Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire

Université Abou Bakr Belkaid– Tlemcen

Faculté des Sciences

Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: *Réseaux et Systèmes Distribués (R.S.D)*

Thème

Tolérances aux Pannes dans les infrastructures Cloud Computing

Réalisé par **:**

Mme Mezouer Leila ***et*** ***Mme Mekkioui Nadjet***

Présenté le 22 Juin 2016 devant le jury d'examination composé de :

M. Benammar Abdelkrim	*Université Abou Bekr Belkaid, Tlemcen*	*Président*
Mme Hamza Cherif Souad	*Université Abou Bekr Belkaid, Tlemcen*	*Encadreur*
Mme Malti Djawida	*Université Abou Bekr Belkaid, Tlemcen*	*Examinatrice*
M. Maatallah Houcine	*Université Abou Bekr Belkaid, Tlemcen*	*Examinateur*

Année universitaire : 2015-2016

Remerciements

Nous tenons à remercier Allah pour sa grâce et sa bénédiction.

Nous tenons tout d'abord à exprimer nos chaleureux remerciements à Mme Souad Hamza-Cherif pour son temps qu'elle a partagé avec nous et ses conseils durant la période de notre projet.

Aussi nous exprimons nos très sincères remerciements à Madame I Hameri qui nous a soutenus.

Nous adressons nos vifs remerciements à M.A.Chikh le doyen de la faculté de technologie pour son aide à la réalisation de ce projet et M.A. Djebbari pour son soutien et ces conceils.

Nous adressons nos remerciements à tous le personnel du centre de calcul de Faculté de technologie.

Nous tenons également à remercier M.A. Benammar d'avoir accepté de présider le jury d'examination de ce mémoire. A Mme D.Malti et à M. H.Maatallah

Enfin, nous remercions toutes les personnes qui nous ont aidés de loin ou de près à réaliser ce travail. Merci à tous.

Mme Mekkioui & Mme Mezouer

Dédicaces

Je dédie ce travail à :

Mes très chers parents qui m'ont beaucoup aidé et soutenus et je leur dis un grand Merci

Je tiens à exprimer ma profonde reconnaissance et tout mon respect à mon cher époux pour son soutien quotidien indéfectible et son enthousiasme contagieux, pour sa patience et sa compréhension dont il a fait preuve à mon égard

A mes enfants Mohamed Wail,Mustapha et mon petit bébé Ramzi.

A mes sœurs Lamia et son mari Abdelghani

A Leila et son mari Ismail

A ma petite sœur Samia

A mes belles sœurs et mes beaux-frères et leurs enfants. Et à toute ma famille. Merci à vous tous.

Mme Mekkioui Nadjet

Dédicaces

Je dédie ce travail à :

A ma mère, qui a œuvré pour ma réussite, par son amour, son soutien, tous les sacrifices et sa présence dans ma vie, reçois à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude

Mes dédicaces vont aussi à mon défunt père

A mon cher époux qui a toujours était à mes cotés

A mes frères, ma sœur ainsi que leurs famille

A ma belle-mère et à toute ma famille et belle famille

A tous mes amis

Mme Bouziani Leila née Mezouer

Sommaire

Liste des figures **8**

Liste des tableaux **9**

Liste des abréviations **10**

Introduction générale **11**

Chapitre I– Etat de l'art sur le Cloud Computing **14**

I.1 Introduction 14

I.2 Définition des concepts 14

I.2.1 Cloud Computing 14

I.2.2 La virtualisation 16

I.3 Comparaison avant et après l'apparition du Cloud Computing 17

I.4 Les cinq caractéristiques fondamentaux du Cloud Computing 20

I.5 Les services du Cloud Computing 21

I.5.1 IaaS 22

I.5.2 PaaS 22

I.5.3 SaaS 22

I.6 Modèle de déploiement des services de Cloud Computing 22

I.6.1 Cloud communautaire (Community Cloud) 23

I.6.2 Cloud privé (Private Cloud Computing) 23

I.6.3 Cloud publique (Public Cloud Computing) 23

I.6.4 Cloud hybride (Hybrid Cloud) 23

I.7 Conclusion 24

Chapitre II – Approche de tolérances aux pannes **25**

II.1 Introduction 25

II.2 Définitions des concepts : 25
II.3 Classification des pannes 26
II.4 La sûreté de fonctionnement 28
II.4.2 Entraves 30
II.4.3 Moyens 30
II.5 Tolérance aux pannes 31
II.5.1 Procédure générale de tolérance aux pannes 31
II.6.Politique de tolérances aux pannes : 33
II.6.1 Politique de tolérance aux pannes proactive : 33
II.6.2. Politique de tolérance aux pannes réactive : 34
II.7 Techniques de tolérance aux pannes 34
II.7.1 Prévention des pannes 34
II.7.2 Masquage des pannes 34
II.7.3 Recouvrement après panne 35
II.8 Travaux connexes 35
II.9 CONCLUSION 36
Chapitre III –Tolérance aux pannes dans le Cloud Computing 38
III.1 Introduction 38
III.2 Problématique 38
III.3 Tolérance aux pannes dans le Cloud Computing par MapReduce 39
III.3.1 MapReduce 39
III.3.2 Tolérance aux pannes grâce au Framework Hadoop MapReduce 40
III.4 Méthodologie et implémentation 41
III.4.1 Phase d'installation et configuration de l'environnement du travail 42
III.4.2 Phase tests : Injection et déroulement de cas de pannes possible dans le Cloud .. 47

III.5 Observations et résultats obtenus 50

III.5.1 Résultats et interprétation 51

III.6 Conclusion 53

Conclusion générale et perspectives 54

Références bibliographique 56

Liste des figures

Figure I.1 : Architecture du Cloud Computing 16
Figure I.2 : Représentation avec et sans la virtualisation 17
Figure I.3 : Comparaison avant et après l'apparition du Cloud Computing 17
Figure I- 4 : les services de Cloud Computing 21
Figure II- 1: Classification des pannes 26
Figure II- 2 : L'arbre de la sureté de fonctionnement 29
Figure II- 3: La chaine fondamentale des entraves à la sureté de fonctionnement 30
Figure II- 4 : Procédure générale de tolérance aux pannes 31
Figure III. 1 : Schéma de l'approche 41
Figure III. 2: Vérification du lancement d'un cluster 43
Figure III. 3 : Page web de la HA du NameNode 47
Figure III. 4: Vérification de lancement du daemons Hadoop 47
Figure III. 5: Résultats de traitement représentés selon une échelle linéaire 51
Figure III. 6: Résultats de traitement représentés selon une échelle logarithmique 51
Figure III. 7: Résultats de traitement représentés selon une échelle logarithmique en axes des abscisses et des ordonnées Temps de traitement réduits pointés par des flèches 53

Liste des tableaux

Tableau I.1 : Comparaison avant et après l'apparition du Cloud Computing.....20

Tableau III. 1 : Tableau de paramétres..... 49

Tableau III. 2 : Expérimentation et résultats……50

Liste des abréviations

HA	High Availibility»,
HDFS	Hadoop Distributed File System
IaaS	Infrastructure as a Service
NIST	National Institute of Standards and Technology
PaaS	Platform as a Service
SaaS	Software as a Service
VM	Virtuelles Machines
ZKFC	ZK Failover Controller

Introduction générale

Les technologies de l'informatique évoluent et révolutionnent nos modes de vie et de travail, Le Cloud Computing ou l'informatique dans le nuage, est apparu ces dernières années comme un nouveau modèle de gestion et d'utilisation des systèmes informatiques. Il est vraiment en train de changer la manière, comment et où le calcul va être effectué. Le concept consiste à accéder à des serveurs distants pour effectuer des opérations de traitements et de stockage.

Le principe de base du Cloud Computing est que les données de l'utilisateur ne sont pas stockées localement mais sont stockées dans le centre de données d'internet (des machines virtuelles). Les entreprises qui fournissent des services du Cloud Computing pourrait gérer et maintenir le fonctionnement de ces centres de données tout est dans la virtualisation. Les utilisateurs peuvent accéder aux données stockées à tout moment en utilisant l'interface de programmation d'application (API). Non seulement ils bénéficient de services de stockage mais aussi les services matériels et logiciels qui sont disponibles pour le grand public et les marchés d'affaires. Les services fournis par les prestataires de service peuvent être dans tous les logiciels, les ressources de l'infrastructure et les plateformes. Chaque service est appelé respectivement Infrastructure as a Service (IaaS), Platform as a Service (PaaS) ou Software as a Service (SaaS). Due à la croissance exponentielle rapide du Cloud Computing la

nécessité de la tolérance aux pannes dans le Cloud est un facteur clé de préoccupation. Pour attribuer des services aux utilisateurs le Cloud Computing doit être fiable et doit gérer les pannes pour donner des résultats et un rendement dans une durée du temps minimal.

L'utilisation d'un grand nombre de ressources de calcul hétérogènes augmente fortement le risque d'apparition des pannes au cours de l'exécution d'une application distribuée. Pour cela la tolérance aux pannes est le grand défi dans un système distribué. La tolérance aux pannes est la capacité d'un système de continuer à fonctionner, même s'il y a une faute, elle est une préoccupation majeure pour garantir la disponibilité et la fiabilité des services essentiels ainsi que l'exécution des applications. Il existe plusieurs types de pannes qui sont classées selon la gravité des défaillances et selon le degré de la permanence. La tolérance aux pannes proprement dite n'est en réalité qu'une partie du concept de sûreté de fonctionnement, où est la propriété qui permet à ses utilisateurs de placer une confiance justifiée dans le service qui leur est délivré.

La tolérance aux pannes est une préoccupation majeure dans le Cloud Computing, divers politiques et techniques tolérantes aux pannes y sont implémentées. Ces techniques sont basées sur la réplication et la redondance. Beaucoup de recherche sont en cours afin d'aboutir à une tolérance satisfaisante.

Ce mémoire présente une étude sur la tolérance aux pannes dans le Cloud Computing en utilisant le Framework MapReduce, qui gère et stocke les données volumineuses (Big Data) et grâce à sa scalabilité et sa tolérance aux pannes il est un moyen très efficace pour le traitement des données dans le Cloud. Il existe des travaux qui ont été déjà fait dans ce contexte.

Notre méthode se base sur un cluster composée de 03 nœuds esclave et 02 maitres basés sur la HA et nous avons testé son influence sur MapReduce en cas de panne du maitre primaire. Pour réaliser cela nous avons fait des expériences en variant des paramètres et en provocant des pannes.

Ce mémoire est organisé en Trois chapitres. Le chapitre 1 intitulé « Etat de l'art sur le Cloud Computing » introduit le Cloud Computing et la virtualisation et présente les concepts, les services, les caractéristiques, et le modèle de déploiement des services de Cloud Computing. Le chapitre 2 intitulé « Approche de tolérances aux pannes » fournit une vue d'ensemble des

différents concepts et problèmes associés aux notions fondamentales de la tolérance aux pannes dans les environnements du Cloud Computing : définition de la panne, types de panne et leur classifications, la sureté de fonctionnement. Nous présenterons aussi dans ce même chapitre les techniques, les approches et les politiques de tolérances aux pannes. Et nous exposerons quelques travaux à la tolérance aux pannes dans le Cloud Computing trouvés dans la littérature.

Enfin dans le dernier chapitre nous présenterons l'approche proposée dans ce mémoire qui se base sur le Framework MapReduce afin de rendre le Cloud plus tolérant aux pannes, dans ce même chapitre nous parlerons des étapes suivies pour l'expérimentation de l'approche et nous discuterons des résultats obtenus.

Chapitre I– Etat de l'art sur le Cloud Computing

I.1 Introduction

Le Cloud est un concept récent dans le domaine informatique, il a connu un grand succès de nos jours non seulement dans les environnements académiques et industriels mais aussi par la concurrence entre les grands prestataires du Cloud dans le monde. L'utilisation potentielle des ressources augmente de plus en plus. C'est un obstacle pour l'évolution des entreprises, organismes…Pour remédier à ce problème, les entreprises se sont orientées vers des fournisseurs possédant des infrastructures énormes reposant sur des technologies de virtualisation et d'automatisation .Ce qui permet l'apparition de la notion du Cloud Computing. Dans ce chapitre nous allons étudier le Cloud Computing en précisant son architecture et ses services.

I.2 Définition des concepts

I.2.1 Cloud Computing

Il existe plusieurs définitions du Cloud Computing, ce paradigme n'est pas encore normalisé, chacun peut le définir selon son point de vue. La plupart des experts définissent le Cloud Computing comme étant un modèle incluant la notion de services disponibles à la demande, plus facilement extensible, virtuels et illimités ne dépendant pas de l'infrastructure physique.

Le NIST (National Institute of Standards and Technology),définit le Cloud comme un modèle qui permet un accès omniprésent, pratique à la demande à un réseau partagé et à un ensemble de ressources informatiques configurables, à titre d'exemple des réseaux, des serveurs, du stockage, des applications et des services, qui peuvent être provisionnées et libérées avec un minimum d'administration. **[1]**

Ce modèle est composé de :

• Cinq caractéristiques essentielles : Libre-service à la demande, accès au réseau large, la mise en commun des ressources, élasticité rapide, service mesuré.

• Trois modèles de services : (SaaS), (PaaS), (IaaS).

• Quatre modèles de déploiement (Cloud privé, Cloud communautaire, Cloud public, Cloud hybride.

Selon Génération NT « Le Cloud Computing est un concept d'organisation informatique qui place l'Internet au cœur de l'activité des entreprises, il permet d'utiliser des ressources matérielles distantes pour créer des services accessibles en ligne ».**[1]**

Selon CISCO Le Cloud Computing est une plateforme de mutualisation informatique fournissant aux entreprises des services à la demande avec l'illusion d'une infinité des ressources.

Le Cloud Computing ou l'informatique dans les nuages signifie l'utilisation des ressources qui n'existe pas dans le Data Center local de l'organisme ou l'entreprise… mais qui sont hébergées dans le monde entier et accessible via le web. L'utilisation de ses ressources tel que la puissance de calcul, la capacité de stockage, la capacité de la mémoire, bande passante, ensemble d'applications et des plateformes, ce fait à la demande sans se soucier de la gestion, l'administration, la sécurité et la maintenance en cas de panne par le client c'est le prestataire de Cloud tel que IBM, Microsoft, Google, SAP, Dell, Amazon … qui se charge de cette tâche.

De manière plus simple, le Cloud Computing est un style d'informatique ou les ressources sont faciles à obtenir et faciles d'accès, simple à utiliser, bon marché et tout simplement fonctionnelles **[2].** Le Cloud est donc un ensemble de matériel, de raccordements réseau et de logiciels qui fournit des services sophistiqués que les individus et les collectivités peuvent

exploiter à volonté depuis n'importe où dans le monde. **[3],** la figure I.1 représente l'architecture du Cloud Computing **[4]**

.

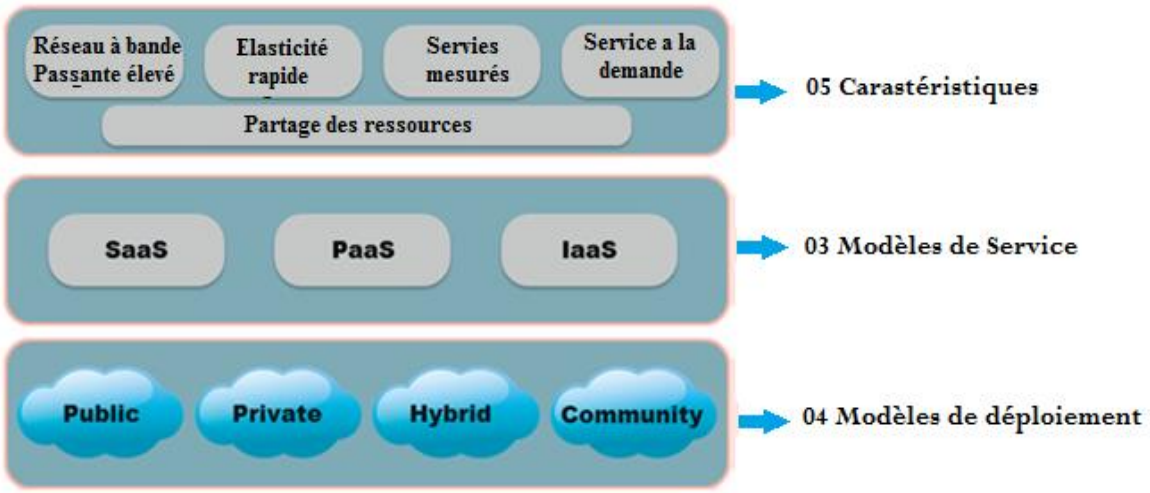

Figure I.1 : Architecture du Cloud Computing

I.2.2 La virtualisation

Ensemble des techniques matérielles et/ou logiciels qui permettent de faire fonctionner simultanément sur une seule machine plusieurs systèmes d'exploitation (appelés VM). A titre d'exemple nous avons VMware, KVM, HyperV **[5].**

Plus formellement, la virtualisation fait référence à l'abstraction physique des ressources informatiques. En d'autres termes, les ressources physiques allouées à une machine virtuelle sont abstraites à partir de leurs équivalents physiques. Les disques virtuels, interfaces réseau virtuelles, réseaux locaux virtuels, commutateurs virtuels, processeurs virtuels et la mémoire virtuelle correspondent tous à des ressources physiques sur des systèmes informatiques physiques **[6],** la figure I.2 est une représentation avec et sans virtualisation **[5]**

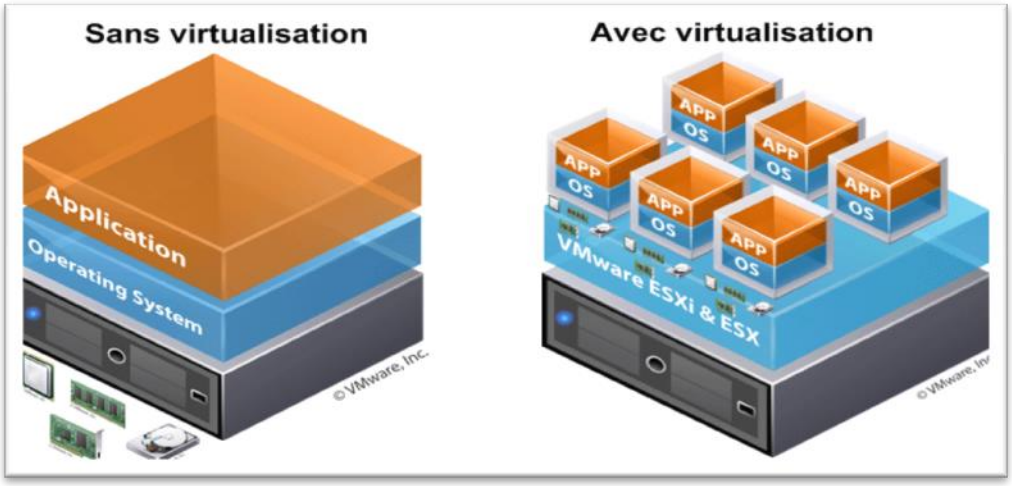

Figure I.2 : Représentation avec et sans la virtualisation

I.3 Comparaison avant et après l'apparition du Cloud Computing

L'apparition du Cloud Computing a bouleversé le monde informatique. Cela crée un grand écart entre l'utilisation ou non de ce concept, ci-dessous un schéma illustrant les étapes de passage de l'évolution en informatique **[3]**

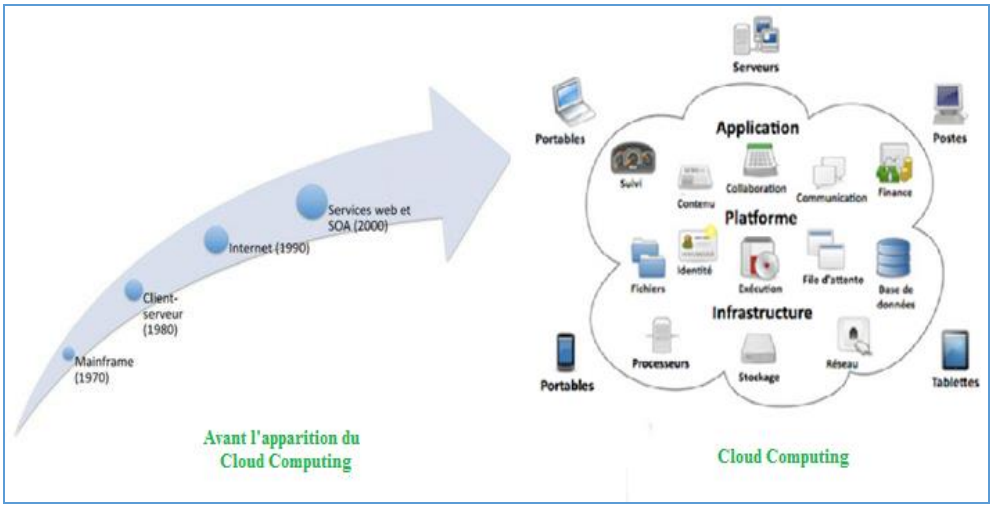

Figure I.3 : Comparaison avant et après l'apparition du Cloud Computing

Le tableau ci- dessus démontre une petite comparaison avant et après l'apparition du Cloud Computing

Avant l'apparition du Cloud Computing	Apres l'apparition du Cloud Computing
Les clients accèdent aux ressources : serveurs, applications, espaces de stockage et services via le réseau LAN ou intranet et internet. Un groupe d'ingénieurs spécialisés est nécessaire pour garantir l'installation, la configuration, la sécurité et la mise à jour du hardware et software. L'accès au serveur ou à l'application sensible se fait depuis l'intranet en passant par l'authentification et selon des droits d'accès bien défini ou depuis Internet en utilisant les VPN « Virtual Private Network ». Si le serveur ou l'application tombe en panne et s'il n'existe pas de mécanisme de reprise après panne ou de sauvegarde toutes les données seront perdues Pour faire la sauvegarde ou les	clients accèdent à des infrastructures informatiques mises à disposition par un prestataire de Cloud via Internet. Le client ne gère aucun matériels, ni logiciels c'est le prestataire de Cloud qui sans charge. Il peut donc se concentrer avant tout sur son travail et son savoir-faire. L'accès au serveur ou à l'application se fait de n'importe où et avec n'importe quel périphérique avec un accès prédéfini a travers le Web. On ne se soucie pas des sauvegardes car la panne est transparente aux clients On dispose d'une capacité de stockage illimité et qui peut augmenter selon les besoins, si le client oublie de sauvegarder ses données ou de faire les backups de

backups de plusieurs serveurs cela nécessite une grande capacité de stockage	ces serveurs, le prestataire de Cloud s'engage à prendre en main cette tache lourde afin de ne pas tomber en panne.
Si le serveur ou l'application n'est pas bien sécurisé, il est facile a attaqué.	le Cloud Computing garantit la sécurité a travers les mécanismes de réplication des données, plan de reprise d'activité,…
Quand le nombre d'utilisateurs augmente dans les entreprises et les organisations…, celle-ci doit investir pour acheter plus d'équipements matériels qui peuvent être couteux.	Le Cloud Computing permet d'accéder plus rapidement à des ressources via un portail web et donc cela nous permet de réduire le cout d'investissement, on a plus a investir dans des équipements matériels très coûteuse en interne.
Le client paie le support même s'il ne l'utilise pas à 100%.	
On ne garantit pas l'accès de n' importe où car ceci représente une faille de sécurité et l'entreprise doit investir et acheter un grand matériels afin de garantir la haute disponibilité.	Le client paie uniquement ce qu'il consomme ou par abonnement mensuel.
La connexion internet est indispensable que pour les clients qui utilisent les VPN, les clients dans l'entreprise accèdent à travers l'intranet.	Le Cloud Computing permet de garantir les accès et la haute disponibilité des services, se facteurs est très important pour les clients nomades.

Les données sont confidentiels est appartiennent totalement à l'entreprise.	La connexion internet est indispensable au client, il l'utilise pour accéder à sa plateforme de travail au niveau du Cloud. Le client n'a pas un accès directe a ses données. Il dépend totalement du fournisseur et doit lui faire entièrement confiance «confidentialité des données, la vie n'est plus privée ».

Tableau I.1 : Comparaison avant et après l'apparition du Cloud Computing

I.4 Les cinq caractéristiques fondamentaux du Cloud Computing

Le Cloud Computing se caractérise par :

• L'accès « à la demande » et en « self-service » par l'utilisateur« On-demand self-service » :

Le Client accède aux services du Cloud Computing selon ces besoins sans avoir recours à l'intervention d'un administrateur interne ou au fournisseur du service en utilisant un accès internet et un périphérique.

• L'accès réseau large bande « Broad network access »:

L'accès au service se fait par des clients lourds, légers, des Smartphones, Tablettes, Mac, PC… via des plates-formes avec une excellente connectivité.

• La mise en commun des ressources « Resource pooling »:

L'attribution des ressources physiques et virtuelles en fonction de la demande est dynamique. La localisation des ressources est transparente aux clients mais celle-ci peut être imposé afin de spécifier l'emplacement à un niveau plus haut d'abstraction (par exemple le pays, l'état ou le Data Center), à ce niveau on n'a plus à s'inquiéter au nombre de serveurs,

taille de disques, nombre de processeurs, en puissance de calcul, capacité de stockage, bande passante disponible...

• L'élasticité rapide « Rapid elasticity » :

Quand les besoins des clients augmentent, il est possible d'obtenir plus de ressources très rapidement donc on s'occupe plus de commander et installer du matériels « gains de cout et de temps ». Quand l'activité des clients diminue, celle-ci peut décider de diminution des ressources allouées.

• Service mesuré en permanence « Measured service » :

L'utilisation des ressources est mesurée de façon précise en fonction de la durée et de la quantité de ressources utilisées, contrôlé et optimisé par le fournisseur du Cloud Computing selon une moyenne estimée de consommation du service. Le client sera facturé selon ces mesures.

I.5 Les services du Cloud Computing

Le Cloud Computing est composé de plusieurs catégories de services, les trois principales catégories sont :(IaaS), (PaaS) et (SaaS). Présenté dans la figure ci-dessous [**6].**

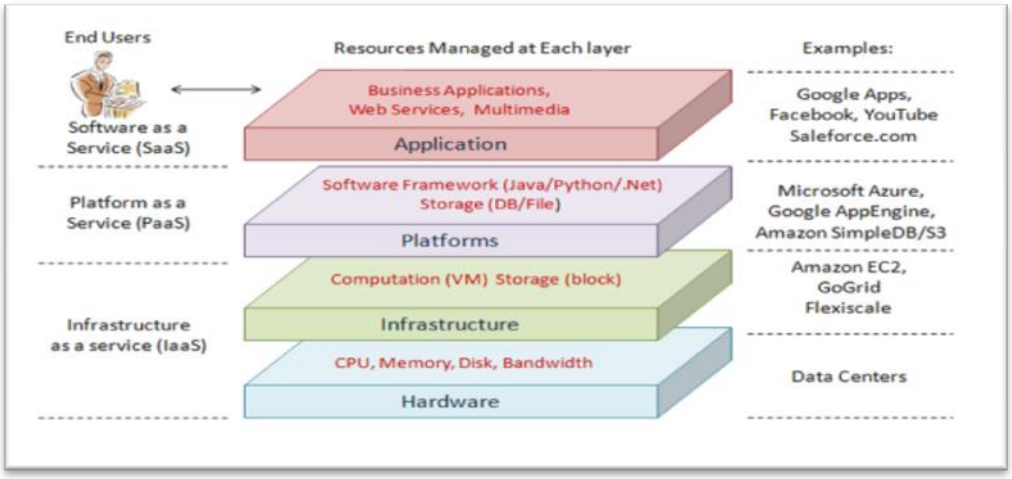

Figure I- 4 : les services de Cloud Computing

I.5.1 IaaS

Le fournisseur offre du matériels informatique vitalisé, tels que les réseaux informatiques, le stockage de données, les systèmes d'exploitation virtuels et la puissance de calcul, donc le client bénéfice de l'utilisation des ressources physiques (hardware) des serveurs, il peut installer son système d'exploitation et ces propres applications dans le Cloud sans se préoccuper des problèmes liés aux pannes et à la sécurité , ces ressources peuvent être allouées dynamiquement en fonction des besoins. A titre d'exemple Amazon Elastic Comput Cloud, RackSpace, Eucalyptus

I.5.2 PaaS

Dans ce mode, le prestataire du Cloud fournit une plate-forme composé d'un ensemble d'outils interactifs qui seront utilisés par les clients pour construire des applications puissantes et flexibles dans un système Cloud Computing. Le client peut installer ses propres applications si besoin. A titre d'exemple Google App Engine, Windows Azure web role, etc.

I.5.3 SaaS

Le prestataire de Cloud fournit aux clients une solution logicielle livrée sur internet, Ces logiciels sont accessibles via différentes interfaces, clients légers, navigateur Web, terminaux mobiles. Saas regroupe IaaS et PaaS dans la même application. Pour le public : on a la messagerie électronique de Gmail, Yahoo, Outlook, Google Apps…Pour les particulier : Le plus utilisé est le stockage de documents (permettant également le partage) comme par exemple :Hubic, DropBox, Google Drive, SalesForce, Le Cloud fournit aussi des logiciels de création de documents, de traitement d'images, de gestion de dessins et bien d'autres qui seront utilisés par les clients selon leurs besoins.

I.6 Modèle de déploiement des services de Cloud Computing

On utilise le terme Cloud pour représenter l'infrastructure gérée par un prestataire. Chaque prestataire propose une solution du Cloud différente, on peut distinguer quatre types principaux de modèles de déploiement des services de Cloud Computing :

I.6.1 Cloud communautaire (Community Cloud)

Dans ce modèle de déploiement, l'infrastructure et les services sont partagés entre un ensemble d'entreprises ou d'organisations clientes indépendantes à l'échelle d'une communauté (ville, académie, …) qui sont reliés par les mêmes intérêts : Ce type de Cloud peut être gérer par l'ensemble des clients ou par un prestataire de Cloud et peut être situées chez les clients qui la partage ou chez le prestataire. A titre d'exemple : UnivCloud.

I.6.2 Cloud privé (Private Cloud Computing)

Dans ce modèle de déploiement, le Cloud est utilisé par un seul client (entreprise ou organisation..) c'est-à-dire que l'entreprise ou l'organisation est propriétaire du Cloud, cette infrastructure est accessible que par les membres de l'entreprise ou l'organisation.

Le Cloud est géré par le client ou par le prestataire et il est situé dans le local du client ou chez le prestataire. Ceci permet de garantir que les ressources allouées ne sont pas partagées avec d'autres clients. Dans ce cas on a deux types du Cloud privé :

a- Cloud Privé interne : Il est géré par l'entreprise ou l'organisation elle-même

b- Cloud Privé externe : Il est géré par le prestataire du Cloud, l'infrastructure est entièrement dédiée à l'entreprise et accessible via réseaux sécurisés de type VPN

I.6.3 Cloud publique (Public Cloud Computing)

Dans ce modèle de déploiement, le prestataire ou fournisseur du Cloud est externe, il possède sa propre infrastructure et ses services sont accessibles par le public. La facturation de ses services se fait selon les ressources allouées aux clients. A titre d'exemple : Amazon web services, Microsoft Azure, Eolas, etc.

I.6.4 Cloud hybride (Hybrid Cloud)

Dans ce modèle de déploiement, l'infrastructure est une combinaison de deux Cloud ou plus (Privé et Publique). Lorsque l'utilisation des ressources du Cloud privé atteint le maximum, l'entreprise peut bénéficier d'autres ressources supplémentaires appartenant au Cloud public. Une autre propriété caractérise ce type de Cloud, c'est la répartition de la puissance de calcul

ou traitement entre des Cloud s publiques et privés ceci permet d'avoir des résultats dans un temps très rapide. Le Cloud Computing permet un équilibrage de charge et évite la surexploitation des ressources quelle que soit le modèle de déploiement choisi.

I.7 Conclusion

Le Cloud Computing est basé sur une supervision centralisée, il ne supporte pas la tolérance aux pannes dans son architecture, ce qui peut rendre le système incohérent et susceptible d'entraîner des ruptures de services. Tout au long de ce chapitre, nous avons abordé la définition du Cloud Computing puis nous avons fait une comparaison avant et après l'apparition de ce concept en précisant les avantages et inconvénients. Nous avons spécifié ces caractéristiques fondamentales, ces services et les modèles de déploiements. Dans le chapitre suivant nous allons introduire les concepts de la tolérance aux pannes, ces types et ces techniques. Pour cela nous exposons quelques travaux similaires.

Chapitre II – Approche de tolérances aux pannes

II.1 Introduction

Ces dernières années de remarquables progrès ont été effectués dans le domaine des systèmes informatiques qui ont permis l'évolution des environnements distribués, avec cette évolution les systèmes sont devenus de plus en plus complexes, de nombreuses pannes surgissent et la continuité de leurs fonctionnement n'est plus garantis. Pour remédier à ces problèmes, des techniques de tolérance aux pannes ont été mis en œuvre. Dans ce chapitre nous allons étudier la tolérance aux pannes d'une façon générale. Nous présentons les concepts de panne et leurs classifications et nous exposons quelques techniques de tolérance aux pannes.

II.2 Définitions des concepts :

- **la panne ou faille** : La panne est l'arrêt accidentel ou l'interruption du fonctionnement d'un système informatique, c'est l'état anormal d'un système ou de l'un de ses composantes matérielles ou logiciels le mettant dans l'impossibilité d'accomplir des fonctions sous des calculs requis.

- **Faute et erreur :** « Une faille (ou panne) du système se produit lorsque son comportement devient inconsistant et ne fournit pas le résultat voulu. La panne est une conséquence d'une ou plusieurs erreurs. Une erreur représente un état invalide du système due à une faute (défaut). La faute est donc la première cause de l'erreur, cette dernière provoque la faille du système. Le but de la tolérance aux pannes est d'éviter la faille totale du système malgré la présence de fautes dans un sous ensemble de ses composants élémentaires » [7].

II.3 Classification des pannes

Les pannes sont classifiées selon différents critères. Le schéma suivant montre une classification générale selon le degré de gravité de la défaillance ou le degré de la permanence de la panne. La figure ci-dessus éclaircis la classification des pannes **[8]**.

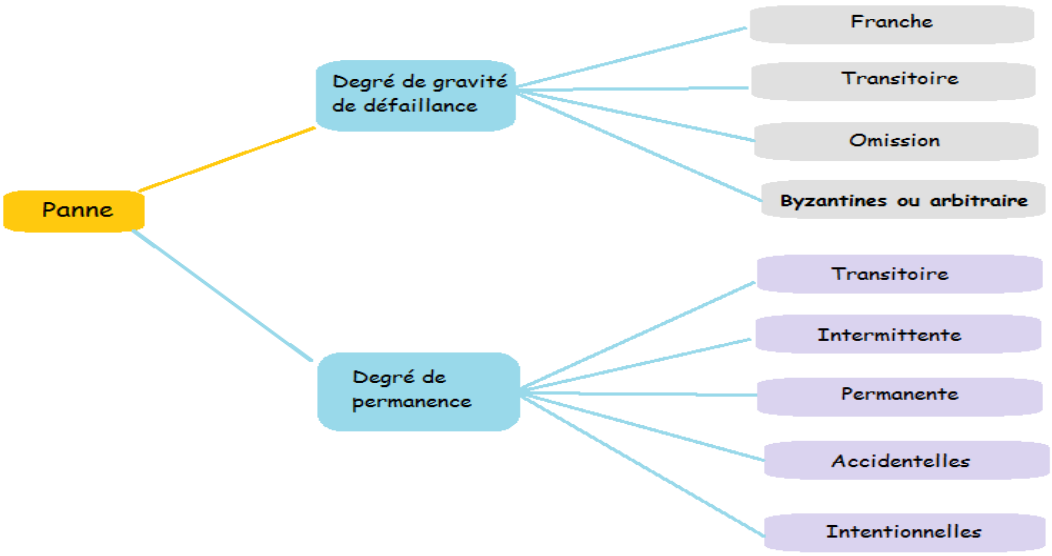

Figure II.1: Classification des pannes

a-Panne selon la gravité des défaillances

1-Panne franche (crash, failstop): Une panne franche est une panne permanente, c'est-à-dire qu'une fois le composant en panne franche, il cesse immédiatement et de façon indéfinie de répondre à toute sollicitation ou de générer de nouvelles requêtes (jusqu'à une réparation)**[8]**

2 -Panne par omission ou transitoire (transient, omission failures): Dans ce type de panne, le système peut perdre des messages en entrée ou en sortie ou les deux, cette panne est considérée comme temporelle de durée infinie. Ce modèle de panne est utilisé pour représenter des défaillances du réseau.

3 -Panne temporelle (timing, performance failures): C'est le comportement anormal du système par rapport à une période de temps, comme par exemple le temps de réponse qui dépasse les exigences des spécifications du système ou l'expiration d'un délai de garde, plus simplement la panne temporelle est une panne franche mais non définitive.

4 - Panne byzantine ou arbitraire (malicious, byzantine failures): Inclut tous les types de pannes, y compris le fait de délivrer un résultat ou un message erroné (intentionnellement ou non), le processus peut alors faire «n'importe quoi », y compris avoir un comportement malveillant. « Les fautes byzantines sont dues à des processus byzantins qui ont un comportement arbitraire, ne suivant plus (nécessairement) le code de leurs algorithmes locaux. Cela peut être du a une erreur matérielle, un virus, ou la corruption du code de l'algorithme » **[9]**

b- Panne selon le degré de la permanence

1-Panne transitoire : C'est un comportement erroné d'un ou plusieurs composants du réseau durant une certaine période. Une fois que cette période est passée, les composants reprennent un comportement correct. Cependant, l'état du système est perturbé. Par exemple, cela peut être des corruptions de mémoires locales de processus ou de contenus de message, ou encore de la duplication de messages » **[9]**

2- Panne intermittent: Ce genre de panne se produit de manière aléatoire c'est-à-dire s'arrête et reprend dans un intervalle irrégulier.

3-Panne Permanente: Une telle panne est stable dans le temps et persiste dès qu'elle apparaît, s'arrête lorsque des spécialistes dans le domaine interviennent et réparent la panne. Un changement physique dans un composant provoque une panne matérielle permanente.

4-Panne accidentelle: Elle se produit de manière accidentelle, soit par une fausse manœuvre, en exécutant du code erroné ou en installant un composant mal configuré ou moins conformant.

5-Panne intentionnelle: C'est l'ensemble des actions malveillantes qui constituent la plus grosse partie du risque pour qu'un système tombe en panne, à titre d'exemple on a les attaques, virus…

II.4 La sûreté de fonctionnement

« La sûreté de fonctionnement des systèmes vise à pouvoir placer une confiance justifiée dans le service qu'ils délivrent »**[10].**La sûreté de fonctionnement s'intéresse entre autre à améliorer la survivance des systèmes, c'est-à-dire leur capacité à continuer de fonctionner pendant et après une perturbation rencontrée durant leur vie opérationnelle. Cette perturbation peut provenir de l'environnement du système (ce qui lui est extérieur) ou du système lui-même (ressources, fautes, ce qui lui est propre).

La figure II.2 présente les trois notions La sûreté de fonctionnement **[10]**

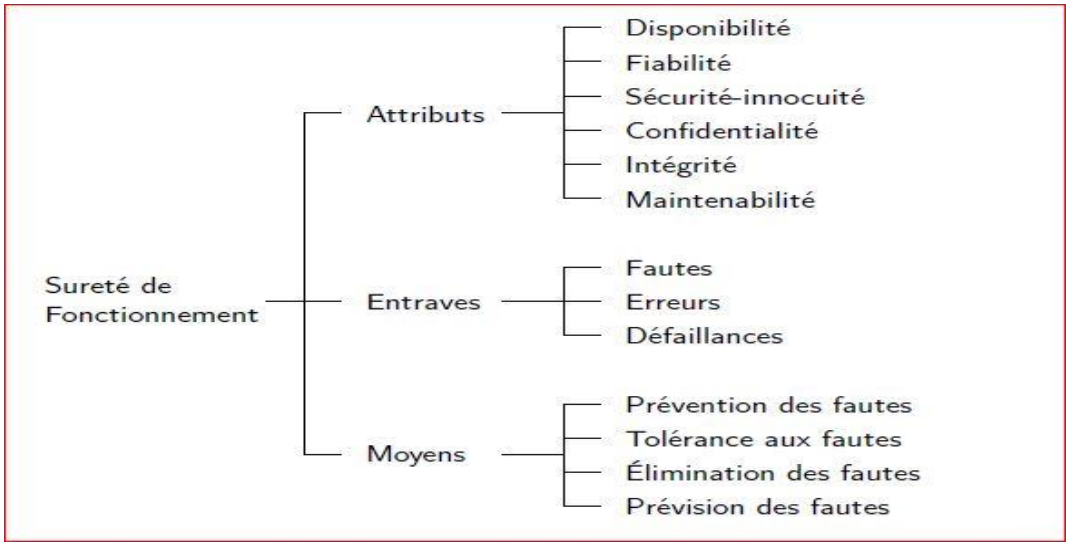

Figure II.2 :L'arbre de la sureté de fonctionnement

II.4.1 Attributs

Les attributs de la sureté de fonctionnement correspondent aux propriétés que doivent vérifier un système. Ces attributs permettent d'évaluer la qualité de service fournie par le système. Il existe quarts attributs de la sureté de fonctionnement **[10]**: disponibilité, fiabilité, sécurité, maintenabilité

- **La disponibilité (Availability**) :c'est la capacité d'un système à rendre un service à tout instant et d'assurer ses fonctions sans interruption et de répondre à une demande à un moment précis
- **La fiabilité(Reliability**) :C'est la capacité du système à se retourner dans un état cohérent après panne et la continuité du service. Pour quantifier la fiabilité on calcule la probabilité en fonction du temps **t**
- **La sécurité (Security) :**C'est l'absence de conséquences catastrophiques sur l'utilisateur ou son environnement avec la Préservation de la confidentialité et de l'intégrité des informations
- **la maintenabilité (Repairability)** : c'est l'aptitude du système à être réparé ou amélioré et sa capacité à restaurer un service correct après une défaillance.

II.4.2 Entraves

Les entraves de la sureté de fonctionnement sont reparties en 3 notions : les fautes, les erreurs et les défaillances qui s'enchainent comme nous le montrons dans la figure suivante **[11]** :

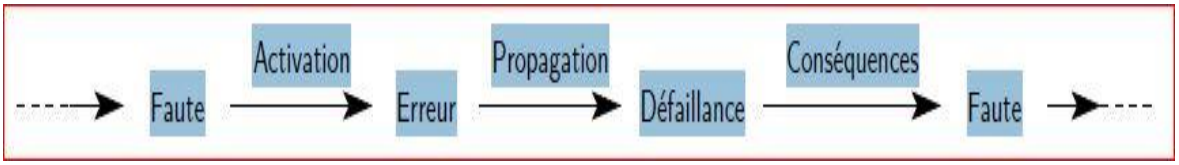

Figure II.3: la chaine fondamentale des entraves à la sureté de fonctionnement

L'activation d'une faute du système provoque la propagation d'erreurs dans le système. Une défaillance a lieu lorsqu'une erreur est observée à la frontière du système**. [11]**

Un système est composé de plusieurs composants élémentaires, et leur défaillance est alors une source d'erreur pour le système, dans le sens où elle peut causer un comportement anormal de ce dernier : il s'agit donc d'une faute pour le système. La chaîne faute, erreur et défaillance deviennent alors récursives.

II.4.3 Moyens

Les moyens sont des solutions éprouvées pour casser les enchainements Faute, Erreur, et défaillance est donc améliorer la fiabilité du système. **[12]** On distingue quatre méthodes principales :

- **La prévention des fautes :** La prévention des fautes est basée sur des méthodes de développement et des techniques d'implémentation pour empêcher que les fautes s'introduisent lors de la phase du développement.
- **La suppression des fautes :** L'élimination de faute peut être divisée en 2 catégories : élimination pendant la phase de développement et élimination pendant la phase d'utilisation. Pendant la phase de développement, l'idée est d'utiliser des techniques de vérifications avancées de façon à détecter les fautes et les enlever avant l'envoi à la

production. Pendant l'utilisation, il faut tenir à jour les défaillances rencontrées et les retirer pendant les cycles de maintenance**. [12]**

- **L'estimation des fautes :** Consiste à faire une estimation qualitative ou probabiliste des fautes qui peuvent se produire et ayant une influence sur le système
- **Tolérance aux pannes :** La tolérance aux fautes est intégrée pour détecter et corriger les erreurs du système. Ce concept sera détaillé par la suite.

II.5 Tolérance aux pannes

La tolérance aux pannes est une méthode de conception qui permet à un système de rester fonctionnelle, éventuellement de manière réduite au lieu de tomber complètement en panne et rester plus ou moins opérationnels lorsque l'un de ses composants ne fonctionne plus correctement **[13]** avec une réduction du débit ou une augmentation du temps de réponse. D'une autre façon le système ne s'arrête pas de fonctionner, s'il y ait défaillance matérielle ou défaillance logicielle **[13,14].**

La propriété de tolérance aux pannes est définie par l'aptitude du réseau à maintenir ses fonctionnalités, en cas de panne de certains de ses nœuds. Elle vise donc à minimiser l'influence de ces pannes sur la tâche globale du réseau **[15].**

II.5.1 Procédure générale de tolérance aux pannes

La conception d'une procédure pour la tolérance aux pannes dépend de l'architecture et des fonctionnalités du système. Cependant, certaines étapes générales sont exécutées dans la plupart des systèmes **[16]** comme illustré dans la figure II-4

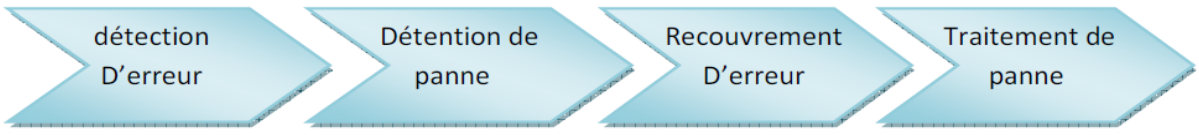

Figure II.4 : Procédure générale de tolérance aux pannes

II.5.1.1 Détection d'erreurs

C'est la phase dans laquelle on détecte qu'un événement inattendu s'est produit dans le système et on signale ou non à l'utilisateur du système à travers un mécanisme de détection qui permet de vérifier l'exactitude du résultat délivré.

Pour détecter les erreurs il existe des techniques de détection qui utilisent la redondance au niveau information ou composant, ou la redondance temporelle ou algorithmique. Parmi les techniques utilisées on a la détention de la panne, le recouvrement d'erreur et le traitement de la panne.

II.5.1.2 Détention de la panne

Cette phase permet de bien cerner la panne dans une zone ou d'un composant particulier c'est-à-dire l'isoler pour ne pas affecter tout le système. A titre d'exemple, en cas de détection d'intrusion, l'isolation des composants affectés minimise le risque d'attaque des composants encore fonctionnels.

II.5.1.3 Recouvrement d'erreur

C'est la phase dans laquelle on procède à l'élimination des erreurs qui cause la panne en utilisant un point de reprise ou par poursuite sur un état prédéfini. Ce qui est sauvegardé lors de la génération d'un point de reprise n'est généralement pas un cliché de l'état de l'ensemble du système, mais seulement l'état d'une partie du système **[17].**Le recouvrement d'erreur se base sur :

- **La reprise**, où le système est ramené dans un état survenu avant l'occurrence d'erreur, ceci suppose qu'on ait sauvegardé auparavant cet état dans un point de reprise.
- **La poursuite**, où la transformation de l'état erroné consiste à trouver un nouvel état à partir duquel le système peut fonctionner (généralement dans un mode dégradé) **[18].**
- **La compensation,** nécessite que l'état du système comporte suffisamment de redondance pour permettre sa transformation en un état cohérent. Elle est transparente vis-à-vis de l'application car elle ne nécessite pas de ré-exécuter une partie de l'application (reprise), ni d'exécuter une procédure dédiée (poursuite). Elle peut par exemple être réalisée en répliquant des composants et en effectuant un vote majoritaire

sur les résultats. Une autre manière de procéder est d'utiliser les codes correcteurs d'erreurs **[17].**

II.5.1.4 Traitement de panne

Dans cette phase, on effectue un diagnostic sur la panne pour déterminer les causes des erreurs (la nature et la localisation) puis on passe à la réparation de la panne en retirant ou isolant les composants fautifs et ceci pour empêcher de retomber dans la même panne. La procédure de réparation dépend du type de la panne. Les pannes permanentes exigent une substitution du composant avec un autre composant fonctionnel **[18].**

II.6 Politique de tolérances aux pannes :

Les techniques de tolérances aux pannes sont basées sur deux types de politiques standard proactive et réactive.

II.6.1 Politique de tolérance aux pannes proactive :

vise à prédire les pannes de manière proactive et remplacer les composants soupçonnés par d'autres composants corrects en évitant ainsi la récupération des fautes et des erreurs. Les techniques basées sur cette politique sont :

- **Software Rejuvenation (Rajeunissement)** : Le système est prévu pour les redémarrages périodiques et chaque fois que le système démarre avec un nouvel État.
- **Proactive tolérance de panne à l'aide d'auto-guérison :** Échec d'une instance d'une application s'exécutant sur plusieurs machines virtuelles est contrôlée automatiquement.
- **Proactive tolérance de panne à l'aide de Préemptive Migration :** Dans cette technique, une application est constamment observée et analysé. Migration préemptif d'une tâche dépend de mécanisme de contrôle de feed-back en boucle.

II.6.2 Politique de tolérance aux pannes réactive :

Cette politique est utilisée pour réduire l'impact des défaillances sur un système lorsqu'une défaillance se produit effectivement. Les techniques basées sur cette politique sont :

- **Check pointing/Restart**. La tâche qui a échoué est relancée depuis le point de contrôle récent plutôt que depuis le début. C'est une technique efficace pour les applications de grandes envergures.
- **Réplication :** afin de rendre l'exécution réussir, plusieurs répliques de tâche sont exécutées sur différente ressources la tâche entière répliquée n'est pas tombé en panne. HAProxy, Hadoop et AmazonEc2 sont utilisés pour l'implémentation de la réplication.

II.7 Techniques de tolérance aux pannes

Les différentes techniques conçues pour tolérer les pannes sont: la Prévention des pannes, le masquage des pannes et le recouvrement après panne.

II.7.1 Prévention des pannes

La prévention des fautes est basée sur des méthodes de développement et des techniques d'implémentation pour empêcher les fautes introduisent lors de la phase du développement. Nous trouvons essentiellement l'approche de migration comme technique de prévention

- **Migration**

Un composant, serveur ou service est susceptible de tomber en panne lors de son exécution, pour éviter l'arrêt du système on migre vers un composant, serveur ou service plus robuste. A titre d'exemple si une baie de stockage tombe en panne, la baie redondante prend le relai avec un temps d'arrêt minimal du système même chose en cas d'arrêt d'un serveur ou service.

II.7.2 Masquage des pannes

Le masquage des pannes est une technique qui consiste à dupliquer les services et serveurs sur différents dispositifs afin de réduire la probabilité que le système tombe en panne. Ainsi

en cas de dysfonctionnement du système la panne est masquée via la duplication. La difficulté de cette technique est de conserver une cohérence forte entre les copies. On peut toujours confondre entre réplication et sauvegarde, ces deux termes sont quasiment différents, la sauvegarde des données ne change pas dans le temps tandis que la réplication est toujours synchronisée avec la copie d'origine.

II.7.3 Recouvrement après panne

Cette approche est basée sur la sauvegarde périodique qui est placé sur un support de stockage stable, persistant et fiable, en cas d'arrêt du système celui-ci sera redémarré à partir point de sauvegarde.

II.8 Travaux connexes

Beaucoup de travaux ont étés fait dans le domaine de la tolérance aux pannes dans les infrastructures du Cloud Computing, dont il y a beaucoup de place de recherche disponible dans ce domaine jusqu'à ce moment-là .Les infrastructures du Cloud ont introduit des nouvelles questions liées à la tolérance aux pannes. Dans ce contexte on va exposer quelques travaux similaires.

Kong et. Coll. **[19]** a présenté un modèle de tolérance aux pannes et de performance dans les infrastructures virtuelles. Mais il n'est pas bien adapté à la tolérance aux pannes des applications dans le Cloud.

Ravi Jhawar, Vincenzo Piuri, Marco Santambrogio, dans leur article**[20]**« A Comprehensive Conceptual System-Level Approach to Fault Tolerance in Cloud Computing», décrit le Framework FTM (Fault Tolerance Manager). Il présente la gestion des techniques de fiabilité pour les utilisateurs dans laquelle il peut spécifier et appliquer le niveau désiré de tolérance de panne sans avoir besoin d'aucune connaissance sur sa mise en œuvre à base sur une couche de service utilisateur dédié.

Une autre approche, utilise un middleware tolérante aux pannes qui est basé sur la réplication a été proposée par W. Zhao, P. M. Melliar-Smith, et L. E. Moser, dans leur article « Fault Tolérance Middleware for Cloud Computing »,**[21].**

Dans «Approach for constructing a modular Fault tolerant protocol » l'article écrit par M.Hiltunen& R.D. Schlichting **[22],** propose des protocoles modulaires et en les combinant à un système à l'aide de techniques hiérarchiques.

Une autre étude vise à fournir une meilleure compréhension des problèmes de tolérance aux pannes et identifie les différents outils et techniques utilisés pour la tolérance aux pannes. Lorsque plusieurs instances d'une application sont en cours d'exécution sur plusieurs VM et l'un des serveurs tombe en panne, il est nécessaire de mettre en œuvre une technique de tolérance aux pannes qui peut gérer ces types de pannes. Pour résoudre ce problème, Anju Bala1, Inderveer a proposé et mis en œuvre la HAProxy. Dans l'article « Fault Tolérance-Challenges, Techniques and Implémentation in Cloud Computing » **[23].**

Un différent modèle de tolérance aux pannes dans le Cloud Computing est proposé par Anjali D.Meshram, A.S.Sambare, S.D.Zade **[24],** Le modèle FTMC (Fault Tolerance Model for Cloud) tolère les pannes sur une base de fiabilité de chaque nœud, ce dernier peut être supprimé si les applications qu'il héberge ne fonctionnent pas bien.

Rajesh.S, KannigaDevi.R, dans leur article « Improving Fault Tolerance in Virtual Machine Based Cloud Infrastructure **» [25]** La méthode qu'ils ont proposée utilise deux ensembles différents de nœuds, l'un est l'ensemble des VM et l'autre est le nœud d'arbitrage principal (nœud serveur). La machine virtuelle utilise le teste d'acceptation pour validité sa logique. Le serveur contient le temps checker qui est l'évaluateur de la fiabilité et le mécanisme de décision pour trouver la VM fiable qui s'occupera pour traiter la demande du client. Pour fournir la tolérance aux pannes, les données peuvent être stockées sur de multiples Cloud utilisant la technique de virtualisation.

II.9 CONCLUSION

Dans ce chapitre, nous avons présenté un aperçu sur les concepts de base de la tolérance aux pannes, la classification des pannes selon la gravité des défaillances et le degré de la permanence. Nous avons aussi abordé d'un point de vue général la sureté de fonctionnement, la procédure générale et les approches conçus pour tolérer les pannes. Il apparait que la tolérance aux fautes n'est qu'une des approches qui permettent d'assurer la sureté de fonctionnement qui est basé sur des techniques de prévention des pannes et de masquage des

pannes utilisant des mécanismes de migration et de duplication et nous avons montré quelques travaux similaires. Dans le prochain chapitre, nous exposons notre méthodologie et les étapes d'installation et de configuration du Framework Hdoop MapReduce et nous étudions les expérimentations et les résultats.

Chapitre III –Tolérance aux pannes dans le Cloud Computing

III.1 Introduction

Les problèmes de sécurité dans une plateforme ou une interruption d'un service, du réseau et d'autres sur le Cloud Computing peuvent affecter les activités et engendrer des pertes économiques et financières pour les clients mais également donner une mauvaise réputation sur ce nouveau concept qui est la cause du retard de la généralisation de cette nouvelle solution. C'est dans ce sens que dans ce chapitre, nous introduisons notre problématique en expliquant notre méthodologie. Nous présenterons le Framework Hadoop MapReduce et les phases de réalisation de notre étude sur la Haute disponibilité (HA) dans HDFS et MapReduce. Nous exposerons par la suite les résultats obtenus suite à l'expérimentation effectuée.

III.2 Problématique

Le Cloud Computing est sujet à divers types de pannes, Parmi les causes de ces pannes, on peut citer les problèmes techniques, les catastrophes naturelles, les pannes de courant, les problèmes de connexion au réseau, le sur chauffage du matériels, les systèmes de stockage défectueux et les erreurs humaines …etc. A titre d'exemple de pannes réelles survenues ces dernières années pour différents prestataires de Cloud nous pouvons citer :

- « Selon une étude, les principales causes de pannes de cloud en 2012 étaient les pannes de courant et les échecs de sauvegardes. Cette même étude indique que le temps moyen d'interruption s'élevait à 7,5 heures.» **[26]**
- Une dizaine de services liés au cloud Azure de Microsoft ont connu une panne mondiale pendant plusieurs heures entre lundi 18 août au mardi 2014. Les services cloud, les machines virtuelles, plusieurs plateformes de restauration et de sauvegarde ont été particulièrement touchées. Les sites web mais également les outils de stockage ont aussi connu des problèmes, mais dans une moindre mesure **[27].**
- En 2015, le Datacenter Belge de Google, à Saint Ghislain, a été frappé par la foudre, qui a grillé des serveurs. Certaines réplications de données n'ont pas eu le temps de s'effectuer. Il y a eu des pertes de données sur 0,000001 % de l'espace de ces serveurs.**[28]**

Ainsi, en cas de panne, relancer tout un traitement n'est pas envisageable économiquement car les services du Cloud sont payants. On ne peut pas demander à un client de payer plusieurs fois les ressources qui ont été mises à sa disposition à chaque panne.
Pour remédier à ces problèmes, les prestataires du Cloud sont en concurrence afin de trouver des solutions à plus bas coût, parmi les solutions proposées on retrouve le standard Hadoop dérivé des travaux de Google (MapReduce). Hadoop est spécialement conçu pour stocker et traiter en parallèle des quantités massives de données appelées Big data. Et c'est dans ce contexte là que nous allons tenter d'introduire des techniques de tolérance aux pannes temporelles lors de la mise en œuvre du Framework *MapReduce.*

III.3 Tolérance aux pannes dans le Cloud Computing par MapReduce

Le Framework MapReduce devient de plus en plus répandu dans le Cloud Computing grâce à ses nombreux points forts notamment pour sa scalabilité et sa tolérance aux pannes. Nous présenterons MapReduce et comment le combiner au Cloud pour remédier aux problèmes de la tolérance aux pannes.

III.3.1 MapReduce

Dans ce projet, nous avons utilisé le Framework « Hadoop MapReduce » sous « Ubuntu ».
Apache Hadoop est un Framework open source de Google écrit en java, conçu pour réaliser des traitements sur des volumes de données massifs (plusieurs petaoctets). Le noyau d'Hadoop s'appuie sur plusieurs briques parmi ces derniers on a les deux grandes briques : la

brique de stockage HDFS et celle du traitement appelée MapReduce **[29]** qui est performante en termes de tolérance aux pannes.

La première brique HDFS : chaque fichier est découpé dans un ensemble de blocs de taille fixe stockés dans des unités indépendantes sur les disques de la plate-forme **[30]**. HDFS fournit une haute disponibilité et la fiabilité en répliquant chaque donnée avec un degré de réplication fixé par défaut à 3. Il est composé du : NameNode, SecondaryNameNode, DataNode.

La deuxième brique du Hadoop est MapReduce qui est un Framework de programmation. Il est principalement utilisé pour la manipulation et le traitement d'un nombre important de données (Big Data) au sein d'un cluster. Ce modèle connaît un vif succès auprès des sociétés possédant d'importants centres de traitement de données telles que **Amazon.com** ou **Facebook…**,**[31**].MapReduce implémente les fonctionnalités suivantes : Parallélisme, gestion transparente du mode distribué et la tolérance aux pannes. MapReduce s'appuie sur deux opérations principales qui traitent les données sous la forme de paires (clé-valeur) et produit en sortie des paires (clé-valeur) Map et Reduce.

III.3.2 Tolérance aux pannes grâce au Framework Hadoop MapReduce

Le système MapReduce offre une solution gérant automatiquement le partitionnement et la duplication des données pour une meilleure tolérance aux pannes, des techniques d'ordonnancement des tâches de calcul visant à améliorer les performances.

Hadoop MapReduce a la capacité de tolérer plusieurs types de panne matérielles ou logicielles, parmi ces pannes on a :

- Panne d'un nœud esclave pour MapReduce (arrêt de NodeManager)
- La panne d'un nœud esclave pour le HDFS (arrêt de DataNode)
- La Panne du nœud maître
- Faute logicielle et arrêt de tâche
- Blockage de tâche

L'évaluation de la tolérance aux fautes de systèmes MapReduce requiert plusieurs éléments tels que définir les fautes à étudier, injecter ces fautes dans un système MapReduce en cours d'exécution et analyser l'impact de ces fautes sur la fiabilité, la disponibilité et la performance du système **[32].**

Dans ce contexte, de nombreux travaux ont été menés pour améliorer la tolérance aux fautes de systèmes MapReduce. Certains travaux se sont intéressés à la tolérance aux fautes en utilisant la réplication de taches **[33],** d'autres se sont orienté à la tolérance aux fautes byzantines en se basant sur la taille du split (le nombre de Map)**.**

III.4 Méthodologie et implémentation

L'implémentation de notre approche c'est effectué en 2 phases : installation et configuration de l'environnement du travail, injection des pannes et déroulement de plusieurs exemples.

La méthode que nous proposons se base sur un cluster composée de 03 nœuds esclaves et 02 maitres, en réalisant la haute disponibilité HA entre les deux maitres, mais est ce que cette HA influe sur le ResourceManager de MapReduce en cas de panne du maitre primaire, pour vérifier cela nous avons effectué des expériences en variant des paramètres et en provoquant une panne.

Pour réaliser ce travail on est passé par les étapes suivantes : tout d'abord nous avons préparé le matériel et le logiciel, déterminé le nombre de nœuds maitres et esclaves, mis en place d'un schéma (la figure ci-dessous). Puis nous avons fixé les adresses IP de chaque nodes appartenant au cluster, Installé et configuré le Framework Hadoop MapReduce, configuré la HA, choisir des paramètres et déroulé plusieurs exemples en utilisant des données en entrées de taille différente dans un environnement avec et sans panne. Enfin de compte nous avons obtenus des résultats représentés sous formes de graphes pour aboutir à la conclusion que MapReduce est tolérant aux pannes.

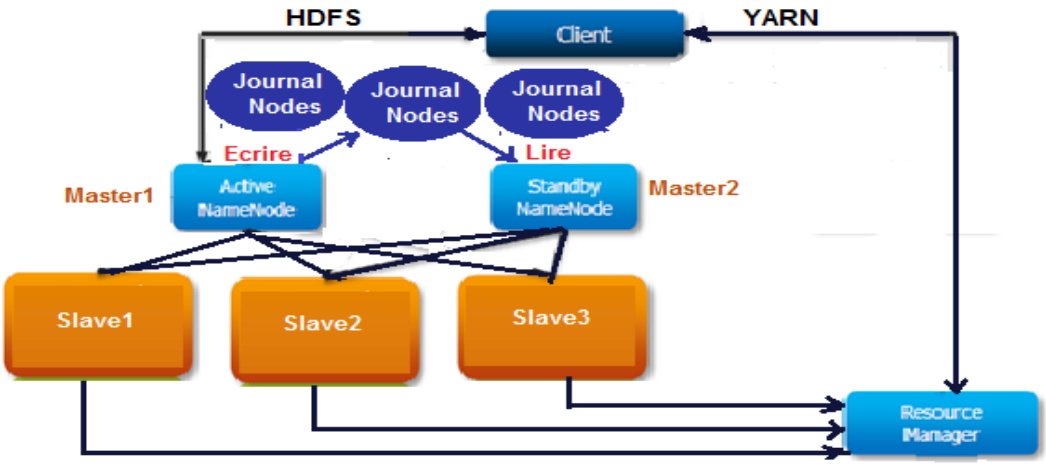

Figure III. 1 : Schéma de l'approche

III.4.1 Phase d'installation et configuration de l'environnement du travail

L'installation se fait en deux étapes : Single Node et Multi Nodes

- Single Node est un cluster Hadoop sur un seul nœud, c'est une instance HDFS avec un Namenode et un seul DataNode et un MapReduce cluster avec un ResourceManager et un seul NodeManager. **[34]**
- Multi Nodes est un cluster Hadoop sur plusieurs nœuds, le Namenode et le ResourceManager se situe sur le master et le DataNode et le NodeManager sur les slaves (le master peut contenir aussi les daemons des slaves),

Nous allons nous basé sur l'installation d'un cluster multi nodes.

- **L'installation d'un cluster Multi Nodes** *:*

- Pour la configuration d'un cluster Multi Nodes consulter le site **[35].**
- Il est très important de modifier le fichier **/etc/hosts :** Ajouter l'adresse IP, le nom des masters et des salves.

172.16.30.31	master
172.16.30.33	master2
172.16.30.35	slave1
172.16.30.36	slave2
172.16.30.37	slave3

- Puis lancer les commandes suivantes :
  hduser@master:~$ **ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser@slave1**
  ## Copier la clé SSH publique du master vers tous les nodes (**slaves**) de l'utilisateur hduser. hduser@master:~**$ssh slave1; (slave2; slave3)**
  ## essayer l'accès (s'il y a aucun problème l'accès SSH se fait sans mot de passe).
- Avant d'exécuter le Namenode (master) pour la première fois, il faut formater tous les nodes du cluster, ceci permettra de supprimer toutes les données(si ce n'est pas la première fois). hduser@master:~$ **hadoop namenode –format**
- Enfin on peut démarrer le cluster multi Nodes :hduser@master:~$ **start-dfs.sh** puis hduser@master:~$ **start-yarn.sh**; respectivement.

start-dfs.sh permet de démarre les daemons Hadoop DFS« NameNode et DataNode, DFSZKFailoverController».

start-mapred.sh permet de démarre les daemons de MapReduce« Resource Manager et NodeManager ».

- Pour vérifier si le cluster et ces services sont bien démarrés lancer la commande :**jps**

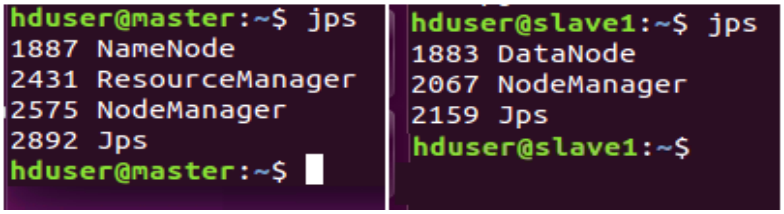

Figure III. 2: vérification du lancement d'un cluster

- Pour arrêter le cluster multi Nodes lancer la commande suivante : **bin/stop-all.sh**
- Consulter les interfaces d'administration du master:

http://master:8088; Visualiser l'état du cluster (vue globale sur les nœuds du cluster)
http://master:50070; Visualiser l'accès aux données contenues dans le nœud NameNode, la capacité totale et l'état de disponibilité des nœuds.

```
<property>
<name>dfs.namenode.rpc-address.myuniv.mas2</name>
  <value>master:8020</value>
<description> l'adresse et le port sur lequel le Namenode mas2 se met  a
l'écoute </description>
</property>
<property>
<name>dfs.namenode.http-address.myuniv.mas1</name>
<value>master:50070</value>
<description> l'adresse http sur laquelle le Namenode mas1 se met a
l'écoute</description>
</property>
<property>
<name>dfs.namenode.http-address.myuniv.mas2</name>
<value>master2:50070</value>
<description> l'adresse http sur laquelle le Namenode mas2 se met a
l'écoute</description>
</property>
```

- **Configuration de la HA pour HDFS**

Pour la configuration de la HA du NameNode, suivre les étapes suivantes :

✓ Ajouter la configuration suivante dans le fichier **hdfs-site.xml**.

```
<property>
<name>dfs.nameservices</name>
<value>myuniv</value>
<description>le nom logique pour ce nouveau nameservice </description>
</property>
<property>
<name>dfs.ha.namenodes.myuniv</name>
<value>mas1,mas2</value>
<description>l'identificateurs uniques pour chaque NameNode dans le
Nameservice </description>
</property>
<property>
<name>dfs.namenode.rpc-address.myuniv.mas1</name>
<value>master:8020</value>
<description>l'adresse et le port sur le quel le Namenode mas1 se met a
l'écoute </description>
</property>
```

Actuellement, on peut configurer seulement un maximum de deux NameNode par Name services. Le Journalnode de notre cluster est lancé sur les nodes "slave1", "slave2 " et "slave3", le Name services est myuniv et le port par défaut du Journalnode est 8485 donc on doit ajouter la configuration suivante dans le fichier **hdfs-site.xml.**

```
<property>
<name>dfs.namenode.shared.edits.dir</name>
<value>qjournal:// slave 1:8485; slave2:8485; slave3:8485/myuniv</value>
<description>le groupe de Journalnodes où les Namenodes vont écrire /
lire les modifications écrites par le Namenode actif et le lut par
le Namenode Standby</description>
</property>
<property>
<name>dfs.client.failover.proxy.provider.myuniv</name>
<value>org.apache.hadoop.hdfs.server.namenode.ha.Configured
FailoverProxyProvider</value>
<description> la classe Java du HDFS clients utilisée pour communiquer  avec le
NameNode actif </description>
</property>
```

Nous avons utilisé la méthode **sshfence** lors du basculement du NameNode actif vers le NameNode**standby**.

-Ajouter les paramètres suivantsdans le fichier core-site.xml:

```
<property>
    <name>fs.defaultFS</name>
    <value>hdfs://myuniv</value>
    <description>définit le chemin par défaut du préfixe utilisé par le client FS de
    Hadoop </description>
</property>
    <property>
    <name>dfs.Journalnode.edits.dir</name>
    <value>/home/hduser/journal/node/local/data</value>
    <description>définit le chemin où le démon Journalnode va stocker son état
    local.</description>
</property>
```

La redondance des données du JournalNode est assurée par l'exécution de plusieurs journalNode (le nombre est de 3 ou plus)

- ✓ **Désactivé Ipv6 pour l'ensemble des nodes existant dans le cluster :**

Apache Hadoop n'est pas actuellement pris en charge sur les réseaux IPv6. Il a été testé et développé sur des piles IPv4 :

hduser@master:~$**Echo '1' > /proc/sys/net/ipv6/conf/default/disable_ipv6**

hduser@master:~$**Echo '1' > /proc/sys/net/ipv6/conf/all/disable_ipv6**

hduser@master:~$**Echo '1' > /proc/sys/net/ipv6/conf/lo/disable_ipv6**

✓ **Démarrer le Namenode master :**

Exécuter la commande suivante : "hduser@master:~$s**tart-dfs.sh**.

Pour s'assurer que le Namenode est démarrer lancer un netstat : **netstat –napgrep 8020**

✓ Une fois que les JournalNode ont été lancés, il faut synchroniser les deux

NameNode pour la HA.

a) Si un nouveau cluster HDFS est configuré, vous devez d'abord exécuter la commande format "**hdfsnamenode –format** " sur le NameNode master.
b) Si le NameNode est formater, copier le contenu de vos répertoires de données NameNode à l'autre, en exécutant la commande suivante : "**hdfsnamenode –bootstrapStandby "** sur le NameNode non formaté « master2 ».
c) Si vous convertissez un NameNode non-HA vers HA, exécuter la commande suivante : "**hdfsnamenode -initializeSharedEdits**", qui initialise ou formate les JournalNode.

✓ Maintenant lancer la HA du NameNode à travers la commande suivante:

hduser@master:~$s**tart-dfs.sh**, celle-ci permet de démarrer les deux Namenode « master et master1 » ainsi que les trois Journalnode lancé au niveau des slaves.

Cette commande va maintenant démarrer automatiquement un démon **ZKFC** (parce que le basculement automatique est activé) sur l'ensemble des NameNode existant dans le cluster, ce daemon sélectionne un seul Namenode active à un moment donné.

✓ Pour vérifier si la HA du cluster est lancé, on accède à la page web de chaque NameNode séparément. On remarque dans la cette page web l'état de la HA du NameNode (soit standby ou active) présenté dans la figure ci-dessus.

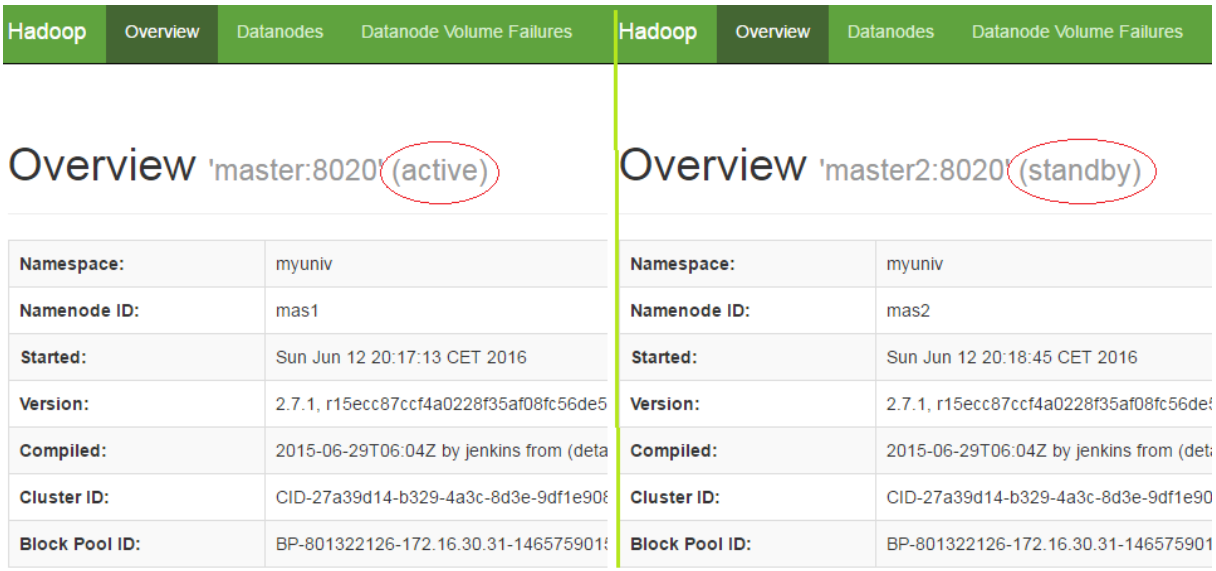

Figure III. 3 : Page web de la HA du NameNode

Pour vérifier si les daemons Hadoop sont lancé, exécuter la commande **jps** :

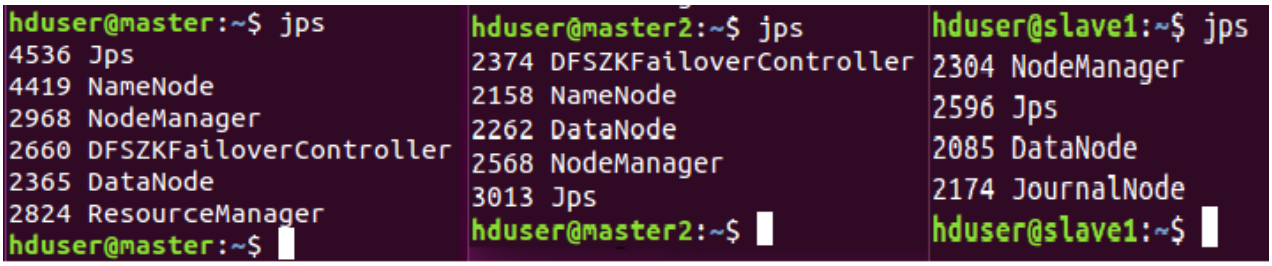

Figure III. 4: Vérification de lancement du daemons Hadoop

III.4.2 Phase tests : Injection et déroulement de cas de pannes possible dans le Cloud

- **Comment injecter une panne**

L'injection d'une faute dans un système MapReduce est traitée différemment selon le type de faute : Panne de nœud, Arrêt de tâche, Faute logicielle, Blocage de tâche.

La panne que nous avons injectée dans notre étude est l'arrêt du master actif de la HA, ceci s'effectue en tuant le daemon NameNode.

- **Déroulement de plusieurs exemples**

Nous avons choisi des mesures différentes pour effectuer nos expériences :

- Latence du MapReduce : temps de réponse à une requête qui représente le résultat en seconde
- Taille d'un Map : **mapreduce.job.maps**
- Taille d'un Reduce : **mapreduce.job.reduces**
- Quantité de données échangées : taille minimal, et taille maximal du split d'entrée **mapreduce.input.fileinputformat-.split.minsize, mapreduce.input.fileinputformat-.split.maxsize**
- Quantité limitée de la mémoire virtuelle pour un Maps :**mapreduce.map.memory.mb**
- Quantité limitée de la mémoire virtuelle pour un Reduce **:mapreduce.reduce.memory.mb**.
-

Taille de fichier (Mo)	N° cas	mapreduce.job.maps (nbr)	mapreduce.job.reduces (nbr)	mapreduce.input.fileinputformat.split.minsize (Mo)	mapreduce.input.fileinputformat.split.maxsize (Mo)	mapreduce.map.memory.mb (Ko)	mapreduce.reduce.memory.mb (Ko)	Latence de MapReduce(sec)
0,265	1	**2**	1	1	128	1024	1024	25
	2	**10**	5	1	150	1024	2048	18
	3	**10**	5	1	150	1024	2048	30
0,545	1	**2**	1	1	128	1024	1024	27
	2	**10**	5	1	150	1024	2048	23
	3	**10**	5	1	150	1024	2048	39
1,74	1	**2**	1	1	128	1024	1024	28
	2	**10**	5	1	150	1024	2048	25
	3	**10**	5	1	150	1024	2048	45
3,38	1	**2**	1	1	128	1024	1024	33
	2	**10**	5	1	150	1024	2048	26
	3	**10**	5	1	150	1024	2048	35
13,5	1	**2**	1	1	128	1024	1024	35
	2	**10**	5	1	150	1024	2048	27
	3	**10**	5	1	150	1024	2048	42
31,5	1	**2**	1	1	128	1024	1024	39
	2	**10**	5	1	150	1024	2048	35
	3	**10**	5	1	150	1024	2048	57
150	1	**2**	1	1	128	1024	1024	140
	2	**10**	5	1	150	1024	2048	90
	3	**10**	5	1	150	1024	2048	136
647	1	**2**	1	1	128	1024	1024	274
	2	**10**	5	1	150	1024	2048	268
	3	**10**	5	1	150	1024	2048	416
1027	1	**2**	1	1	128	1024	1024	355
	2	**10**	5	1	150	1024	2048	318
	3	**10**	5	1	150	1024	2048	468

Tableau III.1 : Tableau de paramètres

Le tableau au-dessus expose les paramètres utilisé dans notre étude et les valeurs des exemples déroulé selon différent cas :

Cas N°1 : Sans pannes, paramètres par défaut (sec)

Cas N°2 : Sans pannes, paramètres modifiés

Cas N°3 : Avec pannes, paramètres modifié

Les tests ont été effectués avec l'exemple du WordCount (sur une masse d'entrée variante entre 500Ko a 1,27Go) dans l'environnement suivant :

- ✓05 machines virtuelle dont : 02 machine maitres pour réaliser la HA et 3 machines esclaves pour la réplication et la journalisation
- ✓Description de la machine physique: 16Go de mémoire, Intel(R) Core (TM) i7-4790 CPU 3,60GHz (08 cœurs) et DDR de 1To.
- ✓En variant les mesures définit précédemment.

III.5 Observations et résultats obtenus

Après avoir exécutés plusieurs exemples, nous récapitulons les résultats de traitement obtenus sur le Tableau III.1. Chaque résultat a fait l'objet d'une moyenne de plusieurs expériences de traitement (3 à 4 expériences), et ce afin d'éviter toute éventuelle valeur pouvant être affecté par la nature multitâche du système.

Cas / Taille de fichier (Mo)	Cas N°1 (sec)	Cas N°2 (sec)	Cas N°3 (sec)	Cas3 – Cas2 (sec)
0,265	25	18	30	12
0,545	27	23	39	16
1,74	28	25	45	20
3,38	33	26	35	9
13,5	35	27	42	15
31,5	39	35	57	22
150	140	90	136	46
647	274	268	416	148
1027	335	318	448	130

Tableau III. 2 : Expérimentation et résultats

III.5.1 Résultats et interprétation

Ces résultats de traitement sont représentés sous forme graphique. Afin d'améliorer la visualisation des résultats illustrés sur la Figure III.5.

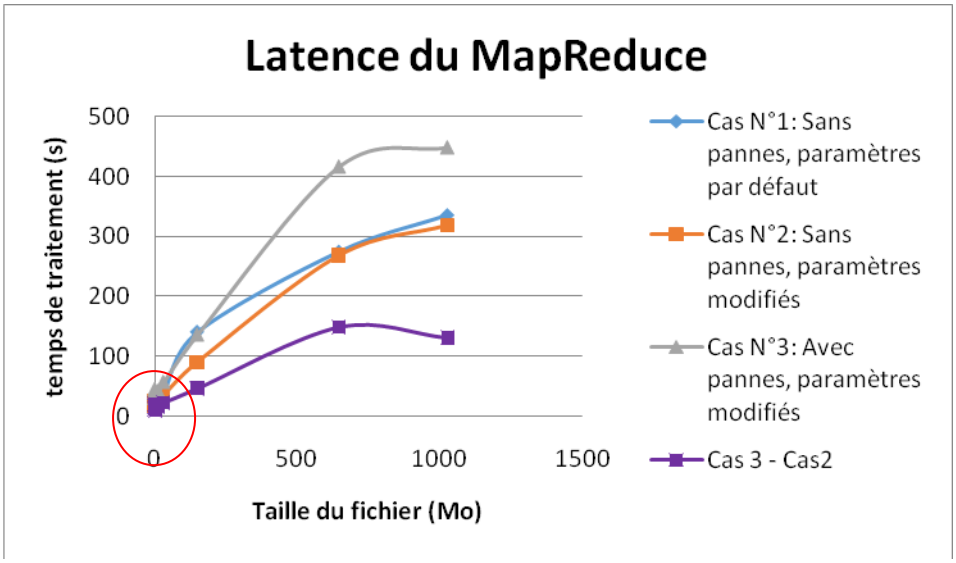

Figure III. 5: Résultats de traitement représentés selon une échelle linéaire

Nous représentons l'axe des abscisses en échelle logarithmique. Nous obtenons les courbes représentées sur la Figure III.6.

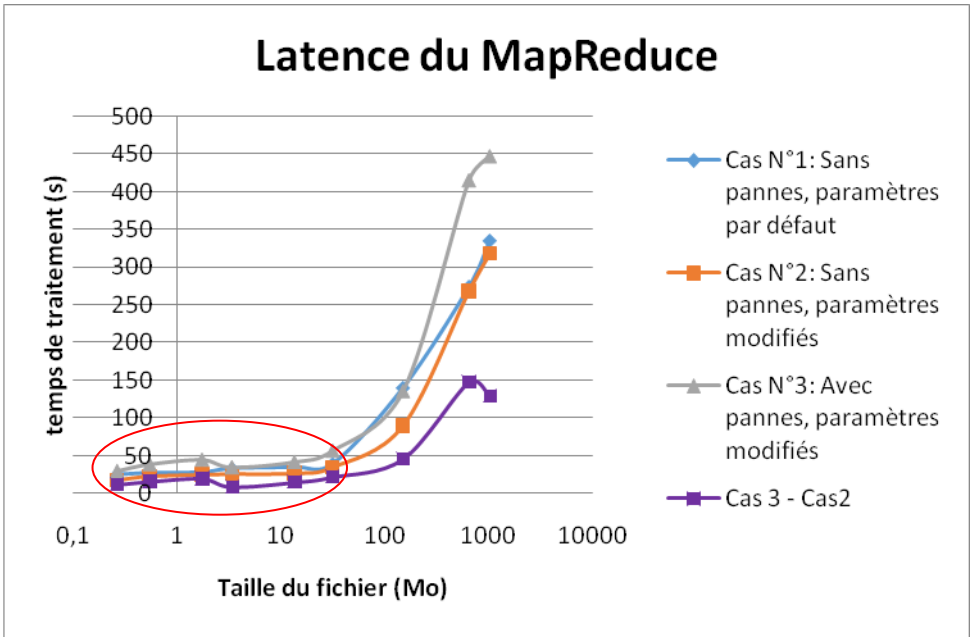

Figure III. 6: Résultats de traitement représentés selon une échelle logarithmique

Nous constatons une meilleure représentation des points cadrés par un cercle en rouge sont mieux mis en évidence sur la courbe en Figure III.5.

- **Interpretation des Courbes :**

Comme illustré sur la Figure III.6, nous constatons que la courbe en rouge du Cas N°2 (sans pannes avec paramètres modifiés) présente le temps de traitement le plus réduit, et ce quelque soit la taille du fichier traité.

La courbe en bleu du Cas N°1 (sans pannes avec paramètres par défaut) présente un temps de traitement relativement élevé par rapport à la configuration du Cas N°2, d'où l'intérêt de l'optimisation des paramètres exploités en vue de l'amélioration du temps de traitement des fichiers.

La courbe du Cas N°3 (avec pannes, paramètres modifiés) dispose du temps de traitement des fichiers le plus élevé en comparaison aux courbes des Cas N°1 et 2.

Malgré les pannes que nous avons introduites, les modifications que nous avons apportées à MapReduce nous ont assuré une haute disponibilité tout en permettant au master Standby de prendre la relève du master active que nous avons mis en panne.

Il est à noter que le temps de traitement n'est pas considérablement affecté par ce procédé. En effet, comme illustré sur la Figure III.6, la courbe en violet représentant la différence entre les temps de traitement du Cas N°3 (avec pannes, paramètres modifiés) et celui du Cas N°2 (sans pannes, paramètres modifiés) montre clairement que pour une taille de fichier donnée, nous aboutissons à une différence de temps de traitement raisonnable de l'ordre de quelques secondes.

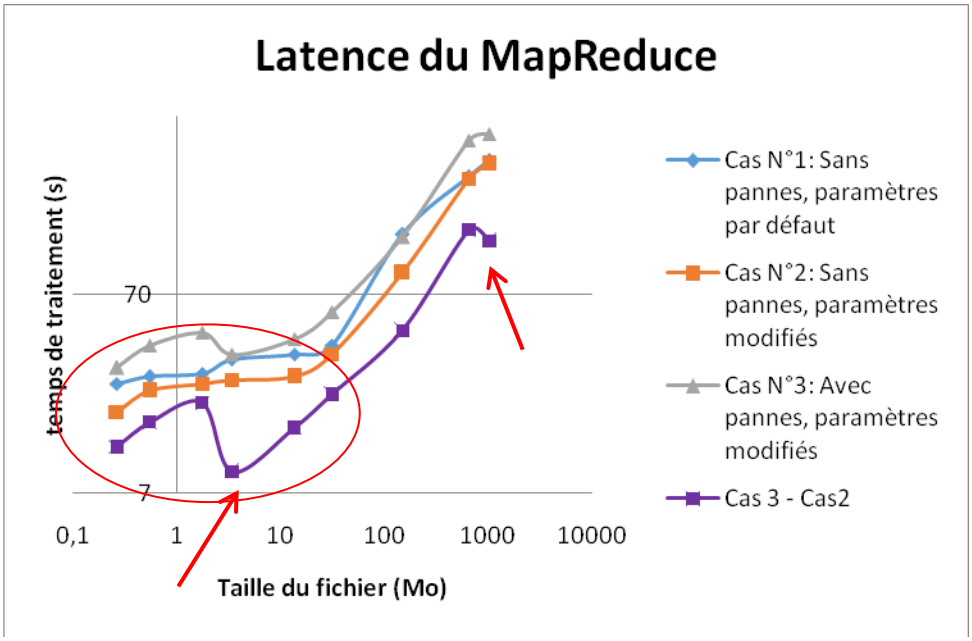

Figure III. 7: Résultats de traitement représentés selon une échelle logarithmique en axes des abscisses et des ordonnées
Temps de traitement réduits pointés par des flèches

Nous avons également remarqué que le temps de différence entre les Cas N°2 et 3 présente certaines valeurs de valeur réduite par rapport aux autres valeurs présentées présenté dans la figure III.7. Ceci est dû au fait que dans certains traitements, il pourrait y avoir des blocs de données qui sont rapidement retrouvés par rapport à d'autres, ce qui permet de raccourcir le temps du traitement.

III.6 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle approche que nous avons conçue permettant d'évaluer la tolérance aux pannes dans le Cloud Computing grâce au Framework Hadoop MapReduce. Nous avons présenté les étapes d'installation et de configuration de cet environnement de traitement, ainsi que la méthodologie suivie pour l'injection de pannes. Nous avons par la suite déroulé plusieurs exemples d'exécutions.

Enfin nous avons présenté les résultats obtenus. Ce qui nous permis de conclure que la tolérance aux pannes est assuré grâce à MapReduce.

Conclusion générale et perspectives

Dans ce mémoire, nous avons procéder à une recherche bibliographique qui nous permis d'accomplir des expériences concernant le nouveau concept du Cloud Computing.

Notre recherche nous a donné connaissance avec les grandes problématiques touchant le Cloud Computing en termes de diverses pannes auxquelles il peut être confronté. Ces pannes peuvent entrainer une perte de performance et peuvent également couter très cher car les services du cloud sont payants. Cette problématique nous a ramené a étudier les travaux connexes traitant la tolérance aux pannes dans le Cloud.
De ce fait nous avons tenté de remédier nous même dans ce mémoire aux pannes dans le Cloud grâce au Framework Hadoop MapReduce et HDFS (Hadoop Distributed File System), en proposant une nouvelle approche qui consiste à assurer la HA (High Availibility) sous une architecture Maitre–Esclave.

Dans cette étude, nous avons exploré les améliorations apportées à Hadoop pour qu'un système soit tolérant aux pannes.

Au cours de nos expérimentations, nous avons pu aboutir à une performance en termes de basculement entre maitre primaire active et maitre secondaire standby. En effet, nous avons constaté que les tâches en cours du traitement ne sont pas interrompues malgré les pannes que nous avons introduites, le maitre secondaire standby à assurer la tâche du maitre primaire active en panne. Il est à noter que l'utilisateur ne remarque par cet état de basculement, ce qui rend la panne inaperçue.

Les expérimentations nous ont permis de regrouper des résultats de traitement en trois configuration ; à savoir les Cas N°1 : sans pannes avec des paramètres par défaut, le Cas N°2 : sans pannes avec des paramètres modifiés, et le Cas N°3 : avec pannes avec des paramètres modifiés. En observant les résultats obtenus, nous avons constaté que la modification des paramètres a permis d'optimiser le traitement avec et sans pannes. Effectivement, le Cas N°2, celui du traitement sans pannes avec modification des paramètres s'effectue à un temps relativement réduit. En plus, le traitement pour le Cas N°3 avec pannes et modification de paramètres s'effectue à une légère augmentation du temps de traitement.

A travers ce résultat, nous pouvons confirmer que notre approche a atteint la haute disponibilité, et ce suite au basculement qui s'est fait dans un temps très réduit et nous avons prouvé que MapReduce tolère les pannes dans le Cloud Computing.

Il est également possible d'étendre la solution dans d'autres travaux au futur avec un matériel plus puissant pour réaliser aussi la HA du ResourceManager. En effet, lorsqu'une panne surgit par exemple au niveau de la HA du HDFS, les deux HA pivots vers le master sain. Cette expérience sera réalisée dans le cas où les deux HA utilisent les mêmes serveurs même si l'un est en état fonctionnel et l'autre dans un état d'arrêt, et ce afin d'aboutir à une meilleure performance de la tolérance aux pannes de MapReduce au sein du Cloud Computing.

Références bibliographique

[1] http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145 vue le 08-03-2016.

[2] F. Rivard, Cloud Computing: Le système d'information sans limite LAVOISIER, 2012.

[3] J. Hurwitz, R.Bloor, M. Kaufmanet FernHalper, Cloud Computing for Dummies, John Wiley& Sons, 2009.

[4] R.Buyya, J.Broberg, A.M.Goscinski, Cloud Computing: Principles and Paradigms, John Wiley& Sons, 2010.

[5] D.Donsez, N.Palma, A.Tchana Panorama du Cloud Computing, Ecole ICAR Université Joseph Fourier 2013.

[6] K.Hess, A.Newman, Virtualisation en pratique, Pearson Education France ,2010.

[7] http://moodle.utc.fr/file.php/498/SupportWeb/co/Module_RCSF_52.html vue le 06-03-2016.

[8] http://beru.univ-brest.fr/~singhoff/DOC/TAP/tolerance-aux-pannes vue le 06-03-2016.

[9] S.Devismes , Tolérance aux fautes , Université Joseph Fourier, Grenoble I 2015.

[10] A.Avizienis, J.Laprie, B.Randellet Carl E. Landwehr, Basicconcepts and taxonomy of dependable and secure computing. IEEE Transactions on Dependable and Secure Computing, 2004.

[11] F.CRISTIAN, Understanding fault-tolerant distributed systems, Communications of the ACM, 1991.

[12] C.Pagetti – ENSEEIHT, Module de sûreté de fonctionnement, 2012.

[13] P.JALOTE., Fault Tolerance in Distributed Systems, Prentice Hall these doctorate, 1994.

[14] P. Narasimhan, T. A. Dumitras, A. M. Paulos, S. M. Pertet, C. F. Reverte, J. G. Slember, and D. Srivastava, "MEAD: support for Real-Time Fault-Tolerant CORBA," in Concurrency and Computation: Practice and Experience, 2005.

[15] J.Paul WALTERS et al., Wireless Sensor Network Security, A Survey", Department of Computer Science Wayne State University, 2006.

[16] http://moodle.utc.fr/file.php/498/SupportWeb/co/Module_RCSF_52.html, Vue le25-03-2016.

[17] M.G Venkata, P. G. Bridges et P. M. Widener: Using application communication characteristics to drive dynamic MPI reconfiguration. In IPDPS Proceedings of the IEEE International Symposium on Parallel Distributed, 2009.

[18] http://homepages.laas.fr/deswarte/Publications/90217, Vue le 02-05-2016.

[19] X. Kong, J. Huang, C. Lin, P. D. Ungsunan, "Performance, Fault-tolerance and Scalability Analysis of Virtual Infrastructure Management System, IEEE International Symposium on Parallel and Distributed Processing with Applications, Chengdu, China, , 2009

[20] R.Jhawar, V.Piuri, M. Santambrogio," A Comprehensive Conceptual System-Level Approach to Fault Tolerance in Cloud Computing, article IEEE, DOI 2012.

[21] W. Zhao, P. M. M.Smith, and L. E. Moser, "Fault Tolerance Middleware for Cloud Computing," Article in Proceedings of the IEEE 3rd International Conference on Cloud Computing, ser. CLOUD "10. Washington, DC, USA: IEEE Computer Society, 2010.

[22] M.Hiltunen& R.D. Schlichting "Approach for Constructing a Modular Fault -Tolerant Protocols "in In Proceedings of the 12th Symposium on Reliable Distributed systems, IEEE,2014.

[23] A.Bala1,Inderveer Chana2,"FaultTolerance- Challenges, Techniques and Implementation in Cloud Computing" Article, Computer Science and Engineering Department, Thapar University, Punjab, India2012.

[24] A.D.Meshram, A.S.Sambare, S.D.Zade, "Fault Tolerance Model for Reliable Cloud Computing", International Journal on Recent and Innovation Trends in Computing and Communication, 2011.

[25] S.Rajesh, K.R.Devi., "Improving Fault Tolerance in Virtual Machine Based Cloud Infrastructure", International Conference on Innovations in Engineering and Technology, 2014

[26] http://www.trendmicro.fr/media/wp/data_security_in_the_time_of_cloud_outages_fr, , vue le 19-05-2016.

.[27] http://www.clubic.com/pro/it-business/cloud-computing/actualite-721673-cloud-azure-panne.html, vue le 19-05-2016.

[28] http://www.silicon.fr/pannes-cloud-donnees-securite-129027.html , vue le 05-05-2016.

[29] https://fr.wikipedia.org/wiki/Hadoop vue le 19-05-2016.

[30] https://fr.wikipedia.org/wiki/MapReduce vue 19-05-2016.

[31] L.Lemkey, A.Sangroya, D.Serrano, S.Bouchenak, Évaluer la tolérance aux fautes de systèmes MapReduce, Université de Grenoble I, Laboratoire LIG, France, janvier 2013.

[32] J.Lejeune, M.Piffaretti, Réplication de taches dans le Framework MapReduce, le 17 mai 2010, https://madeleinepiffaretti.files.wordpress.com/2010/10/rapport_psar.pdf

[33] http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/, vue le 01-06-2016.

RESUME : L'informatique dans les nuages (Cloud Computing) est une tendance actuelle majeure pour répartir les traitements et les données de façon virtuelle sur des environnements d'exécution paramétrables. Le développement et le déploiement de logiciels pour les Cloud proposent un nouveau challenge scientifique en termes d'expression et de prise en compte de la variabilité. En effet, Le Cloud repose sur des principes d'hétérogénéité et d'élasticité, ce qui permet de nombreux choix de configuration et de dimensionnement. Donc, il y a plus de chances d'erreurs, en raison de la latence indéterminée et la perte de contrôle sur les nœuds de traitement. Ainsi, les machines distantes devraient être très fiables. C'est la raison pour qu'une infrastructure Cloud soit bien planifiée lors de l'exécution des tâches de tolérance aux pannes. La tolérance aux pannes est un point majeur et nécessaire dans les infrastructures Cloud. Dans ce projet, nous avons présenté notre approche qui consiste à réaliser une architecture (maitre esclave) tolérante aux pannes qui se base sur la Haute Disponibilité (HA) en utilisant le Framework Hadoop MapReduce.

Mots clés : Cloud, virtuelle, tolérances aux pannes, Haute Disponibilité(HA), Hadoop MapReduce.

ABSTRACT: Computing in the clouds (cloud computing) is a major current trend to distribute processing and virtual data on parameterized execution environments. The development and deployment of software for the Cloud offer a new scientific challenge in terms of expression and consideration of variability. Indeed, the Cloud is based on principles of heterogeneity and elasticity. This allows many configuration and sizing choices. Sothat more chances of errors can occur, because of the indeterminate latency and lose of control over the processing node. Therefore, remote machines should be very reliable. This is the reason for Cloud infrastructures are also well planned when executing fault tolerance tasks. Tolerance to the failure is a major and necessary point in the Cloud infrastructure. In this project, we presented an approach we developed which consists of making a master slave fault tolerant architecture based on high availability using the Hadoop MapReduce Framework.

Key words: Cloud, virtual, failures, tolerances, High availability Hadoop MapReduce.

.

ملخص .إن الحوسبة السحابية هو الاتجاه الرئيسي لتخصيص العلاجات والبيانات في بيئات تنفيذ معلمات تقريبا. تطوير ونشر البرامج للسحابة تقترح تحديا علميا جديدا من حيث التعبير والنظر في التغير. في الواقع، سحابة على أساس مبادئ عدم التجانس ومرونة، والذي يسمح العديد من خيارات التكوين والتحجيم. حتى لا يكون هناك فرصة أكبر من الأخطاء بسبب الكمون غير محدد وفقدان السيطرة على العقد المعالجة. وبالتالي، يجب أن تكون الأجهزة البعيدة موثوق للغاية. وهذا هو السبب في البنية التحتية السحابية يخططون أيضا صحيح عند تنفيذ المهام الخطأ والتسامح. خطأ التسامح هو نقطة مهمة وضرورية في البنية التحتية للسحابة. في هذه الورقة نقدم أسلوبنا في تحقيق بنية (تابع ومتبوع) خطأ تسامحا والتي تقوم على توافر العالي) باستخدام إطار مابريديوس.

كلماتامفتاحيه: سحابة، الظاهري، الخطأ والتسامح، وتوافر عالية مابريديوس.