d'œuvre.

Ce n'est pas juste à titre d'information : le prix est une composante essentielle dans l'évaluation des services AWS.

L'offre de Amazon est, d'une manière générale très compétitive pour ce qui concerne le stockage et la bande passante, le trafic, un peu moins pour ce qui est des serveurs eux-mêmes.

Le tour du marché

Comme on le verra, l'offre d'Amazon a deux principales caractéristiques :

- Elle donne un accès complet aux serveurs au niveau le plus bas (ou presque), (un accès *root* sur les serveurs), ce qui permet d'en faire absolument n'importe quoi.
- La location de CPU s'accompagne d'un ensemble de services plus diversifiés, stockage, file de message, etc.

Amazon a été clairement le précurseur en la matière, mais ces dernières années, de nombreux acteurs s'y lancent également.

Google App Engine

Google avec son App Engine (GAE), a une offre radicalement différente dans l'esprit : au lieu de donner le contrôle complet sur des serveurs, App Engine n'offre que des environnements d'exécution de programmes bien bornés, initialement Python et maintenant Java.

Pour un développeur ordinaire, le principe est plus simple : il n'a pas à être administrateur du serveur en plus d'être développeur. Les outils sont bien packagés et bien documentés, il est possible également de développer et tester sur son poste de travail.

Les deux grands bénéfices du App Engine sont :

- La scalabilité, c'est-à-dire la capacité à monter en puissance, à servir des millions de pages par jour sans se préoccuper le moins du monde d'infrastructure.
- Le ticket d'entrée très bas, avec une utilisation gratuite au début.

Ce sont des conditions qui pourraient séduire des startups du web. Mais notre analyse est qu'elles conviennent beaucoup moins aux entreprises, pour deux raisons :

- La dépendance de l'application à l'environnement Google est extrême. Même si le langage, Python ou Java, est standard, il ne sera pas possible d'extraire une application App Engine

pour la faire tourner ailleurs. On hésitera donc à investir plusieurs dizaines d'années-hommes sur ce socle.

- Il s'agit purement d'un environnement de développement et d'exécution, il n'est pas possible d'y faire tourner des progiciels du marché en l'état. Or les entreprises recourent au moins autant à des progiciels qu'à du développement spécifique. Voire à des progiciels complétés par du logiciel spécifique. Tout cela n'est pas possible avec App Engine.

Une autre offre intéressante est le Secure Data Connector, qui permet d'accéder aux données de l'enterprise à partir des applications GAE.

GoGrid

GoGrid a une offre assez semblable à celle de Amazon, qui semble même en être inspirée. Les serveurs sont alloués à la demande, et loués à l'heure, et comme chez Amazon, le client a un accès *root* sur ses serveurs.

Les caractéristiques principales de GoGrid sont :

- Un fort partenariat avec Microsoft, et donc une offre plus riche autour de ces configurations
- Une console de gestion centrale complète et ergonomique. Mais avec malgré tout une API également pour piloter ses serveurs par programme.
- Une offre de load balancing intégrée, à base de BigIP, alors que le load-balancing n'a été introduit qu'en juin 2009 sur AWS.

Il est assez sain qu'il y ait une compétition sérieuse dans l'offre cloud, mais pour l'instant GoGrid est un peu en retrait par rapport à Amazon :

- Pas de datacenters en Europe
- Le service de stockage sécurisé est encore en béta (mai 2009)
- Moins de services périphériques, pas de CDN.
- Des prix légèrement plus élevés.

IBM, Yahoo, HP et les autres

Courant 2008, le *cloud computing* est devenu à la mode, et quelques grands acteurs se sont aperçu qu'ils avaient laissé Amazon prendre 3 ans d'avance.

Certains ont les moyens de rattraper leur retard rapidement, mais ils ont aussi de sérieux conflits d'intérêt, le fameux « *innovator's dilemna* », qui empêche un acteur d'aller vers un marché nouveau car il lui faudrait sacrifier son lucratif marché antérieur.

Le cas le plus flagrant est celui de IBM, qui a annoncé, presqu'à regrets, une offre cloud à l'été 2008, en décrivant abondamment ses nouveaux datacenters, leur superficie et leur moindre dégagement de CO2. Mais de l'offre elle-même, il n'était guère question. Certaines des annonces qui ont suivi, avec le label « cloud » portaient en fait sur une offre Saas, en matière de groupware. A ce jour (mai 2009), la page « cloud computing » de www.ibm.com (http://www.ibm.com/ibm/cloud/) dit essentiellement « *nous pouvons vous aider à aller vers des infrastructures dynamiques* » *(i.e. élastiques)*, et par ailleurs évoque les images machines faites par IBM pour Amazon EC2.

Pour l'instant, il n'est pas possible de louer un serveur IBM en mode cloud.

A l'été 2008, HP, Yahoo et Intel ont fait une annonce commune promettant de s'intéresser au cloud computing, mais il n'en est rien sorti de concret à cette heure.

Microsoft

Enfin, il faut citer aussi les annonces de Microsoft, à fin 2008, avec la promesse d'un nouveau système d'exploitation dénommé Windows Azure, associé à la *Azure Platform,* un ensemble de services qui devraient être disponible fin 2009, et serait le socle d'une offre *cloud* Microsoft. Tout cela est encore assez théorique donc pour l'instant, mais on peut prévoir que :

- Le fait de s'appuyer sur un OS spécifique ne donnera clairement pas un degré de liberté très grand en termes de configuration
- Malgré tout, cela peut présenter quelques avantages dans le sens d'un accès semi-transparent aux services. On peut imaginer par exemple qu'une application fasse un accès disque standard, qui soit transformé en un appel du service de stockage.
- On ne fera pas tourner des progiciels ordinaires sur cette plateforme, et les applications écrites pour Azure ne pourront tourner que sur Azure, même si différents langages standards seront supportés.
- Microsoft misera sur l'intégration à Visual Studio pour faciliter la prise en main de la plateforme.

SaaS, PaaS, IaaS ...

Le Cloud Computing se développe à une vitesse extraordinaire, et tous les grands acteurs du web y lancent des offres. Mais il ne s'agit pas de produits similaires, tels des baladeurs MP3, dont on pourrait comparer une à une les caractéristiques. Il y a des approches radicalement différentes, sur un spectre qui va du Saas pur jusqu'à l'hébergement traditionnel.

Il faut bien comprendre la distinction entre ces offres:

- Le SaaS, « *Software as a Service* », est l'offre d'un service prêt à l'emploi accessible sur le web. La figure emblématique étant Salesforce.com, service de gestion de la relation client (CRM). Ici le client utilise le service, et ne se soucie ni d'infrastructure, ni d'hébergement, ni même de version de programme. Le plus souvent, il configure lui-même son service, sans faire appel à un prestataire. Le service entre dans un moule relativement simple.

- Le PaaS, terme moins usité, signifie « *Platform as a Service* », et correspond à l'offre AppEngine de Google ou à la future offre Azure de Microsoft. Ici, le client peut développer (ou faire développer) une application pour l'environnement d'exécution cible. Les langages peuvent être standards, mais les APIs sont spécifiques: on ne peut pas porter simplement une application existante. En revanche, dans ce contexte, la question d'infrastructure et de dimensionnement est ensuite oubliée: la plateforme s'en charge de manière transparente.

- Le IaaS, « Infrastructure as a Service », qui correspond à l'offre de Amazon. Ici le client loue de l'infrastructure « on-demand », et des services environnants. Il peut en maîtriser la configuration logicielle, et faire tourner n'importe quel produit. En revanche, si la ressource serveur est facile à obtenir, il appartient au client de savoir en faire usage, c'est à dire de créer des architectures extensibles.

- Enfin l'hébergement traditionnel, où le client a une plus grande maîtrise sur les configurations matérielles, mais une moindre réactivité en montée en charge.

Le SaaS, ami ou ennemi de l'open source ?

[Reproduction d'un article paru dans le magazine « 01 Informatique » du 14 mai 2009, tribune libre signée Patrice Bertrand]

Le Saas s'appuie largement sur l'open source, et réciproquement les éditeurs open source sont attirés par le modèle Saas. Mais « loin des yeux, loin du cœur », si le programme s'estompe en devenant service, fera-t-il oublier les bénéfices de l'open source ?

Le monde de l'open source est partagé face à la montée en puissance du Saas. Le père fondateur Richard Stallman y voit « pire qu'une imbécilité » : la perte de contrôle de l'utilisateur non seulement sur les programmes, mais pire, sur les informations. A l'inverse, bon nombre d'éditeurs open source discernent dans le Saas la possibilité d'un business model enfin limpide, qu'ils cherchent encore à mettre au point.

Regardons-y de plus près.

Le Software as a Service, comme son nom l'indique, substitue le service au logiciel : le programme n'est plus, vive le service. Si le programme disparait, le code source du programme disparait plus encore, et c'est le fondement même de l'open source qu'on abolit. En première analyse, le Saas serait une forme de bundle mêlant application et hébergement, confondus pour offrir un service. Mais, la vraie révolution du Saas tient en partie dans la démarche « DIY », Do It Yourself : l'application arrive totalement prête à l'emploi, en self-service. Il reste si peu de configuration, et d'une telle simplicité, que le client la prend en charge sans aucune expertise. Le Saas entraine aussi la fin du prestataire informatique. Du moins pour les catégories d'applications éligibles, car on ne pourra pas tout faire en mode DIY. Les fournisseurs d'offres Saas s'appuient massivement sur l'open source. Ils veulent construire des applicatifs web de grande envergure sur des socles modernes et extensibles, et l'open source offre le meilleur des socles, une manne. Mais Google, à certains égards numéro 1 du SaaS, et qui raffole d'open source, se trouve parfois accusé de ne pas assez reverser ses œuvres dérivées de l'open source. En effet, vis-à-vis des exigences de la licence GPL (1), mettre à disposition un service, gratuit ou pas, ne s'assimile pas à distribuer un programme, de sorte que l'on peut partir d'un code sous GPL, construire une œuvre dérivée et la proposer en mode SaaS, sans se voir obligé à diffuser cette œuvre sous GPL. Cette faille, sérieuse, a été comblée par la licence AGPL, qui prend en compte ce cas de figure. Mais celle-ci reste modestement utilisée pour le moment.

En contrepoint, le SaaS semble une opportunité pour l'écosystème open source. En effet, si les acteurs du Libre martèlent que « Libre

ne signifie pas gratuit », en réalité, une vraie licence Libre ne permet pas de faire payer un droit d'utilisation. Les éditeurs open source se tournent donc vers d'autres sources de revenus : maintenance, versions « entreprise » non-libres, plus stables ou bien aux fonctionnalités élargies. Or réunir application et hébergement dans un bundle Saas apporte une réponse limpide à la question « comment sera financé un logiciel Libre ?» Le client ne payera ni un droit d'utilisation, ni un serveur, ni un hébergement, ni les jours d'un prestataire : il paye un service. Ainsi, pour les éditeurs open source, le Saas réunit deux attraits : une simplicité de déploiement et d'exploitation qui répond à une réelle attente du marché, et un modèle économique enfin parfaitement lisible.

Parfait, mais que deviennent les sources ? Le risque est réel que les programmes, une fois invisibles, ne deviennent insignifiants. Les sources de l'application mise en Saas resteront-elles disponibles ? Voilà la question que doit se poser le client. Car à certains égards, le verrouillage (lock-in) dans le Saas pourrait s'avérer bien pire que le celui d'un logiciel propriétaire : non seulement changer de programme s'avérera compliqué, mais même la récupération des données posera problème. Et si récupérer ses données est surtout affaire contractuelle, pour qu'elles soient utilisables, il faut encore que le programme reste disponible, et libre.

Les vraies caractéristiques des solutions open source, si elles ne tiennent pas à la gratuité, s'inscrivent dans le respect des standards, dans l'ouverture, dans la dynamique de développement, les contributions communautaires, la rapidité de maturation. Que deviendront ces atouts en mode Saas ? Pourra-t-on faire cohabiter le foisonnement d'une offre communautaire avec l'engagement de l'éditeur du Saas en termes de qualité de service ?

C'est l'enjeu. Mais il serait risqué d'oublier que derrière le Saas, il y a des programmes.

Eucalyptus

Justement, puisque nous avons évoqué le risque de verrouillage, il est intéressant de citer le projet Eucalyptus (www.eucalyptus.com), qui propose une implémentation open source des APIs AWS.

Cela signifie que vous pouvez vous construire votre propre environnement cloud, compatible Amazon.

Bien sûr, vous ne pourrez pas bénéficier ainsi de la totalité des avantages de l'offre AWS:

- Les investissements matériel devront être assumés dès le départ.
- La ressource matérielle ne sera pas infinie.
- Vous aurez plus de difficulté à mettre en place des zones de disponibilité et des régions distinctes et plus encore un CDN.

A l'inverse, vous pourrez peut-être obtenir de meilleurs prix sur le matériel.

Mais en fait, le plus important est ailleurs: l'existence même de la solution Eucalyptus est un gage de liberté. En effet, cela signifie que des clients peuvent construire une infrastructure sur AWS, et en sortir un jour, mais aussi rester dans une logique mixte: AWS d'un côté pour l'allocation à la demande, et Eucalyptus de l'autre pour la partie plus stable de l'infrastructure.

Les services AWS

Amazon a introduit son offre « Amazon Web Services » en 2006, d'abord de manière assez discrète. La première offre était un service de stockage, le « Secure Storage Service », ou S3. Très vite est arrivé le « Elastic Compute Cloud », la location de serveurs à l'heure, qui est restée en béta jusqu'à fin 2008.

Etonnant, pour un libraire ! Mais la démarche avait une certaine logique: Amazon exploite, pour ses besoins des milliers de serveurs, et Amazon a mis au point, pour ses besoins, des outils spécifiques permettant une bonne gestion de ces datacenters. Pourquoi ne pas en faire un business à part entière ?

Petit à petit, l'offre s'est étoffée, et s'est fait connaître plus largement. Lorsque, courant 2008, le cloud computing est devenu le sujet branché du moment, Amazon avait pris quelques années d'avance, tant en richesse des services qu'en maturité de l'offre.

S3: Secure Storage Service

S3 est un service de stockage sécurisé à haute disponibilité, accédé entièrement en *webservices*, et d'un coût très compétitif.

Objects

S3 stocke des « S3 objects », que nous appellerons « objets ». Un objet est plus ou moins un fichier. Comme un fichier il peut être de n'importe quelle taille, de 0 octet à 5 GO, et contenir toute sorte d'information.

Mais il y a de petites différences, qui sont importantes.

Pas d'accès direct

Il n'y a pas d'accès direct sur l'intérieur d'un objet S3 (c'est-à-dire qu'il n'est pas possible d'accéder au 855ème octet, ni de modifier de manière directe certaines parties de l'objet). On ne peut que (a) lire l'objet dans sa totalité, (b) modifier éventuellement tout ou partie de l'objet et (c) écrire l'objet modifié, dans sa totalité.

Propagation asynchrone

Un objet est stocké sur plusieurs serveurs dans les datacenters de Amazon. C'est ce qui assure la forte sécurisation. Mais cette

gestion des copies n'est pas atomique, n'est pas synchrone, c'est-à-dire que à la suite d'une modification d'un objet, il y a une petite latence dans la propagation de l'objet sur les différents serveurs de Amazon. Pendant cet intervalle, il est possible que différentes applications lisant l'objet obtiennent des objets différents. Il est même possible que l'application qui vient d'écrire l'objet lise elle-même la version antérieure !

Ces particularités peuvent être rédhibitoires dans certains contextes, mais le plus souvent elles sont acceptables. D'autant plus que cette propagation est, dans la pratique, de quelques secondes seulement.

Metadonnées

Chaque objet S3 possède des données et des métadonnées. Les données, c'est corps de l'objet, par exemple une image Jpeg. Les métadonnées sont des informations accessoires, qui aident à la gestion de l'objet. Certaines sont renseignées par défaut, par exemple la date de création de l'objet, mais d'autres peuvent être définies de manière spécifique par l'utilisateur.

Bucket

Un *bucket* est un espace de stockage d'objets, un peu comme un répertoire réunit des fichiers. Mais il n'y a pas de sous-niveaux et d'arborescence : tous les *buckets* de tous les utilisateurs de AWS sont au même niveau.

Cela implique que chaque *bucket* doit avoir un nom unique de manière absolue. Comme on le verra plus loin, il peut être utile d'utiliser un nom de *bucket* à la manière d'un nom de sous-domaine. Pour pouvoir faire cela, il faut avoir choisi un nom compatible, c'est-à-dire n'utilisant que des lettres minuscules et chiffres, tirets et points (et quelques autres contraintes).

Chaque *bucket* appartient à un *user* donné, et il est possible de définir les droits d'accès pour l'ensemble des objets du *bucket*.

Contrôle d'accès

Par défaut, un objet S3 ne peut être accédé que par son créateur, mais celui-ci peut en autoriser l'accès à d'autres au moyen d'ACP, « Access Control Policy », équivalent à des ACL, « Access Control List », terme plus usité. La gestion des ACP porte soit sur des *buckets*, soit sur des objets individuels.

Les droits ne peuvent être accordés que de manière positive (i.e. on ne peut pas raisonner par interdiction). Chaque item d'une ACP est un « *grant* », une permission, accordée à un *grantee*. Le *grant*

porte sur une action, qui peut être *read, write,* portant sur la lecture et l'écriture de l'objet lui-même, ou encore *read_acp, write_acp*, portant sur l'accès à la liste ACP. Bien sûr en possédant une permission *write_acp*, un user peut s'accorder toute autre permission.

Le *grantee*, celui à qui la permission est accordée est en général un *user* AWS authentifié, défini par son ID. Il n'est pas possible de définir et de gérer des *groupes* réunissant différents *users*. Il existe toutefois des groupes prédéfinis, mais très généraux : *AllUsers*, *AuthenticatedUsers*.

Nommage et recherche

Comme on l'a vu, il n'y a pas de notion de répertoire, ou de « chemin d'accès ». L'arborescence n'a qu'un niveau : des *buckets*, dans lesquels il y a des objets.

Néanmoins, on peut utiliser le nommage des objets pour reproduire une sorte de chemin, simulant des répertoires, en nommant par exemple des objets

```
"/mesdocuments/images/2009/portrait-michel.jpg"

"/mesdocuments/images/2009/paysage-bretagne.jpg"

"/mesdocuments/images/2010/coucher-soleil.jpg"
```

Recherche

Il est possible de lister le contenu d'un bucket, et il est possible aussi de lister les objets d'un bucket dont le nom commence par une certain chaîne dite préfixe. Ainsi en fournissant comme préfixe « /mesdocuments/images », on obtiendra l'ensemble des images. Mais on pourra aussi demander les « common prefixes », c'est-à-dire les parties de noms qui commencent par un certain préfixe, et se terminent par un certain délimiteur, ici le « / ».

Ce sont les moyens, un peu rudimentaires, mais généralement suffisants, de parcourir les objets.

Accès http

Les objets S3 peuvent être rendus disponibles directement en http, sous réserve bien sûr qu'ils soient publics en termes de contrôle d'accès.

On peut distinguer deux cas d'utilisation :

- Dans un contexte web, les objets peuvent être directement chargés par un navigateur en tant que ressources, s'il s'agit de pages Html, images, css, javascript, flash, etc.

- Dans un contexte non-web, on peut utiliser le protocole http, qui a le mérite d'être simple et largement implémenté, pour accéder à des objets de toutes natures, objets applicatifs sérialisés en Xml ou documents bureautiques par exemple.

Un objet S3, pour autant qu'il soit public, possède une adresse de type :

```
http://s3.amazonaws.com/bucket-name/object-name
```

Mais il est possible de l'accéder aussi en utilisant un sous-domaine du nom du bucket :

```
http://bucket-name.s3.amazonaws.com/object-name
```

Le propriétaire d'un domaine, par exemple smile.fr, peut associer au niveau des DNS un nom de serveur, par exemple `www1.smile.fr` à son nom dans S3 : `bucket-name.s3.amazonaws.com`. Il suffit alors de définir un *bucket* nommé `www1.smile.fr` dans S3, pour que les requêtes adressées à `www1.smile.fr` soient automatiquement dirigées vers S3 sur `www1.smile.fr.s3.amazonaws`.

Typiquement, un site qui manipule de gros volumes d'images par exemple, tout en ayant besoin d'une forte sécurisation dans le stockage des images, pourra très facilement faire servir toutes ses images à partir de S3 de manière pratiquement transparente.

Tarification

Le prix du service S3 est fondé sur trois grandeurs :

- Le volume stocké

- Le transfert de données

- Les requêtes

Les prix sont légèrement différents selon qu'on choisit un datacenter sur la zone Europe ou bien US. La zone Europe a des prix supérieurs d'exactement 20%.

D'une manière générale, les prix sont bas, et c'est une part importante de l'attrait. En particulier, le service est souvent moins coûteux que le stockage facturé par un hébergeur ordinaire. $1.5 par mois pour stocker 10 GO de manière sécurisée, c'est un prix incroyablement compétitif.

Les requêtes proprement dites sont facturées également, mais à un prix très faible : un centime les 1000 en écriture, un centime les 10 000 en lecture. Considérons un site qui reçoit 1 million de visites par mois, soit disons 5 millions de pages vues, et donc 100 millions d'objets servis (images, css, etc.). Cela représente, pour la partie requêtes, une facture de $100 par mois.

Supposons maintenant que la page moyenne, incluant tous ses composants, fasse 200 KO, ce qui dans la bonne moyenne. On obtient un volume mensuel de transfert de 200 x 5 M = 1 TO. Au prix de $0.150 par GO transféré, on obtient un prix mensuel de $150.

Si maintenant la totalité des pages et composants du site représente 1 GO, il faudra ajouter $0.15 pour le stockage.

Au total, l'hébergement de ce site (totalement statique) présente un coût mensuel de $250. Bien sûr, c'est un prix correspondant à l'infrastructure et au réseau, n'incluant aucune exploitation.

Synthèse des prix S3 :

	USA	Europe
Stockage	$0.150	$0.180
Transfert	$0.170	$0.170
Requêtes écriture (pour 1000)	$0.010	$0.012
Requêtes lecture (pour 10 000)	$0.010	$0.012

Il y a par ailleurs quelques réductions sur des tranches importantes de volumes.

EC2

EC2 est un service de *hosting on-demand*, permettant d'allouer et d'utiliser des serveurs virtuels, en location à l'heure.

Une instance EC2

Une instance EC2 est un serveur virtuel d'une capacité donnée. Les outils de virtualisation (Amazon s'appuie sur Xen), masquent totalement les choix d'infrastructure sous-jacents, mais permettent de garantir une capacité spécifiée, en termes de CPU, de mémoire, de disque et de bande passante.

Une instance EC2 est créée en appliquant une image de machine virtuelle, ici appelée *AMI, Amazon Machine Image*, que l'on présentera plus loin. En termes de capacité matérielle, il existe différents types d'instances EC2, distinguées par leur puissance CPU et d'autres caractéristiques. Voir « pricing », plus loin.

Une instance EC2 possède un disque dur, ou du moins un espace de stockage équivalent à un disque dur. Mais ce stockage n'est pas pérenne, il a la même durée de vie que l'instance. Pour un stockage pérenne, il faut se tourner vers S3 ou bien EBS.

Location à l'heure

Dans certains contextes, la location des instances à l'heure est un détail. Il est fréquent que l'on identifie un besoin bien plus stable, à l'échelle d'au moins quelques mois. Et d'ailleurs, le pricing prévoit des réductions importantes pour une location de plus longue durée.

Malgré tout, c'est un détail qui change beaucoup de choses, qui donne tout sont sens à la notion d'élasticité dans le cloud. Le serveur est véritablement « on demand ».

Il n'est pas toujours aisé d'en tirer totalement parti. Dans des contextes de développement ou de test, c'est relativement facile: un administrateur lance et termine les instances en fonction des besoins. Dans ce contexte, il peut en particulier terminer les instances le soir ou les veilles de week-ends.

Pour une plateforme de production, il faut que l'allocation de ressources puisse être totalement automatisée, afin que la plateforme sache grandir à la demande, en fonction de l'audience.

Firewalling

Les instances EC2 bénéficient automatiquement d'un service de firewall assuré par Amazon.

Il est possible de définir deux types de règles : des règles de niveau IP et des règles de niveau groupe.

Les règles IP, comme sur un firewall ordinaire, permettent d'autoriser le trafic sur la base (a) du protocole utilisé au niveau 4 (TCP, UDP, ICMP), (b) du port destinataire, et donc d'une certaine manière du protocole de niveau 7, et enfin (c) de l'adresse IP source.

Il est possible ainsi de n'accepter que des connexions venant du réseau interne de l'entreprise, ou bien même d'un serveur spécifique.

Le firewall ne permet pas de filtrer les connexions sortantes : elles sont toutes autorisées quel que soit le protocole et la cible.

Il faut se souvenir que dans un contexte d'élasticité, les instances d'une plateforme peuvent apparaître et disparaître, et que l'on ne contrôle pas l'allocation de leur adresse IP. Il serait donc difficile de gérer les règles internes au moyen des IP.

Les règles de niveau groupe ne concernent que le trafic « interne à EC2 », c'est-à-dire entre instances EC2. Une instance peut appartenir à un ou plusieurs groupes de sécurité, et des règles peuvent être définies pour l'ensemble du groupe. Les règles de groupe sont moins fines que celles de niveau IP : elles ne distinguent pas les protocoles, port et adresses IP.

Typiquement, on dira que les serveurs d'un même utilisateur, constituant une même plateforme, appartiennent à un même groupe, et ont le droit de communiquer entre eux. Mais les groupes ne sont pas nécessairement limités aux instances d'un user, il est possible d'inclure les groupes d'autres users dans ses permissions.

Notons que l'appartenance d'une instance à un groupe est définie uniquement lors de sa création, elle ne peut pas être modifiée durant la vie de l'instance. Enfin, la notion de groupe est purement administrative, et l'on peut constituer des groupes étendus sur plusieurs datacenters.

Régions et zone de disponibilité

On a beau être « in the cloud », on ne peut faire abstraction de la géographie. La Terre n'est pas une si petite planète, et il n'est pas possible d'offrir un service de grande qualité avec des serveurs trop éloignés de ses clients.

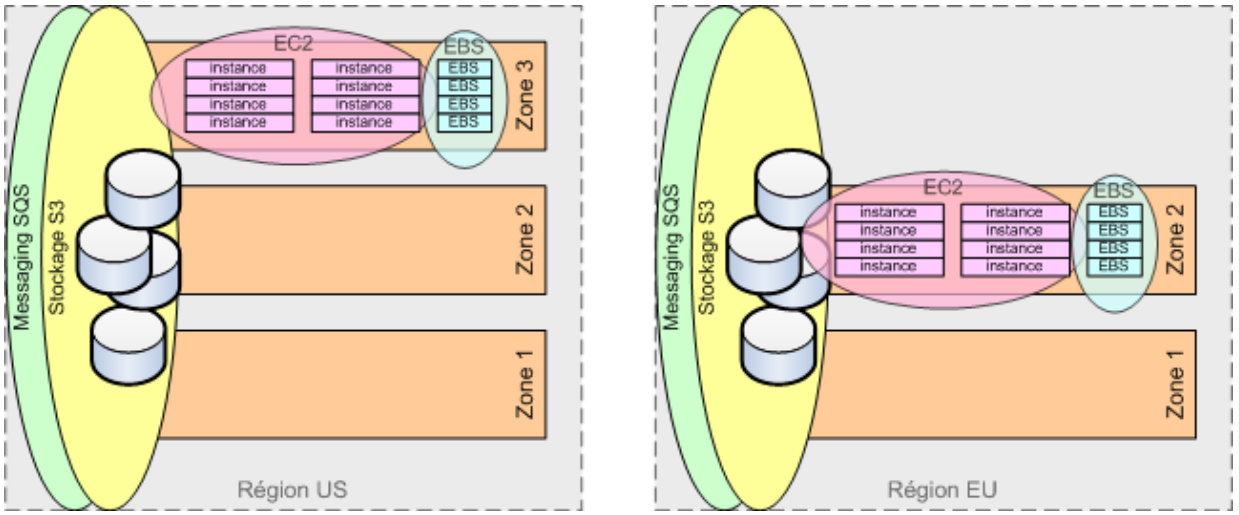
Amazon distingue deux « regions » : Etats-Unis et Europe, et l'on peut choisir dans quelle région on situe ses serveurs, et son stockage S3. La logique est très claire : il faut placer ses instances au plus près de ses utilisateurs. Et ainsi, une entreprise européenne voudra très probablement utiliser la région Europe, même si les prix sont de 20% plus élevés.

Pour un site web dont les utilisateurs seraient sur différents continents, deux voies sont possibles : soit gérer soi-même une répartition des instances entre Europe et Etats-Unis (et gérer les questions de cohérence et de synchronisation associées), soit utiliser le Content-Delivery Network CloudFront proposé également par Amazon.

Au sein d'une région, on distingue des « availability zones », que l'on peut considérer comme des datacenters distincts. Les « availability

zones » ne partagent aucun composant (routeurs, climatisation, électricité ou connexion Internet), de sorte que la probabilité est très faible que deux zones subissent une panne au même moment.

Le client maîtrise la répartition de ses instances entre les zones, il peut donc les placer dans des zones différentes pour améliorer sa qualité de service. Cela suppose toutefois de bien réfléchir aux interactions entre les instances : il ne servirait à rien de les placer dans des zones différentes si elles restent dépendantes les unes des autres de manière synchrone.



Les notions de régions et de zones ne sont pas transparentes pour les services AWS, ainsi :

- Les flux d'une région à une autre sont facturés exactement comme des flux externes : en sortie d'une région, en entrée de l'autre. Ceci s'applique pour des flux entre instances EC2, comme pour des flux entre S3 et instances EC2.
- Les flux internes à une région, en revanche ne sont pas comptabilisés.
- Le stockage S3 est, par construction, réparti entre plusieurs zones d'une même région, pour en garantir la sécurisation. En revanche, le stockage EBS est limité à une zone particulière : une partition EBS ne peut être montée que sur une instance de la même zone.

A mi-2009, il y avait trois zones dans la région US et deux zones dans la région EU.

AMI

EC2 est basé sur une virtualisation XEN, qui est totalement invisible pour le client.

La création d'une instance EC2 précise quelques paramètres, en particulier le type d'instance (puissance), la région et éventuellement la zone de disponibilité, et finalement l'image machine, ou encore « AMI », *Amazon Machine Image*.

L'AMI définit la configuration complète de la machine : son OS, et tous les composants qui peuvent constituer la pile logicielle.

L'AMI peut être choisie dans la librairie d'AMIs prêtes à l'emploi proposée par Amazon et gratuites pour la plupart. Il existe aussi des AMIs proposées par des tiers, certaines gratuites, d'autres payantes selon un système de location, c'est-à-dire que le prix à l'heure de l'AMI vient s'ajouter au prix à l'heure de l'instance, et Amazon reverse ce montant à l'auteur. Lorsqu'une AMI inclut des logiciels propriétaires, les droits d'utilisation des programmes sont payés au travers de la location de l'AMI. On peut donc trouver des AMIs avec une base Oracle, ou des logiciels Microsoft.

Une AMI peut rester privée, réservée à celui qui l'a créée, ou bien elle peut être partagée avec d'autres utilisateurs désignés de AWS, ou laissée en accès libre.

Adresse IP

Chaque instance EC2 reçoit une adresse IP de manière dynamique, valable jusqu'à sa destruction. Voir également le service d'*Elastic IP Address*, décrit plus loin.

Informations de contexte et metadonnées d'instance

Lorsqu'une instance est créée, elle ne connaît, au départ, que ce qui est inclus dans son AMI. Ce n'est pas suffisant, car l'instance a aussi un *contexte d'utilisation*. Typiquement, supposons que l'on définisse une AMI correspondant à une configuration de serveur web autonome (dédié). On souhaite utiliser cette AMI pour mettre en œuvre différents serveurs, correspondant à différents sites. On ne va donc pas placer toute l'arborescence du site, ou bien sa base de données s'il s'agit d'un CMS, dans l'AMI, car il nous faudrait une AMI par site. Si l'on optait pour une AMI par site, on pourrait se retrouver à gérer 10 AMIs différentes pour 10 sites qui ont en fait la même configuration matérielle. Cela impliquerait qu'un patch de sécurité Apache par exemple, serait à appliquer sur les 10 AMIs : on voit bien que ce n'est pas une bonne idée.

Il faut donc considérer que une AMI correspond à une configuration *logicielle*, et que toutes les données et le contexte d'utilisation sont obtenus par ailleurs, dans la phase d'initialisation de l'instance, le plus souvent à partir de S3.

Il faut donc au minimum pouvoir transmettre à une instance, lors de sa création, un paramètre qui lui fera choisir le bon *bucket* S3 pour lire ses données contextuelles.

Il est possible ainsi de fournir des « *instance data* », spécifiées lors de la création de l'instance. Ces données pourront être obtenues par l'instance en adressant une requête http à une adresse particulière (http://169.254.169.254/).

Cette même adresse est utilisée pour obtenir différentes métadonnées, « *instance metadata* », qui permettent à l'instance de mieux s'identifier elle-même, parmi lesquelles :

- Identifiant de l'AMI dont elle est issue
- Identifiant de l'instance elle-même
- Type d'instance (small, large, ...)
- Local hostname et local IP, le nom DNS privé et l'IP privée.
- Public hostname et public IP : le nom DNS public et l'IP publique.

Pricing

Synthèse des prix EC2, région Europe, en mode « on demand », à l'heure :

Type d'instance	Unités CPU	/ heure	/ mois
Small	1	$0.11	$79
Large	4	$0.44	$317
Extra-Large	8	$0.88	$634
High CPU / Medium	5	$0.22	$158
High CPU / Extra Large	20	$0.88	$634

Depuis mars 2009, Amazon a introduit un prix pour des « reserved instances », c'est-à-dire lorsqu'il y a un engagement d'un an ou de trois ans. Le prix est constitué d'un paiement initial, puis d'un prix à l'heure sensiblement plus faible. Par souci de simplicité, nous le ramenons ici à un prix au mois, et convertissons en Euros, sur la base de 1 EUR = 1,3 USD, ce qui donne :

Type d'instance	On demand	Reserved 1 an	Reserved 3 ans
Small	€61	€43	€33
Large	€244	€172	€132
Extra-Large	€488	€344	€263
High CPU / Medium	€122	€86	€65
High CPU / Extra Large	€488	€344	€263

Il faut se souvenir que pour un usage typique en serveur web, il faudra ajouter au minimum les transfers, sur la base de $0.170 par GB transféré.

Comparatif

Dans l'offre AWS la partie EC2 est celle dont le pricing est le moins agressif. En fait, le prix n'a pas évolué depuis l'introduction du service en béta, en 2006. A l'époque il n'existait qu'une configuration unique, correspondant à l'instance small d'aujourd'hui. De nouvelles instances sont apparues, plus puissantes et plus chères, mais le prix de base n'a pas changé, alors que la célèbre Loi de Moore nous dit que le prix d'achat des serveurs à puissance constante a été divisé par plus de 2 dans l'intervalle.

Une instance High-CPU extra-large correspond sensiblement un double quadcoeur Intel 2009, que l'on peut trouver en hébergement entre 100 et 250 € / mois, alors qu'il sera à 344 €/mois chez Amazon avec engagement d'un an.

C'est clairement dans l'élasticité que l'offre peut être économiquement attractive. Si une plateforme utilise un serveur pendant 60% du temps, 2 serveurs pendant 30% et 3 serveurs pendant 10%, alors le calcul est le suivant (base serveurs extra-large) :

- Serveur réservé un an: 344 €
- Deuxième serveur commandé on-demand, pendant 30% des heures (soit 2628 heures par an)148 € / mois
- Troisième serveur commandé on-demand, pendant 10% des heures (soit 876 heures par an): 50 €/an.

- Total: 542 €/an (hors trafic).

Avec un dimensionnement fixe à 3 serveurs, en hébergement 'traditionnel', on peut arriver à un prix comparable.

En revanche, si les pointes sont plus fortes encore, et peuvent requérir occasionnellement 5, 6 serveurs par exemple, alors seulement l'écart se creusera en faveur du Cloud.

D'une manière générale, on peut dire que le compromis prix / réactivité s'analyse ainsi:

- Si la réactivité requise, et l'engagement d'utilisation, sont *à l'échelle de l'heure*, alors AWS est imbattable.
- Si on est *à l'échelle du mois*, alors l'hébergement traditionnel pourra être moins coûteux – mais avec moins de services environnants.
- Si on est *à l'échelle de l'année*, alors un hébergement internalisé pourra être moins coûteux, mais avec moins de prestations environnantes.

Et parmi les paramètres du choix, il faut citer aussi la relative rigidité des configurations: certes Amazon a introduit 5 types d'instances différentes, mais il demeure que l'on ne maîtrise pas plus finement les configurations. Si vous avez besoin d'un serveur avec 50 Giga-Octets de RAM, vous ne l'aurez pas.

EBS

Elastic Block Store ou EBS est le dernier des services introduits dans le Cloud Amazon. En bref, c'est un stockage sur disque qui peut être dissocié des instances, c'est-à-dire des serveurs.

EBS n'est pas un NAS : les disques ne sont pas partagés entre différentes instances AWS. Il est plus proche d'une partition sur SAN, qui peut être montée successivement sur une instance puis une autre. Si vous voulez mettre en place un NAS, rien de plus facile, il suffit d'utiliser une instance EC2 à cet effet.

Contrairement à S3, EBS offre un véritable accès disque direct. Un disque EBS peut être monté sur un serveur et utilisé de manière transparente, à la manière d'un disque local.

Ainsi, contrairement à S3, EBS est *transparent pour les applications*.

Contrairement aux disques des instances EC2, un disque EBS est persistant : il est préservé lorsque l'instance est terminée, et pourra être monté plus tard sur une autre instance.

Les disques EBS sont gérés en mode RAID 1, et donc à l'abri de pannes simples. Pour autant, ils n'offrent pas le même niveau de sécurisation que le stockage S3. En particulier ils sont soumis à une seule « zone de disponibilité ».

Sauvegarde S3 et snapshots

La bonne pratique est donc en général de sauvegarder un disque EBS sur S3. Amazon offre pour cela un outil de snapshot incrémental.

Le principe du snapshot est que l'on prend une image, un instantané, du disque à un instant très précis. Même si la copie effective de tous les blocs peut prendre longtemps, la sauvegarde sera l'image exacte de l'état du disque à l'instant où le snapshot a été pris. Dans la pratique, cela signifie que à compter de l'instant du snapshot et jusqu'à ce que la copie soit terminée, les blocs modifiés par les applications sont en fait dupliqués : la version antérieure est conservée pour être copiée sur la sauvegarde, et la version nouvelle est utilisée par les applications.

La sauvegarde de EBS vers S3 est aussi incrémentale, c'est-à-dire qu'une seconde sauvegarde ne prendra que les blocs modifiés depuis le premier snapshot. Et ainsi de suite pour les sauvegardes ultérieures, de sorte que plus les sauvegardes sont fréquentes, plus elles sont rapides.

Lors de la restauration, on prendra la dernière version de chaque bloc, quel que soit son instant de sauvegarde. Mais on pourra aussi restaurer à l'état précis de l'un des snapshots intermédiaires.

On peut ainsi faire une sauvegarde quotidienne en ne consommant que le minimum nécessaire en termes d'espace sur S3.

A noter que si techniquement le snapshot peut être pris à n'importe quel instant, il est préférable de le prendre dans un état cohérent du disque par rapport aux applications.

Ainsi, si le disque est utilisé par une base de données, il vaut mieux qu'il n'y ait pas de transaction en cours. S'il y en avait, alors elles devraient être déjouées (rollbacked) lorsque le disque serait restauré, ce qui n'est pas impossible, mais un peu dommage. Sur une base MySql, on choisira une heure creuse pour faire un rapide « *flush tables with read lock* », suivi du snapshot, suivi du *unlock*.

Elastic IP Address

Quand une instance est créée, elle dispose automatiquement d'une adresse IP publique, stable, qui pourra être utilisée pour administrer le serveur ou invoquer ses services.

Il est possible de déclarer cette adresse IP au niveau des DNS, mais c'est assez peu flexible. Si l'instance est supprimée, puis recréée un autre jour, elle reçoit une adresse IP différente, et la propagation de cette nouvelle adresse dans les DNS peut prendre plusieurs heures. En d'autres mots, la réactivité des DNS n'est pas au même niveau que la réactivité de l'élasticité AWS.

D'où la notion d'Elastic IP address. Une Elastic IP est une adresse IP publique, routable, qui est allouée à un utilisateur AWS indépendamment de ses instances. L'utilisateur demande une Elastic IP, puis il crée une instance et affecte l'adresse à cette instance. Il peut à tout instant réaffecter l'adresse, l'associant à une autre instance. Il peut aussi terminer toutes ses instances, et l'adresse IP se retrouve alors non-affectée, mais reste malgré tout allouée à l'utilisateur.

Du fait que les Elastic IP sont pérennes par rapport aux instances, elles ont aussi un coût spécifique, afin d'encourager une utilisation responsable.

L'affectation d'une Elastic IP à une instance est une opération quasi-instantanée, du moins qui prend au plus quelques secondes. C'est donc un moyen simple et rapide de gérer un passage en secours d'un serveur vers un autre. Mais ce passage en secours n'est pas automatique, ce n'est pas un service proposé par Amazon : il incombe à chacun de surveiller ses serveurs et de gérer la bascule de ses IP.

SQS

SQS est un service de file de message, ou *message queue*, qui entre donc dans la catégorie des MOM, les *Message Oriented Middleware*, qui fournissent un moyen d'échange à la fois fiable et asynchrone entre différents serveurs, ou process sur ces serveurs.

Ce type d'outil met en œuvre un couplage lâche entre différents process : un ou plusieurs process écrivent des messages dans une queue, un ou plusieurs process lisent les messages de la queue. Les process récepteurs et émetteurs ne doivent pas se connaître en aucune manière les uns les autres, ils ne connaissent que l'identifiant de la queue.

Par rapport à des MOM plus sophistiqués, tels que Apache Active MQ par exemple, SQS est un service très limité dans ses fonctionnalités. La caractéristique essentielle est celle de fiabilité : une fois qu'un message est confié à SQS, il ne sera pas perdu.

Vie des messages

Dans une utilisation typique de SQS, les messages ne doivent pas seulement être *lus*, ils doivent être *traités*. C'est à dire que le lecteur doit effectuer certaines tâches sur la base du message reçu.

C'est pourquoi un message qui est lu n'est pas automatiquement supprimé de la file d'attente : il est conservé dans la queue, mais passe dans un état *invisible*, c'est à dire qu'il ne sera pas servi à d'autres lecteurs de la queue. Cet état *invisible* du message est d'une durée limitée, qui peut être définie. Lorsque le délai fixé est écoulé, si le message n'a pas été traité avec succès, il repassera dans un état *visible*, et pourra donc être servi lors de requêtes de lecture.

Lorsque le process qui a lu le message a fini de le traiter, il le supprime enfin de la queue (en définissant son délai d'invisibilité à 0). Si le process devait se terminer brutalement avant d'avoir traité le message, alors le message redeviendrait visible automatiquement à l'expiration du délai, et il pourrait donc être traité par un autre process lecteur.

On voit que le principe du SQS vise en premier lieu la fiabilité : rien ne doit être perdu.

Habilitations

Il est possible de définir le contrôle d'accès à une liste.

Trois permissions peuvent être définies :

- Ecriture
- Lecture
- Contrôle total.

Chacune des permissions est accordée à un utilisateur AWS, les requêtes étant toujours adressées *au nom d'un utilisateur*.

Limitations

Les messages SQS ont une taille maximale de 8 KO. Elle était de 256 KO dans la version initiale, mais a été réduite dans la version '2008' du service. Cela signifie que SQS ne doit pas être utilisé

pour transmettre des données en volume ; pour une telle utilisation, la bonne pratique serait de transmettre dans SQS l'identifiant d'un objet S3, qui serait lu, et éventuellement détruit, par le récepteur.

En revanche, il n'y a pas de limite au nombre de messages que peut contenir une queue. Mais les messages ont une durée de vie maximale de 4 jours (anciennement 15), c'est à dire qu'un message non-traité sera supprimé au bout de 4 jours.

Il y a quelques autres limitations, en particulier si on compare SQS à des MOM haut de gamme. SQS garantit qu'un message ne sera pas perdu, mais ne garantit pas grand chose de plus, en particulier :

- Il est possible dans certains cas qu'une requête de lecture d'une queue ne retourne rien alors qu'il y a des messages dans la queue !
- Il n'y a pas de garantie de délivrance dans l'ordre, c'est à dire que la queue SQS ne se comporte pas toujours en FIFO (*first-in, first-out*). En fait c'est le cas presque toujours, mais ce n'est pas garanti.
- Il n'y a pas de garantie qu'un message ne sera pas délivré plusieurs fois (indépendamment même de la gestion de sa visiblité, vue plus haut), même si dans la pratique c'est très improbable.
- Enfin, il n'y a pas de garantie en termes de délai d'acheminement, même si dans la pratique il est de quelques secondes.

Toutes ces limitations sont dues au caractère distribué de l'ensemble de l'implémentation, qui privilégie la fiabilité et l'extensibilité, sur les autres caractéristiques.

Pricing

Le prix est de $0,01 pour 10 000 requêtes, les requêtes comprenant toutes les opérations possibles, y compris création et suppression de queues. Mais bien sûr, les requêtes les plus structurantes pour le prix sont les écritures et les lecteurs.

Le trafic est gratuit au sein d'une même région (US ou EU), entre une instance EC2 et le service SQS.

Recommandations d'utilisation

Comme nous l'avons dit en préambule, une recommandation d'ordre général est de réfléchir aux implications avant de construire de grandes applications qui s'appuient structurellement et lourdement sur ce type de service. La fiabilité est extrême et le coût est faible, mais il faut bien mesurer le degré de dépendance que l'on est prêt à accepter.

Ainsi par exemple au 1er janvier 2008, une nouvelle version de SQS a été introduite, qui modifie certaines caractéristiques essentielles du service. La taille maximale des messages passe de 256 K à 8 K, et la durée de vie de 15 jours à 4, et quelques autres changements. La validité de l'ancienne version est maintenue jusqu'en mai 2009, soit 17 mois. Celui qui aurait construit une application d'envergure qui aurait un besoin impérieux de messages de 256 KO, serait dans l'obligation incontournable de refondre son application en profondeur, dans le délai de 17 mois. On voit que, malgré l'excellence du service, la dépendance a un prix.

CloudFront

CloudFront est le dernier né des services AWS. C'est un service de type *Content Delivery Network*, qui vise à servir des contenus au plus près de l'utilisateur final, l'internaute. C'est-à-dire à servir les contenus depuis un datacenter situé au Japon lorsque l'internaute est lui-même au Japon.

On se reportera au livre blanc Smile traitant des *« Architectures Web Hautes-Performances »*, pour plus d'information sur les CDN en général, dont le précurseur et la figure emblématique est Akamai.

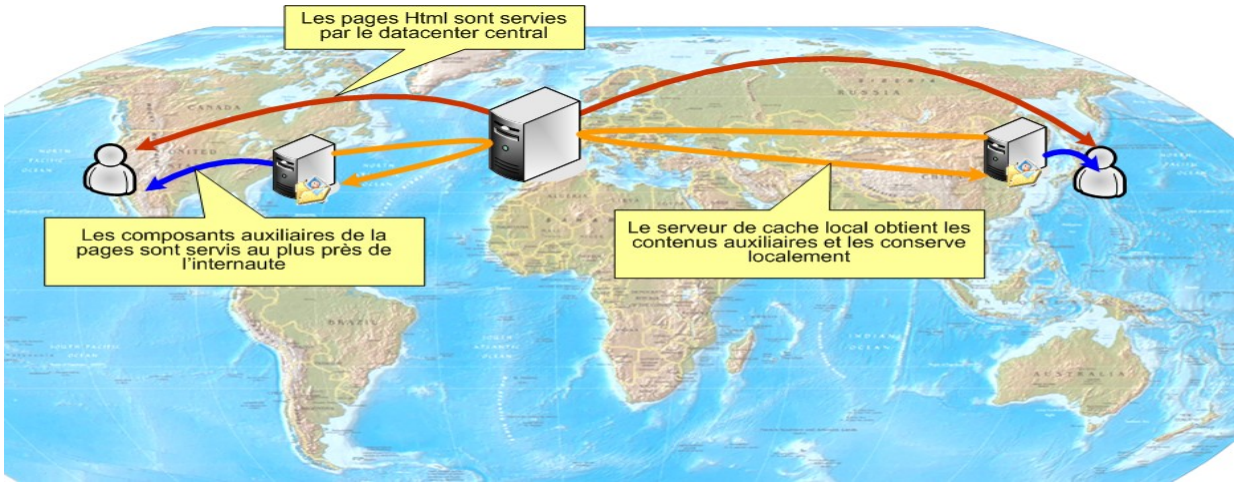
Tous ceux qui gèrent un site web avec des utilisateurs en Asie savent que le temps de latence Europe Asie (150 ms aller-retour) dégrade les performances de manière spectaculaire. Pour de tels services, le recours à un CDN est quasiment obligatoire.

CloudFront se distingue par sa facilité de mise en œuvre et son coût réduit, alors que Akamai et ses émules sont relativement coûteux.

Le principe d'utilisation est le suivant :

- Le client dépose ses fichiers de ressources (images, css, javascript, videos, etc.) dans un bucket S3, en accès public.
- Ces ressources ont une URL par défaut, mais il est possible de leur associer une URL sur le nom de domaine du client.

- Lorsque la ressource est demandée, la requête est dirigée vers le datacenter le plus proche du client, d'où elle est servie.



On notera toutefois qu'il ne fonctionne pas en mode cache, à la manière par exemple de Akamai. Dans Cloudfront, les objets doivent en premier lieu être placés sur S3, ce qui ne permet pas de l'utiliser à partir d'applications qui ne seraient pas conçues pour fonctionner avec S3.

Cloudfront fait appel à des datacenters situés :

- Aux Etats-Unis
- En Europe
- Au Japon
- A Honk-kong.

Le service est uniquement facturé à la bande passante utilisée, avec un prix sur US et EU, égal à celui des transferts S3 et EC2, soit $0.170 par GO transférés. Il est un peu plus coûteux depuis l'Asie.

CloudFront est encore en béta, mais étant donné que c'est un service relativement simple, il est probable qu'il sera à maturité très vite.

SimpleDB

SimpleDB est un service de gestion de données extrêmement rudimentaire. Il n'est pas comparable à une base de données relationnelle telle que MySql, et il n'est donc pas imaginable d'appuyer un progiciel standard sur SimpleDB, on peut uniquement écrire des applications spécifiques utilisant SimpleDB.

Fondamentalement, nous ne pensons pas qu'il soit prudent d'utiliser ce service dans la sphère des applications d'entreprise, et c'est pourquoi nous n'en dirons pas davantage ici.

Amazon Cloud Watch

Amazon Cloud Watch ou ACW est la réponse d'Amazon aux problématiques de monitoring. Encore en version bêta, il permet de surveiller précisément l'état de son cloud à l'aide de Webservices.

Les seuls éléments fournissant des statiques sont les instances EC2 et les Load Balancer (LB).

Contrairement à des solutions de monitoring « maisons », celle-ci ne requiert aucune installation. Les statistiques sont fournies par le système de gestion du cloud d'Amazon. Quel que soit l'état de la machine, il suffit d'activer le monitoring d'une instance et le tour est joué.

La visibilité et la facilité d'utilisation font d'AWC un outil incontournable. Un des seuls regrets est l'impossibilité de mettre en place des alertes.

A ce jour, ce service n'est pas disponible sur les datacenters Européens, mais ça ne devrait pas tarder.

ACW introduit ses terminologies et concepts que nous allons étudier.

Dimensions

Les *dimensions* représentent un regroupement de grandes caractéristiques de la plateforme. Chaque *dimension* englobe des *métriques*. Cela permet de filtrer les métriques par dimensions. Les métriques, comme nous le verrons plus loin, sont les représentations des mesures récupérées du Cloud.

Les dimensions des métriques d'instances EC2 sont :

Dimension	Description
ImageId	Dimension filtrant par rapport à l'identifiant d'une image.
AutoScalingGroupName	Dimension filtrant par rapport à un nom de groupe d'auto répartition (Auto Scaling Group). Un groupe d'auto Scaling est une collection d'instances définie par l'utilisateur utilisée par le service d'Auto Balancing.

InstanceId	Dimension filtrant par rapport à l'identifiant d'une instance.
InstanceType	Dimension filtrant par rapport au type d'une instance. Les types d'instance sont, par exemple, m1.small ou encore m1.large.

Les dimensions des métriques de Load Balancer sont :

Dimension	**Description**
LoadBalancerName	Dimension filtrant par rapport au nom d'un Load Balancer
AvailabilityZone	Dimension filtrant par rapport à la Zone de disponibilité des instances EC2.

Lors d'une recherche, plusieurs dimensions peuvent être combinées en même temps. Un cas d'utilisation typique est l'obtention des métriques d'une instance particulière. Dans ce cas, nous filtrons les métriques par la dimension ImageId.

Mesures

Les mesures représentent les données à l'état brut des machines surveillées. Elles disposent toutes :

1) d'un nom unique
2) d'un ensemble de dimensions
3) d'un namespace
4) d'une unité
5) d'un timestamp

Le mode Push étant de mise, ce sont les instances et les LB qui poussent leurs statistiques dans le système de surveillance. Elles peuvent ainsi être récupérées à n'importe quel moment et dans n'importe quel ordre. Cette caractéristique explique une mise en cache d'une minute avant leur publication.

Métriques

Une *métrique* est l'agrégation de données brutes. Elle est publiée sur ACW avec une unité de mesure. C'est l'attribut fondamental de stockage des données agrégées.

Une métrique répond aux caractéristiques suivantes :

6) Elle possède un nom de mesure

7) Elle peut avoir plusieurs dimensions

8) Elle possède un seul namespace

Les métriques des instances EC2 sont les suivantes :

Nom	Description
CPUUtilization	Utilisation du CPU (en %)
NetworkIn	Activité réseau en réception (en bits)
NetworkOut	Activité réseau en envoie (en bits)
DiskWriteOps	Activité disque en écriture (en nombre d'opération)
DiskReadBytes	Activité disque en lecture (en bits)
DiskReadOps	Activité disque en lecture (en nombre d'opération)
DiskWriteBytes	Activité disque en écriture (en bits)

Les métriques des LB sont les suivantes :

Nom	Description
Latency	Durée entre une requête et sa réponse vue par le Load Balancer en secondes
RequestCount	Nombre de requêtes par seconde
HealthyHostCount	Nombre d'instances aptes à recevoir des requêtes
UnHealthyHostCount	Nombre d'instances non aptes à recevoir des requêtes

Namespace

Les *namespace* donnent la possibilité de catégoriser les flux d'information disponibles. Cela permet l'isolation des mesures.

Tout nom de *namespace* commence par « AWS/ ».

Par exemple, une entreprise hébergeant plusieurs plateformes web sur le Cloud Amazon souhaitera séparer les statiques issues de chacune d'elles.

Statistique

Les statiques représentent l'attribut principal d'une métrique. Elles permettent de récupérer la distribution statistique de celle-ci.

Actuellement, il existe 4 types de statistiques :

- Maximum
- Minimum
- Somme
- Moyenne

Du fait du cache des mesures, la période minimale prise en compte est la minute.

Chaque mesure reçue par AWC est exprimée selon une unité.

Pricing

Le coût est de 0.015 dollars par heure et par élément surveillé.

A titre d'exemple, si l'on surveille dix instances EC2 en 24/7 et pour une durée de 30 jours, le coût engendré est de 108 dollars : 0.015 * 10 * 24 * 30.

Elastic Load Balancing

Amazon fournit un service de « *Load Balancing* » ou ELB. Encore en version bêta, il permet de distribuer les requêtes vers une machine ayant la possibilité de répondre.

Le « *Load Balancer* » ou LB, est intégré à l'infrastructure Amazon, supposée être stable, performante et fiable. Néanmoins, Amazon n'en donne aucune information précise.

Le *LB* est *élastique,* c'est à dire que l'on peut programmer l'ajout et la suppression d'instances en fonction du besoin.

Les bénéfices d'un tel service sont entre autres :

- La réparation de la charge
- La gestion des pannes

En effet, en gardant à jour une liste de machines opérationnelles, si une des instances venait à mourir spontanément, le LB s'en apercevra et arrêtera de lui envoyer des requêtes. Dans cette

optique, l'utilisateur doit paramétrer les conditions pour qu'une machine soit considérée comme « en bonne santé » ou « *healthy* ».

A ce jour, ce service n'est pas disponible sur les Datacenters Européens.

On se reportera au livre blanc Smile traitant des *« Architectures Web Hautes-Performances »*, pour plus d'information sur les architectures et mécanismes de répartition de charge.

Les paramètres du LB

Un LB agit comme une instance EC2. Il dispose ainsi d'une adresse IP et d'un DNS. Évidemment, la personnalisation de ce nom de domaine est possible.

Contrairement à des instances EC2, tout LB est identifié par un nom unique que l'utilisateur lui attribue.

Tout au long de la vie d'un LB, les trois propriétés suivantes ne peuvent pas être modifiées :

Nom	Description
LoadBalancerPort	Le port TCP sur lequel le LB écoute
InstancePort	Le port TCP sur lequel les instances reliées écoutent
Protocol	Le protocole de routage : TCP ou HTTP

Comme nous le verrons plus loin, chaque LB dispose d'une liste de « *Availability Zone* » ou AZ sur lesquelles il opère. Ces zones doivent être dans la même région que le LB.

Vérifier l'état des instances

Deux méthodes de vérification sont disponibles : par TCP ou par HTTP.

Par TCP, le LB essayera d'ouvrir une connexion vers un port. L'instance est « *healthy* » ou en bonne santé si tout se passe bien dans le temps imparti, sinon elle est « *unhealthy* ».

Par HTTP, le LB essayera de demander une page Web. Si tout se passe bien, c'est-à-dire, après l'obtention d'une réponse de code 200 dans le temps imparti, l'instance est « *healthy* » sinon elle est « *unhealthy* ».

Cependant, une instance est considérée et affichée « *unhealthy* » uniquement après un certain nombre d'échec de la vérification. Ce nombre représente le *Unhealthythreshold*.

Il en ait de même avec le fait qu'une instance soit considérée et affichée « *healthy* » avec « *Healthythreshold* ».

Les vérifications peuvent être réalisées au minimum toutes les 5 secondes et au maximum toutes les 600 secondes.

Le *timeout*, représentant le temps maximum d'une vérification, peut aller de 2 à 60 secondes.

Répartition de charge par zones

Un LB distribue le trafic équitablement à travers toutes les zones configurées. C'est-à-dire, que le trafic sera réparti d'une façon indépendante du nombre d'instances de chaque zone.

Notons qu'il faut impérativement avoir le même nombre de machines dans chaque zone.

Quoique bientôt débridée, la limitation la plus importante est celle de l'impossibilité de faire de la répartition de charge à travers des régions, par exemple entre Europe et États-Unis.

Pricing

Le pricing suit celui d'EC2. L'utilisation d'un LB est facturée à 0.025 dollars l'heure accompagnée de 0.008 dollars par Giga Octets transférés à travers celui-ci. Rappelons que le transfert de données entres services Amazon d'une même région n'est pas facturé. Ainsi, la bande passante consommée par les instances suite aux requêtes d'un LB n'est pas facturée.

A titre d'exemple, un site internet disposant d'une dizaine de frontaux utilise le serve ELB. Il finit le mois avec un transfert de 100 Go. En plus de la facture des instances (et de leurs frais annexes), il sera facturé 18 dollars : 0.025 * 24 * 30 + 0.008 * 100.

Auto Scaling

« *Auto Scaling* » ou AS est le service Amazon mettant en relation ACW et ALB. Encore en version bêta, il permet de dimensionner automatiquement un parc d'instances EC2 selon des conditions issues des métriques ACW. De plus, tout ceci peut s'imbriquer à ALB.

Ses valeurs ajoutées sont les suivantes :

- Obtenir une plateforme élastique : Ce service ajoute et supprime automatiquement des instances selon le besoin, et donc s'adapter à une pointe de charge.
- Économiser de l'argent : Ce service permet de supprimer automatiquement des instances lorsqu'on en a plus besoin.
- Obtenir une plateforme facilement maintenable : Ce service remplace automatiquement les instances mortes ou « *unhealthy* ». Il permet de garder un nombre minimal d'instances actives.

A ce jour, ce service n'est pas disponible sur les Datacenters Européens.

Scaling Group ou groupe de dimensionnement

Vu de haut, un *Scaling Group* ou SG représente *une application* tournant sur plusieurs instances.

Identifié par un nom, il opère sur une seule Availability Zone (zone de disponibilité)

Un SG peut être lié à un LB identifié aussi par son nom.

Le démarrage d'une instance peut être paramétré. Dans la suite, nous regarderons ce point plus en détail.

Les autres valeurs numériques paramétrant un SG sont :

Nom	Description
Cooldown	Durée en seconde de la pause après ajout ou suppression d'une instance
MinSize	Taille minimale du groupe
MaxSize	Taille maximale du groupe. Un SG peut contenir un maximum de 10 000 instances.

Triggers

Identifié par un nom, le « *trigger* » *(déclencheur)* représente le mécanisme par lequel un AS détermine s'il faut rajouter ou supprimer une instance. Un *trigger* est directement lié à un SG.

Il reprend les attributs définis par ACW et en y rajoutant des bornes.

Par exemple, pour une application Web utilisant le Framework « Magento », nous ne voulons ajouter une instance de frontal que si

la charge moyenne sur 5 minutes du parc de frontaux dépasse 60 %. Nous voulons, et ceci en même temps, ajouter une instance si la charge moyenne du parc de frontaux sur 1 minute est supérieur à 80 %. Quant à la suppression, nous ne souhaitons en supprimer que si la charge moyenne du parc de frontaux est inférieure à 20 %. Ceci est possible avec les couples AS, ACW et ALB de Amazon. Nous obtenons ainsi une plateforme réagissant très rapidement à un pic de charge et ceci tout en revenant à son état de stabilité.

Configuration de démarrage

Une configuration de démarrage est nécessaire pour qu'AS puisse connaitre les paramètres concernant la composition du parc.

Ces paramètres se révèlent être un atout car ils permettent vraiment de personnaliser chaque SG.

Imaginons que nous ayons à disposition une image AMI générique représentant une application Web utilisant Jahia. Selon les besoins, sa configuration s'effectue lors du démarrage pour s'adapter aux exigences. Ainsi, nous avons la possibilité d'utiliser la même image dans différents AG. Il suffira juste de créer une configuration de démarrage par groupe.

Pricing

Ce service est gratuit. Il requière, au minimum, l'utilisation d'ACW qui lui est facturé.

Conclusion

L'offre AWS est accompagnée d'une documentation de qualité, et il ne sera pas difficile pour le lecteur d'en savoir davantage. Notre but ici était de présenter une courte synthèse de cette offre, ses principes et ses possibilités.

Comme on l'a vu, il ne faut pas mélanger les offres Saas et les offres de type cloud, quoique les unes comme les autres sont amenées à prendre des parts de marché.

Une réticence forte des entreprises vis à vis du Saas est liée au risque d'une trop forte dépendance, de la perte de contrôle tant de ses programmes que de ses données.

En matière de Cloud computing, des risques semblables existent, même s'ils sont moindres. Il nous semble que l'offre AWS de Amazon est celle qui combine au mieux:

- Une totale maîtrise de ses configurations logicielles
- Une liberté et une indépendance préservées
- Une grande maturité
- Des prix attractifs, en particulier en termes de bande passante.

Smile a construit, depuis près de deux ans – alors même que l'offre était en partie en bêta – une expertise complète sur les services AWS. Et Smile a mis au point des solutions permettant de construire des plateformes qui soient à la fois:

- Élastiques, c'est à dire allouant des serveurs à la demande.
- Auto-correctrices (self-healing), c'est à dire remplaçant automatiquement des serveurs défaillants.
- Planifiables, c'est à dire qui peuvent allouer des ressources supplémentaires par anticipation, de manière programmée.

Bien entendu, ces solutions impliquent en premier lieu de s'appuyer sur des architectures extensibles.

Nous espérons que ce petit ouvrage vous aura donner envie de profiter de ces offres, et nous sommes à votre disposition pour vous y aider.

Les livres blancs Smile

Les livres blancs Smile sont téléchargeables gratuitement sur www.smile.fr

- **Introduction à l'open source et au Logiciel Libre**

  Son histoire, sa philosophie, ses grandes figures, son marché, ses modèles économiques, ses modèles de support et modèles de développement. [52 pages]

- **Gestion de contenus : les solutions open source**

  Dans la gestion de contenus, les meilleures solutions sont open source. Du simple site à la solution entreprise, découvrez l'offre des CMS open source. [58 pages]

- **Portails : les solutions open source**

  Pour les portails aussi, l'open source est riche en solutions solides et complètes. Après les CMS, Smile vous propose une étude complète des meilleures solutions portails. [50 pages]

- **200 questions pour choisir un CMS**

  Toutes les questions qu'il faut se poser pour choisir l'outil de gestion de contenu répondra le mieux à vos besoins. [46 pages]

- **Conception d'applications web**

  Synthèse des bonnes pratiques pour l'utilisabilité et l'efficacité des applications métier construites en technologie web. [61 pages]

- **Les frameworks PHP**

  Une présentation complète des frameworks et composants qui permettent de réduire les temps de développement des applications, tout en améliorant leur qualité. [77 pages]

- **Les 100 bonnes pratiques du web**

  Cent et quelques « bonnes pratiques du web », usages et astuces, incontournables ou tout simplement utiles et qui vous aideront à construire un site de qualité. [26 pages]

- **ERP/PGI: les solutions open source**

  Des solutions open source en matière d'ERP sont tout à fait matures et gagnent des parts de marché dans les entreprises, apportant flexibilité et coûts réduits. [121 pages]

- **GED : les solutions open source**

  Les vraies solutions de GED sont des outils tout à fait spécifiques ; l'open source représente une alternative solide, une large couverture fonctionnelle et une forte dynamique. [77 pages]

- **Référencement : ce qu'il faut savoir**

  Grâce à ce livre blanc, découvrez comment optimiser la "référençabilité" et le positionnement de votre site lors de sa conception. [45 pages]

- **Décisionnel : les solutions open source**

  Découvrez les meilleurs outils et suites de la business intelligence open source. [78 pages]

- **Collection Système et Infrastructure :**
  - Virtualisation open source [41 pages]
  - Architectures Web open source [177 pages]
  - Firewalls open source [58 pages]
  - VPN open source [31 pages]
  - Cloud Computing [42 pages]
  - Middleware [91 pages]

Smile
OPEN SOURCE SOLUTIONS

Contactez-nous, nous serons heureux de vous présenter nos réalisations de manière plus approfondie !
+33 1 41 40 11 00 / sdc@smile.fr