

Optimisation d'infrastructures de cloud computing sur des green datacenters

Ibrahim Safieddine

▶ **To cite this version:**

Ibrahim Safieddine. Optimisation d'infrastructures de cloud computing sur des green datacenters. Base de données [cs.DB]. Université Grenoble Alpes, 2015. Français. NNT : 2015GREAM083 . tel-01680265

HAL Id: tel-01680265
https://theses.hal.science/tel-01680265

Submitted on 10 Jan 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Informatique**

Arrêté ministériel : 7 août 2006

Présentée par

Ibrahim SAFIEDDINE

Thèse dirigée par **Noël DE PALMA**

préparée au sein du **Laboratoire d'Informatique de Grenoble (LIG)**
et de **l'Ecole Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique (MSTII)**

Optimisation d'Infrastructures de Cloud Computing dans des Green Datacenters

Thèse soutenue publiquement le **29/10/2015**,
devant le jury composé de :

Mr. Daniel HAGIMONT
Professeur, INPT/ENSEEIHT, Rapporteur
Mme. Vania Marangozova
Maître de conférence, Université Joseph Fourier - LIG, Examinateur
Mr. Jean-Marc Menaud
Professeur, Ecole des Mines de Nantes , Rapporteur
Mr. Noël DE PALMA
Professeur, Université Joseph Fourier - LIG, Directeur de thèse

Abstract

With the development of cloud computing, green data center managers face two significant challenges : providing a higher quality of service (availability, security, performance) and improving energy efficiency and environmental footprint. A data center is composed of computing resources, cooling systems, and power distribution. Many research studies have focused on reducing the consumption of data centers to improve the power usage effectiveness (PUE), while respecting the different service level agreements (SLA). Optimizing the energy efficiency of a data center can be achieved on software level using consolidation and on hardware level using optimized power distribution and cooling equipments. Moreover, the complexity of administration and sensitivity of the hosted data raise the issue of security and its impact on performance. Malfunctions should be detected as soon as possible in order for the system to react rapidly and minimize their damage.

In this thesis, we propose two contributions in both the management of energy efficiency and security SLA. In the first contribution, we propose an autonomous system for the global optimization of cooling, which is based on external data sources such as the outside temperature and weather forecasting, coupled with a prediction module of the overall IT load to absorb activity peaks. This system allows us to plan the use of different cooling systems in order to reduce the PUE, while maintaining the quality of service. In the second contribution, we propose a distributed and scalable architecture, to detect anomalies in real time. We use a module based on Big Data technologies to create a statistical model, capable of detecting complex abnormal behavior, difficult to detect with conventional monitoring tools.

We evaluate the performance of our contributions using real applications' data from an operating data center.

Keywords. Autonomic Computing, Green Datacenter, Cloud Computing, Big Data, Real-time Processing.

Résumé

Avec le développement du Cloud Computing, les gestionnaires de centre de données verts font face à des défis de taille : fournir une plus haute qualité de service (disponibilité, sécurité, performance) et améliorer le rendement énergétique et l'empreinte écologique. Un centre de données est constitué de ressources informatiques, de systèmes de refroidissement et de distribution électrique. De nombreux travaux de recherche se sont intéressés à la réduction de la consommation des centres de données afin d'améliorer le PUE, tout en garantissant le même niveau de service Service Level Agreement (SLA). L'optimisation de l'efficience énergétique d'un centre de données peut être faite au niveau logiciel avec des systèmes de consolidation et au niveau matériel avec des équipements de distribution électrique et de refroidissement optimisés. D'autre part, la complexité d'administration et la sensibilité des données hébergées posent de nouveaux problèmes de sécurité et de performance. Il faut détecter les dysfonctionnements le plus rapidement possible pour agir vite.

Dans cette thèse, nous proposons deux contributions pour la gestion des SLA efficience énergétique et sécurité. Dans la première contribution, nous proposons un système autonome d'optimisation globale du refroidissement, qui se base sur des sources de données externes telles que la température extérieure et les prévisions météorologiques, couplé à un module de prédiction de la charge informatique globale pour absorber les pics d'activité. Ce système permet de planifier l'utilisation des différents systèmes de refroidissement, afin de réduire le PUE, tout en préservant la qualité de service. Dans la deuxième contribution, nous proposons une architecture distribuée et scalable, pour la détection des anomalies en temps réel. Nous utilisons un module de calcul se basant sur des technologies BigData pour la création d'un modèle statistique, capable de détecter des comportements anormaux complexes, difficiles à détecter avec des outils de supervision classiques.

Nous évaluons les performances de nos contributions sur des données provenant d'un centre de données en exploitation hébergeant des applications réelles.

Mots-clés. Administration autonome, Centres de données, Cloud Computing, Big Data, traitement temps réel.

Remerciements

Je tiens à adresser mes plus sincères remerciements et toute ma reconnaissance à mon directeur de thèse, Mr Noël DE PALMA, pour son soutien constant, sa patience, sa présence et sa confiance. Je tiens à remercier Mr Gérald Dulac, mon responsable scientifique à Eolas pour tout son suivi tout au long de la thèse. Un grand merci à Yann Vernaz, notre expert en statistique, pour son aide et tout le temps qu'il m'a consacré.

Je remercie les membres du jury, qui ont accepté de juger mon travail : Mr Daniel HAGIMONT et Mr Jean-Marc Menaud pour avoir également accepté de rapporter la thèse, Mme Vania Marangozova pour avoir accepté d'examiner en profondeur le travail réalisé, ainsi que pour tous leurs commentaires constructifs.

Je remercie tous mes collègues à Eolas, que je ne peux pas tous les citer, mais plus particulièrement ma deuxième famille à l'hébergement. Je remercie également mes collègues de l'équipe ERODS, pour leur aide et leur soutien. Merci à tous ceux ont fait le déplacement pour la soutenance, et ceux qui n'ont pas pu venir.

Je ne saurai oublier ma très grande famille qui m'a toujours soutenu dans tous mes choix. Vous constituez une grande partie de mes motivations et j'espère toujours vous faire honneur. Un merci particulier à mes parents qui m'ont toujours soutenu, à plusieurs milliers de km de là, depuis le Liban, mais néanmoins si proches. Enfin, je remercie tous les gens que j'aurais involontairement oublié de remercier.

Table des matières

Résumé **i**

Remerciements **v**

Table des matières **vii**

1 Introduction **1**

1.1 Contexte 1

1.2 Approche 4

1.3 Contribution 6

1.4 Organisation du document 7

2 État de l'art **9**

2.1 Cloud Computing et niveaux de service 10

2.1.1 Le Cloud Computing 10

2.1.2 Les niveaux de service SLA/SLO/iSLO 11

2.1.3 La définition des niveaux de service 13

2.1.4 La gestion des niveaux de service 14

2.2 Optimisation énergétique d'un datacenter Green 16

2.2.1 Indicateurs d'efficience énergétique 17

2.2.2 Cloud Computing et gestion des ressources 18

2.2.3 Gestion optimale des Facilités 19

2.2.4 Prédiction de l'activité d'un datacenter 21

2.2.4.1 Méthodes de prédiction de charge 22

2.3 Le traitement des données Big Data 23

2.3.1 Hadoop 24

2.3.1.1 HDFS 24

2.3.1.2 MapReduce 25
2.3.1.3 YARN 27
2.3.1.4 Spark 28
2.3.2 Distributions Hadoop 30
2.3.3 Configuration optimale des traitements Hadoop 31
2.4 Le traitement temps réel des flux Big Data 32
2.4.1 Apache Storm 33
2.4.1.1 Architecture Storm 33
2.5 Détection des anomalies 34
2.5.1 Détection des attaques de type DDoS 35
2.5.2 Méthodes statistiques pour la détection d'intrusion 37
2.6 Synthèse 39

3 Optimisation du système de refroidissement dans un Datacenter Green 41
3.1 Contexte 42
3.2 Approche et contribution 44
3.2.1 Approche 44
3.2.2 Contribution 46
3.3 Le datacenter Green de Mangin 48
3.3.1 Le système de distribution électrique 48
3.3.2 Le système de refroidissement 50
3.3.2.1 Les pompes d'eau de la nappe phréatique 51
3.3.2.2 Les groupes de production de froid 53
3.3.2.3 L'ultime secours : l'eau de ville 55
3.3.2.4 Le refroidissement des cubes 55
3.4 Réglementations et contraintes 56
3.5 Méthodologie 59
3.5.1 Étude de l'activité d'un datacenter 59
3.5.2 Prédiction de la charge avec SARIMA 60
3.5.3 Utilisation des prévisions météorologiques 60
3.5.4 Algorithme d'optimisation du refroidissement 63
3.5.4.1 Programme d'optimisation linéaire 63
3.5.4.2 Optimisation réactive et prédictive 66
3.6 Expérimentation 68

3.6.1 Évaluation 69
3.6.1.1 Choix manuel d'un seul système de refroidissement 70
3.6.1.2 Optimisation du refroidissement avec une température peu variable 72
3.6.1.3 Exécution de l'algorithme avec des températures variables 73
3.7 Conclusion 75

4 Un système scalable de détection des anomalies en temps réel 77
4.1 Contexte 78
4.2 Approche et contribution 80
4.3 Méthode ACP 82
4.4 Architecture 84
4.4.1 La couche de traitement par lots 84
4.4.2 La couche de traitement temps réel 86
4.5 Conception 88
4.5.1 Collections de données 88
4.5.2 Création du modèle avec Hadoop 90
4.5.2.1 Extraction des données et construction des séries temporelles 90
4.5.2.2 Calcul de la covariance 91
4.5.2.3 Calcul du modèle ACP 92
4.5.3 Détection en temps réel des attaques avec Storm 92
4.6 Expérimentation 94
4.6.1 Clusters d'évaluation 94
4.6.1.1 Cluster Hadoop 95
4.6.1.2 Cluster Apache Storm 96
4.6.2 Architecture réseau des datacenters Eolas 97
4.6.2.1 Journaux des pare-feux 97
4.6.3 Évaluation 98
4.6.3.1 Mesures de performance du traitement par lots . 98
4.6.3.2 Mesures de performance de la topologie Storm . . 102
4.6.3.3 Détection des attaques 103
4.7 Conclusion 105

5 Conclusion **107**
5.1 Rappel du contexte 107
5.2 Conclusion 108
5.3 Perspectives 110
5.3.1 Perspectives court terme 110
5.3.2 Perspectives long terme 111

Glossaire **113**

Bibliographie **115**

Liste des figures **123**

Liste des tableaux **125**

1

Introduction

Contents

1.1 Contexte 1

1.2 Approche 4

1.3 Contribution 6

1.4 Organisation du document 7

1.1 Contexte

Avec l'augmentation de la demande des ressources informatiques, notamment par le développement du Cloud Computing ou "l'informatique dans le nuage", les gestionnaires des centres de données (datacenters) font face à une explosion du besoin de la puissance de calcul. Le marché des datacenters augmente globalement de 15% tous les ans [45]. Les Green datacenters (centre de données verts) de dernière génération sont conçus pour un "Power Usage Effectiveness (PUE)" [4] optimisé. L'indicateur d'efficience énergétique PUE, reconnu internationalement, est le rapport entre la consommation totale du datacenter et celles des équipements informatiques. L'amélioration continue des Green datacenters vise la réduction du PUE pour s'approcher de la valeur idéale PUE=1, c'est à dire que toute la puissance électrique consommée est utilisée par les serveurs informatiques. Un datacenter est considéré Green, si son PUE est inférieur à 1,7. Pour cela, il

faut limiter la part de la consommation énergétique non directement destinée à produire des traitements informatiques. On pense bien évidemment aux systèmes de refroidissement et de distribution électrique, pour réduire la consommation d'électricité.

D'autre part, les datacenters doivent fournir un très haut niveau de service et une meilleure disponibilité, grâce à des systèmes de refroidissement et de distribution électrique redondés, ce qui augmente la facture électrique et fait grimper la valeur du PUE. L'impact économique et écologique devient problématique, principalement à cause de la hausse de la consommation électrique et de l'impact CO2 "Carbon Usage Effectiveness (CUE)" [3] et de l'augmentation de la consommation d'eau "Water Usage Effectiveness (WUE)" [2].

Les prestations informatiques sont de plus en plus dématérialisées et réclament une très haute qualité de service : disponibilité, temps de réponse, sécurité, confidentialité... Les avancées en terme de virtualisation et de grilles de calcul ont permis une évolution des infrastructures classiques vers des infrastructures d'hébergement mutualisées et virtualisées. Le Cloud Computing représente des ensembles de ressources fournies à la demande pour des utilisateurs à travers internet. Les ressources sont exploitées selon un modèle du type « paiement à l'utilisation ». Les garanties sont mises à disposition par le fournisseur de l'infrastructure au moyen d'un contrat de qualité de niveaux de services "SLA" définissant des engagements de performance, de sécurité et de disponibilité. Le non respect du SLA peut entraîner des coûts pour le fournisseur de service (par exemple, l'utilisation temporaire de matériel de secours, la surréservation de ressources ou bien les pénalités en cas de non respect des engagements). Quatre types d'offres sont associées aux infrastructures de Cloud Computing (Figure 1.1) :

- Service logiciel "Software-as-a-Service (SaaS)" qui délivre des applications complètes prêtes à l'emploi.
- Service de plateformes "Platform-as-a-Service (PaaS)" qui délivre des environnements logiciels et matériels permettant de bâtir des applications complètes prêtes à l'emploi.
- Service d'infrastructure "Infrastructure-as-a-Service (IaaS)" qui délivre des ressources " Calcul - Stockage - Réseau " virtualisées dans un datacenter, ou

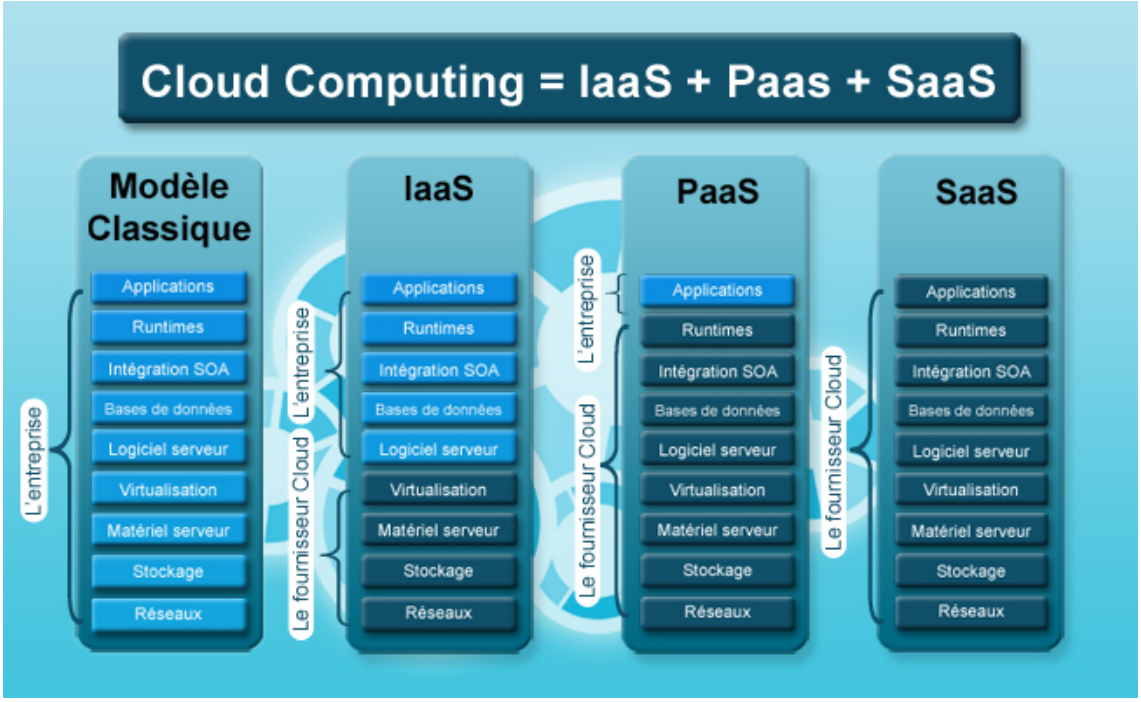


Figure 1.1 – Gestion des plateformes dans le Cloud (http ://www.cloud-serveur.fr/fr/le-cloud/cloud-kesako)

dans plusieurs datacenters c'est alors le service de datacenter "DataCenter-as-a-Service (DCaaS)".

- Service de facilité "Facilities-as-a-Service (FaaS)" qui délivre l'environnement (bâtiments, équipements de sécurité...) en termes d'énergie et de refroidissement pour supporter les services d'infrastructures informatique et de télécommunication.

Du point de vue du fournisseur, l'enjeu majeur de ce type d'infrastructure concerne la gestion des niveaux de services SLA. Certains niveaux de services peuvent être contradictoires (sécurité et disponibilité, performance et efficacité énergétique...). Il n'existe pas une standardisation suffisamment générale pour toutes les applications au sein des datacenters. La virtualisation et les couches SaaS/IaaS/PaaS/FaaS complexifient les modèles de SLA et multiplient les différents niveaux de contrôle ainsi que les politiques de gestions/coût. La détection des anomalies de fonctionnement des applications en temps réel devient primordiale, et notamment les anomalies de type dysfonctionnement ou vulnérabilité aux attaques

applicatives et réseaux. Mais l'hétérogénéité des ressources informatiques et des applications, et l'important volume de données générées rendent l'exploitation des données plus difficile, et nécessite donc une méthodologie Big Data.

Dans cette thèse, nous nous intéressons à la problématique de la gestion du SLA "efficacité énergétique" d'un datacenter. Le coût du refroidissement représente un part important de la consommation totale d'un datacenter. La plupart des systèmes existants pour la gestion du refroidissement fonctionnent en mode réactif, ce qui ne permet pas de prévoir les pics de charge et peut entraîner une dégradation du SLA. Le manque d'une vue globale qui couvre les facilités (refroidissement et distribution électrique) et de l'IaaS, peut amener à des décisions parfois contradictoires et pas optimisés.

D'autre part, nous nous focalisons sur le SLA sécurité et notamment la détection des attaques réseaux. Les attaques réseaux de type Déni de service sont de plus en plus nombreuses, et les solutions existantes utilisent des systèmes de détection parfois faciles à contourner. De même, la détection des attaques doit être faite au plus tôt pour éviter la dégradation du SLA disponibilité et performance. Dans un contexte d'un datacenter hébergeant des milliers d'applications, le défi de détecter une alerte en temps réel est difficile à assurer.

Les travaux réalisés durant cette thèse ont été directement appliqués sur le Green datacenter d'Eolas[1] [26] à Grenoble, en termes de conception et d'exploitation. Ce datacenter Green réparti sur trois étages, a une capacité totale de 13000 U (un "U" correspond à une unité d'hauteur de 1.75 pouces empilés dans les baies). La particularité de ce datacenter est l'utilisation de ressources naturelles dans son système de refroidissement, et notamment l'eau de la nappe phréatique, ainsi qu'une centrale solaire d'une puissance crête de 75kW.

1.2 Approche

Un datacenter est composé de ressources informatiques (serveurs, matériels réseau, baies de stockage...), de circuits de distribution électrique et de systèmes de refroidissement. La consommation du système de refroidissement dans un datacenter représente une part importante qui fait grimper le PUE. L'organisation

[1]Eolas (www.eolas.fr) est une entreprise française qui fait partie du groupe international Business & Decision

"Uptime Institute" classe les datacenters en quatre niveaux : Tier I, II, III et IV (Table 1.1). L'usage montre aussi un niveau de tier III+ ayant un niveau de disponibilité proche du Tier IV, mais avec une conception tolérante aux pannes. Chaque niveau définit des garanties de niveau de disponibilité, de fiabilité et de redondance des équipements déployés dans le datacenter en cas de panne. Un datacenter à très haute disponibilité (Tier IV) est équipé de plusieurs systèmes de refroidissements, parfois hétérogènes, qui assurent la redondance afin de garder dans des conditions optimales les équipements informatiques. Plus le niveau de Tier est élevé, plus le PUE est important à cause des équipements redondants actifs qui font augmenter les coûts de fonctionnement. Dans la majorité des datacenters, des automates industriels contrôlent le choix des systèmes de refroidissement à démarrer en fonction de la puissance demandée et de l'état de chacun des systèmes. Ces automates sont connus pour leur fiabilité et leur réactivité suite à des situations d'urgence, et permettent aux gestionnaires de datacenters de garantir les SLA. Par contre, chaque automate possède une vue limitée aux équipements qu'il gère et prend en compte très peu de paramètres, ce qui rend leur décision moins efficiente globalement. La prise en compte des paramètres telles que la température externe ou les prévisions météorologiques, permet de protéger les équipements contre les cycles de démarrage/arrêt successifs et offre la possibilité de démarrer les systèmes de refroidissement les plus économes tout en respectant les SLA. L'optimisation du choix de système de refroidissement aide à réduire le PUE et permet de réduire également la consommation électrique.

Afin de valider le respect des SLA, des outils de monitoring sont mis en place afin de récolter un maximum de métriques sur l'état des équipements et des services. Les outils de monitoring classiques alertent en cas de dépassement de seuil ou de l'indisponibilité d'une ressource, ce qui permet aux administrateurs de réagir et de lancer des actions correctives. Ces outils restent quand même peu intelligents et se basent uniquement sur des règles prédéfinies par l'utilisateur. De même, les alertes sont émises au moment de la panne ou du dépassement de limite haute ou basse, ce qui s'avère trop tard pour garantir le taux de disponibilité. Pour une meilleure gestion des SLA, une détection intelligente des comportements anormaux aux niveaux des différentes couches de gestion d'un datacenter est une alternative crédible. La détection des anomalies doit être faite le plus rapidement possible, avec une vue globale sur l'ensemble des ressources (physiques et logicielles)

Table 1.1 – Récapitulatif des niveaux de Tier dans un datacenter

Niveau de tier	Caractéristique	Taux de disponibilité	Arrêt annuel	Maintenance à chaud	Tolérance aux pannes
Tier I	Non redondant	99.671%	28.8h	Non	Non
Tier II	Redondance partielle	99.749%	22h	Non	Non
Tier III	Redondance actif/passif	99.982%	1.6h	Oui	Non
Tier III+	Redondance actif/passif	Entre Tier III et Tier IV		Oui	Oui
Tier IV	Redondance actif/actif	99.995%	0.4h	Oui	Oui

d'un datacenter afin d'en tirer des corrélations entre les différents événements et alerter les administrateurs. Ces derniers peuvent mettre en place des boucles de contrôle qui réagissent aux événements afin de réduire le temps de réparation et réduire encore l'erreur humaine. La mise en oeuvre de la détection des anomalies nécessite la mise en place de nouvelles sources de données et l'implémentation de modèles qui permettent de décrire un état normal global du système, c'est à dire les comportements qui peuvent être observés durant l'exécution du système. Ce modèle permet donc de facilement identifier et contrôler les dérives et de corriger les anomalies.

1.3 Contribution

Face aux défis et aux problématiques auxquels font face les gestionnaires des Green datacenters pour se conformer aux SLA de chaque niveau de Tier tout en réduisant l'empreinte écologique et les coûts de leurs infrastructures, nous proposons deux contributions pour les Green datacenters. Dans cette thèse, nous proposons un contrôle réactif et prédictif pour l'optimisation du SLA efficience énergétique afin de minimiser le coût de fonctionnement. Nous proposons aussi un système de détection d'attaques réseaux pour assurer le SLA sécurité dans le contexte de Cloud Computing. L'objet de la thèse est d'apporter une partie de solutions à ces problèmes dans le cadre des deux projets de recherche : CtrlGreen

[16] et Datalyse [18].

La première contribution [69], "**Optimisation du système de refroidissement dans un Datacenter Green**" consiste à optimiser les "Dépenses d'exploitation (OPEX)" et notamment la part importante de la consommation d'un datacenter : le refroidissement. Indépendamment des "Dépenses d'investissement de Capital (CAPEX)" et du type des systèmes de refroidissement, nous proposons un système autonome pour l'automatisation du choix des systèmes de refroidissement. Avec un contrôle réactif et prédictif à la fois, notre système permet d'améliorer le choix du système de refroidissement le plus efficient énergétiquement, afin de baisser le PUE tout en garantissant le même niveau de service. La mise à disposition de nouvelles ressources pour répondre à des pics de charge ne peut pas être instantané ce qui menace la dégradation du SLA. La dimension prédictive permet de répondre aux demandes imprévues et d'anticiper les montées en charge et donc de mieux gérer la qualité de service.

Dans le même spectre du respect de SLA, la deuxième contribution de cette thèse "**Un système scalable de détection des anomalies en temps réel**" consiste à la mise en place d'une architecture Big Data temps réel pour la détection des anomalies de fonctionnement d'un datacenter. L'architecture se décompose en deux parties :

1. Module de calcul d'un **modèle** statistique qui décrit le comportement normal d'un système à partir des historiques de données, avec le Framework Big Data Apache Hadoop [33] de calcul distribué et scalable.

2. Module de traitement des flux de données en temps réel en utilisant le **modèle** calculé dans le module Hadoop. Cette solution a été développée avec la solution Open Source Apache Storm [76] pour le traitement de flux en temps réel.

L'application que nous avons développée vise la détection des anomalies réseaux de type attaque de déni de service distribué "Distributed Denial of Service (DDoS)".

1.4 Organisation du document

Le document est organisé de la manière suivante :

D'abord, nous présentons dans le **chapitre 2** l'état de l'art scientifique sur les travaux de recherche effectués sur les SLA, les systèmes autonomes et la gestion des "Facilités" (refroidissement et distribution électrique). Nous portons une attention particulière sur les travaux de recherche réalisés sur les SLA Cloud, les Green datacenters ainsi que l'utilisation des technique de traitement temps réel et scalable. Nous étudions les mécanismes et les politiques de contrôle FaaS pour optimiser l'efficience énergétique dans un environnement Cloud. Pour le développement de l'outil de détection des anomalies, nous avons étudié l'état de l'art technologique des solutions de traitement des données Big Data et des solutions de traitement des flux en temps réel.

Le **chapitre 3** décrit notre première contribution pour l'optimisation de l'efficience énergétique d'un datacenter avec des systèmes autonomes. Nous avons travaillé sur des boucles de contrôles niveau "Facilités" permettant d'automatiser le choix du système de refroidissement le plus économe et réduire ainsi la facture électrique, en fonction des capteurs internes et externes. Une couche de prévision de l'activité du datacenter permet de protéger le système des oscillations (permutation excessive entre les différents systèmes de refroidissement). Une expérimentation du système et des mécanismes de contrôle, en vraie grandeur a été réalisée sur le cas du Green datacenter d'Eolas.

Ensuite, dans le **chapitre 4** nous détaillons notre deuxième contribution sur la détection des anomalies de fonctionnement en temps réel, qui se base sur des modèles statistiques appliqués sur des historiques de données Big Data. Le modèle calculé, sur lequel se base l'architecture de traitement des flux de données datacenter, permet la détection en temps réel des anomalies afin d'agir au plus vite. Nous présentons ensuite une évaluation de l'architecture sur un cas d'usage : la détection des attaques réseaux de type Déni de Service distribué DDoS sur des données réelles issues d'un datacenter en exploitation.

Finalement, dans le **chapitre 5** nous rappelons le contexte et les objectifs de la thèse et nous donnons la conclusion des travaux et les perspectives court et long terme des travaux que nous envisageons.

2

État de l'art

Contents

2.1 Cloud Computing et niveaux de service 10

2.1.1 Le Cloud Computing 10

2.1.2 Les niveaux de service SLA/SLO/iSLO 11

2.1.3 La définition des niveaux de service 13

2.1.4 La gestion des niveaux de service 14

2.2 Optimisation énergétique d'un datacenter Green . . 16

2.2.1 Indicateurs d'efficience énergétique 17

2.2.2 Cloud Computing et gestion des ressources 18

2.2.3 Gestion optimale des Facilités 19

2.2.4 Prédiction de l'activité d'un datacenter 21

2.2.4.1 Méthodes de prédiction de charge 22

2.3 Le traitement des données Big Data 23

2.3.1 Hadoop 24

2.3.1.1 HDFS 24

2.3.1.2 MapReduce 25

2.3.1.3 YARN 27

2.3.1.4 Spark 28

2.3.2 Distributions Hadoop 30
2.3.3 Configuration optimale des traitements Hadoop 31
2.4 Le traitement temps réel des flux Big Data 32
2.4.1 Apache Storm 33
2.4.1.1 Architecture Storm 33
2.5 Détection des anomalies 34
2.5.1 Détection des attaques de type DDoS 35
2.5.2 Méthodes statistiques pour la détection d'intrusion . . 37
2.6 Synthèse 39

Dans cette thèse, nous nous intéressons à l'optimisation énergétique des datacenters Green et au respect des niveaux des services SLA des applications hébergées, grâce à la détection des anomalies de fonctionnement en temps réel. Dans la première partie de l'état de l'art nous nous intéressons à l'étude des technologies existantes pour la définition et la mise en oeuvre de notre architecture, avec des solutions de calcul Big Data et de traitement de flux en temps réel. Ensuite, dans la deuxième section nous étudions les travaux de recherche autour de l'optimisation énergétique des datacenters et les solutions réactives et prédictives proposées. Nous présentons également une étude bibliographique des solutions de détection des anomalies dans un datacenter, et notamment les attaques réseaux.

2.1 Cloud Computing et niveaux de service

2.1.1 Le Cloud Computing

Le "Cloud Computing" [27, 74] ou "Informatique dans le nuage" représente des ensembles de ressources fournies à la demande pour des utilisateurs, qu'ils peuvent réserver et libérer selon leurs besoins. C'est un service avec payement à l'utilisation. L'utilisateur paye uniquement les ressources qu'il utilise. En plus des services IaaS, PaaS et SaaS ordinaires, nous voyons apparaître de nouveaux "X" as a Service qui sont des services quelconques fournis à travers internet et qui

remplacent des services gérés habituellement en local. Des nouveaux services ont vu le jour : HaaS (Hadoop as a Service), TaaS (Internet of Things as a Service), MaaS (Monitoring as a Service)...

Les ressources dématérialisées dans le Cloud sont accessibles partout dans le monde à travers internet. Les fournisseurs de services Cloud proposent également un hébergement multi-datacenters pour une meilleure fiabilité, ce qui permet d'assurer la haute disponibilité de leurs applications.

Grâce à la virtualisation des datacenters, les utilisateurs réservent des machines virtuelles qui peuvent être distribuéss sur des serveurs physiques différents, ce qui permet aux fournisseur d'adapter l'utilisation des ressources physiques en fonction de la demande et faire donc des économies d'énergie en arrêtant les serveurs inutiles.

Les services de Cloud Computing sont payants, et donc des garanties sont offertes par le fournisseur de l'infrastructure au moyen d'un contrat de qualité de niveau de services définissant des engagements de performance, de sécurité et de disponibilité. Pour cela, le client dispose d'un accès à une interface graphique qui offre un monitoring de l'ensemble de ces ressources, qui permet en même temps la gestion de la facturation.

2.1.2 Les niveaux de service SLA/SLO/iSLO

Les fournisseurs de services font face au défi de devoir se conformer à des contrats de niveaux de services "SLA" [15, 19] contractualisés entre le client et le fournisseur hébergeur. Ces contrats sont des accords qui régissent la performance garantie que les clients peuvent attendre d'un service en ligne. Ces contrats énoncent également les mesures d'anticipation, de correction et d'organisation comme le Garantie de Temps d'Intervention (GTI) et le Garantie de Temps de Réparation (GTR) ainsi que toutes les conséquences qui entreront en vigueur si les performances ne sont pas au rendez-vous.

Le SLA [15, 19] est un contrat entre le fournisseur et le client qui garantit un niveau de performances spécifiques et la fiabilité à un certain coût. Un SLA complet est un document mentionnant les parties impliquées, les termes de l'accord, les coûts, les modalités d'adaptation, les rapports, les responsabilités de chaque partie... Ce contrat spécifie aussi ce que le fournisseur de service attend de son client en termes de charge d'utilisation de ressources. Si le niveau de service

n'est pas respecté, le client peut demander une indemnisation. Les pénalités de non-respect du SLA peuvent coûter cher aux fournisseurs de services. Beaucoup de travaux de recherches visent l'augmentation de l'efficience énergétique des datacenters tout en évitant la violation des SLA.

Un service en ligne est composé de plusieurs dimensions :

- Son cycle de vie : développement, pré-production, production, archivage.
- Ses fonctionnalités : Serveur critique, stratégique, PRA (Plan de Reprise d'activité)

Chaque fonction (cycle de vie et fonctionnalité) peut avoir un SLA associé. Les objectifs quantitatifs des SLA sont définis grâce à des Service Level Objectives (SLO) [15] qui permettent de spécifier les différents aspects de niveaux de service. Les SLO sont assurés par la tenue de iSLO (SLO interne) qui représentent les outils de mesure des différents processus. Les SLA sont gérés en 5 familles :

- La gestion du projet
- Le support : Suivi des demandes (par ticket ou par téléphone)
- les performances techniques : Bande passante, latence, fiabilité, disponibilité, maintenabilité, sécurité
- l'empreinte écologique du datacenter, mesurée grâce à des indicateurs d'efficience énergétique (PUE [4], WUE [2], CUE [3]...)
- les consommations : Ressources IT, bande passante réseau, électricité

Le respect des SLO peut entraîner des coûts pour le fournisseur de service (par exemple, par l'utilisation temporaire de matériel de secours, la sur-réservation de ressources ou bien des pénalités en cas de non respect). Le fournisseur de service peut faire face à un choix difficile : remplir à tout prix le SLA ou accepter des violations temporaires. Pour limiter les violations de SLA, le fournisseur de service a 2 solutions : (i) Dimensionner les ressources pour tenir compte des pics de demande les plus exigeants ou (ii) Adapter les ressources pour tenir compte des sollicitations du client. La solution i est très onéreuse en termes de coût IT et d'énergie. La solution ii est intéressante car elle permet d'assurer des SLA tout en minimisant les ressources requises. Elle correspond au modèle du Cloud

où les ressources peuvent être approvisionnées dynamiquement. Cependant elle accroît les risques de violation de SLA. Assurer des SLA grâce à des mécanismes dynamiques de gestion de ressources permettant de minimiser les coûts d'utilisation de ressources matérielles et de consommation des "facilités" (eau, électricité ...) est un enjeu majeur pour les exploitants d'infrastructure de Cloud Computing. Les solutions actuelles dans le Cloud gèrent des SLA avec une granularité importante négocié de manière ad hoc sans avoir une vision globale du risque de violation de SLA. Les SLA liés aux performances sont extrêmement difficiles à assurer dans ces environnements à cause de la grande variabilité en terme de performance de machines virtuelles (Virtual Machine (VM)). Les applications installées sur ces VM sont des boites noires. Par exemple, seuls des SLA de disponibilité sont assurés par un opérateur industriel leader de Cloud comme Amazon.

2.1.3 La définition des niveaux de service

La notion de SLA est généralisée mais elle entraîne des difficultés. Certains niveaux de services peuvent être contradictoires tel que la sécurité et la disponibilité, ou la performance et l'efficacité énergétique. Il n'y a pas de standardisation suffisamment générale pour toutes les applications au sein de datacenters. En plus, la virtualisation et les couches IaaS/PaaS/FaaS complexifient les modèles de SLA et multiplient les différents niveaux de contrôle ainsi que les politiques de gestions et de coûts.

Le comité C-SIG de l'union européenne, qui regroupe des acteurs du Cloud de toute l'Europe, a travaillé sur des directives de standardisation des SLA [15] pour les fournisseurs de services Cloud et leurs clients. Cet article fournit un dictionnaire des termes utilisés dans le contexte SLA du Cloud Computing et qui peuvent révéler des ambiguïtés. Le comité a défini les différentes familles de SLA ainsi que la description de leurs SLO et leurs méthodes de mesure, en mettant l'accent particulièrement sur les aspects sécurité :

- Performance : Disponibilité, temps de réponse, capacité, le support et la réversibilité en cas de résiliation du contrat.
- Sécurité : Fiabilité du service (niveau de redondance), le niveau d'authentification et d'autorisation, la protection cryptographique, la gestion des incidents de sécurité et le reporting, la gestion des traces et de la supervision...

Les lois varient selon les pays, ce qui peut mettre en question l'accessibilité aux données dans le cloud. Le BackUp fait partie de la sécurité.

- La gestion des données : Classification des données, sauvegarde et restauration, durée de vie des données, gestion des données personnelles...

2.1.4 La gestion des niveaux de service

Beaucoup de travaux de travaux autour de la gestion des SLA ont été proposés afin de réduire les pénalités. La faible utilisation des serveurs dans les datacenters est une préoccupation des fournisseurs. Optimiser l'utilisation de serveurs permet de réduire la consommation électrique, le nombre de serveurs nécessaires, le coût du refroidissement et la surface nécessaire.

Dans [49], Lee et al. présentent les fonctionnalités et les composants d'un système de gestion de SLA (SLM : SLA Management System). Un SLM possède les fonctionnalités suivantes :

- Service Level Metric : Gestion de la qualité des mesures.
- Service Level Objectives : Valeur minimale de la mesure de qualité.
- Service Level Measurement : méthode pour évaluer le SLO incluant les métriques à évaluer, le but et la période de l'évaluation.
- Service Level Reporting : type et méthode de rédaction des rapports sur le niveau de service.
- Biling Adjustments : En cas de non-respect du niveau de service, il est important de décider des modalités des pénalités.

Dans cet article est proposé une architecture d'un SLM qui comporte un composant de collecte de données, un composant de surveillance pour détecter la violation du contrat SLA à partir des données collectées et envoyer une alerte en cas de violation, et un composant d'analyse et de statistique qui génère des statistiques à partir de données stockées.

Pour mieux gérer les SLA et définir le dimensionnement d'une plateforme, les fournisseurs de services passent les applications sur un banc de test pour évaluer leurs performances à l'aide d'outils de test de charge. Plusieurs outils d'injection

de charge permettent de mesurer les limites d'une application tout en récoltant des mesures de performance (Neoload [58], JMeter [39], ...). Clif [78] est une solution scalable d'injection de charge Open Source qui permet de redimensionner automatiquement la plateforme d'injection en fonction du profil de charge afin de réduire les coûts.

L'approvisionnement dynamique est très utile pour répondre à l'augmentation de charge sur les applications. Les applications web sont les meilleurs candidats puisqu'il est difficile d'estimer les pic de charge dans le Cloud. Il faut trouver un moyen de réduire les coûts dans les offres de type paiement à la demande. La plupart des travaux faits pour le redimensionnement dynamique des applications multi-tiers se focalisent sur le dimensionnement des serveurs frontaux, ce qui est relativement simple puisqu'un serveur frontal n'a pas d'état (une fois démarré, un front est prêt).

Les solutions proposés dans [7,11,38,43] implémentent des méthodes de gestion des ressources de type machines virtuelles (VM) pour limiter les violation des SLA tout en réduisant les coûts : c'est l'objectif principal des datacenters. [38] propose une solution pour optimiser le SLA temps de réponse d'une application web Multi-tier en dupliquant les noeuds surchargés.

[7] propose une autre approche pour limiter les violations de SLA en proposant un Framework de migration et de consolidation dynamique de machines virtuelles pour limiter les coûts. La consolidation des serveurs peut être statique ou dynamique. La consolidation statique se base sur la prédiction de ressources nécessaires d'une machine virtuelle en se basant sur la l'historique récent des ressources demandées. La consolidation dynamique se base sur la variation de charge de la VM sur des périodes de temps bien définies. L'algorithme propose alors de déplacer la VM si les capacités de la machine physique sont inférieures aux besoins de la VM. L'algorithme range les VM en ordre décroisssant de leur future demande de ressource et essaye ensuite de la placer la première VM sur le serveur en utilisant le bin-packing. L'algorithme s'assure que la somme des ressources demandées par les VMs ne dépasse pas la capacité de chaque serveur physique. Si la liste des serveurs physique est épuisée, l'algorithme place la VM sur le serveur le moins chargé. Dans les tests faits sur un ESX VMware, il a constaté que la durée de migration est indépendante de l'utilisation CPU. Pour limiter le taux de violation des SLA, l'algorithme prend en compte un seuil à ne

pas dépasser.

La base de données reste problématique pour l'approvisionnement dynamique parce qu'il faut répliquer toutes les données de la base. La méthode proposée dans [12], présente une technique de clonage de VM pour créer des réplications de base de données dans le Cloud afin de réduire ainsi la latence de démarrage d'une nouvelle réplication. Dolly utilise des mécanismes de réplication de machines virtuelles pour déployer automatiquement des réplications de base de données. L'algorithme utilise une méthode de prédiction de charge qui se base sur l'historique des mesures pour estimer le nombre de réplications nécessaires pour tenir la charge. La fenêtre de prédiction n'a pas une largeur fixe et bouge avec le temps. Nous avons le choix entre utiliser un ancien clone, ce qui va augmenter le temps de synchronisation après le démarrage de la VM, ou créer un nouveau snapshot ce qui va charger les I/O pendant la création mais réduit le temps de synchronisation des réplications. Par contre cette méthode ne prend pas en compte la charge réseau dans le temps de réplication et ne propose pas un dimensionnement vertical.

2.2 Optimisation énergétique d'un datacenter Green

Un Green datacenter est un bâtiment dans lequel les ressources informatiques et les systèmes de refroidissement et de distribution électrique ont été conçus pour un maximum d'efficience énergétique et un minimum d'impact environnemental. Plusieurs aspects permettent de classer un datacenter comme Green :

- L'utilisation de sources d'énergie renouvelables (photovoltaïque, barrages électriques...) et la réutilisation de la chaleur dissipée par le fonctionnement du datacenter pour chauffer les locaux.

- L'utilisation des systèmes de refroidissement à faible consommation tel que le Free Cooling ou le Natural Cooling permet de réduire les émissions de CO2, et le choix de la température de fonctionnement des salles informatiques.

- Le choix des processeurs des serveurs informatiques efficients énergétiquement et à fort rendement.

- La méthode et le cycle de recyclage du matériel informatique.

Business & Decision a lancé en 2009 une initiative de bonne conduite pour les clients finaux des datacenters baptisée GreenEthiquette [22]. Cette charte engage les fournisseurs à réduire leur empreinte écologique. Malgré les réductions du coût d'exploitation, la mise en oeuvre d'infrastructures éco-responsables demande un investissement conséquent de la part des fournisseurs de services hébergés.

2.2.1 Indicateurs d'efficience énergétique

Pour mesurer l'efficacité énergétique d'un datacenter, plusieurs indicateurs ont été proposés pour comparer les datacenters et suivre leur évolution, dans une démarche d'informatique éco-responsable. Le consortium "Green Grid" propose l'indicateur qui fait référence en la matière, le PUE [4] qui est l'indicateur d'efficience énergétique. Il permet aussi de mesurer les progrès, l'enjeu étant de tendre au fil du temps vers un PUE de 1, l'idéal théorique. Il est utilisé maintenant comme un élément de mesure des progrès réalisés plus globalement. Un Green datacenter est caractérisé par un PUE entre 1.2 et 1.4. Il est ainsi envisageable d'atteindre un PUE inférieur à 1 en incorporant des équipements de production d'énergie, comme les panneaux solaires.

Il existe actuellement plusieurs versions du PUE [31]. Le PUE v0 se base sur des mesures de puissance instantanée en kW : La mesure IT se fait au niveau des onduleurs et la mesure totale se fait au niveau du transformateur. Les PUE v1, v2 et v3 se basent sur des mesures d'énergie cumulé sur 12 mois. Le PUE v3 prend en compte toutes les pertes puisqu'il mesure la consommation IT au niveau des prises des serveurs et la consommation totale au niveau du poste Haute Tension A (HTA). Il est aussi préconisé de calculer le PUE par zone du datacenter afin d'identifier l'endroit où l'énergie est consommée et ainsi améliorer le PUE global.

"Green Grid" propose également deux indicateurs pour améliorer la durabilité des datacenters en calculant le rendement carbone et la consommation d'eau des infrastructures énergétiques qu'ils contiennent. Le CUE [3] permet d'extrapoler un volume d'émissions de gaz à effet de serre (GES) à partir de la consommation électrique du datacenter. On le calcule en divisant le total des émissions d'équivalent CO2 (kg CO2eq) par le total de la quantité d'énergie consommée (kWh) par le datacenter. Le WUE [2] combine la quantité totale d'eau consommée par un datacenter et la consommation d'eau des activités strictement informatiques.

L'efficacité est évidement une composante importante. Il est une autre, com-

plémentaire et tout aussi importante, mais beaucoup bien moins bien maîtrisée, il s'agit du rendement. Même si d'importants progrès ont été réalisés en la matière par les constructeurs, la consommation des serveurs n'est pas directement proportionnelle à leur taux d'activité. Ainsi, le rendement énergétique d'une ressource informatique dont le taux d'occupation est faible est bien plus important que celle d'une ressource utilisée à plein régime. Alors que le PUE est l'indicateur incontesté pour mesurer l'efficacité énergétique, il n'y a pas encore d'indicateur de mesure qui fasse l'unanimité pour le rendement informatique. La difficulté est d'obtenir un consensus sur une métrique qui soit à la fois simple à collecter et représentative des différents cas d'usage. Le projet EnergeTIC [24] propose un nouvel indicateur de mesure d'efficience énergétique : l'EUE_{CPU} (EnergeTIC Usage Effectiveness pour le rendement de l'IT) qui est un indicateur qui représente le rendement des serveurs, obtenu en divisant le PUE par la charge moyenne de l'IT en marche (Équation 2.1). Cet indicateur est pertinent pour la mesure du rendement et les prises de décision d'optimisation, tout en restant assez simple à mettre en oeuvre puisqu'il se base sur le PUE.

$$EUE_{CPU} = PUE/\%CPU \tag{2.1}$$

Pour réduire la facture électrique d'un Green datacenter et utiliser au mieux les ressources, plusieurs solutions ont été proposées. Nous avons étudiés les solutions dans le contexte de Cloud Computing et de l'optimisation du système de refroidissement.

2.2.2 Cloud Computing et gestion des ressources

Dans un datacenter, la charge réelle des applications est très dynamique. Dans le Cloud Computing, l'utilisation des VM permet d'isoler les applications et offre la possibilité de faire de la consolidation afin de réduire le gaspillage des ressources physiques. Cela permet de réduire la consommation tout en conservant la même qualité de service.

La virtualisation a été introduite comme solution pour répondre à l'augmentation de la demande de puissance de calcul en consolidant plusieurs VMs sur un même serveur. Cela permet de réduire la consommation électriques des datacenters

et l'émission de CO2 avec une politique de gestion des datacenters virtualisés en consolidant les machines virtuelles par migration à chaud et en arrêtant les serveurs inactifs tout en gardant un niveau de service acceptable.

Beaucoup de travaux de recherche se sont focalisés sur la réduction de la consommation d'un datacenter dans un contexte de Cloud Computing [5]. Les serveurs physiques consomment la majeur partie (52-56%) [41, 62]. ElRedhanne et al. [66] proposent une solution dynamique de consolidation de machines virtuelles sur un minimum de serveurs physiques, ce qui adapte automatiquement la taille du Cloud. Cette solution est basée sur Entropy, un outil Open Source de consolidation. VMWare intègre nativement dans son outil de virtualisation VSphere [54], la solution DRS/DPM [32] qui permet de consolider les VMs, tout en arrêtant les serveurs inutiles. D'autre part, [24] propose une solution similaire basée sur la prédiction de la charge informatique en fonction de l'historique. La prédiction permet de mieux gérer les pics de charge et prévoir les ressources à l'avance pour éviter la violation du SLA.

Dans [79], une nouvelle approche est proposée pour réduire encore plus les ressources utilisées. La consolidation dynamique des services permet de déplacer des services appartenant à des applications différentes, sur une même VM. La consolidation des services, couplée à la consolidation de VM peut être utilisée dans un Cloud privé pour réduire la consommation électrique en réduisant le nombre de serveurs physiques démarrés. La solution a été évaluée sur deux applications distinctes et a démontré un gain pouvant aller jusqu'à 40%.

Le temps de déploiement et de migration des machines virtuelles a été toujours un point faible dans les algorithme de consolidation, ce qui peut dégrader la qualité du service durant ces processus. Docker [23], la nouvelle solution Open Source de déploiement des applications dans des containers logiciels, sur n'importe quel serveur Linux, permet de remplacer les solutions de virtualisation actuelles, avec un temps de déploiement relativement très court. Le déploiement d'applications distribuées devient plus simple et nous pouvons même déployer Docker sur des machines virtuelles.

2.2.3 Gestion optimale des Facilités

L'infrastructure de refroidissement et d'alimentation électrique représente aujourd'hui souvent plus de la moitié de la consommation totale d'un datacenter

[41, 62]. Avec l'augmentation de la concentration des serveurs physiques par m^2 (un châssis de 10 lames occupe 4U), les besoins de refroidissement augmentent constamment. L'amélioration de l'efficience énergétique d'un datacenter consiste donc à réduire au maximum la consommation énergétique des infrastructures, pour réduire donc le PUE. Le choix des systèmes de refroidissement lors de la conception d'un datacenter est capital et fait partie du calcul du CAPEX. Un refroidissement à l'eau permet d'absorber plus de calories que l'air, mais l'emplacement d'un datacenter dans des zones froides donne une préférence au refroidissement à l'air. La réduction du coût de refroidissement et donc de la facture énergétique d'un datacenter, permet de réduire efficacement son PUE.

L'optimisation du système de refroidissement dans un datacenter, afin de réduire le coût du refroidissement, a amené à beaucoup de travaux de recherche. Certaines solutions se basent sur la variation de la température de l'eau ou de l'air dans la salle serveur pour réduire le coût du refroidissement. D'autres solutions sont couplées à la consolidation des serveurs pour réduire la dissipation de la chaleur. D'autres approches plus globales ont pour objectif de réduire la consommation de l'infrastructure de refroidissement en changeant régulièrement de système de refroidissement en fonction des températures externes :

Shaoming et al. [13] discutent l'impact de la consolidation des serveurs et de la variation de la température de l'allée froide des salles serveurs, sur la consommation totale du datacenter. Ils se sont focalisés sur l'optimisation de la consommation de l'infrastructure de refroidissement et des coûts de la maintenance des équipements informatiques. L'augmentation de la température de la salle de 1°C peut réduire la consommation du refroidissement de 2 à 5%, pourtant les températures élevées réduisent la fiabilité des composants électroniques (augmenter la température de 10°C au-delà de 21°C, réduit de 50% la fiabilité des composantes électronique) et augmentent le risque de panne, ce qui fait par conséquent grimper le coût de la maintenance des processeurs et de la mémoire. De même, la consolidation qui consiste à mettre en place des cycles de démarrage/arrêt des serveurs, augmente le coût du refroidissement et en même temps réduit la durée de vie des disques durs, sachant que le disque est le composant le plus fragile. C'est une problématique à double tranchant, puisque toute optimisation peut avoir un impact sur un des composants électroniques d'un serveur (disque, CPU, mémoire...). Malgré les risques liés, les gains annoncés dans ce papier sont de l'ordre de 18% avec une

garantie du temps réponse de 99% des requêtes.

Tandis que dans l'article précédent ils se sont focalisés sur l'augmentation de la température dans la salle serveurs, Jungsoo et al. [44] proposent un système d'optimisation du refroidissement des datacenters équipés de systèmes de refroidissement hybrides. Ils utilisent la consolidation des serveurs en se basant sur la variation de la charge des serveurs en fonction du temps et les conditions climatiques externes pour réduire la consommation électrique totale du datacenter. Ce système est basé sur l'utilisation maximale du refroidissement par FreeCooling et la limitation du nombre de transitions entre les différents systèmes de refroidissement pour éviter de les endommager. Les expérimentations réalisées montrent un gain de plus de 25% par rapport à un système basé uniquement sur la température externe pour démarrer le système de FreeCooling.

Le Framework proposé dans [72] permet une gestion dynamique de la température basé sur le placement asymétrique de la charge qui permet une distribution uniforme de la température et réduit la concentration de la température dans des endroits. Ce système réagit rapidement en fonction de l'urgence de la variation de la température, réduit la consommation d'énergie ainsi que les coûts d'investissement CAPEX des systèmes de refroidissement et améliore la fiabilité des équipements. Ce Framework n'est pas liée à la capacité des systèmes de refroidissement. Ce Framework permet de gérer des scénarios catastrophe de panne partielle du refroidissement en grâce au placement dynamique de la charge. Les auteurs démontrent une économie allant jusqu'à 14%.

2.2.4 Prédiction de l'activité d'un datacenter

Avec l'optimisation du rendement des ressources informatiques actives, l'infrastructure est moins prête à répondre aux demandes imprévues. L'initialisation de nouvelles ressources pour répondre à des pics de charge est une opération qui prend un certain temps, avec le risque de dégrader le SLA du datacenter. La prédiction permet d'anticiper les montées en charge, et permet donc de réduire le risque de ne pas savoir répondre à cette charge croissante sans dégrader la qualité de service.

2.2.4.1 Méthodes de prédiction de charge

Suivant les applications hébergées sur les serveurs, la charge présente une périodicité caractéristique (la journée, la semaine...). Il est donc important d'utiliser un algorithme prédictif capable de donner l'évolution de cette charge pour les heures ou les jours à venir. Cette prédiction est basée sur l'historique des mesures présenté sous forme d'un série temporelle. Une série temporelle, ou série chronologique, est une suite de valeurs numériques représentant l'évolution d'une quantité spécifique au cours du temps.

Dans [21], la méthode Seasonal Autoregressive Integrated Moving Average (SARIMA) permet d'avoir de très bons résultats de prédiction de la charge des serveurs. La série temporelle correspondent à la charge CPU du serveur toutes les 30 minutes, pendant une semaine. L'étude d'un trafic réel provenant d'une application en production permet de voir la périodicité de la charge, ce qui permet de faire des prédictions sur des périodes plus longues. La fréquence des mesures peut être d'une granularité fine, ce qui permet d'avoir des prédictions sur des périodes plus petites (1 minute). Le taux d'erreur moyen est de l'ordre de 11%, comparé aux valeurs réelles.

Le module de prédiction, développé dans EnergeTIC [24] permet lui aussi d'imaginer des développements futurs. Les prévisions de charge sont élaborées à partir des observations passées. L'utilisation de ces mesures "historisées" permet d'identifier les journées types et d'en prédire la journée de même nature à venir. Les statistiques des journées similaires peuvent être exploités pour déterminer la probabilité qu'un seuil de charge IT soit dépassé pour une heure donnée. Le seuil en question pourra être défini comme le seuil au-delà duquel un serveur doit être remis en service. On pourra alors anticiper l'heure idéale de démarrage des équipements, sans risque de compromettre la qualité de service.

Un datacenter héberge des applications très hétérogènes, mais la charge globale est périodique. Nous pouvons mesurer des différences entre les jours ouverts et les week-ends, ainsi que des pics de charge réguliers sur chaque journée. La prédiction de la charge peut être appliquée sur la globalité de l'activité du datacenter pour prédire sa charge future et donc estimer sa consommation. Cela permet une utilisation plus efficiente des ressources tout en respectant la qualité du niveau de services.

2.3 Le traitement des données Big Data

Avec l'augmentation exponentielle du volume des données des SI, nous avons vu apparaître le terme "grosses données" ou "Big Data" qui désigne un ensemble de données très volumineuses (plusieurs Péta-Octets) et très difficile à traiter avec des outils traditionnels de gestion des données. Le nouveau défi auquel sont confrontés les entreprises sont la vitesse et le coût de traitement de ces données. Les perspectives du traitement et de l'analyse rapide des données Big Data sont énormes pour les entreprises. Le Big Data permet de réduire les risques grâce à l'apprentissage et permet de faire de l'analyse prédictive afin de faciliter la prise de décision.

Aujourd'hui, grâce à la virtualisation et à l'augmentation de la concentration de la puissance informatique, les gestionnaires des datacenters gèrent un nombre de plus en plus important de capteurs. Les entreprises cherchent également à collecter un grand nombre de mesures avec une faible granularité, pour détecter les défaillances en temps réel et faire l'analyse détaillée de leur infrastructure. Le volume de données produit devient énorme et nécessite des outils de calcul distribué à visée analytique.

Dans le Big Data nous parlons essentiellement des 3V :

- **Volume :** Le volume décrit la quantité des données (plusieurs dizaines de téraoctets) générées.
- **Variété :** La variété des données est un nouveau défi auquel font face les datacenters. Les données collectées sont brutes (xml, csv, Json...) et parfois non structurées, provenant de sources très hétérogènes (Open Data, internet des objets, Web...).
- **Vélocité :** Cela représente la fréquence à laquelle les données sont générées et partagées. Dans certains cas, l'accès et le partage doivent se faire en temps réel.

Aujourd'hui, à ces 3V s'ajoutent aussi la **Véracité** des données (précision, validité et valeur des données collectées) et la **Valeur** des données pour le client et pour l'entreprise.

Hadoop [33] est l'une des premières solutions de traitement de données Big Data.

2.3.1 Hadoop

Tout a commencé en 2004, avec l'article publié par Google sur son algorithme de calcul de gros volumes de données "MapReduce" [20], ainsi que son système de fichiers distribué "GoogleFS" (Google File System) [29]. Plus tard, en 2005, une version Open Source inspirée de la publication de Google a vu le jour, qui deviendra le projet Hadoop [33]. Hadoop est composé du système de fichiers Hadoop Distributed File System (HDFS) et d'une implémentation complète de l'algorithme de calcul MapReduce.

D'après le cabinet IDC [37], le marché de Hadoop et MapReduce devrait croître de plus de 60% par an jusqu'en 2016 pour s'établir à presque 813 millions de dollars à cette date. Le cabinet estime toutefois que cette progression sera limitée, d'un point vue économique, par la pénurie de compétences et par une concurrence acerbe entre Open Source et éditeurs propriétaires.

2.3.1.1 HDFS

Inspiré du GoogleFS [20], HDFS [83] est un système de fichiers distribué conçu pour stocker de très gros fichiers (plusieurs centaines de giga-octets) de données, sur des serveurs peu onéreux et peu performants, avec le principe d'écrire une fois et lire plusieurs fois. Ce système de fichier peut tourner sur des clusters avec une forte probabilité de défaillance des noeuds. Il permet l'abstraction de l'architecture physique du stockage, afin de manipuler un système de fichiers distribués comme s'il s'agissait d'un disque dur unique, ce qui permet d'avoir de gros fichiers qui dépassent la taille d'un disque.

Dans un cluster HDFS, les données sont découpées et distribuées sous forme de blocs de données selon deux paramètres :

- Taille des blocs : Chaque fichier de données est découpé en plusieurs blocs de taille réduite, généralement 64 ou 128 Mo. Par exemple, un fichier de 1Go sera divisé en 8 blocs. Cela permet d'accélérer le calcul avec des fichiers de taille plus petite.

- Nombre de réplications : C'est le nombre de copies d'un même bloc de données, réparties sur les différents noeuds du cluster, pour la tolérance aux pannes en cas de perte d'un noeud du cluster.

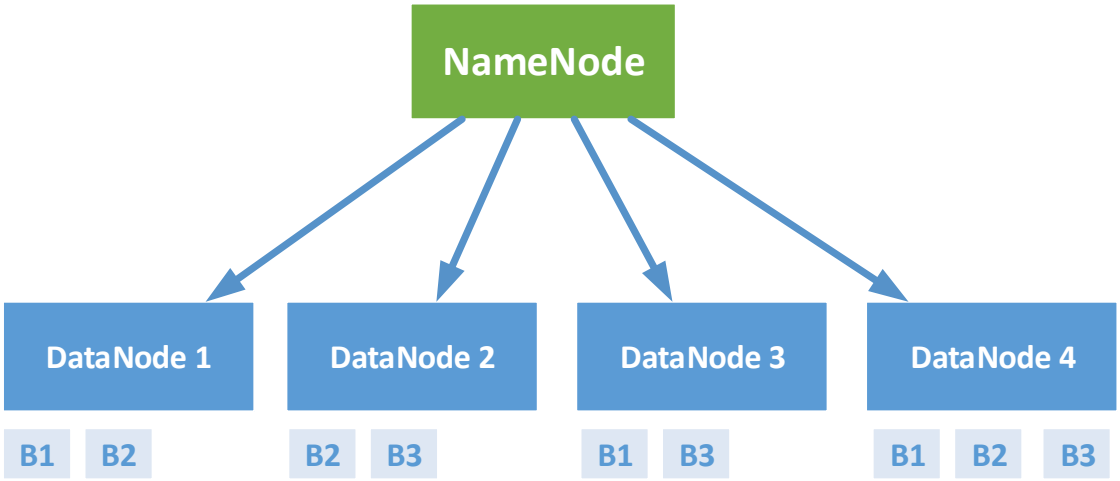


Figure 2.1 – Architecture du système de fichiers HDFS

L'architecture d'un cluster HDFS [83] repose sur deux composants (Figure 2.1) :

- NameNode : C'est le noeud master unique de l'architecture qui est responsable de la localisation des données sur le cluster. Il gère les arborescences ainsi que les méta-données des fichiers et des répertoires. Le NameNode constitue le point d'entrée du cluster, sans lui le système de fichiers ne fonctionne pas.

- DataNode : Ce sont les noeuds qui stockent les données et les restituent. Ces noeuds communiquent régulièrement au NameNode la liste des blocs de données qu'ils hébergent. La réplication des données entre les DataNodes se fait d'une manière asynchrone. La perte d'un DataNode n'entraîne pas la perte des données qu'il contient.

2.3.1.2 MapReduce

MapReduce [83] est un Framework de calculs parallèles et souvent distribués, de données très volumineuses. Ce Framework se base sur la décomposition d'un calcul important en plusieurs tâches plus petites, qui seront distribuées sur un cluster de machines pour être traitées et qui produiront chacune une partie du résultat final.

Le noeud maître de l'architecture MapReduce est le "JobTracker" (Figure 2.2), et les noeuds esclaves sont les "TaskTracker". Le JobTracker s'occupe de la coordination des différents calculs en ordonnançant les tâches qui s'exécutent sur les TaskTrackers. Ces derniers exécutent les tâches et envoient un état d'avancement au JobTracker, qui contient un registre de suivi de tous les calculs en cours. Si une tâche échoue, le JobTracker peut la replanifier sur un autre TaskTracker.

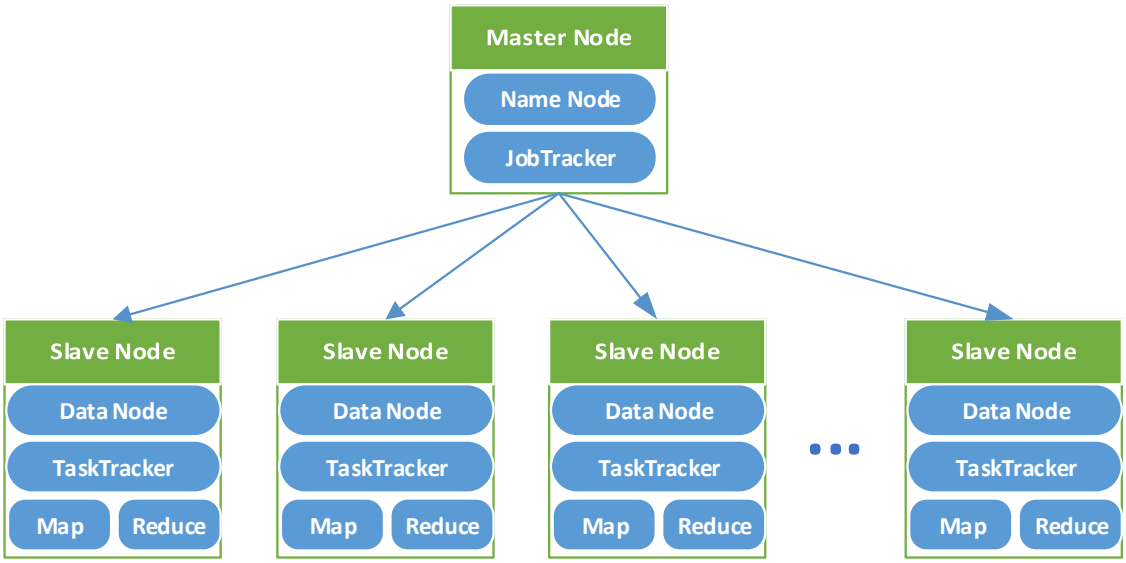


Figure 2.2 – Architecture Hadoop MapReduce

MapReduce décompose le calcul en deux phases (Figure 2.3) : La phase Map et la phase Reduce. Chaque phase possède des paires de clés-valeurs en entrée et en sortie, avec des fonctions de Map et de Reduce spécifiées par le programmeur. La phase Map permet de découper le problème en sous problèmes, et les délègue à d'autres noeuds. L'ensemble des résultats obtenus de la phase Map, seront traités dans phase reduce (agrégation, filtrage...) qui associe les valeurs correspondantes à chaque clé.

MapReduce est une très bonne solution pour les traitements à passe unique mais n'est pas la plus efficace pour les cas d'utilisation nécessitant des traitements et algorithmes à plusieurs passes. Chaque étape d'un flux de traitement étant constituée d'une phase de Map et d'une phase de Reduce, il est nécessaire d'exprimer tous les cas d'utilisation sous forme de programmes MapReduce. Les données résultant de l'exécution de chaque étape doivent être stockées sur un système

de fichiers distribués avant que l'étape suivante commence. Cette approche a tendance à être peu rapide à cause de la réplication et du stockage sur disque. Chacun des tâches présentant une latence élevée et aucun ne pouvant commencer avant que le précédent n'ait tout à fait terminé.

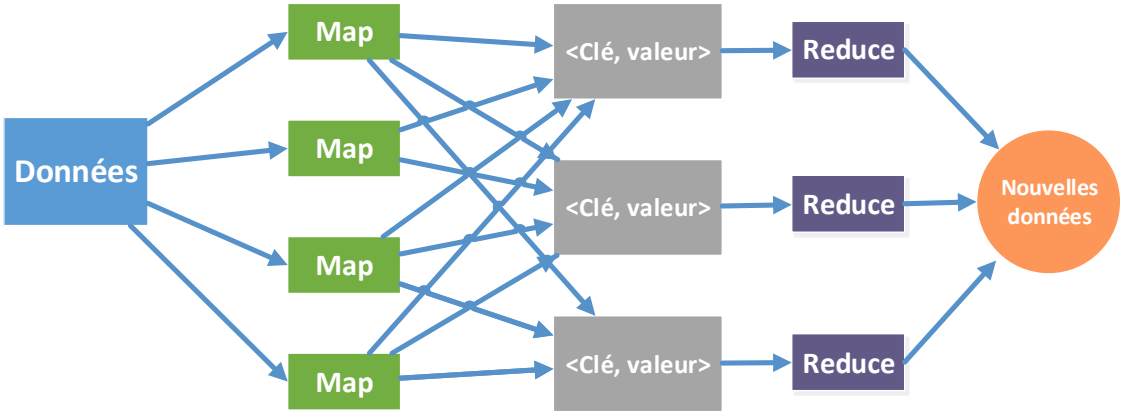


Figure 2.3 – Fonctionnement de MapReduce

2.3.1.3 YARN

Yet Another Resource Negotiator (YARN) [84] ou MapReduce 2.0 (MRv2), est le successeur de MapReduce. L'idée fondamentale de YARN est la séparation des deux fonctionnalités majeures du JobTracker (Figure 2.4) :

- La gestion du cluster et des ressources (Resource Manager – RM).
- Le pilotage de l'exécution des tâches et des applications (Application Master – AM).
- **Resource Manager :** C'est essentiellement un ordonnanceur. Il arbitre l'attribution des ressources disponibles dans le cluster en fonction des demandes concurrentes. Il optimise l'utilisation du cluster (garder une utilisation permanente des ressources) selon les contraintes définis de SLA.
- **Application Master :** est un Framework disposant d'une librairie spécifique qui négocie les ressources RM avec le NodeManager (NM) afin d'exécuter et de surveiller les tâches. Le principe est que l'AM négocie avec le RM des ressources sur le cluster, décrites sous forme de nombre de

containers, avec une limite mémoire pour chacun de ces containers, et lance ensuite les tâches de cette application dans ces containers.

Les containers sont supervisés par les NodeManagers qui sont exécutés sur les noeuds de calcul, ce qui permet de limiter l'utilisation des ressources par rapport à celles allouées. YARN est compatible avec les anciennes versions de MapReduce.

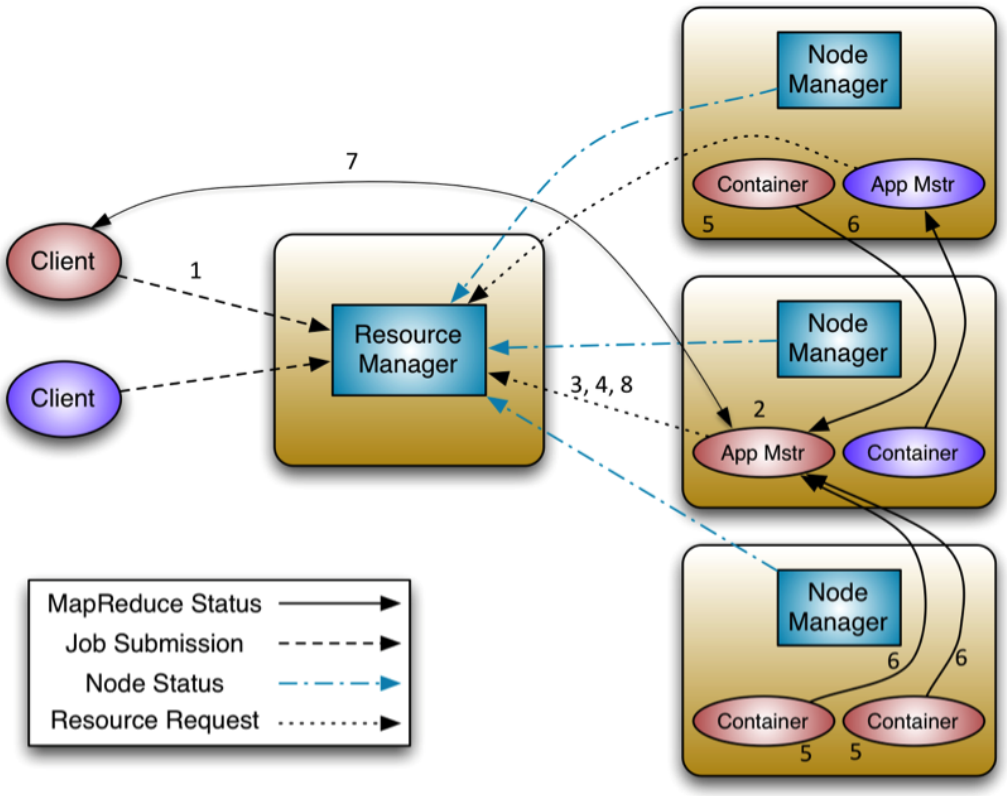


Figure 2.4 – Architecture Yarn (MRv2) - (http ://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html)

2.3.1.4 Spark

Spark [75] est un Framework Open Source de calcul distribué en mémoire (in-Memory), contrairement à MapReduce qui utilise les disques. Cela permet

d'atteindre des performances 100 fois supérieures. Il peut être utilisé en mode Standalone ou utiliser Hadoop YARN pour la gestion de la distribution du calcul.

Spark permet de développer des pipelines de traitement de données complexes, à plusieurs étapes, en s'appuyant sur des graphes orientés acycliques (DAG). Spark permet de partager les données en mémoire entre les graphes, de façon à ce que plusieurs jobs puissent travailler sur le même jeu de données. Spark s'exécute sur des infrastructures HDFS et propose des fonctionnalités supplémentaires. Cette solution n'a pas été prévue pour remplacer Hadoop mais pour mettre à disposition une solution complète et unifiée permettant de prendre en charge différents cas d'utilisation et besoins dans le cadre des traitements Big Data.

Spark est constitué de plusieurs composants (Figure 2.5) :

- Spark Core : C'est le coeur du projet Spark, il contient les fonctionnalités de base comprenant l'ordonnanceur de taches, le gestionnaire de mémoire, l'accès aux systèmes de stockage de données... Les données utilisées dans Spark Core sont stockées sous forme de "Resilient Distributed Datasets" (RDD). Nous pouvons assimiler un RDD comme une table dans une base de données. Les RDD permettent de réarranger les calculs et d'optimiser le traitement. Ils sont aussi tolérants aux pannes car un RDD sait comment recréer et recalculer son ensemble de données. Pour obtenir une modification d'un RDD, il faut y appliquer une transformation, qui retournera un nouveau RDD, l'original restera inchangé.

- Spark Streaming : Peut être utilisé pour le traitement temps-réel des données en flux. Il s'appuie sur un mode de traitement en "micro batch" et utilise pour les données temps-réel DStream, c'est-à-dire une série de RDD (Resilient Distributed Dataset).

- Spark SQL : Il permet d'exposer les jeux de données Spark via API JDBC et d'exécuter des requêtes de type SQL en utilisant les outils BI et de visualisation traditionnels. Spark SQL permet d'extraire, transformer et charger des données sous différents formats et les exposer pour des requêtes ad-hoc.

- Spark MLlib : MLlib est une librairie de "Machine Learning" qui contient tous les algorithmes et utilitaires d'apprentissage classiques, comme la

classification, la régression, le clustering, le filtrage collaboratif, la réduction de dimensions, en plus des primitives d'optimisation sous-jacentes.

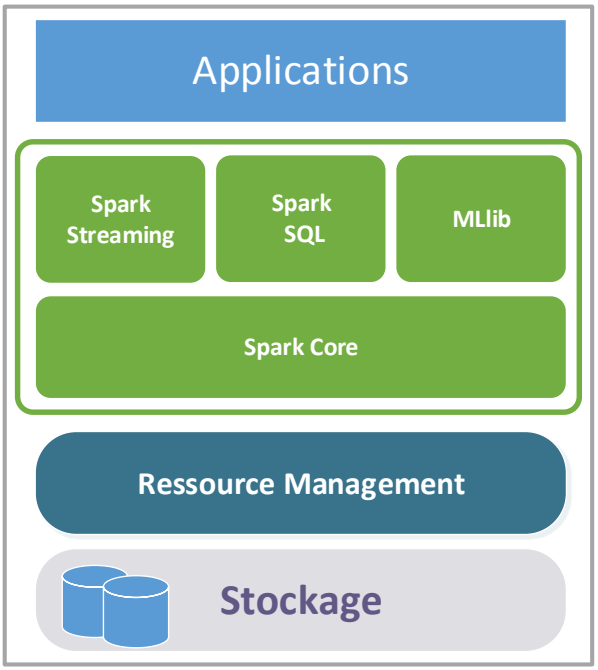


Figure 2.5 – Composants Apache Spark

2.3.2 Distributions Hadoop

Il existe plusieurs distributions qui se basent sur Apache Hadoop et fournissent un package complet des composants de l'éco-système Hadoop avec une facilité d'installation et d'administration. Les distributions majeures sont :

- Cloudera : Cloudera [14] a été fondée par des experts Hadoop en provenance de Facebook, Google, Oracle et Yahoo. Leur solution est basée sur Apache Hadoop et complétée par des logiciels propriétaires pour l'installation automatisée et la gestion du cluster. Cloudera propose une version entièrement Open Source avec quelques limitations.
- HortonWorks [35] : Cette entreprise a été formée en Juin 2011 par des membres de l'équipe Yahoo en charge du projet Hadoop. Tous les composants sont en Open Source sous licence Apache. HortonWorks est un grand

contributeur Hadoop et propose une distribution 100% conforme à la version d'Apache Hadoop. En plus, HortonWorks utilise Ambari pour l'installation automatique du cluster Hadoop et la gestion centralisée du cluster.

- MapR [56] : MapR a été fondée en 2009 par d'anciens membres de Google. MapR propose son propre système de fichiers distribués et une version modifiée de MapReduce : MapR FS et MapR MR. MapR annonce de meilleures performances et a été entièrement optimisé pour HBase [28]. L'installation d'un cluster MapR est essentiellement manuelle, et il n'y a pas de système de déploiement fourni.

- Intel Hadoop : Cette distribution a été fournie par Intel et a été optimisée pour les processeurs Intel pour maximiser les performances. Finalement, cette version a été abandonnée et substituée par un accord avec Cloudera.

2.3.3 Configuration optimale des traitements Hadoop

Le Framework MapReduce comporte plus d'une centaine de paramètres de configuration. La performance d'un traitement est très sensible aux valeurs de ces paramètres. Étant donné un traitement P, une ensemble de données d et un cluster r, le but est de trouver la configuration optimale $c_{optimal}$ pour ce traitement. Il existe au moins deux études existantes pour l'optimisation des calculs MapReduce en explorant l'ensemble des paramètres de configuration. Le problème a été abordé par Babu et al. in [34] et par Wilke et al. in [52].

Pour un traitement P donné et une configuration initiale des paramètres c, Babu et al. commencent par exécuter le job P avec la configuration C et lui donne un profil. Ils mesurent ensuite d'une manière empirique plusieurs statistiques sur les traitements et les flux de données, tel que le nombre moyen des pairs de clés valeurs émis par chaque Map, le temps de traitement d'un Map sur chaque noeud... Les statistiques obtenues de l'étape de profilage seront utilisés pour prédire la performance de P sur un point aléatoire dans l'espace de paramètres c'. Les statistiques sont utilisées pour estimer le temps total d'exécution de $< P, c' >$. La configuration optimale obtenue sera $c_{optimal}$.

[52] propose un algorithme appelé Gunther/Active Tuner pour obtenir la configuration optimale de l'ensemble des paramètres. Gunther commence par choisir M configurations aléatoires de l'ensemble de paramètres. Le traitement

P sera exécuté avec l'ensemble des configurations M. Par la suite, Active Tuner adopte une approche génétique de l'algorithme dans lequel il maintient une liste de configurations L déjà évaluées. Dans cet ensemble de configurations, certaines des meilleures configurations seront choisies pour générer de nouvelles configurations enfants, en croisant des paramètres provenant de configurations parentes différentes. Chaque configuration fille sera évaluée pour mesurer sa performance en exécutant le traitement P avec sa configuration. L'algorithme a été conçu pour fonctionner pendant un maximum de N tours, où chaque tour génère M nouvelles configurations. Ce processus sera répété jusqu'à ce que les critères d'arrêt soient satisfaits, ce qui présente un temps de validation relativement long.

Ces travaux proposent un grand apport pour le choix des paramètres très complexes et nombreux du Framework MapReduce. Cependant, les évaluations ont été réalisées sur des plateformes statiques, et il aurait été intéressant de proposer une approche avec un cluster élastique qui prend en compte le gain/perte de l'ajout/suppression des noeuds du cluster. D'autre part, garder une base de données de l'historique des exécutions permettrait de comparer les nouveaux traitements à l'existant pour fournir des résultats plus rapides

2.4 Le traitement temps réel des flux Big Data

Le Big Data peut être traité de deux moyens différents. Dans le traitement en mode lots (Batch), les données Big Data sont stockées dans des systèmes de stockage distribués ou dans d'énormes bases de données. Les calculs seront lancés sur des clusters de calculs distribués tel que MapReduce [83]. La durée de calcul peut être longue, selon la taille des données et les caractéristiques du cluster. Dans certains contextes, les données doivent être traitées en temps réel, c'est le cas par exemple dans la finance où l'information a une durée de vie très limitée. D'où le deuxième mode de traitement Big Data : Le traitement de flux.

Les plateformes de traitement de flux permettent d'analyser les données au fil de l'eau, ce qui offre une meilleure réactivité. Plusieurs solutions de traitement de flux Big Data existent : Yahoo S4, Spark Streaming [75], Apache Storm [76]. La solution Apache Storm développée initialement par Twitter est la plateforme de traitement de flux la plus utilisée.

2.4.1 Apache Storm

Apache Storm [40,76] est un Framework distribué de traitement de flux scalable, performant (plus de 100 millions de messages de 100 octets par seconde et par noeud) et tolérant aux pannes. Il permet de déployer, de définir et de déployer des chaînes de traitement appelées "Topologies" (Figure 4.5). Une topologie est un graphe de calcul : chaque noeud de la topologie contient une logique de traitement et des liens d'échange de données entre les noeuds. La topologie Storm est composée de deux types de composants : Les "Spouts", sources de flux, qui permettent de récupérer les données depuis une ou plusieurs sources et les formate sous forme de tuples (liste d'objets), et les "Bolts" qui consomment les flux en entrées, lancent des traitements, et émettent de nouveaux flux. Il est possible de définir le degré de parallélisme d'un composant et la stratégie de routage des flux de données.

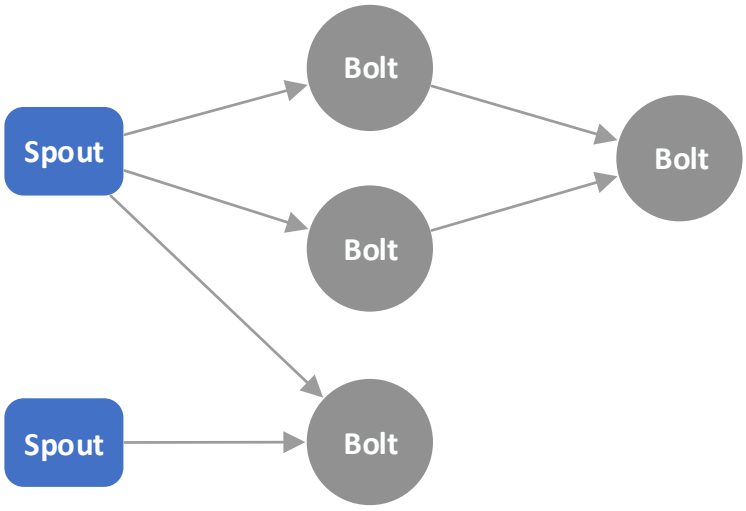


Figure 2.6 – Exemple de topologie Storm

2.4.1.1 Architecture Storm

Storm possède un noeud central appelé "Nimbus", auquel la topologie est soumise, c'est l'équivalent d'un JobTracker dans le Framework MapReduce. Ce noeud est responsable de la distribution du code dans le cluster. Il assigne les tâches aux workers et supervise les "Supervisors". Les "Supervisors" reçoivent les tâches envoyées par le Nimbus à travers Zookeeper et gèrent l'arrêt/démarrage

des processus. Zookeeper gère la coordination entre le Nimbus et les Supervisors. Les Nimbus et les Supervisors sont sans état, les états sont stockés dans le cluster Zookeeper ou sur le disque local. Un cluster Storm est tolérant aux pannes, la perte d'un noeud n'entraîne pas la panne de tout le système. Même si le noeud Nimbus tombe, le cluster Storm continue à fonctionner normalement. La figure 2.7 représente l'architecture d'un cluster Storm et ses différents noeuds.

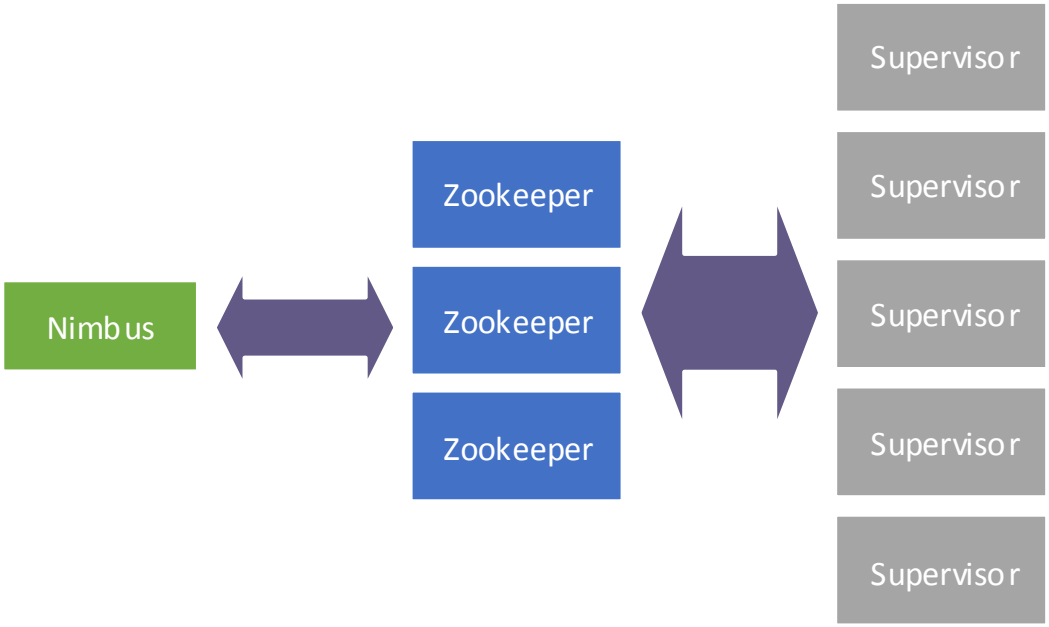


Figure 2.7 – Architecture d'un cluster Storm

2.5 Détection des anomalies

Le meilleur moyen de gérer efficacement un datacenter, est de détecter les dysfonctionnements au plus tôt et d'une manière intelligente. Les outils de supervision mis en place dans la plupart des datacenter permettent de détecter rapidement les anomalies simples : dépassement de seuil bas ou haut, changement d'état... Les anomalies de fonctionnement complexes, qui lient plusieurs mesures ou indicateurs, est plus difficile à calculer et demande un temps de calcul plus long. Augmenter la vitesse de traitement des données afin d'agir plus vite lorsqu'une alerte de fonctionnement est détectée est primordial, même pour les anomalies complexes.

Avec un service à la demande et un risque associé à la migration des services vers le cloud, les clients cherchent à avoir plus de garanties. D'où l'importance des efforts dans la définition des niveaux de sécurité de service (SLA Sécurité). Nous nous intéressons à la détection des anomalies de fonctionnement du réseau et notamment les attaques de type Déni de service distribué (DDoS). Les méthodes de détection d'intrusions utilisées à l'heure actuelle reposent essentiellement sur la mise en place d'outils d'observation d'événements et leur analyse à l'intérieur du datacenter. Elles sont basées sur la surveillance des systèmes et de leur environnement réseau proche. Elles ne s'attachent que peu à la corrélation entre les menaces connues et les événements externes : menaces publiques sur des réseaux sociaux, géolocalisation d'une "IP" dans un pays à risque...

De nos jours, le trafic réseau d'un datacenter est de plus en plus important, nous passons d'un réseau 1Gb/s à un réseau de plusieurs Gb/s. A cause de la forte augmentation du volume de trafic, la taille des fichiers de traces a considérablement augmenté, et les techniques de fouille de traces actuelles qui permettent de détecter les attaques et identifier les attaquants et les localiser, ne sont pas capables de gérer toutes ces données. De plus, lors d'une attaque DDoS, le trafic augmente fortement et peux entraîner la saturation de l'outil de détection.

2.5.1 Détection des attaques de type DDoS

Un système de détection d'intrusion (Intrusion Detection System (IDS)) est un système permettant de détecter des fonctionnements anormaux ou suspects sur la cible supervisée. Il existe trois grandes familles distinctes d'IDS : Les NIDS (Network Based Intrusion Detection System) qui surveillent l'état de la sécurité au niveau du réseau [59, 73, 82], les HIDS (HostBased Intrusion Detection System) qui surveillent l'état de la sécurité au niveau des hôtes et les IDS hybrides, qui utilisent les NIDS et HIDS pour avoir des alertes plus pertinentes. Les HIDS sont particulièrement efficaces pour déterminer si un hôte est contaminé et les NIDS permettent de surveiller l'ensemble d'un réseau de datacenter contrairement à un HIDS qui est restreint à un hôte ou une application.

La collecte d'informations constitue la première étape dans tout système de détection d'intrusions. Les informations collectées sont fournies par : le journal système, des journaux propres à certaines applications ou des "sondes" installées par les outils de détections eux-mêmes. Les outils de détection d'intrusions ana-

lysent ensuite cette masse d'informations de manière à y détecter de potentielles intrusions. Deux principales approches sont à ce jour utilisées : l'approche comportementale et l'approche par scénario. La première se base sur l'hypothèse que l'on peut définir un comportement "normal" de l'utilisateur et que toute déviation par rapport à celui-ci est potentiellement suspecte. La seconde s'appuie sur la connaissance des techniques employées par les attaquants : on en tire des scénarios d'attaques et on recherche dans les traces d'audit leurs éventuelles occurrences.

- Approche comportementale : La détection d'anomalies consiste à définir, dans une première phase, un comportement normal du système, des utilisateurs et des applications. Dans une seconde phase, l'entité modélisée est observée et tout écart par rapport au comportement de référence est signalé comme suspect. Cette approche regroupe en fait deux problèmes distincts : la définition du comportement "normal" (profil) et la spécification des critères permettant d'évaluer le comportement observé. La mise en service d'un détecteur d'anomalie est précédée d'une phase d'apprentissage au cours de laquelle le profil initialement construit évolue, afin que toute utilisation jugée normale soit reconnue comme telle. Dans certains cas, cet apprentissage continue également après la mise en service ; le profil évolue constamment afin de suivre au mieux l'utilisation réelle du système. Les différentes approches de détection d'anomalies se distinguent essentiellement par le choix des entités modélisées dans le profil et l'interprétation des divergences par rapport à ce profil. (Deux cas typiques : la détection probabiliste et la détection statistique).

- Approche par scénario : Cette seconde approche vise à détecter des signes de scénarios d'attaques connus. Le principe consiste à utiliser une base de données contenant des spécifications de scénarios d'attaques bien connus (on parle de signatures d'attaque et de base de signatures). Le détecteur d'intrusions confronte le comportement observé du système (l'audit) à cette base et lève une alerte si ce comportement correspond à l'une des signatures.

Snort [73] est une application Open Source de détection d'intrusion de type NIDS, avec une communauté importante et très active. Snort est capable de faire de l'analyse en temps réel de paquets pour détecter des attaques de type DDoS et d'autres types d'attaques. Ce logiciel sur base sur les règles prédéfinis pour son

analyse. Grâce à sa bibliothèque de signatures (règles) de filtrage du trafic et les règles saisis manuellement depuis son interface, Snort peut analyser efficacement l'ensemble du trafic réseau. Cependant, ces règles doivent être maintenues à jour et adaptées pour chaque environnement : le trafic d'un datacenter est variable et de nouvelles applications sont mises en place régulièrement. Dans [65], une étude statistique sur les alertes détectés par Snort pour identifier le taux de faux positifs. Afin de réduire le taux de faux positifs, l'utilisation des méthodes de prédiction de la charge telle que SARIMA a été proposé comme solution.

Un autre outil de détection d'intrusion de type HIDS : Ossec [10]. Cet outil permet de détecter les intrusions système par analyse de logs locaux système et applicatif. Il permet de détecter la mauvaise utilisation, les violations de politiques et autres formes d'activités inappropriées sur un système (intégrité des fichiers et du registre, anomalies système). Le blocage automatique des accès non légitimes se fait avec "Iptables". Ossec se base également sur des règles prédéfinies et une base de signatures fournies à l'installation.

La plupart des solutions Open Source proposés utilisent la recherche de contenu grâce à l'utilisation des règles prédéfinies, et proposés par défaut. Ces règles contiennent des listes noires d'IPs à bloquer et des règles qui changent régulièrement. La mise à jour régulière de ces règles est indispensable pour le bon fonctionnement du système. Le blocage du trafic intrusif peut être fait d'une manière automatique, mais les pirates peuvent utiliser des IPs très variées et parfois utiliser des IPs innocentes ce qui peut causer des dommages en cas de blocage d'une tel IP. Les systèmes de détection d'intrusion basés sur des règles prédéfinies se basent sur le fait que l'attaquant ne connaît pas le profil d'un trafic normal. Cependant, lors d'attaques DDoS sophistiqués, l'attaquant peut mettre en échec le système de détection en découvrant les règles de filtrage pour ensuite inonder le réseau avec des attaques qui ne seront jamais détectées. Le filtrage par des méthodes statistiques permet de résoudre ce problème puisqu'ils sont très difficiles à identifier.

2.5.2 Méthodes statistiques pour la détection d'intrusion

L'état de l'art des solutions d'analyse de trafic et de détection d'intrusion est très riche. La plupart des méthodes proposées détectent les attaques superficiellement et partiellement en inspectant le trafic sur des liens individuels isolés au

lieu d'une vue globale à travers tout le réseau. Il existe plusieurs types d'attaques réseaux, parmi lesquelles l'attaque DDoS est la plus redoutée. En effet, ce type d'attaque peut être dévastatrice pour l'ensemble des applications d'un datacenter.

Lakhina et al. [47] utilisent l'analyse par composantes principales ACP (Principal Component Analysis (PCA)) pour séparer le trafic légitime et le trafic anormal. Le trafic réseau IP est partagé en deux sous-espaces distincts (normal et anormal), ce qui permet de détecter une anomalie (attaque réseau) quand la magnitude de la projection sur le sous-espace anormal dépasse un certain seuil. Ringberg et al. [67] ont étudié la méthode proposée par Lakhina et ont observé qu'elle est très sensible aux données utilisées pour la construction du modèle. Quand les données d'apprentissage contiennent beaucoup d'anomalies, le sous-espace normal est contaminé. Dans ce cas, Rubinstein et al. [68] ont montré que la méthode ACP est vulnérable aux attaques. Ils ont alors proposé une version robuste [77] du modèle ACP afin de le rendre moins sensibles à la présence d'anomalies dans les données. Puisque l'attaque DDoS est établie lorsque de nombreux robots commencent à inonder les réseaux en renvoyant un grand nombre de paquets, le trafic d'attaque sur des liens multiples tient habituellement les mêmes caractéristiques telle que l'heure de début, l'intensité et le type des paquets. Cependant, le problème principal du ACP traditionnel, comme utilisé aujourd'hui, est qu'il ne prend pas en compte les corrélations de trafic, et le fait qu'en réalité le trafic est corrélé casse l'hypothèse sous-jacente de la méthode de détection des anomalies basée sur ACP. Par conséquent, dans la détection des attaques DDoS, la corrélation du trafic d'attaque est généralement allouée au sous-espace normal. La méthode basée sur le modèle ACP proposé par [77] montre son efficacité pour la détection du trafic non légitime même avec un trafic statique, dynamique ou adaptative. La sensibilité du modèle ACP dépend du nombre de composantes principales calculées et du seuil de détection choisi [68], et la variation de ces limites peut augmenter le taux de faux positif.

Il existe plusieurs études qui utilisent Hadoop et MapReduce pour implémenter des méthodes de détection des attaques DDoS [50, 51]). Par contre, aucune de ces études n'utilise le modèle basé sur ACP. D'autre part, la scalabilité du Framework Hadoop permet de lancer un calcul en mode distribué mais ne permet pas une détection en temps réel des attaques, ce qui ne remplit pas l'objectif principal d'un système de détection d'intrusion.

2.6 Synthèse

L'optimisation de l'efficience énergétique et la réduction de l'empreinte écologique des datacenters Green tout un gardant un niveau de service acceptable, est un sujet qui suscite beaucoup d'intérêt dans le domaine de la recherche. Les travaux déjà effectués se situent dans deux domaines : (i) la réduction de la consommation des serveurs en utilisant la consolidation des machines virtuelles et des applications, pour limiter le nombre de serveurs démarrés et (ii) optimalisation du système de refroidissement pour réduire la consommation du datacenter et optimiser le PUE. Les solutions actuelles d'optimisation du refroidissement se limitent à des types spécifiques de systèmes de refroidissement d'un datacenter. De plus, l'optimisation du refroidissement indépendant de l'activité des équipements informatiques peut entraîner des décisions contradictoires. Certaines contributions combinent des solutions d'optimisation du refroidissement avec des solutions de consolidation pour optimiser les gains [44]. La consommation de la plupart des systèmes de refroidissement dépend de la température extérieure, mais très peu de solutions utilisent les prévisions météorologiques pour estimer les variations de consommation. La prédiction de l'activité des applications a été utilisée pour absorber les pics de charge dans le cas de la consolidation des serveurs, mais cette approche n'a pas été abordée dans l'optimisation du refroidissement.

La détection des anomalies du réseau dans un datacenter en temps réel, permet une meilleure réactivité pour éviter la dégradation du niveau de service. Les outils de détections des attaques réseaux sont nombreux, et parmi eux beaucoup de solutions Open Source. Les majorité des solutions utilisant le filtrage à travers des règles prédéfinies, permettent une détection en temps réel des attaques. L'attaquant est capable de contourner ces outils en envoyant des attaques complexes, susceptibles de découvrir les règles de filtrage. Les travaux de recherches utilisant des modèles statistiques, montrent que ces modèles dont difficiles à contourner. Le modèle est calculé à partir de très gros volumes de données, difficiles à gérer avec les outils d'analyse de données ordinaires. Ces solutions ne sont pas capables de détecter les attaques réseaux en temps réel.

Dans ce travail de thèse, nous nous intéressons à l'optimisation du refroidissement d'un Green datacenter en exploitation, d'une manière réactive et prédictive. En se basant sur des données de prévisions météorologiques et sur un algorithme de prédiction de l'activité du datacenter, nous pouvons absorber les pics de charge

et limiter les transitions entre les différents systèmes de refroidissement. Avec une vue globale de l'activité informatique et des capteurs, notre système offre de meilleurs gains par rapport aux solutions réactives. Dans la deuxième contribution nous utilisons les technologies Big Data de traitement de données pour construire un modèle statistique de détection des attaques réseau de type DDoS, à partir de très gros volumes de donnée. En utilisant des historiques très longues et des mesures à faible granularité, note modèle permet de représenter 100% de la variance des données. Ce modèle sera utilisé dans une topologie Storm [76] pour détecter en temps réel les attaques sur l'ensemble du datacenter. L'architecture proposée utilisant Hadoop et Storm peut être utilisée dans d'autres cas d'usage de détection d'anomalies en temps réel.

Dans les chapitres suivants nous détaillons nos contributions et les résultats des expérimentations.

3

Optimisation du système de refroidissement dans un Datacenter Green

Contents

3.1 Contexte 42

3.2 Approche et contribution 44

3.2.1 Approche 44

3.2.2 Contribution 46

3.3 Le datacenter Green de Mangin 48

3.3.1 Le système de distribution électrique 48

3.3.2 Le système de refroidissement 50

3.3.2.1 Les pompes d'eau de la nappe phréatique . . 51

3.3.2.2 Les groupes de production de froid 53

3.3.2.3 L'ultime secours : l'eau de ville 55

3.3.2.4 Le refroidissement des cubes 55

3.4 Réglementations et contraintes 56

3.5 Méthodologie 59

3.5.1 Étude de l'activité d'un datacenter 59

3.5.2 Prédiction de la charge avec SARIMA 60

3.5.3 Utilisation des prévisions météorologiques 60

3.5.4 Algorithme d'optimisation du refroidissement 63
3.5.4.1 Programme d'optimisation linéaire 63
3.5.4.2 Optimisation réactive et prédictive 66
3.6 Expérimentation 68
3.6.1 Évaluation 69
3.6.1.1 Choix manuel d'un seul système de refroidissement 70
3.6.1.2 Optimisation du refroidissement avec une température peu variable 72
3.6.1.3 Exécution de l'algorithme avec des températures variables 73
3.7 Conclusion 75

Dans ce chapitre, nous présentons notre système d'optimisation du refroidissement d'un datacenter, avec une vue globale, des sources de données externes et un module de prédiction de charge, dans un contexte de "Green Computing" et ciblant différents domaines de gestion (performance, disponibilité, énergie...). Au début, nous présentons le contexte et les motivations autour du sujet d'optimisation, nous détaillons ensuite les contributions et les verrous technologiques rencontrés pour aboutir à cette solution. Finalement, nous validons notre approche à travers un cas d'étude réel sur un datacenter Green en production.

3.1 Contexte

Avec l'extension du Cloud Computing, nous avons vu apparaître le terme "Green Computing" ou informatique verte, visant à réduire la consommation énergétique et l'empreinte écologique des datacenters en réduisant les émissions de gaz à effet de serre. Cela reste une priorité pour les gestionnaires des datacenters à cause de l'augmentation du prix du kWh (Augmentation de la facture d'électricité de 50%, d'ici 2020 en France [57]). Actuellement, l'émission de CO_2 provenant des datacenters représente 2% de l'émission mondiale de CO_2, soit autant que l'aviation civile [8].

Sachant que la facture électrique est la principale charge opérationnelle dans un datacenter, les nouveaux défis des datacenters sont la maîtrise de la distribution électrique, le choix du meilleur système de refroidissement (le refroidissement air-air, eau-air...) et son optimisation pour un meilleur rendement. Les datacenters Green de nouvelle génération ont été conçus pour un PUE optimisé. L'amélioration continue de ces datacenters cible la réduction du PUE en fonction de la charge des serveurs, d'où l'intérêt d'utiliser les ressources informatiques d'une manière optimisée grâce à la consolidation des serveurs et des services. Les datacenters doivent respecter leur engagement en terme de SLA (disponibilité, sécurité, performance...) en utilisant des architectures redondantes, tout en optimisant la consommation électrique et le coût de refroidissement, et en réduisant l'empreinte écologique. L'intelligence dans un datacenter se repose sur des réseaux de capteurs qui fournissent des mesures en temps réel et sur des automates industriels robustes de commande, pour trouver le point optimal de fonctionnement.

L'objectif des datacenters est d'assurer une disponibilité proche de 100%, grâce aux éléments redondés qui permettent de le maintenir en fonctionnement, même en cas de panne d'un élément actif ou pour faire de la maintenance. Pour assurer la haute disponibilité des services en cas de pannes catastrophiques des "facilités" (systèmes de distribution électrique et de refroidissement), il est important d'avoir un plan de gestion des scénarios de crise pour assurer la continuité des services critiques le plus longtemps possible et permettre le retour rapide à la normale à la fin de l'incident. La concentration de la puissance informatique par m^2 a augmenté très rapidement ces 10 dernières années (15 à 40kW par baie) [9], ce qui se traduit par une forte dissipation thermique et donc un coût supérieur pour le refroidissement. Cela impacte la consommation globale du datacenter et le coût de l'exploitation, et nécessite donc l'optimisation du refroidissement.

Le datacenter est une architecture complexe de logiciels et matériels difficiles à fiabiliser et à optimiser au niveau Facilités. Les gestionnaires des datacenters utilisent les automates industriels connus pour leur fiabilité, pour gérer les équipements de distribution électrique et de refroidissement. Le point de fonctionnement optimal et le choix du système de refroidissement dépend d'un très grand nombre de paramètres. Le système de refroidissement est géré par des automates, chacun d'entre eux gérant un nombre limité d'équipements. Sans interconnexion et vue globale, chacun de ces automates prend des décisions indépendamment des autres

à cause de sa vue limitée aux équipements qu'il supervise. Il est intéressant de compléter ces automates physiques par une couche logicielle, pour optimiser le système de refroidissement et apporter plus d'intelligence et de réactivité dans les décisions.

La charge informatique d'un datacenter peut être très variable, mais aussi très périodique. Sans prédiction, il est difficile de prévoir les pics de charge et donc le risque de violation des SLA est important. D'autre part, l'utilisation de plusieurs systèmes de refroidissement hétérogènes, ayant des profils de charge différents selon la température extérieure, nécessite une connaissance des prévisions météorologiques. Sans ces données, la transition entre les différents systèmes de refroidissement est très fréquente, au risque d'endommager les circuits électriques.

3.2 Approche et contribution

3.2.1 Approche

Nous utilisons un système de gestion autonome "Autonomic Computing", basé sur des boucles de contrôle autonomiques avec une vue globale du datacenter. Ces boucles permettent une régulation efficace des différentes métriques avec une coordination globale entre elles. Les systèmes de gestion autonomes ont été proposés comme une solution pour la gestion des infrastructures distribuées afin d'automatiser les tâches et réduire la charge d'administration. De tels systèmes peuvent être utilisés pour déployer et configurer des applications dans un environnement distribué. Ils peuvent également surveiller l'environnement et réagir à des événements tels que les pannes ou les surcharges, et reconfigurer les applications et/ou les infrastructures en conséquence et de manière autonome. Le but principal du système que nous proposons est d'assurer la disponibilité des services et des applications critiques même en cas de pannes graves (coupure d'électricité et/ou défaillance du refroidissement) et d'optimiser le système de refroidissement. En cas de panne, il faut agir très vite pour assurer la continuité des services. Cela laisse très peu de temps d'analyse et nécessite l'exécution d'actions parfois contradictoires. L'automatisation des tâches d'administration permet une meilleure réactivité en cas de problèmes et une optimisation continue des états stables du système. Quand la panne est résolue, un plan de retour à la normale pour chaque scénario doit être mis en place pour redémarrer les équipements

et les applications qui ont été impactés. Une fois de plus, la connaissance de l'ensemble de l'infrastructure physique et logicielle du datacenter par les boucles FaaS, permet de décider de l'ordre et de la priorité de démarrage des serveurs et des applications. L'optimisation du système de refroidissement consiste à fournir des recommandations aux automates pour démarrer le(s) système(s) de refroidissement le(s) plus efficient(s) énergétiquement à un instant donné, en prenant en compte un très grand de paramètres (prévisions météorologiques, réglementations, charge serveurs...) et grâce à la vue globale de l'ensemble du datacenter.

Le coeur du datacenter doit être maintenu en marche le plus longtemps possible en évitant l'augmentation de la température dans les cubes. Le coeur du datacenter représente l'ensemble des équipements réseaux et des serveurs critiques (serveurs d'administration, baies de stockage, applications prioritaires...). Ceci nécessite une vision globale du datacenter (Facilités et infrastructure) et de la pile logicielle qui s'exécute dessus (machines virtuelles, applications, services...). Nous pouvons classer les équipements informatiques (serveurs de traitement, serveurs de stockage et équipements réseaux) en deux niveaux de criticité :

- Les équipements niveau 0 : Ils représentent le coeur du datacenter. Ce sont les équipements critiques, indispensables pour le fonctionnement du datacenter, qu'il faut garder le plus longtemps possible en cas de coupure générale d'électricité et fonctionnement des cubes sur batteries. Cela couvre l'ensemble des équipements réseau, stockage et serveurs d'administration et de monitoring ainsi que les serveurs hébergeant des applications critiques.

- Les équipements niveau 1 : Couvre les équipements niveau 0 et les applications hébergées en Tier IV (Applications prioritaires).

La vision globale multi-niveaux est fournie par la Configuration Management Database (CMDB). La CMDB est une base de données qui unifie tous les équipements du datacenter. Elle contient les informations sur les différents composants du système d'information et les relations entre eux.

Les applications peuvent avoir différents niveaux de priorité selon l'offre souscrite et le degré de criticité. Le niveau de criticité d'une application est défini dans la CMDB :

- Priorité haute : Applications hébergées en Tier IV

- Priorité moyenne : Applications hébergées en Tier III

- Priorité basse : Les serveurs et les applications de test (banc de test, plate-formes de développement ou de pré-production, ...)

Pour résumer, les boucles FaaS ayant une vision globale de tout le datacenter permettent de compléter les automates industriels déjà mis en place, en prenant en compte IaaS, PaaS et les boucles de contrôle respectives, pour optimiser le système de refroidissement et gérer au mieux les pannes.

Le passage d'un système de refroidissement à un autre n'est pas instantané, donc en cas d'une forte augmentation du besoin de puissance de refroidissement, due à une augmentation de la charge des serveurs, le système ne peut pas fournir cette puissance rapidement et nous risquons d'endommager les serveurs à cause de l'augmentation de la température des salles, ce qui impactera le SLA. De même, la variation très fréquente de la puissance informatique peut entraîner une permutation fréquente entre les différents systèmes de refroidissement et donc une augmentation du risque d'endommager ces systèmes. Pour remédier à ces risques, nous proposons un module de prédiction de la charge, qui permet à partir des historiques de mesures, d'anticiper les pics de charge.

3.2.2 Contribution

Nous proposons une couche logicielle, permettant de prendre des décisions plus intelligentes pour le choix du système de refroidissement. Cette couche est constituée (i) d'une vue globale de l'ensemble de l'infrastructure FaaS, IaaS et PaaS (applications et services) et (ii) d'une boucle FaaS d'optimisation du système de refroidissement grâce à une meilleure connaissance de l'ensemble du système. Cette boucle permet de gérer des situations de panne et de grandes variations pour protéger les équipements et les applications critiques. Une décision au niveau Facilités peut impacter directement les équipements IT, d'où l'importance de garder un environnement (température, humidité) stable pour les équipements IT. Cette couche logicielle reçoit des informations hétérogènes de sources variées (capteurs de température et d'humidité, points de mesures électriques, serveurs, VM, applications...) et dans des formats très variés (Modbus, monitoring applicatif, VCenter...). Cela représente un gros volume de données qu'il faut rendre exploitable

pour l'analyser ensuite. Le but est l'intégration des mécanismes de niveau FaaS et avec prise en compte de SLA multicritère énergie/performance/fiabilité.

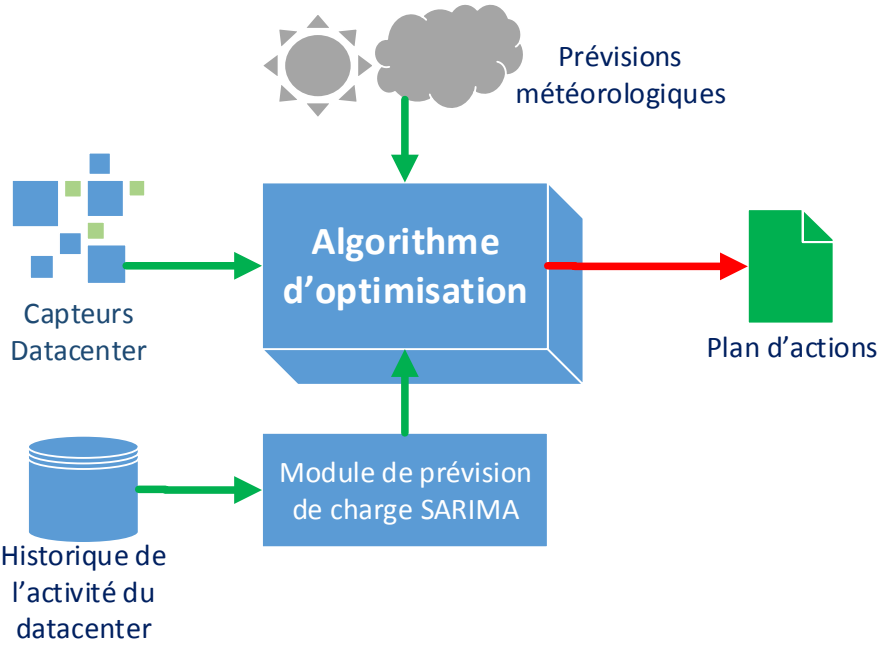


Figure 3.1 – Système d'optimisation du refroidissement

Nous avons défini une boucle de contrôle qui coordonne le fonctionnement des automates du datacenter pour l'optimisation du système de refroidissement : Cette boucle nécessite une analyse continue des données internes (charge des serveurs, température salles...) et externes (température et humidité extérieurs, prévisions météo...) pour réduire les coûts de refroidissement du bâtiment. Cette boucle peut agir d'une manière prédictive grâce au module de prédiction de l'activité du datacenter à partir de l'historique (Figure 3.1).

Pour évaluer notre algorithme, nous étudions la mise en place de boucles de contrôle logicielles dans le datacenter Green de "Business & Decision Eolas". L'objectif est d'évaluer notre système autonome et les techniques de contrôle réactif et prédictif dans la vraie vie d'un datacenter sur lequel tourne des applications réelles.

3.3 Le datacenter Green de Mangin

Dans cette partie nous détaillons l'architecture et les mécanismes de monitoring et de contrôle de bas niveau du datacenter de Business & Decision Eolas ayant une capacité totale de 13 cubes informatiques. Ce datacenter Green est équipé de plusieurs circuits de distribution électrique et de refroidissement. Sa particularité est le refroidissement des serveurs par l'eau de la nappe phréatique. Il offre deux niveaux de redondance et de disponibilité :

- Tier IV : A ce niveau, les équipements sont super-redondés en alimentation électrique et en refroidissement. Ils sont alimentés par deux arrivées de courant principales, en plus du groupe électrogène de secours. Grâce aux deux circuits de refroidissement : le groupe froid et les pompes d'eau de la nappe phréatique, ainsi que le refroidissement de secours par l'eau de ville, le refroidissement est maintenu même en cas de panne électrique. Ce niveau de disponibilité couvre l'équivalent de deux salles informatiques à pleine charge avec le système de refroidissement nécessaire. La disponibilité est très proche de 100%.

- Tier III : Les équipements sont alimentés par deux arrivés redondantes du courant électrique principal mais ne sont pas reliés au groupe électrogène de secours. Ils sont refroidis par deux circuits de refroidissement. Ce niveau couvre 8 cubes à pleine charge en plus du système de refroidissement.

Pour garantir une meilleure disponibilité de l'infrastructure, plusieurs milliers de points de mesure sont répartis sur le site pour fournir des mesures en temps réel et détecter les défauts de fonctionnement. Le volume de données généré est très important, ce qui rend l'analyse plus compliquée. Le basculement d'une source d'alimentation à une autre et le démarrage des systèmes de refroidissement sont gérés automatiquement grâce à des automates industriels (automates Schneider TSX). Dans la suite nous détaillons l'architecture complète du système de distribution électrique et de refroidissement dans le datacenter Mangin.

3.3.1 Le système de distribution électrique

Le datacenter Mangin est alimenté essentiellement en électricité verte (garantie 100% énergie renouvelable) par l'entreprise GEG (Gaz Électricité de Grenoble).

L'alimentation par GEG du poste HTA à 20 kV en coupure d'artère (le poste HTA est inséré en série sur un départ HTA. Il est équipé de deux interrupteurs et d'un dispositif de protection HTA, qui protège le réseau des défauts provenant de l'installation de l'utilisateur consommateur) depuis 2 autres postes Haute Tension B (HTB) ou HTA distincts, assure une redondance 2N favorisant la continuité du service. Le poste HTA reste alimenté si une des lignes électriques était défaillante. La figure 3.2 illustre l'architecture complète du système de distribution électrique du datacenter en 2N+1 : Deux arrivées principales du courant GEG et une arrivée du groupe électrogène de secours.

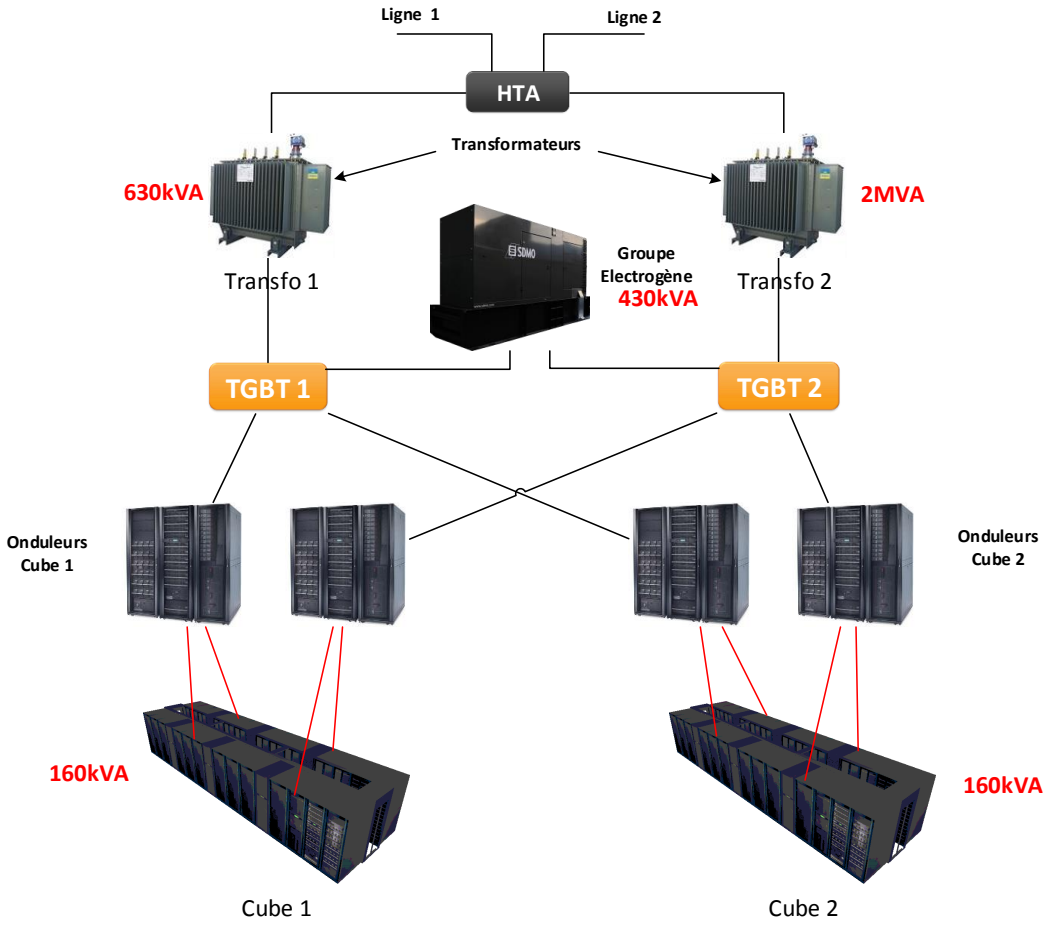


Figure 3.2 – La distribution électrique du datacenter Eolas

Deux transformateurs de capacités distinctes sont reliés au poste HTA :

- Le transformateur 1 de capacité 630 kVA est capable d'alimenter trois cubes à pleine puissance en plus du système de refroidissement

- Le transformateur 2 de capacité 2 MVA peut alimenter jusqu'à 8 cubes à pleine charge en plus du système de refroidissement

Le groupe électrogène de secours est capable d'alimenter deux cubes à pleine puissance en plus du système de refroidissement correspondant. Il est maintenu à chaud et démarre automatiquement en moins de 8 secondes en cas de coupure du courant électrique GEG. Ce groupe électrogène est alimenté par un réservoir de fuel offrant jusqu'à 24 heures d'autonomie sachant que ce réservoir peut être alimenté en continu. Les tableaux électriques Tableau Electrique Basse Tension (TGBT) permettent de passer d'une source d'alimentation à une autre automatiquement et fournissent en temps réel l'état des différents disjoncteurs, des mesures de puissance électrique et des alertes de fonctionnement. Chaque cube est équipé de deux onduleurs, chacun relié à un TGBT distinct pour assurer la redondance. Ces onduleurs sont équipés de batteries pour maintenir les serveurs allumés en cas de coupure d'électricité ou de changement de source d'alimentation, avec une autonomie pouvant aller jusqu'à 40 minutes. La plupart des serveurs sont équipés d'une double alimentation, chacune reliée à un onduleur différent et donc restent allumées en cas de défaillance d'un onduleur ou d'un transformateur.

3.3.2 Le système de refroidissement

Dans cette partie, nous allons détailler l'architecture du système de refroidissement et les différentes sources de production de froid du datacenter d'Eolas. Les trois circuits de refroidissement offrent une meilleure redondance pour le système de refroidissement en cas de défaillance de l'un de ces systèmes :

- Les deux pompes d'eau de la nappe phréatique (voir paragraphe 3.3.2.1).
- Le groupe froid 1 et le groupe froid 2 avec un module "Freecooling" (voir paragraphe 3.3.2.2). Le "Freecooling" est une méthode économique qui utilise la basse température d'air extérieur pour refroidir l'eau dans le système à moindre coût.
- L'eau de ville en ultime secours avec une ouverture manuelle et un coût important (voir paragraphe 3.3.2.3).

La figure 3.3 montre l'architecture du système de refroidissement par eau, jusqu'à l'arrivé dans les InRows des cubes. Un InRow est une armoire (Rack) de

refroidissement qui joue le rôle d'un échangeur thermique entre l'allée chaude et l'allée froide et qui s'ajuste automatiquement (ouverture et fermeture des vannes d'eau et vitesse de ventilation) pour un meilleur rendement énergétique.

Arrêter le système de refroidissement dans ce datacenter peut être dramatique. En l'absence de la climatisation, chaque minute, la température de la salle informatique augmente de 1°C. Sachant que la température de l'allée chaude est fixée à 35°C, la salle peut atteindre les 60°C en moins de 30 minutes, une température pouvant causer des dommages importantes aux serveurs.

Dans la suite, nous détaillons le fonctionnement de chacun des systèmes de refroidissement de l'architecture, ayant des modes et des capacités très hétérogènes. Tous ces systèmes sont alimentés par deux arrivés électriques distinctes pour la redondance.

3.3.2.1 Les pompes d'eau de la nappe phréatique

C'est le système de refroidissement primaire qui consiste à utiliser l'eau de la nappe phréatique à 14-15°C pour refroidir les cubes. L'eau passe par un échangeur thermique pour refroidir le circuit d'eau secondaire avant d'être envoyée aux InRows (Figure 3.4). L'eau récupérée est utilisée à la même température sans être refroidie et seules les pompes de circulation d'eau consomment de l'énergie : c'est le refroidissement passif ou "Natural Cooling". L'indicateur WUE dans ce datacenter est égale à 0 puisque l'eau utilisée pour refroidir est rejetée ensuite dans la nappe.

Deux pompes à eau de puissances différentes récupèrent l'eau de la nappe phréatique à 30 mètres de profondeur. La pompe Nappe 1 peut produire jusqu'à 300kW de froid et consommer jusqu'à 7kW d'électricité pour refroidir jusqu'à 2 cubes à pleine charge. La pompe Nappe 2 peut produire jusqu'à 800kW de froid avec une consommation électrique maximale de 25kW et peut refroidir jusqu'à 6 cubes à pleine charge. Les pompes d'eau glacée permettent d'alimenter le circuit secondaire d'eau glacée qui passe par les InRows. Ces pompes ayant des puissances différentes, sont contrôlées par des automates matériels. Les pompes d'eau glacée 1 et 2 ont des puissances équivalentes et chacune permet d'alimenter en eau glacée les deux cubes en Tier IV. La pompe 3 est plus puissante que les pompes 1 et 2, et permet de refroidir jusqu'à 6 cubes. Les automates contrôlent l'état et la puissance des pompes pour avoir un meilleur rendement. L'automate 1 contrôle le

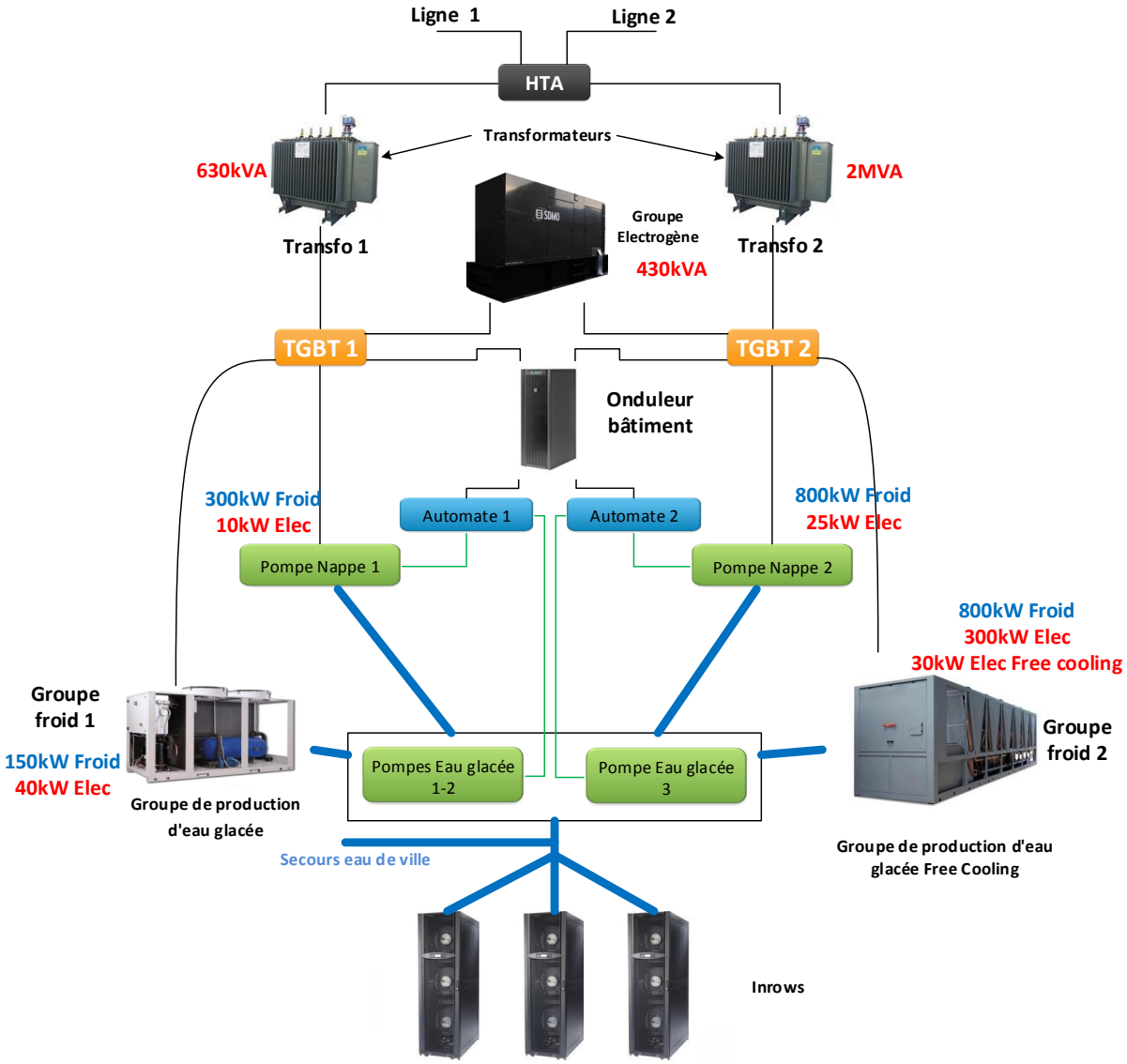


Figure 3.3 – Vue détaillé des systèmes de refroidissement redondés électriquement

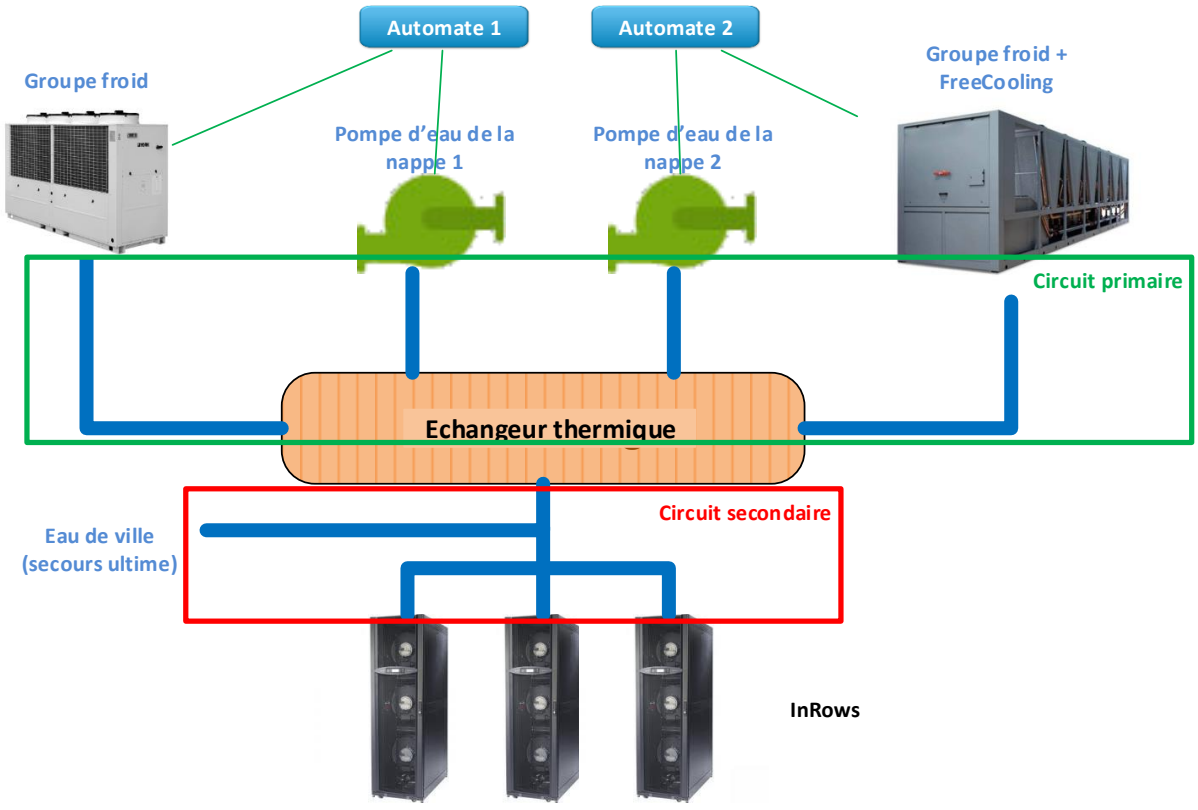


Figure 3.4 – Circuits de refroidissement du datacenter Eolas

démarrage et la puissance de la pompe Nappe 1 et des pompes d'eau glacée 1 et 2. L'automate 2 contrôle le démarrage est la puissance de la pompe nappe 2 et de la pompe d'eau glacée 3. Actuellement, la pompe nappe 1 (ou 2) et la pompe d'eau glacée 1 (ou 2) sont suffisantes pour refroidir les deux cubes à pleine charge.

3.3.2.2 Les groupes de production de froid

En cas de dysfonctionnement dans le système de refroidissement par l'eau de la nappe phréatique, l'un des deux groupes froids sera démarré. Le groupe froid 1 peut produire jusqu'à 150kW de froid permettant de refroidir 2 cubes, tandis que le groupe froid 2 peut produire jusqu'à 800kW pour refroidir jusqu'à 6 cubes. Des sondes de température dans le circuit d'eau primaire et secondaire déclenchent le démarrage des groupes froids. Cela peut être dû à l'augmentation de la température de la nappe phréatique qui est incontrôlable (selon les saisons), à une panne des pompes de la nappe phréatique ou à une grande différence de

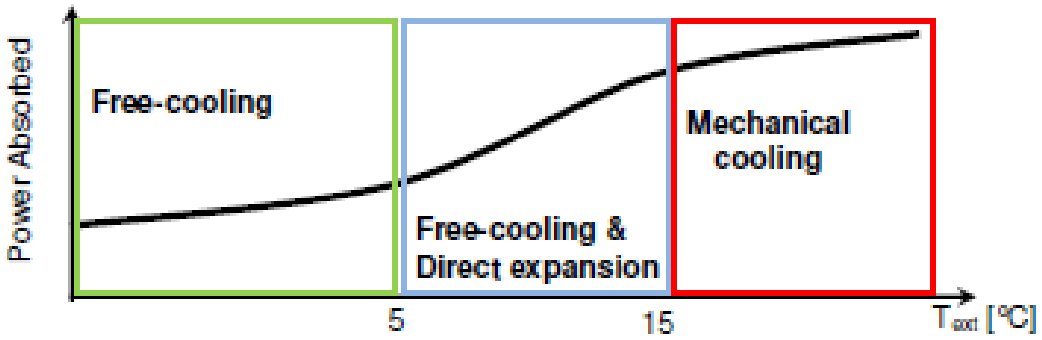


Figure 3.5 – Permutation de mode de refroidissement en fonction de la température externe

température entre l'eau en entrée et sortie de la nappe suite à une forte charge informatique. Les groupes froids remplacent le refroidissement par l'eau de la nappe phréatique aussi pour respecter les réglementations de la DREAL "Direction Régionale de l'Environnement, de l'Aménagement et du Logement" (voir paragraphe 3.4 pour les règles d'utilisation de l'eau de la nappe), qui impose un débit maximal par heure et une limitation de la quantité d'eau sur l'année. Les groupes froids refroidissent l'eau entrant dans les salles selon un cycle thermodynamique (compression, condensation, détente, évaporation). Ce système de refroidissement consomme beaucoup d'électricité, donc il faut éviter l'utilisation de ce système le plus possible. Le Groupe Froid 2 est équipé d'un module FreeCooling permettant de refroidir l'eau à moindre coût quand la température d'air extérieur est basse (moins de 5°C) (Figure 3.5) : Nous pouvons alors parler de système de refroidissement hybride. En mode freecooling, ce groupe froid consomme 10 fois moins qu'un refroidissement par compression de gaz et peut consommer moins d'énergie que les pompes de la nappe phréatique pour une même production de froid. Le Groupe Froid 2 fonctionne aussi en mode mixte quand la température extérieure est entre 5 et 15°C. Dans ce mode, le groupe froid utilise à la fois le mode compression et le FreeCooling pour refroidir l'eau, donc la consommation est très variable. Les automates industriels permettent de gérer un nombre limité d'entrées sorties et ont une vision locale du système. D'où l'importance de mettre en oeuvre des automates logiciels avec une vision de l'ensemble de l'infrastructure et prenant en compte un grand nombre de paramètres (prévisions météo, température extérieure...) pour décider du choix du système le plus efficient.

Les températures basses favorisent l'utilisation des systèmes de FreeCooling même si le "Natural Cooling" reste le système de refroidissement le plus économe la majorité des jours de l'année.

3.3.2.3 L'ultime secours : l'eau de ville

En cas d'absence des deux systèmes de refroidissement primaires (par eau de la nappe phréatique et par groupe froid) due à une panne, une fuite d'eau ou une défaillance électrique, le seul système de refroidissement possible est l'eau de ville. L'ouverture des vannes d'eau de ville est manuelle. Une fois alertés, les équipes d'astreintes peuvent intervenir 24/24h dans un délai de 20 minutes. Sachant que la température de la salle de serveurs augmente de 1°C par minute, il est donc important d'agir vite pour limiter cette augmentation de température. L'eau de ville peut refroidir jusqu'à 3 cubes à pleine charge. Par contre, ce mode de refroidissement coûte très cher et fait grimper l'indicateur WUE du fait que l'eau utilisée pour le refroidissement est rejetée sans être réutilisée.

3.3.2.4 Le refroidissement des cubes

Un cube est composé de deux rangés de baies informatiques. Les cubes sont conçus avec un système de confinement par allée chaude [53]. L'air froid rentre par la face avant des serveurs et l'air chaud est ensuite rejeté à l'arrière des serveurs dans l'allée chaude. Les cubes sont refroidis grâce aux armoires de climatisation « InRows » (solution APC) qui jouent le rôle d'échangeur thermique entre allée froide et allée chaude. La figure 3.6 montre le principe de fonctionnement du refroidissement des cubes par confinement d'allée chaude.

Le circuit d'eau glacée secondaire alimenté par le circuit d'eau secondaire, amène l'eau glacée directement dans les InRows. L'eau entre dans les échangeurs des InRows à 15°C et sort à 24°C. Nous utilisons l'eau de la nappe phréatique à 14-15°C directement sans le refroidir. Les deux paramètres qui permettent de contrôler la température de la salle sont :

- La température de consigne de l'allée froide (25°C).
- La différence de température Δt maximale entre allée froide et allée chaude (10°C).

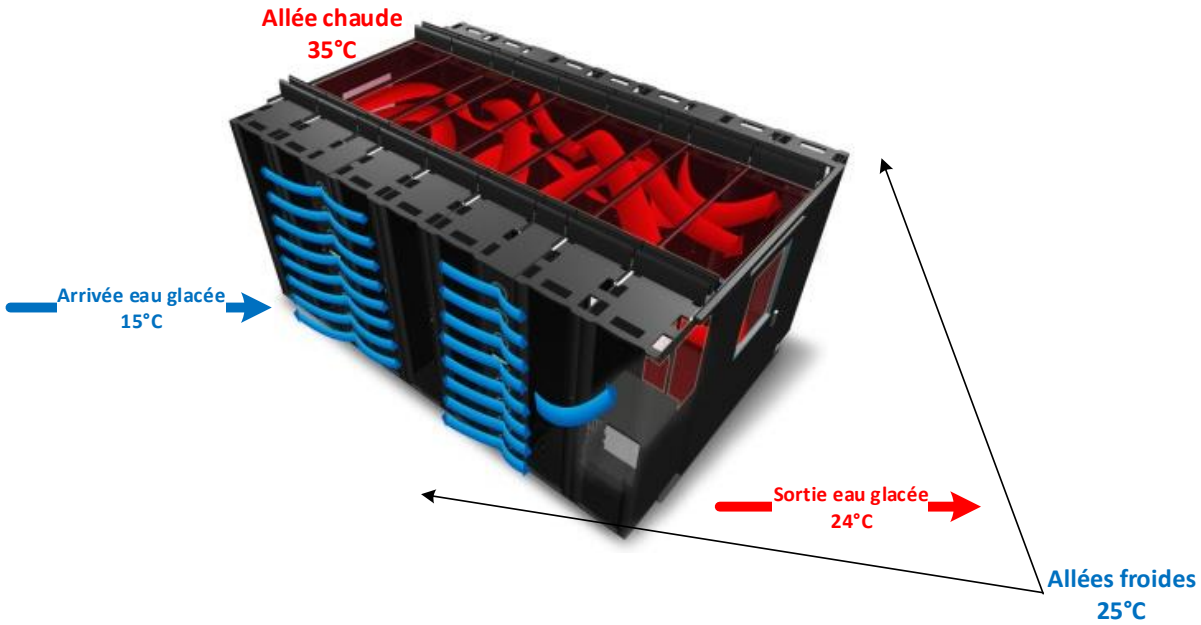


Figure 3.6 – Un cube informatique avec confinement d'allée chaude

L'ouverture des vannes d'eau à l'entrée des InRows et la vitesse de ventilation des InRows sont ajustées automatiquement selon ces deux paramètres pour avoir le meilleur rendement tout en répondant à l'augmentation de la charge informatique dans les cubes. Nous étudions dans la suite les boucles de contrôles visant à optimiser le choix du circuit de refroidissement le plus efficient et le plus économe en énergie dans le datacenter de Mangin.

3.4 Réglementations et contraintes

Actuellement, les pompes d'eau de la nappe phréatique représentent le système principal de refroidissement du datacenter. Les groupes Froids sont démarrés automatiquement en cas de problème. Les automates industriels déjà mis en place ne permettent pas de choisir le système de refroidissement le plus efficient, mais gèrent uniquement les pannes. L'idée de la boucle d'optimisation est d'automatiser le choix du système de refroidissement primaire le plus économe en énergie pour réduire la consommation du datacenter. Cette boucle doit prendre en compte les conditions extérieures et intérieures, et les réglementations en vigueur pour générer des plans d'actions, ce qui n'est pas possible avec un automate industriel.

Le nouveau groupe de production d'eau glacée Free Cooling peut refroidir l'eau à moindre coût selon les conditions extérieures. Quand la température d'air extérieur est basse et que les conditions météo sont stables, le module FreeCooling est plus économe en énergie que les pompes de la nappe phréatique. D'autre part, les réglementations définis dans le dossier ICPE (Installation Classée pour la protection de l'environnement) déposé à la DREAL (Direction Régionale de l'Environnement, de l'Aménagement et du Logement), imposent une limitation du débit par heure et de la quantité d'eau de la nappe phréatique pour refroidir le bâtiment, ainsi qu'une limite de différence de température de l'eau en entrée et en sortie de la nappe de 10°C au maximum. Le contrat est divisé en deux phases : La phase 1 est donnée sur 4 ans depuis le 1er Avril 2011. Le tableau 3.1 donne les limitations du débit et de la quantité d'eau durant les deux phases :

Table 3.1 – Réglementation pour l'utilisation de l'eau de la nappe phréatique

	Débit	Quantité
Phase 1	130 m3/h	1138000 m3/an
Phase 2	260 m3/h	1138000 m3/an

Le choix du système de refroidissement peut donc être optimisé selon :

- La température et l'humidité de l'air extérieur.
- Les saisons et la stabilité de la météo : En se basant sur les prévisions météorologiques sur une journée et la différence de température entre jour et nuit
- Le taux d'utilisation du refroidissement par l'eau de la nappe phréatique (débit et quantité annuelle).
- L'état des différents systèmes de refroidissement : arrêté, démarré, en panne, en maintenance, en dépassement.
- La source d'alimentation électrique primaire : GEG ou groupe électrogène.
- Les prévisions de charges informatiques calculées en fonction des historiques de mesure de l'activité globale.

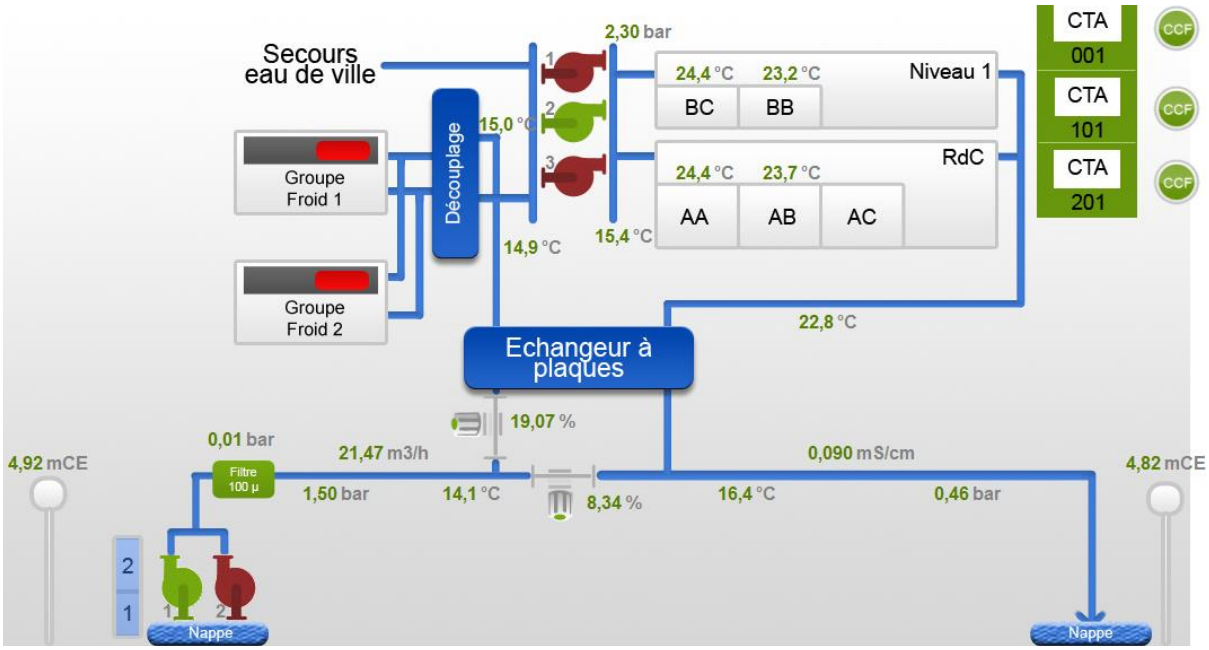


Figure 3.7 – Vue temps réel de l'état et des métriques du système de refroidissement

- L'ensemble des métriques remontées par les capteurs déployés à différents niveaux du circuit de refroidissement. La figure 3.7 montre une vue temps réel de l'ensemble des équipements et des mesures du système de refroidissement (exemple : température de la nappe en entrée et en sortie).

Cette variété de paramètres ne peut pas être gérée par un automate industriel avec un comportement isolé, d'où l'importance d'ajout d'une couche logicielle plus complexe. Cette couche va pouvoir agir sur les automates industriels pour privilégier le groupe froid 1, le groupe froid 2 (en mode compression ou Freecooling) ou les pompes de la nappe phréatique pour minimiser les coûts, tout en respectant les réglementations. Cette couche logicielle fonctionne dans un environnement stable pour optimiser la consommation du système de refroidissement. Le système est dit stable si les équipements informatiques concernés fonctionnent dans les conditions normales d'exploitation (alimentation électrique en continu et refroidissement suffisant). Il est important de garder stable la température de la salle pour éviter d'endommager les équipements suite aux changements de température. Pour cela, nous évitons la modification de la température de consigne de l'allée froide, sachant que la consommation des ventilateurs d'un serveur représente 5 à

10% de la consommation totale du serveur et que cette consommation augmente avec l'augmentation de la température.

3.5 Méthodologie

3.5.1 Étude de l'activité d'un datacenter

Les datacenters hébergent des applications très hétérogènes avec un profil de charge très variable sur une journée. Certaines applications, telles que les applications e-commerce présentent des pics de charges réguliers (au début, à moitié et en fin de journée) ou saisonnier (périodes de soldes, fêtes...). D'autres applications de type Big Data sont sollicitées en permanence avec une forte charge qui sature l'ensemble des machines. La figure 3.8 montre l'évolution de l'activité globale du datacenter en nombre de flux par minute sur une journée ordinaire. Nous pouvons identifier deux grand pics de charge correspondant aux périodes de 10h et de 14h, et un petit pic de charge vers 20h.

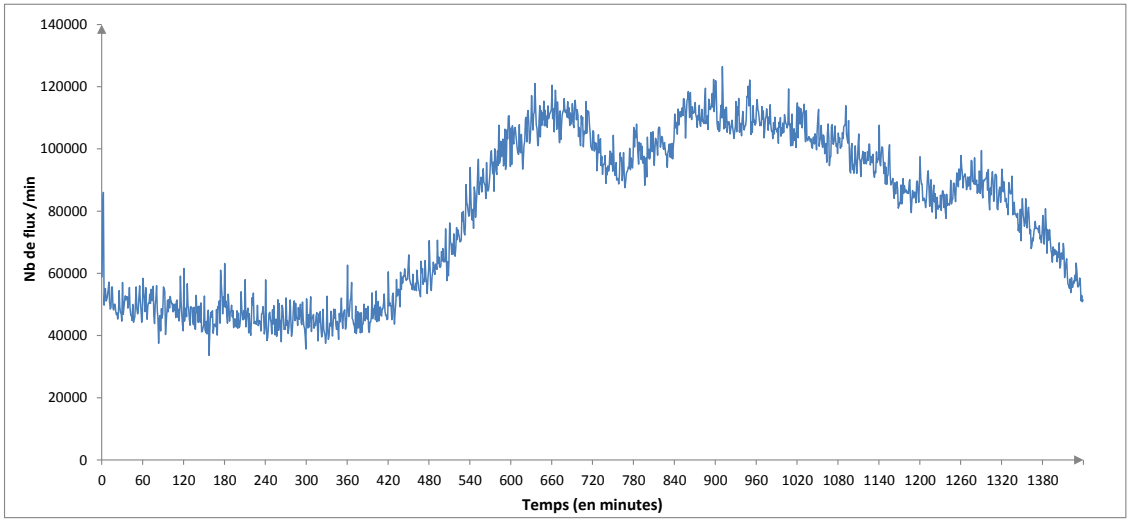


Figure 3.8 – Variation de l'activité du datacenter sur une journée

Pour identifier la variation de la charge sur une période plus longue, la figure 3.9 montre l'évolution de l'activité du datacenter sur une période de 14 jours (avec une fréquence de mesure d'une minute) qui démarre un Lundi. Nous remarquons une activité importante pendant les jours ouvrés (Lundi à Vendredi), et une

activité moins importante les week-ends (Samedi et Dimanche), ce qui est normal puisque les clients sont moins nombreux les week-ends. L'activité globale du datacenter est périodique sur une période de 7 jours, puisque la semaine suivante on distingue un trafic semblable à celui de la semaine précédente.

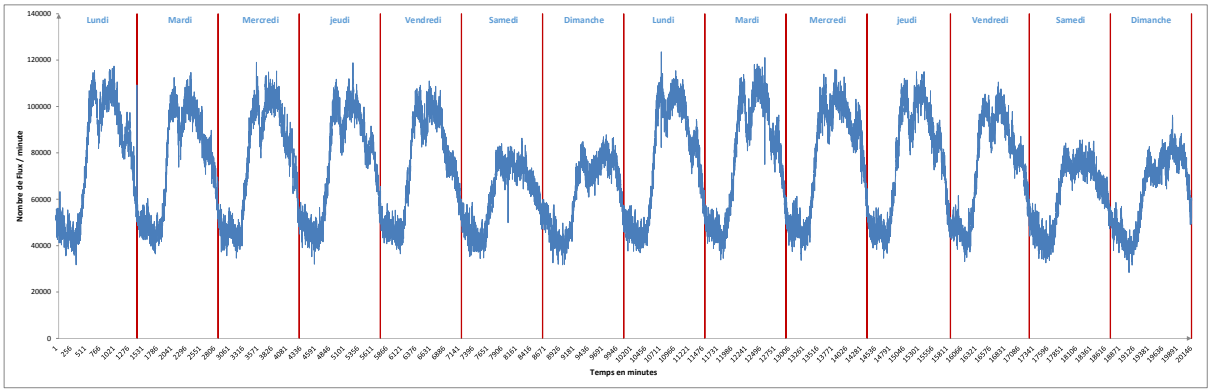


Figure 3.9 – Variation de l'activité du datacenter sur 14 jours

3.5.2 Prédiction de la charge avec SARIMA

Nous appliquons le modèle SARIMA [21] pour prédire l'activité du datacenter. La méthode SARIMA permet de prédire des périodes futures, à partir des séries temporelles stationnaires (varient en fonction du temps) et saisonnières. Nous utilisons 14 jours d'historique de mesures pour estimer les paramètres du modèle. Nous avons 672 mesures, qui correspondent à 48 mesures par jour pour des périodes de 30 minutes. La saisonnalité de la série temporelle a été fixée à une semaine, soit 336 mesures (48 mesures x 7 jours). La figure 3.10 montre la charge réelle sur 7 jours et la prévision issue du modèle SARIMA. L'erreur absolue relative (taux d'erreur moyen) est inférieur à 7%, avec un écart maximal de 15%.

Le modèle SARIMA génère une prédiction qui est suffisamment bonne (faible taux d'erreur) pour prévoir à 7 jours la charge du datacenter. Cette estimation est suffisante pour notre cas d'usage.

3.5.3 Utilisation des prévisions météorologiques

Le choix de la méthode de refroidissement dépend de beaucoup de paramètres externes et notamment de la température de l'air à l'extérieur du bâtiment. Tandis

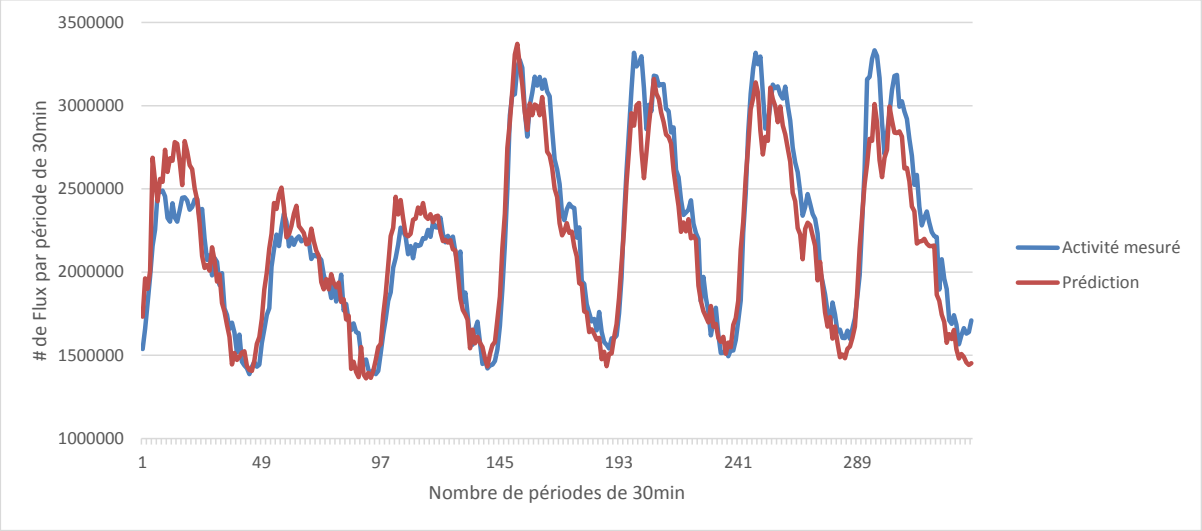


Figure 3.10 – Prédiction de l'activité du datacenter sur une semaine avec SARIMA.

que l'eau de la nappe phréatique garde une température constante tout au long de l'année (température entre 13 et 14°C), la température de l'air extérieur peut varier très rapidement pendant une même journée. Le climat de Grenoble est atypique, nous mesurons une grande variation de température sur une journée et une grosse différence entre les saisons. Le refroidissement par l'eau de la nappe phréatique reste privilégié, sauf si la température extérieure est très basse, ce qui rend le refroidissement en mode FreeCooling plus avantageux. L'impact de la température extérieure est aussi calculable sur les systèmes de refroidissement de type groupe froid. Les variations très fréquentes de la température entraînent des changement de mode de refroidissement sans cesse, ce qui peut endommager les composants électriques des systèmes de refroidissement avec des cycles de démarrage/arrêt. Par exemple, la pompe de la nappe phréatique peut être redémarrée au maximum deux fois par jours, au risque d'effondrer les parois de la nappe phréatique.

Pour limiter les permutations entre les différents systèmes de refroidissement en fonction de la température extérieure, nous nous basons sur les prévisions météorologiques pour estimer le nombre de permutations et la durée d'utilisation de chaque système. Cela permet de limiter le nombre de permutations, tout en minimisant la consommation du refroidissement.

La figure 3.11 montre le choix du système de refroidissement le plus économe

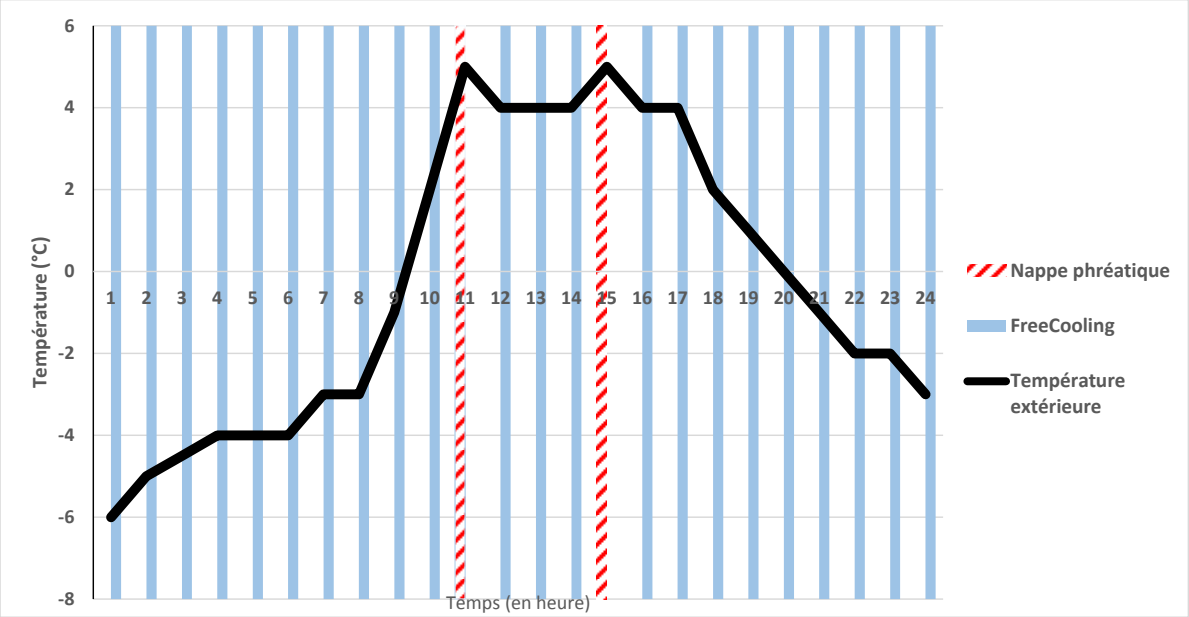


Figure 3.11 – Choix du système de refroidissement en fonction de la température extérieure

en fonction de la température extérieure sur un jour du mois de Janvier, sur des périodes d'une heure. Nous remarquons que durant la journée, deux changements de système de refroidissement seront nécessaires vers le système de refroidissement par l'eau de la nappe phréatique suite à l'augmentation de la température externe. Quand la température extérieure est très basse, le système de FreeCooling est plus économe (Le rendement "Energy Efficiency Ratio (EER)" est très haut). Quand la température augmente, le EER du système de FreeCooling baisse et le refroidissement par l'eau de la nappe phréatique devient plus économe électriquement. La pompe de la nappe sera démarrée deux fois quand la température dépasse les 5°C, pendant des périodes très courtes, ce qui ne permet pas de faire beaucoup d'économies d'énergie et peut réduire la durée de vie des composants électriques, ce qui rend le coût de la maintenance plus importante. Dans cet exemple, maintenir le refroidissement par le système de FreeCooling pendant toute la journée est la meilleure solution.

Dans l'exemple de la Figure 3.12, il est plus efficient énergétiquement d'utiliser le refroidissement par les pompes d'eau de la nappe phréatique sur les périodes 11 et 13, mais cela entraîne deux cycles de démarrage/arrêt du système. Même si énergétiquement il est plus efficient d'utiliser le système de refroidissement par

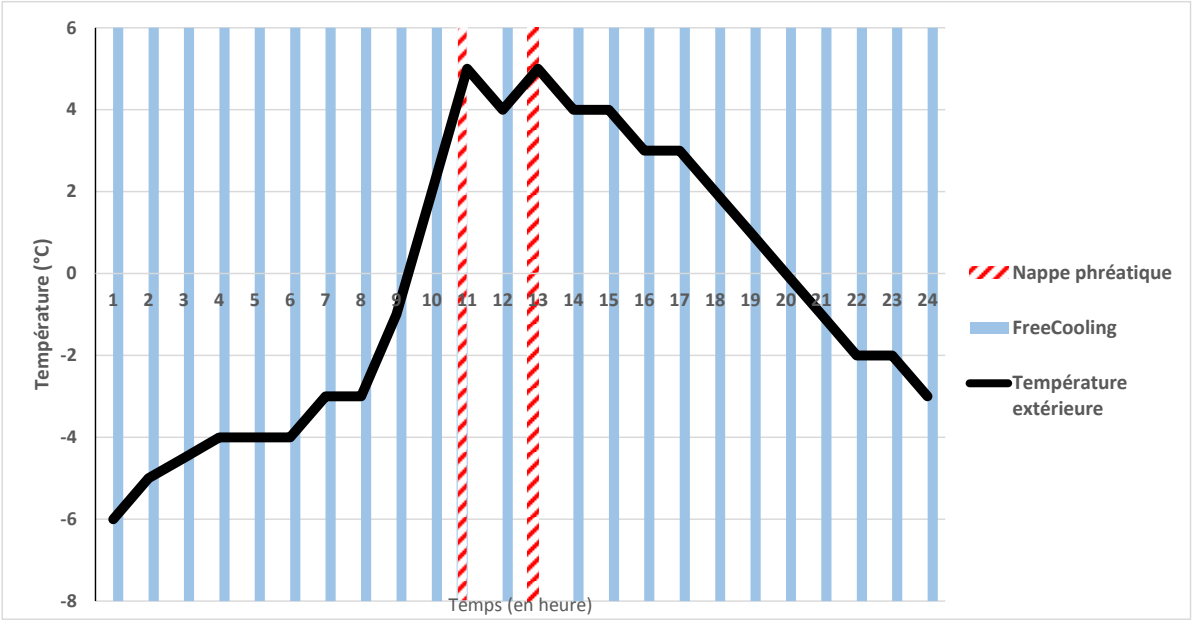


Figure 3.12 – Limitation des transitions des systèmes de refroidissement en fonction de la température extérieure

FreeCooling sur la période 12, nous choisissons d'utiliser le système de refroidissement par l'eau de la nappe sur les périodes 11, 12 et 13 afin de réduire à un le nombre de cycle de démarrage/arrêt et ainsi prolonger la durée de vie des composantes électriques.

3.5.4 Algorithme d'optimisation du refroidissement

3.5.4.1 Programme d'optimisation linéaire

Pour réduire la consommation électrique du refroidissement et ainsi réduire le PUE, nous avons estimé la puissance de refroidissement nécessaire pour refroidir le datacenter en fonction de la consommation des serveurs informatiques comme présenté dans la formule (3.1) :

$$\sum C_P(kW) = 0.9 * \sum S_C(kW) \tag{3.1}$$

Avec C_P la puissance de refroidissement nécessaire en kW et S_C la consommation électrique totale des équipements informatiques. Presque 90% de la puissance électrique consommée par les serveurs est transformée en puissance calorifique

qu'il faut refroidir.

Pour un résultat optimal du problème, nous avons utilisé un programme d'optimisation linéaire, qui a pour objectif de réduire la consommation électrique du système de refroidissement tout en respectant les contraintes opérationnelles. La programmation linéaire est une technique mathématique d'optimisation (maximisation ou minimisation) de fonction objectif linéaire sous des contraintes ayant la forme d'inéquations linéaires.

L'équation (3.2) représente le coût en euro (€) pour produire un kW de froid. P_{elec} et k_{elec}€ représentent respectivement la puissance électrique consommée par le système de refroidissement pour produire 1kW de froid, et le prix d'un kWh sachant qu'en France 1kWh coûte 6centimes en moyenne pour les entreprises.

$$C = P_{elec} * k_{elec}€ \tag{3.2}$$

Le EER est le coefficient d'efficacité frigorifique. Il représente la performance énergétique d'un système de refroidissement, plus l'EER est haut, l'efficacité énergétique est meilleure. La consommation électrique d'un système de refroidissement peut être calculée avec la formule 3.3. P_{elec} et P_{froid} représentent respectivement la puissance électrique consommée par le système de refroidissement et la puissance frigorifique fournie par ce système. La valeur du EER est fournie par les constructeurs et dépend de plusieurs facteurs (température de l'eau et de l'air, l'humidité...).

$$P_{elec} = P_{froid}/EER \tag{3.3}$$

Le coût d'utilisation de l'eau de la ville est présenté dans la formule (3.4). $Debit$ représente le débit d'eau utilisé en m^3 et k_{eau}€ le coût d'un m^3 d'eau facturé.

$$C = Debit * k_{eau}€ \tag{3.4}$$

Le débit d'eau de ville peut être déduit de la puissance de refroidissement produite, grâce à la formule (3.5). $P_{cooling}$ représente la puissance de refroidissement

fournis et Δt la différence de température de l'eau entre l'entrée et la sortie du datacenter.

$$C = P_{cooling}/(1.16 * \Delta t) \tag{3.5}$$

Le programme d'optimisation se présente sous la forme suivante :

$$Min \sum_{i=0}^{n} a_i.C_i \tag{3.6}$$

Sous les contraintes :

$$\begin{cases} \sum_{i=0}^{n} a_i >= C_P & i \in 0, n \\ \sum Cm_i \in \{\text{modes de refroidissement possibles}\} \end{cases} \tag{3.7}$$

C'est un problème d'optimisation linéaire, avec des contraintes linéaires qu'on peut résoudre simplement et efficacement avec un solveur numérique (type simplex). L'équation (3.6) représente la fonction objectif du programme d'optimisation linéaire où a_i représente la quantité de froid généré par le système de refroidissement i en kW et C_i le coût en euro (€) d'un kW de froid en utilisant le système de refroidissement i. n est le nombre maximal de systèmes de refroidissement qui peuvent être démarrés simultanément. On vise à minimiser la consommation totale des systèmes de refroidissement en fonctionnement tout en minimisant le nombre de systèmes démarrés. En réduisant la consommation des systèmes de refroidissement, nous réduisons la consommation globale du datacenter et réduisons donc le PUE. L'équation (3.7) représente les contraintes du problème d'optimisation linéaire. Nous ciblons la satisfaction des SLA en produisant suffisamment de froid par rapport à la puissance demandée, et donc la puissance de refroidissement totale générée doit être idéalement égale à la puissance de refroidissement demandée. Cm_i est le mode de refroidissement du système de refroidissement i. Les modes de refroidissement possibles sont limités par des paramètres tel que la température externe ou la température de l'eau de la nappe phréatique, c'est à dire si la température de l'eau est très haute, il n'est plus possible d'utiliser les pompes de

la nappe pour refroidir le datacenter. Les systèmes de refroidissement "en panne" ou "en maintenance" seront éliminés de la liste.

3.5.4.2 Optimisation réactive et prédictive

Pour éviter d'endommager les circuits électriques et électroniques avec des cycles de démarrage/arrêt répétitifs, nous avons définis une période minimale entre deux transitions et un nombre limité de transitions sur une journée. Nous nous basons sur les prévisions météorologiques et les prévisions de charge des serveurs pour valider ces limitations même en cas de très fortes variations de la température. Sachant qu'il est possible de démarrer plusieurs systèmes de refroidissement simultanément, il est important de réduire le nombre de systèmes de refroidissement démarrés simultanément pour réduire la consommation.

A chaque nouvelle période, soit une fois toute les heures dans notre évaluation, nous récupérerons les prévisions météorologiques sur 24h avec une période d'une heure, à travers un web-service et calculons les prévisions de charge globale des serveurs sur 24 heures afin de définir un plan d'action pour l'optimisation du refroidissement.

L'algorithme 1 représente notre programme d'optimisation ainsi que les entrées et les sorties. A chaque période, nous estimons la consommation de chaque système de refroidissement et identifions la/les système(s) les plus économes sur 24 heures. Nous calculons ensuite le nombre de transitions ainsi que la durée de chaque transition pour déterminer la pertinence d'une transition à un instant t pour respecter nos contraintes.

A chaque itération, en utilisant des programmes d'optimisation linéaires, nous calculons la consommation électrique pour toutes les combinaisons possibles d'utilisation des systèmes de refroidissement disponibles. Les systèmes en panne, en cours de maintenance ou en dépassement de taux d'utilisation (exemple : dépassement débit d'utilisation de l'eau de la nappe phréatique) sont exclus du calcul d'optimisation. L'objectif est d'utiliser un seul système de refroidissement si possible, ou d'en utiliser deux en cas de forte charge.

L'algorithme est appelé à chaque nouvelle période, en mode prédictif, pour prévoir les changements de charge et de température sur toute la journée. En cas de panne du système de refroidissement utilisé, ou en cas d'augmentation non prévu de la charge informatique, l'algorithme agit en mode réactif pour trouver

Data :

- L'ensemble des mesures des capteurs de température, d'humidité, de consommation électrique...
- L'état des différents systèmes de refroidissement (démarré, arrêté, en panne, en maintenance, autorisé)
- Les prévisions météorologiques sur 24 heures par période d'une heure

Result : Le système de refroidissement le plus économe à utiliser

```
==========================================
liste-prédictions[] = prédiction-charge-serveurs-SARIMA(24h);
prev-météo[] = prévisions-météorologiques(24h);
liste-systèmes[] = liste-système-disponibles();
for i← 0 to 24 do
    //La fonction "optim-linéaire" calcul pour chaque période le(s)
    meilleur(s) système(s) de refroidissement en évaluant toutes les
    combinaisons possibles;
    liste-systèmes-opt[i] = optim-linéaire (liste-systèmes[], list-prédictions[i],
    prev-météo[i]);
end
// meilleurs-transitions renvoie les transitions les plus longues
liste-transitions[] = meilleurs-transitions( liste-systèmes-opt[] ) ;
====================
Pour chaque nouvelle période de la journée, lancer le code suivant :
====================
//nb-transitions-restants représente le nombre de transitions restants sur
une journée;
if liste-système-opt[k] == système-actuel || nb-transitions-restants == 0
then
    retourner système-actuel;
else
    // Si la transition actuel est rentable;
    if k dans liste-transitions[] then
        retourner liste-système-opt[k];
    else
        retourner système-actuel;
    end
end
==========================================
```

Algorithme 1 : Algorithme d'optimisation du refroidissement

une nouvelle solution.

3.6 Expérimentation

L'objectif de ces expérimentations est de mesurer le gain réalisé grâce à notre algorithme d'optimisation et l'avantage des prévisions météorologiques et de la prédiction de charge pour éviter les redémarrages très fréquents des systèmes de refroidissement.

Nous avons réalisé des expérimentations de notre algorithme d'optimisation du refroidissement, sur le datacenter Green d'Eolas en exploitation. Ce datacenter héberge actuellement plus de 2000 applications très hétérogènes (e-commerce, banques, site de collectivités locales...). Malgré une activité très variable sur une journée, comme présenté dans le paragraphe 3.5.1, la consommation globale des serveurs varie très peu. La figure 3.13 montre la variation de la consommation de l'ensemble des équipements informatiques du datacenter (serveurs, équipements réseaux et équipements de stockage). Nous pouvons remarquer que la variation de la consommation est très faible sur une journée ($\approx$ 2kW), ce qui implique un besoin en refroidissement peu variable puisque la puissance de froid demandée est directement liée à la puissance électrique consommée par les serveurs.

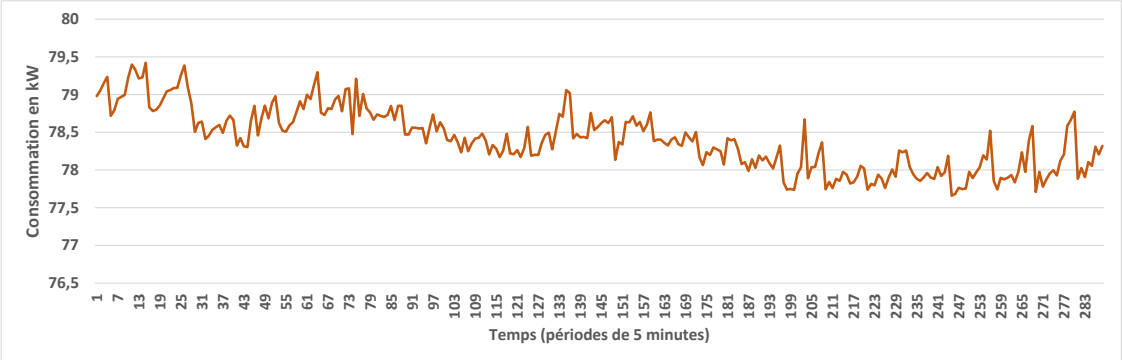


Figure 3.13 – Consommation informatique globale du datacenter d'Eolas sur une journée

Dans la littérature, la consommation électrique d'un serveur est présentée sous forme d'une équation linéaire en fonction de l'utilisation CPU du serveur. La figure 3.14 montre la variation de la consommation d'un serveur en fonction de la charge CPU sur un processeur Intel Haswell.

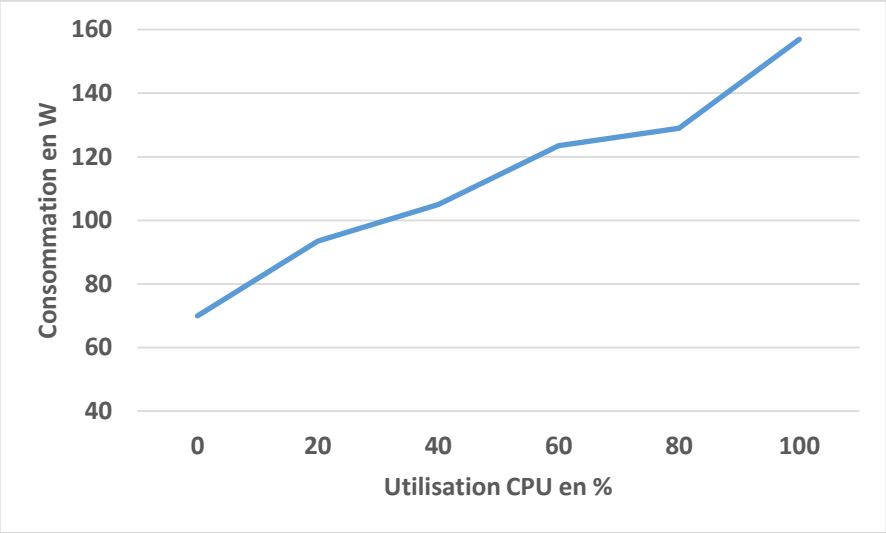


Figure 3.14 – Variation de la consommation d'un serveur équipé d'un processeur Intel Haswell en fonction de la charge CPU

D'autre part, en étudiant la courbe de la variation de la charge CPU dans le datacenter d'Eolas, comme d'ailleurs dans la plupart des datacenters, nous remarquons que le CPU des serveurs est très peu sollicité (en moyenne 15% d'utilisation). La figure 3.15 montre sur une journée le pourcentage CPU moyen dans un cluster en production, contenant 19 hôtes et plus de 340 machines virtuelles. La variation moyenne de la charge CPU est très faible ce qui justifie une consommation globale peu variable tout au long de la journée.

3.6.1 Évaluation

Pour évaluer notre algorithme, nous utilisons un profil de charge variable proportionnel à l'activité du datacenter. La charge électrique du datacenter étant constante, nous démontrons l'efficacité de cet algorithme avec une charge variable sur une journée.

Nous expérimentons le comportement de notre algorithme avec différents profils de température, pour voir comment les prévisions météorologiques permettent d'éviter les effets "yo-yo" de démarrage/arrêt des différents systèmes de refroidissement.

Le but de cette évaluation est de calculer le gain que nous pouvons réaliser grâce à notre algorithme avec une vision globale, par rapport à un système non

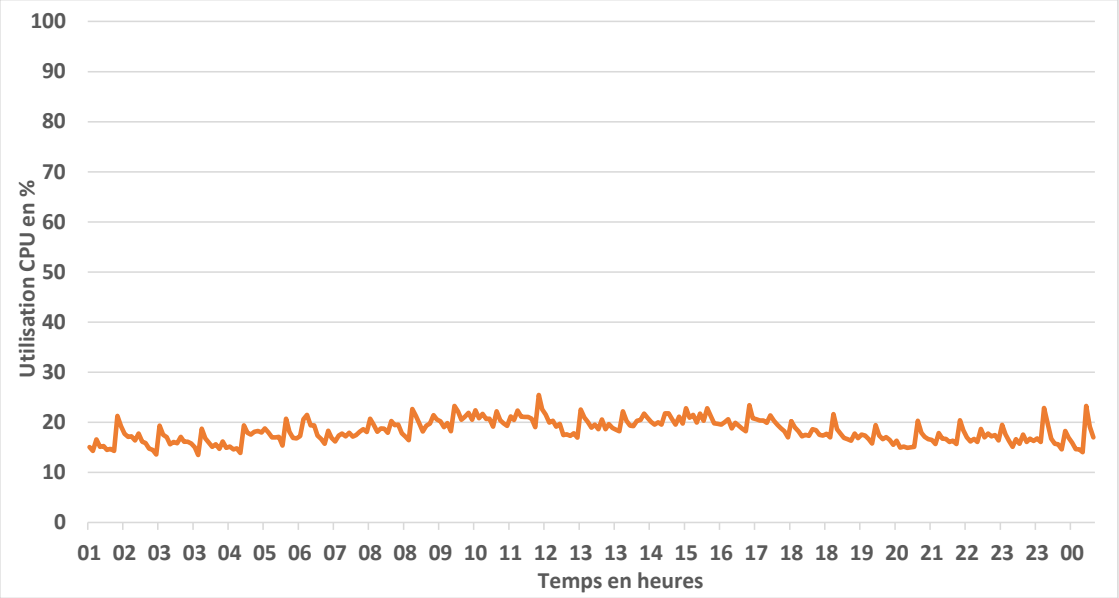


Figure 3.15 – Variation de la charge CPU en % dans un cluster en production

automatisé et géré par des automates industriels ayant une vision très limitée.

3.6.1.1 Choix manuel d'un seul système de refroidissement

Actuellement, le refroidissement du datacenter d'Eolas fonctionne principalement en mode "Natural Cooling" avec les pompes à eau 1 et 2 de la nappe phréatique. Le choix entre les deux pompes d'eau est en fonction de la puissance de refroidissement demandée. Il est possible de démarrer les deux pompes en même temps. Également, La bascule vers les groupes froid se fait automatiquement mais uniquement en cas de panne ou de la maintenance des pompes d'eau. En fonction de la température extérieure, le FreeCooling peut être plus économe que le Natural Cooling.

La figure 3.16 présente la courbe de variation de la consommation totale du système de refroidissement pour une utilisation de la pompe à eau 2 de la nappe phréatique sur une journée complète avec des périodes d'échantillonnage d'une heure. La courbe de température représente une journée froide de l'hiver (25 Février 2015) à Grenoble, avec une température moyenne de 3.34°C. La puissance de refroidissement demandée est variable sur la journée selon l'activité des équipements informatiques. En utilisant les pompes de la nappe phréatique, la

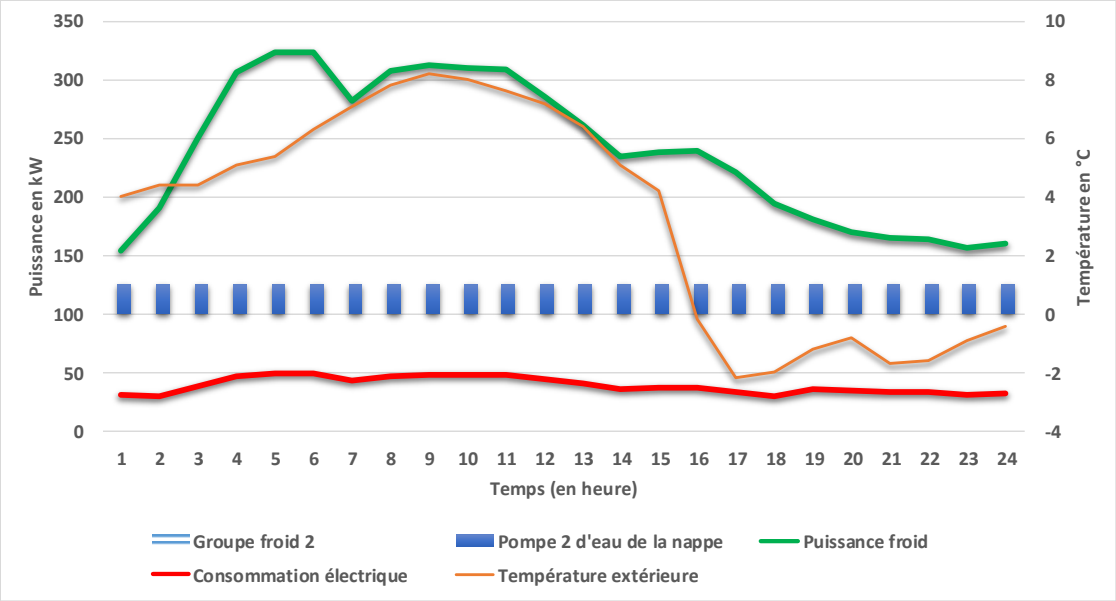


Figure 3.16 – Fonctionnement du datacenter en mode "Natural Cooling"

température n'a pas d'impact direct sur la consommation électrique du "Natural Cooling". La consommation des pompes est directement liée à la chaleur dissipée par les équipements informatiques du datacenter et donc au besoin en refroidissement. La consommation totale du refroidissement sur la journée est de 935,81 kWh pour l'utilisation de la pompe 2 qui est capable de répondre au pic de puissance de refroidissement nécessaire sur la journée.

Dans la figure 3.17, nous utilisons uniquement le groupe froid 2 équipé du module FreeCooling pour refroidir le datacenter. La courbe de puissance de refroidissement fournis ainsi que la courbe de la température est identique à celle de la figure 3.16. Pendant les périodes froides de l'hiver, et en fonction de la puissance informatique, il est plus efficient énergétiquement d'utiliser les systèmes de FreeCooling qui utilisent uniquement les ventilateurs pour refroidir l'eau du datacenter grâce au froid de l'extérieur. La consommation totale du refroidissement sur la journée est de 465,31 kWh soit presque la moitié de la consommation en utilisant la pompe d'eau 2 de la nappe phréatique.

Sans connaissance de la variabilité de la température sur la journée et de la variation de la consommation des serveurs, il est difficile de déterminer manuellement le choix du système de refroidissement le plus économe sur une journée complète. Un mauvais choix de système peut coûter cher et faire grimper la facture

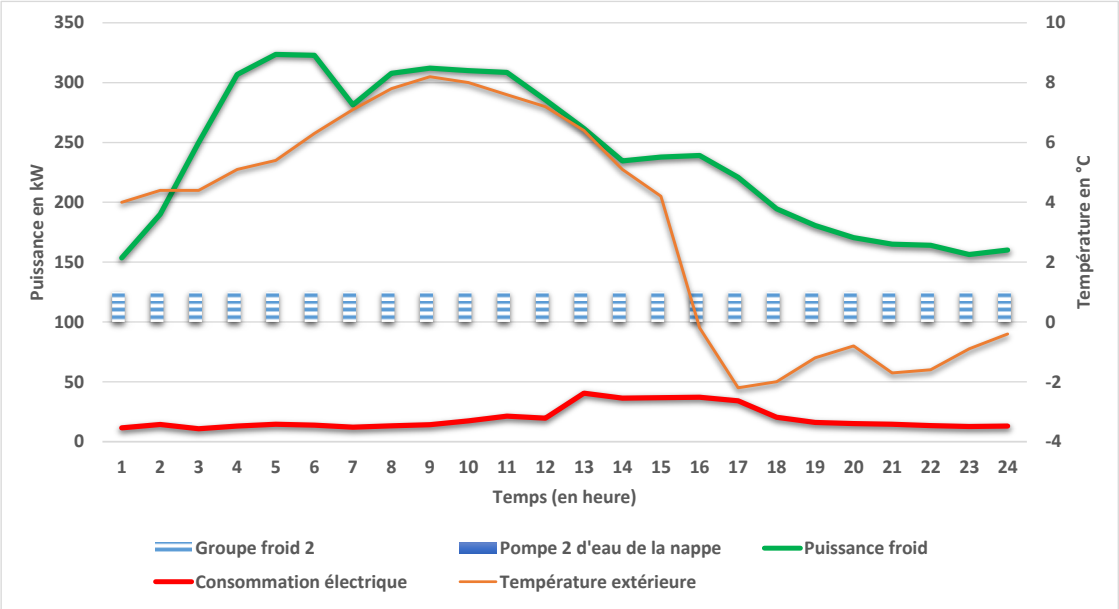


Figure 3.17 – Fonctionnement du datacenter en mode "FreeCooling"

électrique. Sur l'exemple précédent nous passons d'un PUE à 1.27 en utilisant la pompes d'eau 2 de la nappe phréatique, à 1.19 pour l'utilisation du groupe froid équipé du module FreeCooling. Ces premiers résultats montrent l'efficacité de l'algorithme d'optimisation pour améliorer l'efficience énergétique et donc faire baisser le PUE.

3.6.1.2 Optimisation du refroidissement avec une température peu variable

La figure 3.18 montre le résultat de l'application de notre algorithme sur un profil de charge et une température identiques à ceux du paragraphe 3.6.1.1. Durant la journée, et à cause de la variation de la température et du besoin en puissance de refroidissement, notre algorithme propose de passer au refroidissement en mode "Natural Cooling" avec la pompe 2 de la nappe phréatique entre les périodes 13 et 17, soit pendant 5 heures. Nous avons limité à deux le nombre de transitions entre les systèmes de refroidissement pour éviter l'endommagement des équipements électriques. La consommation totale du système de refroidissement passe à 439,68 kWh sur toute la journée, soit un gain de 25.5 kW ($\approx 6\%$) par rapport à l'utilisation du groupe froid 2 toute la journée. Le PUE passe donc de

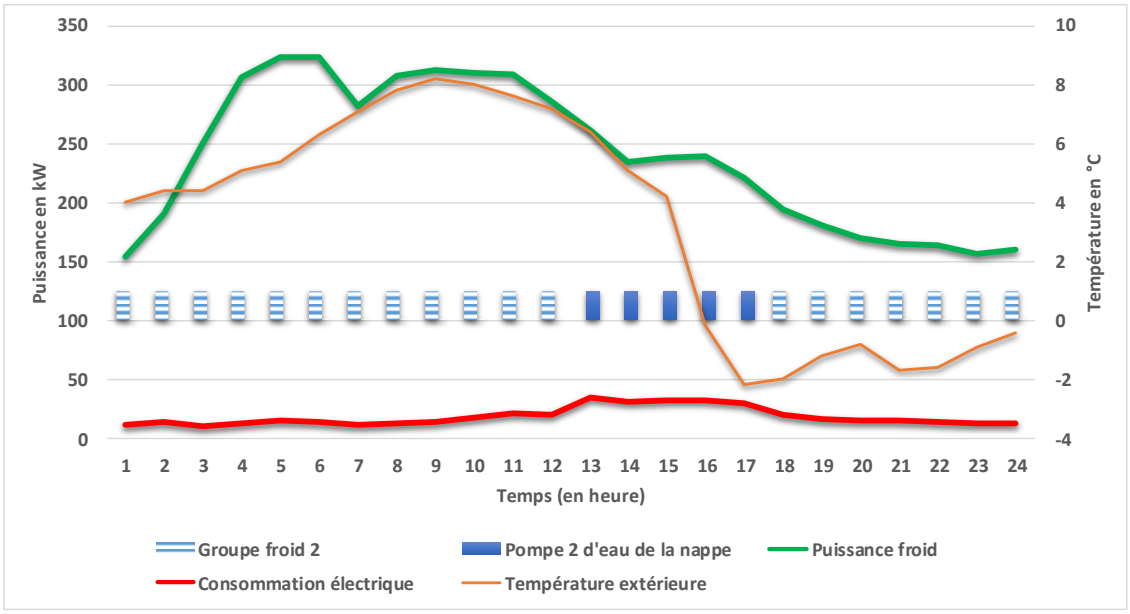


Figure 3.18 – Optimisation du refroidissement avec une température stable

1.19 à 1.18.

3.6.1.3 Exécution de l'algorithme avec des températures variables

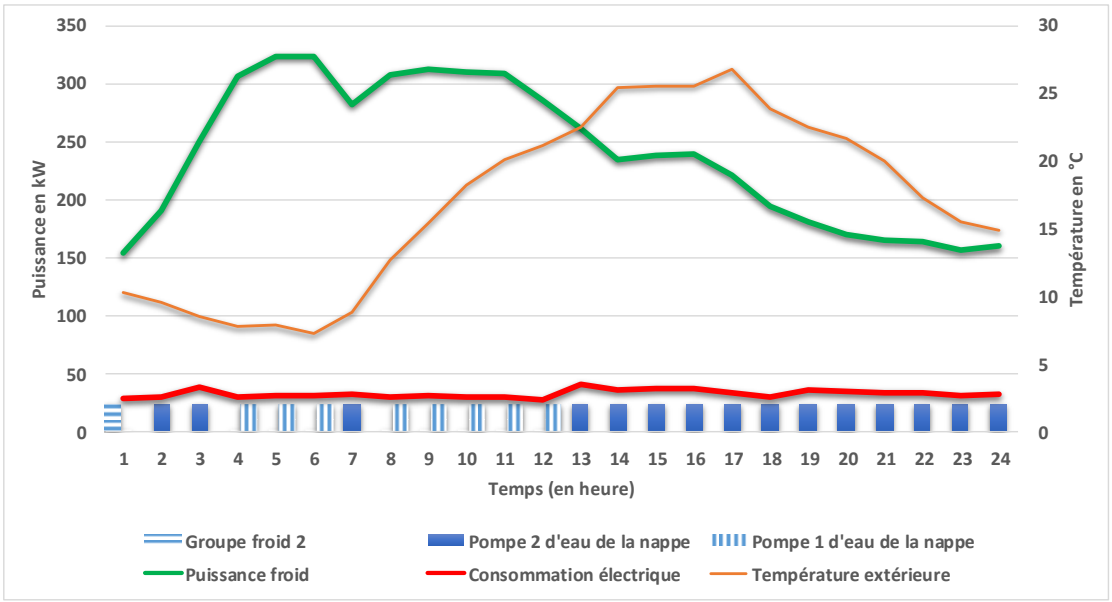


Figure 3.19 – Exécution de l'algorithme sans limitation des transitions

Nous avons évalué notre algorithme sur des journées avec des fortes variations de température, ce qui a pour effet de changer de système de refroidissement très régulièrement. Chaque système de refroidissement à un comportement différent en fonction de la température extérieure. La figure 3.19 montre le résultat de l'exécution de notre algorithme sur une journée avec une charge identique à celle des évaluations précédentes, mais une courbe de température très variable du 25 Avril 2015 à Grenoble. Sans limitation du nombre de transitions entre les différents systèmes de refroidissement, une forte variabilité de la température peut entraîner beaucoup de changement des modes de refroidissement, et donc un coût d'entretien très élevé. Dans L'exemple de la figure 3.19, il faut minimum 5 changements de systèmes de refroidissement sur la journée pour optimiser la consommation, ce qui peut endommager les systèmes électriques et augmenter le risque de pannes.

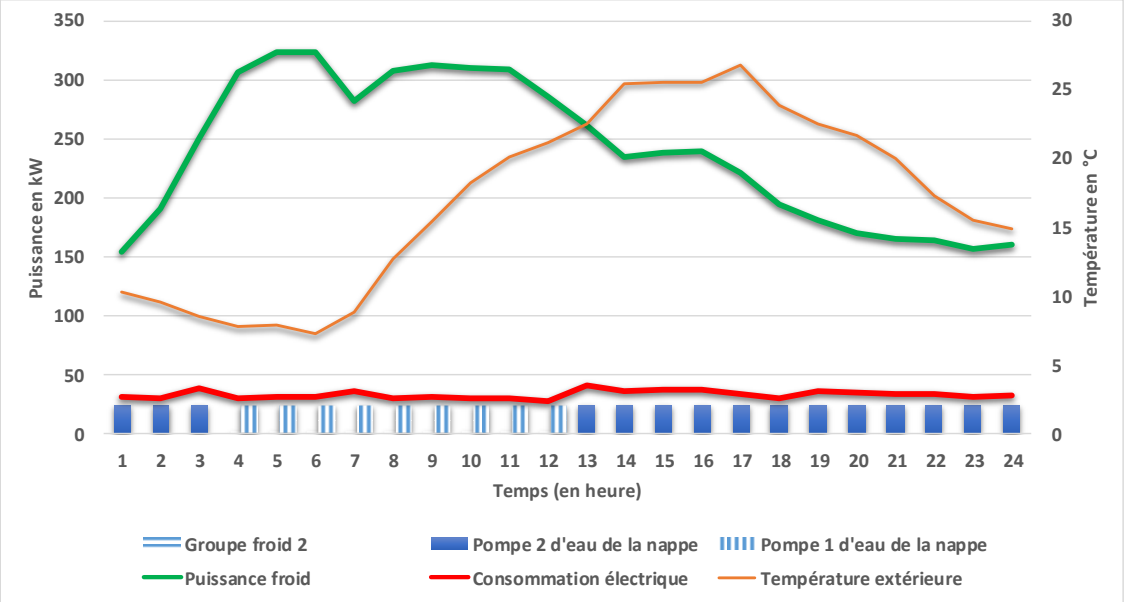


Figure 3.20 – Exécution de l'algorithme avec limitation du nombre de transitions

Dans notre algorithme, nous limitons le nombre de transitions entre les différents systèmes de refroidissement pour réduire le taux de panne. Grâce aux prévisions météorologiques, dans la figure 3.20, les transitions de système de refroidissement qui sont de courte durée ont été éliminés par rapport à la figure 3.19. Nous avons donc uniquement deux transitions entre les pompes 1 et 2 de la

nappe phréatique. La consommation totale sur la journée a légèrement augmenté (moins de 1%), et le PUE est resté le même à 1.24.

3.7 Conclusion

En étudiant les limitations des automates industriels et les redondances dans les systèmes de refroidissement et d'alimentation électrique, nous avons défini un algorithme permettant d'optimiser le système de refroidissement tout en réduisant les pannes. Avec une vision globale couvrant les FaaS, IaaS et même PaaS, cet algorithme permet de compléter les automates industriels déjà en place par du logiciel, en ajoutant plus d'intelligence pour améliorer et optimiser le fonctionnement du datacenter. Le projet vise à concevoir un système de gestion d'un datacenter énergétiquement optimisé, par la mise en adéquation permanente des ressources énergétiques fournies par l'infrastructure (électricité et climatisation) avec les besoins applicatifs de l'informatique.

Garder les équipements informatiques dans un environnement stable (température et humidité constants) permet de les protéger contre les pannes. Par ailleurs, le changement fréquent de la température de la salle informatique peut avoir un impact sur les performances et peut donc affecter le SLA des applications du datacenter.

Actuellement, le changement du mode de refroidissement se fait manuellement. Notre algorithme permet une transition automatique entre les différents systèmes de refroidissement, en utilisant des paramètres externes tel que les prévisions météorologiques. Nous utilisons deux modes dans notre algorithme : réactif et prédictif. Le mode réactif a pour effet d'adapter la consommation énergétique au besoin courant de l'application informatique, en quasi temps réel. Le mode prédictif permet de prendre en compte des prévisions d'activité informatique pour limiter le décalage entre la demande IT (CPU par exemple) et la fourniture d'énergie. Enfin, le mode continu offre une gestion très globale de l'énergie, en proposant à l'administrateur des stratégies de planification des tâches applicatives ou de sélection des sources de froid. Par ces différents biais, le projet estime pouvoir engendrer des économies d'énergie notables que ce soit dans les datacenters récents ou dans les datacenters qui font l'objet d'une rénovation.

Un démonstrateur en condition réelle d'exploitation dans un datacenter (Le

datacenter Green d'Eolas) permet de démontrer de façon tangible les résultats obtenus et fournit des informations complémentaires sur son fonctionnement afin d'affiner les actions d'optimisation. Les gains réalisés montrent l'apport d'un système coordonné par rapport aux automates industriels qui contrôlent les systèmes de refroidissement séparément.

4

Un système scalable de détection des anomalies en temps réel

Contents

4.1 Contexte 78

4.2 Approche et contribution 80

4.3 Méthode ACP 82

4.4 Architecture 84

4.4.1 La couche de traitement par lots 84

4.4.2 La couche de traitement temps réel 86

4.5 Conception 88

4.5.1 Collections de données 88

4.5.2 Création du modèle avec Hadoop 90

4.5.2.1 Extraction des données et construction des séries temporelles 90

4.5.2.2 Calcul de la covariance 91

4.5.2.3 Calcul du modèle ACP 92

4.5.3 Détection en temps réel des attaques avec Storm . . . 92

4.6 Expérimentation 94

4.6.1 Clusters d'évaluation 94

4.6.1.1 Cluster Hadoop 95

4.6.1.2 Cluster Apache Storm 96
4.6.2 Architecture réseau des datacenters Eolas 97
4.6.2.1 Journaux des pare-feux 97
4.6.3 Évaluation 98
4.6.3.1 Mesures de performance du traitement par lots 98
4.6.3.2 Mesures de performance de la topologie Storm 102
4.6.3.3 Détection des attaques 103
4.7 Conclusion 105

Dans ce chapitre, nous présentons notre deuxième contribution pour la détection des anomalies dans un datacenter, en utilisant des outils d'analyse et de traitement de flux de données Big Data. Nous détaillerons une implémentation de l'architecture pour la détection en temps réel des attaques réseau de type Déni de Service Distribué DDoS sur des données réelles provenant d'un datacenter en exploitation.

4.1 Contexte

Un datacenter est constitué d'un ensemble d'équipements matériels sophistiqués et à grande échelle pour fournir une puissance de calcul et une capacité de stockage très élevées. La puissance de calcul et la capacité de stockage sont fournies par une plateforme formée par un nombre important de serveurs physiques puissants interconnectés via un réseau à haut débit. De nos jours, le trafic réseau d'un datacenter est de plus en plus important (plusieurs dizaines de Gb/s). A cause de la forte augmentation du volume de trafic, la taille des fichiers de journaux a considérablement augmenté, et les techniques de fouille de données actuelles qui permettent de détecter les attaques et identifier les attaquants et les localiser, ne sont pas capables de gérer toutes ces données.

La détection du trafic anormal est une partie critique de la gestion des datacenters. Un des problèmes majeurs est la détection des attaques distribuées de type déni de service DDoS. Une attaque DDoS vise à rendre un serveur, un service ou une infrastructure indisponible en surchargeant la bande passante du

serveur, ou en saturant ses ressources. Il peut s'agir par exemple de l'ouverture d'un grand nombre de nouvelles sessions TCP dans un intervalle de temps très court. Lors d'une attaque DDoS, une multitude de requêtes sont envoyés simultanément, parfois en provenance de multiples machines souvent compromises. C'est le type d'attaque commun à la plupart des applications en ligne et présente de graves menaces pour les serveurs. Avec l'augmentation rapide du volume de trafic internet, les technologies actuelles de détection des attaques DDoS ont rencontré de nouveaux défis pour faire face efficacement à l'énorme quantité de trafic dans un délai de réponse raisonnable. La clé pour se protéger contre l'attaque DDoS est de la détecter le plus rapidement possible et neutraliser son effet rapidement pour restaurer la qualité des différents services à des niveaux acceptables par les utilisateurs.

Les attaques par déni de service distribué DDoS sont aujourd'hui fréquentes, notamment du fait de la relative simplicité de leur mise en oeuvre, et de leur efficacité contre une cible non préparée. Ces attaques peuvent engendrer des pertes financières non négligeables par l'interruption de service ou encore indirectement, par l'atteinte portée à l'image de la cible. Les opérateurs français ont constaté jusqu'à plus d'un millier d'attaques par jour en 2014. Outre l'augmentation en nombre, l'ampleur des attaques a augmenté de manière significative au cours des dernières années [25]. Ainsi, en 2014, des opérateurs français ont constaté des attaques dont le volume était de l'ordre de la centaine de gigabits par seconde, et le débit, en nombre de paquets par seconde, de l'ordre de la dizaine de millions.

Pour toutes ces raisons, il est nécessaire d'anticiper cette menace, et de prendre un certain nombre de mesures techniques et organisationnelles afin d'y faire face. La première étape est la collecte d'informations multiples, indispensable pour connaître l'état de santé du datacenter, que l'on peut regrouper en deux types :

- Les informations issues d'une source interne au datacenter : les journalisations (pare-feu, routeur, commutateur, serveurs, HIDS, applications, contrôle d'accès...), le système de monitoring temps réel et de métrologie

- Les informations issues d'une source externe au datacenter : réseaux sociaux, IP malveillantes...

Avec la croissance de l'activité du datacenter, et par conséquent le nombre de clients et de plateformes hébergés, la volumétrie des données de trafic ne

cesse d'augmenter et la possibilité d'attaque devient plus importante. Une attaque importante peut paralyser tout le trafic du datacenter et peut donc avoir un impact important si elle n'est pas détectée et bloquée rapidement. Les responsables de la sécurité de datacenter font face au défi de respecter les SLA, et notamment la disponibilité et la sécurité des services. Les attaques réseaux qui ciblent une application peuvent avoir un impact sur la disponibilité des applications d'autres clients du datacenter. De même, certaines attaques exploitent des failles de sécurité sur des systèmes vulnérables, dans le but de récupérer des données parfois sensibles.

En utilisant des systèmes de détection basés sur des règles, l'attaquant est capable de contourner ces outils et peut donc identifier les règles de filtrage. L'utilisation des modèles statistiques a été proposée comme solution pour rendre le contournement plus difficile. Mais les outils d'analyse actuels ne sont pas capables d'analyser les gros volumes de données produites à l'échelle d'un datacenter. L'utilisation des solutions Big Data pour le calcul du modèle statistique est une nouvelle approche. Pour ensuite détecter les attaques en temps réel, les outils de traitement de flux de données Big Data, se basant sur le modèle statistique, fournissent un résultat fiable sans latence.

4.2 Approche et contribution

Avec l'augmentation du trafic réseau et donc de la taille des fichiers de journaux des pare-feux (plusieurs dizaine de Go par jour), il est impossible de traiter ces données avec des outils traditionnels de fouille de données. Les outils de traitement des données Big Data sont adaptés au traitement par lots de gros fichiers de données pour les analyser et ensuite détecter les attaques et les géolocaliser. Les outils de traitement de flux Big Data viennent compléter notre architecture pour la détection en temps réel des anomalies.

La plupart des solutions Open Source de détection des attaques réseaux utilisent l'analyse de paquets basée sur des règles et des signatures prédéfinies [73, 82]. L'attaquant est capable de contourner ces outils en envoyant des attaques sophistiqués susceptibles de découvrir les règles de filtrage et donc générer ensuite un trafic qui ne sera jamais détecté par ces logiciels. L'utilisation de modèles statistiques pour le filtrage en se substituant aux règles, en calculant des corrélations entre différents attributs, empêche l'attaquant de dévoiler les règles statistiques de

détection et la rend très robuste et impossible à contourner.

Le modèle de détection des anomalies que nous utilisons est basé sur l'analyse de composants principaux ACP (PCA) [36], une méthode puissante pour détecter une grande variété d'anomalies. Le système de protection des attaques DDoS basé sur la méthode ACP extrait les caractéristiques du trafic nominal en analysant les dépendances intrinsèques entre les plusieurs valeurs d'attributs. Le schéma basé sur ACP différencie les paquets d'attaques de ceux légitimes en vérifiant le volume du trafic actuel de la valeur d'attribut associée qui viole la dépendance intrinsèque du trafic nominal.

Nous proposons une plateforme distribuée et scalable pour l'analyse des journaux des pare-feux en temps réel afin de détecter des attaques de type DDoS. Nous utilisons une architecture lambda [48] de calcul, unifiant deux système de calcul, le traitement par lots et le traitement de flux en temps réel. Cette architecture est composée de deux couches (Figure 4.1) : la couche de calcul du modèle utilisant le Framework Hadoop MapReduce [33] permettant d'analyser les données Big Data rapidement grâce au traitement par lots, distribué et parallèle, et la couche de traitement des flux de données de trafic en temps réel utilisant la solution Storm [76].

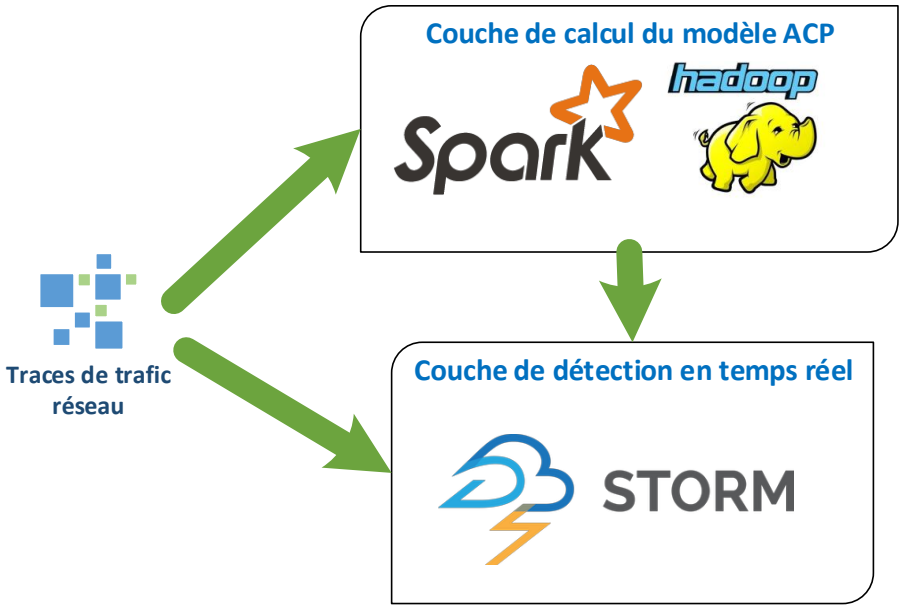


Figure 4.1 – Architecture générale de la plateforme de détection

La couche Hadoop est orientée traitement par lots et se focalise sur la scalabilité de la création du modèle statistique à partir de gros volumes de données, et permet donc de minimiser le temps de traitement. Le temps de calcul du modèle ne peut pas être fait en temps réel et dépend du volume des données et de la taille du cluster. D'où l'importance de la couche Storm de traitement en temps réel. Cette deuxième couche utilise le modèle calculé, et permet de réaliser un système de protection en temps réel, pour détecter le trafic anormal plus tôt pour et réagir afin d'éviter les latences sur le service client et préserver ainsi la qualité du SLA.

L'architecture que nous proposons est très générique et peut être utilisée pour la détection d'anomalies d'autres processus, tel que les anomalies de fonctionnement des "Facilités" dans un datacenter en fournissant en entrée les données des capteurs et en adaptant le modèle de détection des anomalies.

Nous validons notre solution en l'appliquant sur un trafic réseau réel provenant d'un datacenter en production, avec des anomalies connues et identifiés. Les expérimentations montrent que la méthode présentée est efficace pour la détection des attaques DDoS et possède un niveau de détection élevé en temps réel.

4.3 Méthode ACP

La méthode d'analyse de composantes principales ACP, est une méthode statistique d'analyse de données qui repose sur l'analyse des relations entre un nombre important de variables, et consiste à transformer des variables liées entre elles ("corrélées") en nouvelles variables décorrélées les unes des autres. Ces nouvelles variables sont nommées "composantes principales", ou axes principaux. Cette méthode permet de réduire le nombre de variables et de rendre l'information moins redondante tout en gardant un maximum d'informations. Nous décomposons le trafic en deux sous-espaces qui résultent de l'analyse ACP :

- Trafic usuel et prévisible (variations périodiques, quotidiennes) : projection sur les k premières composantes principales.
- Trafic résiduel ou "anormal" : projection sur le sous-espace complémentaire.

La figure 4.2 montre un exemple de décomposition du trafic réel en deux sous-espaces (trafic prévisible et résiduel) avec la méthode ACP en utilisant 10 jours d'historique.

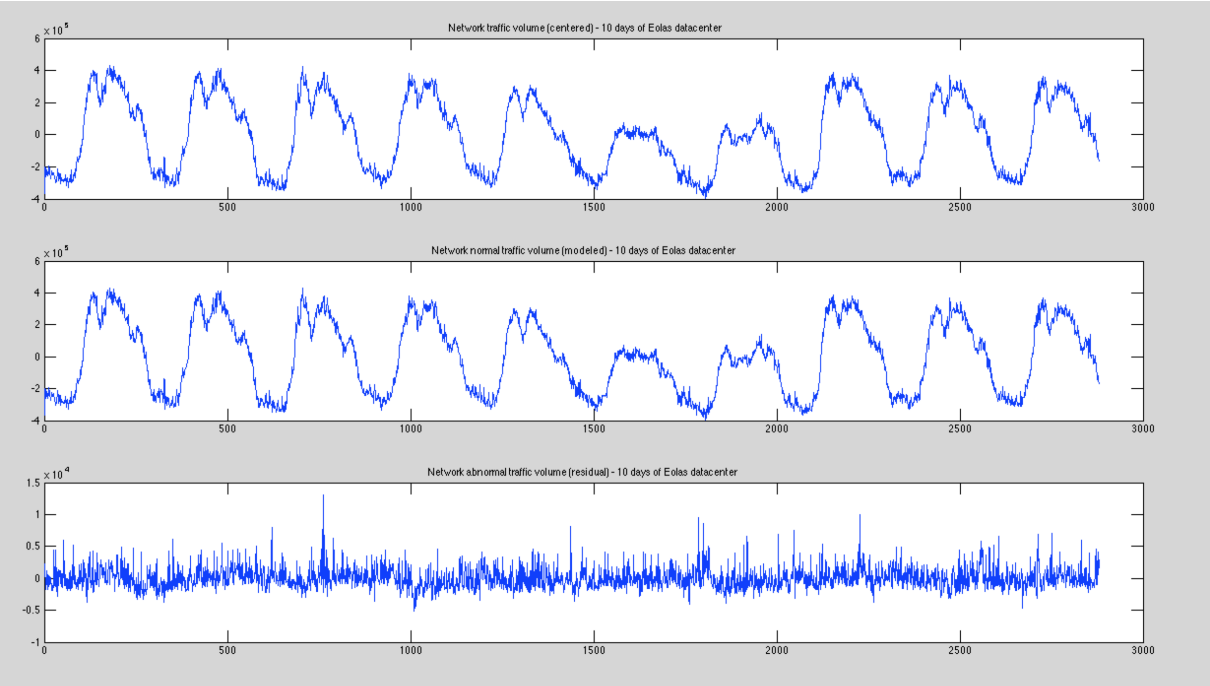


Figure 4.2 – Décomposition du trafic réel en trafic normal et anormal (Données du datacenter Eolas)

En se basant sur les statistiques des mesures, on peut détecter les attaques DDoS, même celles de type "never before seen" (jamais vu). Les attaquants ne connaissent pas le profil du trafic nominal de la victime et donc ne peuvent pas truquer un trafic normal. La détection des attaques DDoS basé sur l'analyse de composantes principales ACP, permet à partir des données de trafic nominal (normal), d'analyser les dépendances entre plusieurs valeurs d'attributs. Le schéma de détection d'anomalies est construit sur la composante "anormale" du trafic : des anomalies visibles sur les projections sur les composantes résiduelles alors qu'elles sont noyées quand le flux est projeté sur les k premières composantes. Le schéma ACP permet de différencier un trafic légitime d'une attaque en comparant le volume de trafic actuel d'un attribut par rapport au trafic nominal.

Les variables dans notre modèle représentent les clés distinctes d'une trace réseau (IP source, IP destination, protocole...) et les métriques correspondent aux valeurs de chaque variable à une période donnée. Nous avons au minimum plusieurs centaines de milliers de variables dans le trafic réseau du datacenter. Le modèle ACP permet de choisir le niveau de variance des données qu'on peut expliquer.

C'est le nombre de composantes (paramètre k) qui détermine ce niveau. Dans notre cas, avec $k = 10$ on arrive à expliquer plus de 99% de la variance des données brutes. A partir des journaux de pare-feux pouvant atteindre plusieurs centaines de gigaoctets, nous obtenons un modèle ACP réduit de quelque mégaoctets, qui représente 99% des données, qu'on peut ensuite utiliser pour filtrer le trafic en temps réel.

4.4 Architecture

Dans cette section nous présentons notre architecture pour la collecte et la transformation des données, le calcul du modèle et la détection en temps réel de nouvelles attaques réseaux de type DDoS. Notre objectif est de proposer une architecture générique qui permet de détecter les attaques en utilisant le modèle ACP.

La figure 4.3 montre que les traces de trafic réseau sont collectées depuis plusieurs pare-feux et stockés dans un système de fichier distribué. Le volume de données peut être énorme et augmente avec l'augmentation du nombre de pare-feux. Chaque pare-feu peut traiter des milliers de paquets par seconde et donc produire plusieurs Go de données par jour. Notre architecture est composée de deux couches de traitement, distribuées, scalables et avec tolérance aux pannes : traitement par lots et en temps réel. Les deux sections suivantes présentent ces deux couches et la communication entre elles.

4.4.1 La couche de traitement par lots

La couche de traitement par lots utilise l'historique des traces de trafic transitant par des pare-feux, stockés en HDFS, pour extraire l'empreinte du trafic normal pour un environnement donné et créer le modèle en utilisant des méthodes numériques de calcul. Dans un environnement hétérogène, tel qu'un datacenter, plusieurs pare-feux seront déployés pour filtrer l'énorme trafic réseau et pour assurer la redondance en cas de panne. Plus les journaux de trafic réseau sont grands, plus la durée de calcul du modèle est longue, et donc le processus de création du modèle ne peut pas être accompli sur une seule machine.

Généralement, dans la plupart des datacenters, le trafic réseau suit un cycle journalier ou hebdomadaire : trois pics de charge par jour, avec un pic plus

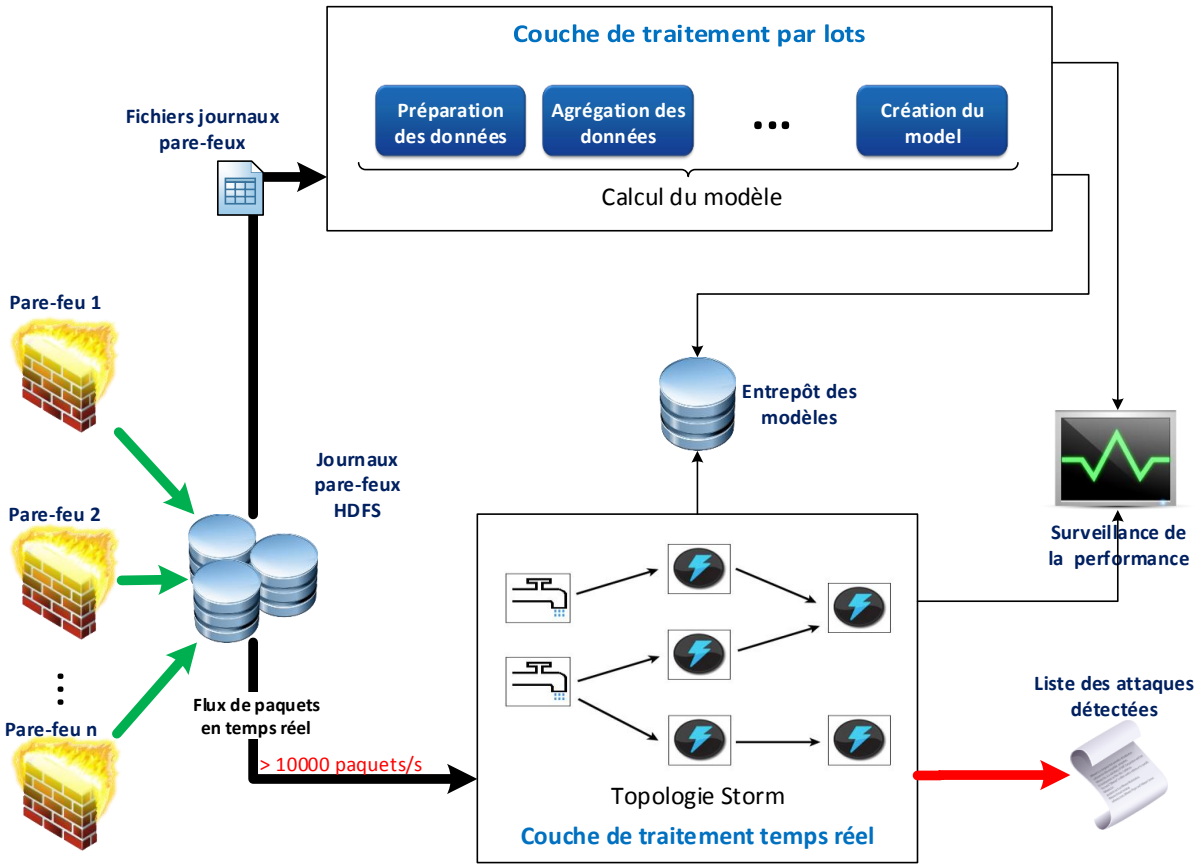


Figure 4.3 – Architecture globale de l'architecture Lambda

important vers midi, et un trafic similaire entre les jours travaillés de la semaine (du lundi au vendredi) et entre les jours du week-end. Nous pouvons parfois identifier des cycles mensuels ou annuels. De plus, nous aurons besoin de créer plusieurs modèles pour chaque clé d'une trace de paquet (IP source, IP destination, protocole, ...) et par type d'agrégation (nombre total de paquets, taille moyen d'un paquet...). Nous aurons alors besoin de calculer plusieurs modèles selon le cycle du trafic et les différentes clés des traces. La méthode de calcul distribuée que nous proposons permet de calculer plusieurs modèles sur un très gros volume de données, de manière rapide et optimisée. Dans cette couche, nous utilisons le Framework Big Data Hadoop [33] pour le stockage et le traitement à large échelle de gros volumes de données sur des clusters de noeuds distribués.

Cette couche récupère les journaux des pare-feux et calcule un modèle statistique grâce au modèle de programmation Map-Reduce. Le système de fichier

d'Hadoop HDFS est un système de fichiers distribués avec tolérance aux pannes, conçu pour le stockage de données volumineuses et que nous avons utilisé pour le stockage de tous les archives des journaux des pare-feux. Une journée de traces de pare-feux peut atteindre plusieurs dizaines de gigaoctets, et la décomposition d'un fichier en plusieurs blocs distribués en HDFS ainsi que la réplication des blocs représente un atout pour notre système. Le modèle de programmation MapReduce est adapté pour les traitements de gros ensembles de données grâce au calcul distribué et parallèle. MapReduce accède au système de fichiers HDFS pour effectuer les calculs sur les traces pare-feux et peut ensuite générer plusieurs modèles en parallèle. La tolérance aux pannes et la scalabilité en stockage et en puissance de calcul, rendent cette couche adaptable pour un hébergement mutualisé et variable. Les phases de calcul du modèle seront détaillées dans la section 4.5.2.

En utilisant cette couche de traitement, nous pouvons calculer et mettre à jour le modèle régulièrement dans l'entrepôt des modèles. Généralement, les modèles générés sont trop petit (quelques méga-octets), par exemple 5Mo pour le modèle ACP. L'entrepôt des modèles utilise un système de fichier distribué simple, adapté pour l'accès rapide aux petits fichiers de données. Nous pouvons générer plusieurs modèles pour chaque clé de flux et pour chaque type d'agrégation sans impact sur le système de détection temps réel des attaques. Donc, la détection ne peut pas être faite à l'intérieur de la couche de création du modèle, d'où l'importance de dissocier la couche de traitement par lots de la couche de détection en temps réel. Grâce à la séparation en couches et l'utilisation de modèles pré-calculés, la couche temps réel peut détecter rapidement les attaques DDoS sur les nouveaux flux temps réel des pare-feux.

4.4.2 La couche de traitement temps réel

Le but de cette couche est la détection en temps réel des attaques réseau de type DDoS sur un trafic réseau continu. Cette couche doit permettre l'utilisation des modèles calculés dans la couche de traitement par lots, pour détecter rapidement les attaques. Les pare-feux envoient les journaux du trafic de paquets filtrés en temps réel à cette couche. Les flux seront filtrés et agrégés pour permettre la détection du trafic anormal en utilisant le modèle stocké dans l'entrepôt de modèles. Il existe plusieurs solutions de streaming Open Source (Flume, Apache Storm, Yahoo S4...). Nous avons choisis le Framework de streaming Apache Storm [76],

un système de traitement temps réel, distribué et scalable avec tolérance aux pannes, capable de traiter plusieurs centaines de milliers de messages en parallèle.

Apache Storm garantit que chaque tuple sera traité dans la topologie en temps réel, même sur d'importants flux de données. Ce Framework est très facile à mettre en place et à configurer. En utilisant Storm, nous pouvons construire des topologies. Une topologie est un graphe de Spouts et de Bolts connectés avec des groupes de flux. Un flux est une séquence illimitée de tuples. Un Spout est une source de flux et le Bolt est un composant de traitement dans la topologie.

Le Bolt qui permet la détection des attaques, récupère les modèles calculés dans la couche de traitement par lots à travers l'entrepôt des modèles. Le processus de détection des attaques est indépendant du processus de création du modèle et peut tourner pendant que la couche de traitement par lots met à jour les modèles. Selon la clé choisie ou le type d'agrégation, on peut utiliser le modèle calculé correspondant dans l'entrepôt des modèles. En récupérant les nouveaux journaux des pare-feux, cette couche peut fournir des alertes sur des possibles attaques de type DDoS en temps réel. Ces alertes peuvent déclencher des notifications ou lancer des actions spécifiques basées sur ce qui a été identifié (blocage ou contournement du trafic vers un trou noir).

La couche de traitement temps réel peut fonctionner avec tout modèle statistique, il est donc possible de modifier le modèle généré ou la méthode d'analyse dans la couche de traitement par lots sans avoir à modifier la couche de détection temps réel. Il est aussi possible de détecter plusieurs types d'attaques basées sur la clé ou sur la corrélation de comportements anormaux sur un ensemble de clés. Il est tout à fait envisageable d'utiliser plusieurs modèles statistiques ou même des modèles basés sur les règles (exemple : Snort [73]) en parallèle dans notre architecture sur un même flux. Chaque modèle possède un atout pour un type spécifique d'attaque.

Notre couche de traitement temps réel est capable de fournir des mesures statistiques en temps réel (nombre de paquets par seconde, taille totale des paquets en octets...) sur le trafic, et peut même fournir des statistiques par plateforme et par client en la connectant à une base d'inventaire (CMDB).

4.5 Conception

Cette section détaille notre implémentation des deux couches de traitement par lots et en temps réel. La figure 4.4 montre l'architecture globale du système et comment les données sont récupérées, ensuite transformées avant d'être analysées pour le modèle ACP généré par Hadoop.

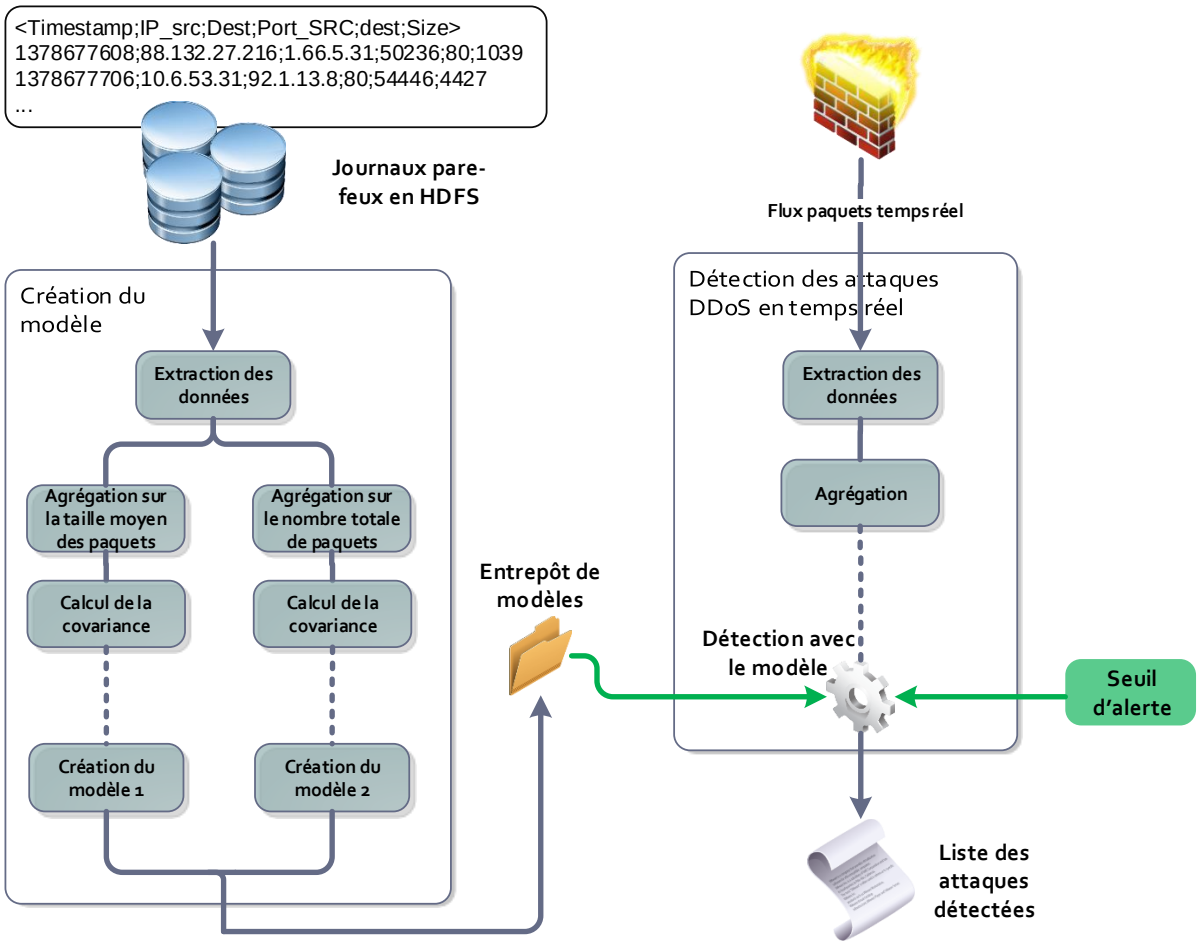


Figure 4.4 – Chaine de traitement globale données de pare-feux

4.5.1 Collections de données

Les données collectées pour alimenter notre architecture proviennent des journaux des pare-feux d'un datacenter hébergeant des milliers d'applications clientes hétérogènes et avec des offres de Cloud IaaS, PaaS et SaaS. L'accès

aux applications hébergées est filtré à travers plusieurs pare-feux qui assurent la redondance et qui gèrent une quantité très importante de trafic. Chaque pare-feu génère plusieurs gigaoctets voir plusieurs dizaines de gigaoctets de traces de trafic par jour, ce qui correspond à plusieurs centaines de millions de lignes. Les traces des paquets sont stockés dans les deux directions, en provenance et à destination du datacenter. Notre architecture récupère des fichiers texte contenant la liste des flux réseaux échangés, avec une date et plusieurs champs séparés par des ';' comprenant des informations sur ce flux. Voici un exemple d'un journal de pare-feu ordinaire renvoyé par l'outil "Syslog" :

```
<TIMESTAMP;IP_SRC;IP_DEST;PORT_SRC;PORT_DEST;SIZE>
1378677608;88.132.27.216;1.66.5.31;50236;80;1039
1378677706;10.6.53.31;92.1.13.8;80;54446;4427
1378677739;10.6.1.2;88.19.55.16;53;64687;144
...
```

Chaque ligne du journal correspond à un flux de paquets, soit une succession de paquets ayant les mêmes source et destination. Le champ **Timestamp** correspond à la date Linux de la réception du paquet en secondes (nombre de secondes depuis le 01/01/1970). En moyenne, un pare-feu peut traiter plusieurs dizaines de milliers de paquets par seconde. **IP_SRC** et **IP_DEST** correspondent respectivement aux IP source et destination de chaque flux (l'IP peut être privée ou publique). Dans le cas d'Eolas, les plateformes utilisent des adresses réseau en "/24", donc avec les trois premiers octets d'une IP, nous pouvons identifier une plateforme ou un client. Nous utilisons des préfixes IP communs pour créer nos modèles. **PORT_SRC** et **PORT_DEST** correspondent aux numéros de port, source et destination du flux. Le numéro de port peut donner une indication sur le type d'application et le protocole utilisé (exemple : 80 pour le protocole HTTP et 21 pour FTP). Dans une application Web, seuls les serveurs frontaux sont accessibles depuis internet, pour cela seules les communications des serveurs web sont tracées. Le dernier champ **SIZE** correspond à la taille du flux en octets.

Les traces peuvent être enrichies avec des champs supplémentaires permettant de mieux connaître le flux. Le protocole (TCP, UDP, ICMP, ...) permet d'identifier certains types d'attaques qui ciblent les protocoles les moins utilisés tels que le protocole UDP. Le nombre de paquets dans un flux peut donner une idée sur la taille des paquets et donc identifier des attaquants qui essayent de voler les

données d'un client ou ceux qui génèrent un grand trafic de petits fichiers pour saturer le réseau.

4.5.2 Création du modèle avec Hadoop

Dans cette sous-section, nous allons détailler la chaîne de création du modèle ACP. Le processus de création du modèle est composé de plusieurs étapes, commençant par la récupération des données du trafic en entrée jusqu'à la génération du modèle ACP final en sortie. Chaque étape de la chaîne correspond à un traitement MapReduce distinct, dont les données proviennent du résultat du traitement précédent. Des phases intermédiaires de préparation des données ne seront pas détaillées dans la suite.

4.5.2.1 Extraction des données et construction des séries temporelles

Nous collectons l'ensemble de fichiers de journaux des pare-feux depuis le système de fichier HDFS sur une période continue suffisamment longue pour prendre en compte la variabilité du trafic entre les différentes périodes de la journée et entre les jours de la semaine. Nous extrayons ensuite les données utiles pour chaque clé et calculons les agrégations. Trois paramètres sont nécessaires pour cette phase :

- La clé d'agrégation : IP source, IP destination, port source, port Destination, protocole...
- Le préfixe des clés de type adresse IP (/8, /16, /24, /32) source ou destination. Plus le préfixe est long, plus le nombre de clés distinctes est important.
- La période d'échantillonnage pour l'agrégation : 30 secondes, 1minute, 5minutes, 1heure...
- Le type d'agrégation : taille moyenne d'un flux, nombre total de flux...

Pour chaque clé, nous disposons d'une mesure agrégée correspondante à chaque période (Equation 4.1).

$$nombrePeriodes = dureeTotale/dureePeriodeEchantillonnage \tag{4.1}$$

Exemple : Si on choisit une période d'échantillonnage de 60 secondes sur une durée totale de 1 jour, nous aurons 1440 valeurs agrégées pour chaque clé correspondante aux 1440 périodes de 60 secondes sur les 24 heures.

En utilisant les données extraites de cette étape, nous construisons la matrice X complète du trafic du réseau agrégé sur la période totale. Chaque ligne de la matrice correspond à une clé, et chaque colonne correspond à une mesure agrégée par clé (ou variable) sur une période donnée. Le nombre totale de lignes de la matrice est égal au nombre de clés distinctes et le nombre maximal de colonnes est égale au nombre maximal de périodes. Le temps de calcul nécessaire pour la création de la matrice dépendra de la taille de données en entrée. La taille totale de la matrice dépendra du nombre de clés distinctes (lignes) et du nombre de périodes totales (colonnes), et peut atteindre plusieurs gigaoctets, voir plusieurs dizaines de gigaoctets.

A partir de la matrice X, nous calculons la matrice moyenne $X - m$ qui sera utilisée pour le calcul de la covariance dans la phase suivante.

4.5.2.2 Calcul de la covariance

Pour les calculs mathématiques et statistiques, nous utilisons la librairie java Mahout [55] d'Hadoop qui se base sur le Framework MapReduce. Apache Mahout est un projet qui implémente des algorithmes d'apprentissage automatiques distribués, ce qui permet de lancer des calculs complexes rapidement sur de très gros volumes de données. Nous utilisons la fonction Mahout "matrixmul" de multiplication de matrices pour calculer la covariance de la matrice $X - m$ générée précédemment après avoir transformé la matrice en un fichier séquentiel pris en charge par Mahout. La matrice covariance (Equation 4.2) résultante sera le point d'entrée de la méthode de création du modèle ACP. Le temps de calcul de la covariance est très long puisqu'il consiste à multiplier deux énormes matrices, difficiles à découper dans un calcul de type MapReduce basé sur les couples de clés/valeurs. Avec l'augmentation du nombre de clés, le temps de calcul de la covariance augmente exponentiellement. La complexité du calcul est de l'ordre de $O(n^3)$.

$$Cov = (X - m)^t * (X - m) \tag{4.2}$$

Pour accélérer le calcul de la covariance, nous avons utilisé la librairie Spark "MLlib", qui grâce au calcul "In Memory" permet de charger les matrices en mémoire et donc réduire considérablement le temps de calcul.

4.5.2.3 Calcul du modèle ACP

Pour créer le modèle ACP, nous utilisons la fonction Singular Value Decomposition (SVD) "Décomposition en valeurs singulières" de Mahout pour décomposer la matrice covariance carrée calculée auparavant. Selon le rang de décomposition choisi, la fonction générera la liste des valeurs propres et des vecteurs propres. Plus le nombre de composantes principales k est important, la représentation de la variance des données est meilleure. Avec un k très grand, nous représentons 100% de la variance des données et pouvons donc détecter plus d'attaques, au risque d'augmenter le taux de faux positif. Au contraire, avec un k trop petit, nous détecterons beaucoup moins d'attaques. La matrice résultante, qu'on appellera dans la suite "modèle ACP" est trop petite et le nombre de lignes ne dépassera pas le nombre de clés, ce qui permet de l'utiliser pour le filtrage en temps réel du trafic réseau. Le temps de traitement pour le calcul des composantes principales dépend du nombre de composantes principales choisi.

Pour partager les modèles ACP calculés entre les différents noeuds du cluster Storm, nous stockons tous les modèles générés sous forme de fichiers texte dans un entrepôt de modèle NFS (Network File System) partagé qui offre un accès rapide et simultané aux petits fichiers matrices.

4.5.3 Détection en temps réel des attaques avec Storm

Pour détecter le trafic anormal en temps réel, nous avons implémenté une topologie Storm qui utilise le modèle créé dans la couche de traitement par lots et stocké dans l'entrepôt des modèles. Nous pouvons choisir le seuil de détection des alertes de l'algorithme au démarrage de l'application pour déterminer le niveau de sensibilité que nous voulons mettre en place. Le choix de la période de calcul d'agrégation doit être identique à la période de calcul du modèle choisi. La Figure4.5 représente notre topologie Storm qui récupère en temps réel les traces du trafic des pare-feux et fournit une liste d'attaques DDoS possibles.

Pour alimenter la topologie avec les journaux des paquets filtrés par les pare-

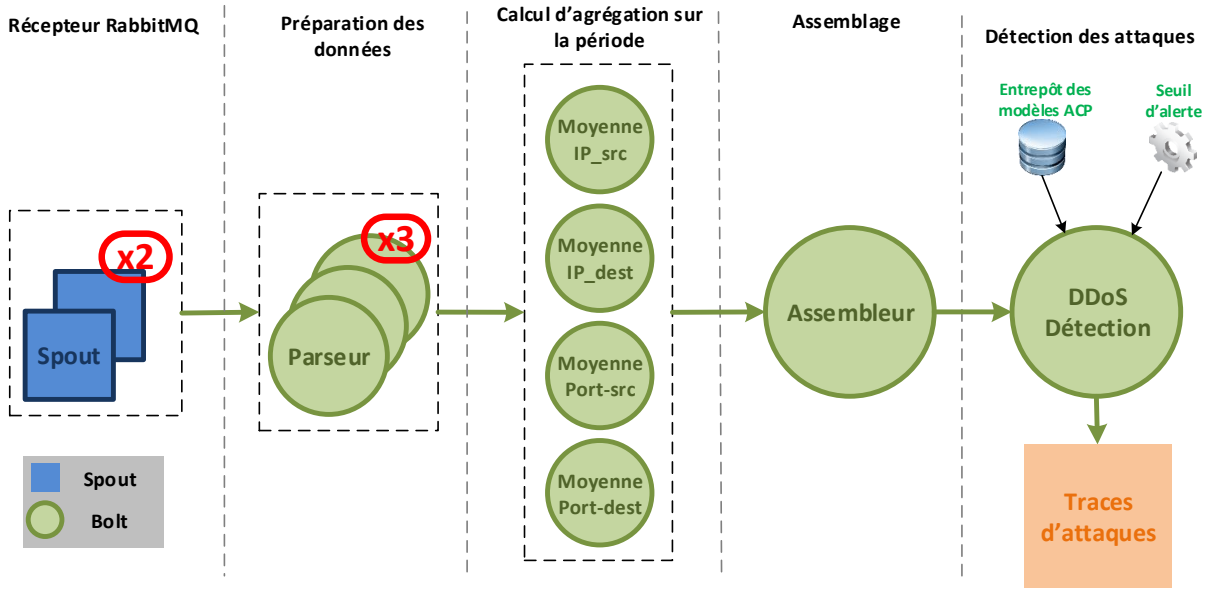


Figure 4.5 – La topologie Storm de détection des attaques en temps réel

feux, nous avons mis en place un serveur de messages MOM (Middle Oriented Middleware) qui permet l'échange de données entre les pare-feux et la topologie Storm. Notre choix s'est porté sur RabbitMQ [64], qui est un serveur de message basé sur le protocole MQTT très robuste (peut traiter plus d'un million de messages par seconde) et très facile à utiliser et configurer. Dans la topologie, le Spout d'entrée est abonné à la queue de messages provenant des pare-feux, et les messages reçus sont identiques aux flux décrit dans la section 4.6.2.1. Chaque message reçu correspond à un flux de données transitant sur le pare-feu. Ce Spout reçoit les données depuis le serveur RabbitMQ et les transmet au prochain Bolt sans apporter des modifications. Le but est de faire en sorte que le Spout constitue un point d'entrée des données sans le surcharger avec des tâches de calcul. En cas de gros flux de données, nous pouvons instancier plusieurs Spout qui alimentent la topologie pour éviter de saturer la file d'attente du serveur RabbitMQ. Dans la phase **Préparation des données**, le Bolt **Parseur** découpe chaque message correspondant à une trace d'un flux et envoi les tuples correspondants au Bolt suivant selon la clé (IP_SRC, IP_DEST, PORT_SRC, PORT_DEST...). Quand la clé est de type IP, nous prenons en compte uniquement le préfixe et donc les n premiers octets de l'IP. Nous pouvons définir des préfixes différents pour les IP privées et publiques. Chaque tuple en sortie de ce Bolt contient trois champs

<Clé,Timestamp,taille_paquet>. Ensuite, dans la phase **Calcul d'agrégation sur la période**, les Bolts **Moyenne_**CLE récupèrent les tuples et calculent le nombre et la taille totale des messages pour chaque clé. A partir de ces données nous pouvons calculer les moyennes dans la phase suivante, une fois toutes les traces de la période reçues. A la fin de chaque période, les Bolts envoient un tuple au prochain Bolt avec une liste de couples clés / données agrégées (<Clé,Timestamp,taille_totale,Nombre_paquets>). Les Bolt des phases un et deux peuvent être très sollicités avec l'augmentation du trafic, d'où l'importance de la scalabilité de Storm permettant l'utilisation de multiples instances de chaque Bolt. Dans la phase **Assemblage**, les calculs d'agrégations sur les différentes clés sont assemblés à la fin chaque période pour initier la phase de détection. Finalement, dans la phase **Détection d'attaques**, le Bolt **DDoS Détection** reçoit à la fin de chaque période un seul message concaténant les résultats des agrégations sur l'ensemble des clés. Ce Bolt utilise le modèle ACP stocké dans l'entrepôt des modèles pour détecter les attaques, en filtrant le trafic agrégé en temps réel à travers le modèle correspondant en fonction du seuil défini. Il est possible de définir des actions qui seront lancées en cas d'attaque pour les bloquer ou pour notifier l'administrateur du réseau d'un éventuel risque.

4.6 Expérimentation

L'objectif des expérimentations est de montrer l'efficacité de l'algorithme de détection des attaques par rapport aux solutions existantes, et la capacité de détecter les attaques en temps réel sans latence. Les expérimentations ont été réalisées sur des données provenant des datacenters d'Eolas. Dans cette section, nous présentons les sources de données utilisées pour l'expérimentation, les clusters déployés et dédiés aux tests et l'implémentation de notre architecture pour la détection des attaques réseau de type DDoS. Finalement, nous présenterons les résultats obtenus et nous les discuterons. Nous comparerons ensuite les résultats de notre système par rapport à ceux de la solution Wanguard [82].

4.6.1 Clusters d'évaluation

Nous avons utilisé deux clusters de traitement Big Data : Un cluster Hadoop pour les traitements par lots et le calcul du modèle, et un cluster Apache Storm

pour la détection en temps réel des attaques de type DDoS.

4.6.1.1 Cluster Hadoop

Pour calculer les modèles, nous utilisons un cluster Hadoop Cloudera de 6 noeuds sur une plateforme VMware ESXi virtualisée. La figure 4.6 présente l'architecture physique du cluster.

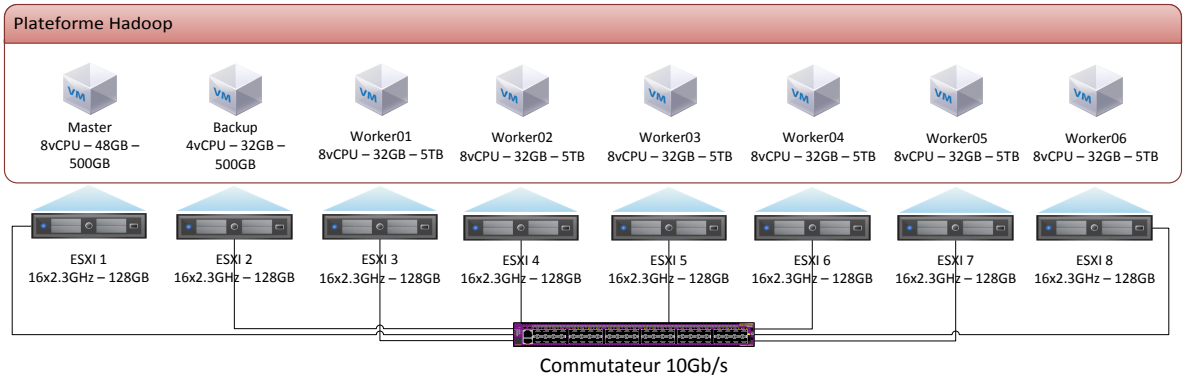


Figure 4.6 – La plateforme Hadoop d'évaluation

Le cluster est composé d'un ensemble de 6 noeuds de calcul (DataNode + TaskTracker) avec 8 coeurs à 2.3GHz, 32Go de RAM chacun et une capacité totale de stockage de 30To (5To/nœud). Tous les serveurs sont reliés à l'aide un commutateur 10Gb/s pour réduire les latences réseau. Le noeud master héberge le NameNode, le JobTracker et Spark Master, et le noeud Secondary héberge le SecondaryNameNode.

Nous utilisons la distribution Cloudera CDH 5.0.1 avec CentOS 6.5-64bits. Sur ce cluster, nous avons installé Yarn 2.0.3 et HDFS 2.3.0. Nous avons modifié la configuration du cluster Hadoop pour pouvoir optimiser les tâches de calcul et ainsi lancer jusqu'à 6 tâches Map et 5 reduce par noeud, ce qui revient à lancer 36 tâches Map en parallèle et 30 Reduce en parallèle sur l'ensemble des noeuds du cluster. Ce choix de configuration a été fait suite à de nombreux tests de performance avec des configurations distinctes. MapReduce comporte une centaine de paramètres de configurations qu'il faut adapter en fonction des données et du type de calcul. Le nombre de tâches Map et Reduce dans MapReduce 1.x peut être déterminé au lancement du traitement. Dans MapReduce 2.x, le nombre maximal de tâches Map et Reduce dépend de la taille mémoire des noeuds et de la mémoire

allouée aux conteneurs de calcul Map et Reduce. Le but de l'optimisation est d'utiliser la totalité des ressources mémoire et CPU de l'ensemble des machines du cluster pour accélérer le calcul, sachant que la plateforme est dédiée à 100% pour les calculs de modèle et qu'un seul modèle est calculé à la fois pour éviter les pertes de performance dus à l'ordonnancement des tâches. Dans Spark, les principaux paramètres d'optimisation des traitements sont le nombre d'exécuteurs qui peuvent être lancés sur les workers et le nombre de coeurs du CPU à attribuer à chaque exécuteur. Les configurations dépendent de la capacité CPU et mémoire des noeuds du cluster Spark et du type de traitement.

4.6.1.2 Cluster Apache Storm

Pour le filtrage du trafic en temps réel à travers le modèle ACP, le cluster Apache Storm que nous avons utilisé pour le traitement en temps réel est composé de 5 noeuds sur une plateforme VMware ESXi virtualisée. La figure 4.7 montre l'architecture physique du cluster Storm. Chaque noeud est une machine virtuelle avec 4 coeurs à 2.3 GHz et 16 Go de RAM. Les hyperviseurs sont connectés grâce à des commutateurs à 10Gb/s pour une meilleure bande passante. Nous utilisons la version 0.9.1-incubating d'Apache Storm avec Zookeeper-3.4.6. Les instances de Zookeeper et des Supervisors sont installés sur les 4 noeuds de calcul, et le serveur Nimbus est installé sur le noeud master.

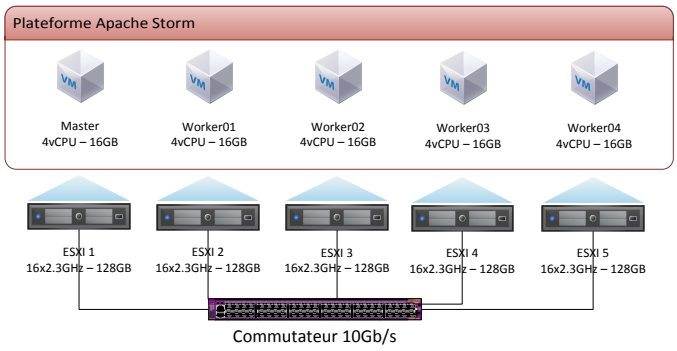


Figure 4.7 – La plateforme Apache Storm d'évaluation

Nous avons taillé le nombre d'instances des Spout et des Bolt de la topologie (Figure 4.5) pour pouvoir absorber les pics de trafic correspondant au profil de charge du datacenter d'Eolas, soit plus de 20000 flux par seconde.

4.6.2 Architecture réseau des datacenters Eolas

Eolas dispose de trois datacenters à Grenoble, interconnectés entre eux avec des liens 10Gb/s (Figure 4.8). Deux des trois datacenters disposent de liens directs avec des opérateurs de télécommunication, et le trafic internet des applications des datacenters peut transiter par n'importe quel lien pour une meilleure redondance. Les données de trafic réseau d'Eolas proviennent de plusieurs pare-feux répartis sur les trois datacenters.

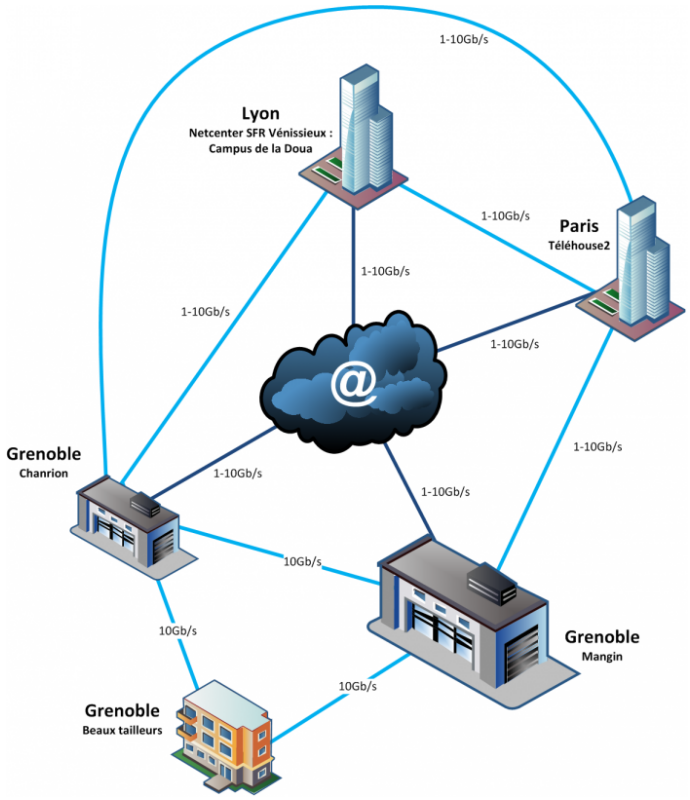


Figure 4.8 – Architecture réseau des datacenters Eolas

4.6.2.1 Journaux des pare-feux

Actuellement, Eolas compte plus de 2000 clients avec des offres d'hébergement variées : IaaS, PaaS, SaaS et FaaS. Le principal datacenter d'Eolas est le datacenter Green de Mangin qui peut héberger jusqu'à 13 cubes informatiques. Actuellement seuls deux cubes de ce datacenter sont utilisés, donc la volumétrie des données peut être multipliée par 6 d'ici quelques années avec la mise en production des cubes

restants. Nous avons utilisé deux ans d'historique de traces de trafic des pare-feux d'Eolas pour construire et évaluer notre modèle. Les données sont composées de traces de flux réseau pour plus de 2,000 applications hétérogènes (Web, Business Intelligence, Big Data, ...) hébergées dans les trois datacenters d'Eolas. Un total de 3To de données de traces ont été stockés en HDFS, avec 3 réplications à travers les DataNodes. Les journaux de logs sont fournis sous forme de fichiers texte (section), par l'outil "Syslog" installé sur tous les pare-feux. Tous les flux sont tracés dans Syslog sans échantillonnage. Chaque pare-feu produit plus de 5Go de journaux sur les flux filtrés par jour. Un journal peut contenir plus de 200000 adresses IPs distinctes internes et externes qui transitent sur les liens externes. Avec l'enrichissement des journaux avec des informations sur les protocoles, le nombre de paquets par flux... le journal d'un pare-feu dépasse les 8Go par jour. Dans le cas d'Eolas, les plateformes utilisent des adresses réseau en "/24", donc on avec les trois premiers octets d'une IP, nous pouvons identifier une plateforme cliente. Nous utilisons des préfixes IP communs pour créer nos modèles.

Eolas nous a fourni une liste d'attaques réelles observées et détectées manuellement dans les archives des traces qu'ils ont fournies. Cette liste nous aidera à valider notre implémentation et à faire un meilleur choix pour la valeur de seuil de l'algorithme.

4.6.3 Évaluation

L'objectif de cette section est d'évaluer les performances de notre architecture et des optimisations réalisées sur les deux couches : la couche de traitement par lots et la couche de traitement temps réel. Nous étudions ensuite l'efficacité de notre modèle pour détecter les attaques réseau en temps réel.

4.6.3.1 Mesures de performance du traitement par lots

Le calcul du modèle ACP dans la couche de traitement par lots est fait occasionnellement pour maintenir à jour le modèle (tous les jours, toutes les semaines ou même tous les mois). En fonction du volume de données et du nombre de variables, le temps de calcul peut être très long (plusieurs heures). Pour évaluer la scalabilité de la plateforme Hadoop, nous avons lancé le calcul du modèle ACP sur un fichier de journaux de pare-feux de 5Go avec 1440 périodes de 1 minute

et des clés IP en /16 pour limiter le nombre de clés distinctes. Le cluster est constitué de 3 workers identiques. La matrice de trafic calculée est de 1Go.

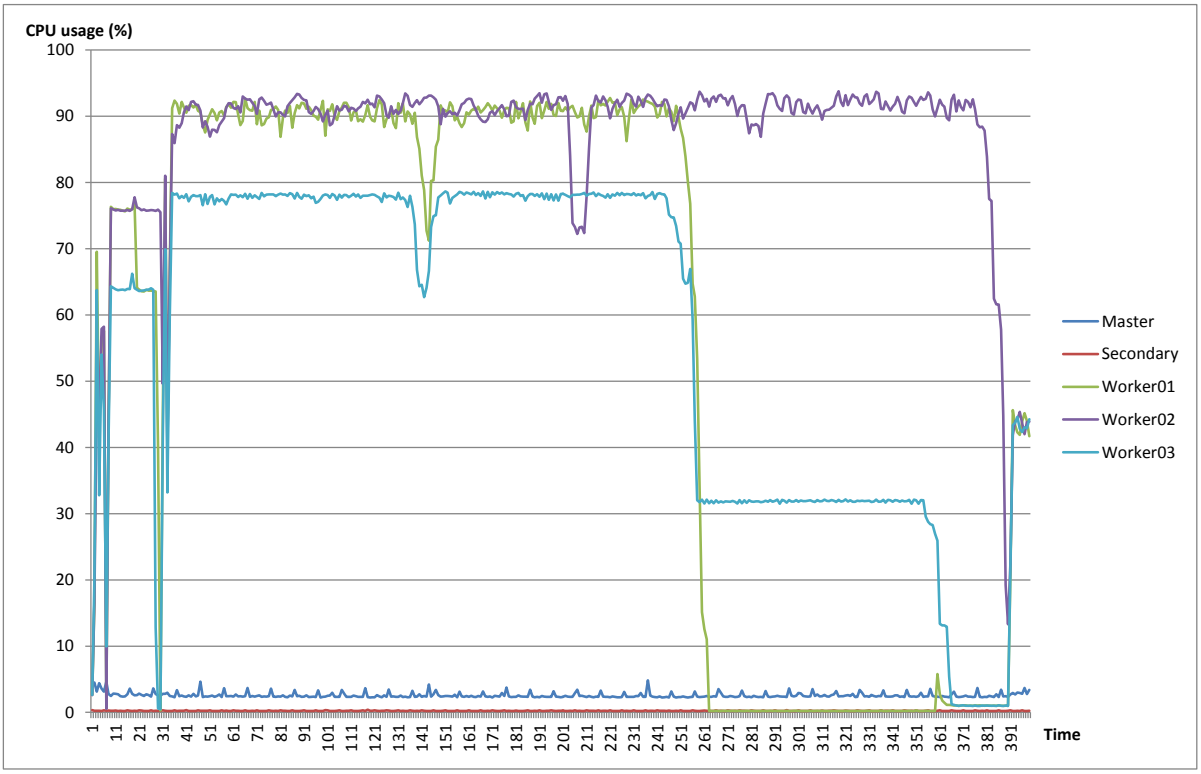


Figure 4.9 – Activité CPU des noeuds durant la création du modèle (3 workers)

La figure 4.9 montre les courbes de variation de l'activité CPU de l'ensemble des machines du cluster Hadoop, dont les 3 workers. Nous pouvons identifier les différentes phases de calcul du modèle. La première phase consistant à construire la matrice de données à partir des fichiers de journaux, qui dure 4 minutes. Ensuite le calcul de la matrice moyenne qui dure 22 minutes. Puis la phase de calcul de la matrice covariance qui dure plusieurs heures, et finalement le calcul des composantes principales dont la durée dépend du nombre de composantes principales à calculer. Au final, la création du modèle a pris plus de 4heures. Le temps de calcul peut être réduit en ajoutant encore plus de noeuds. Nous avons démontré qu'avec 5 composantes principales, nous pouvons représenter 99.96% de la variance des données Eolas, ce qui est très rassurant. Dans les évaluations, nous avons donc généré 5 composantes principales pour notre modèle.

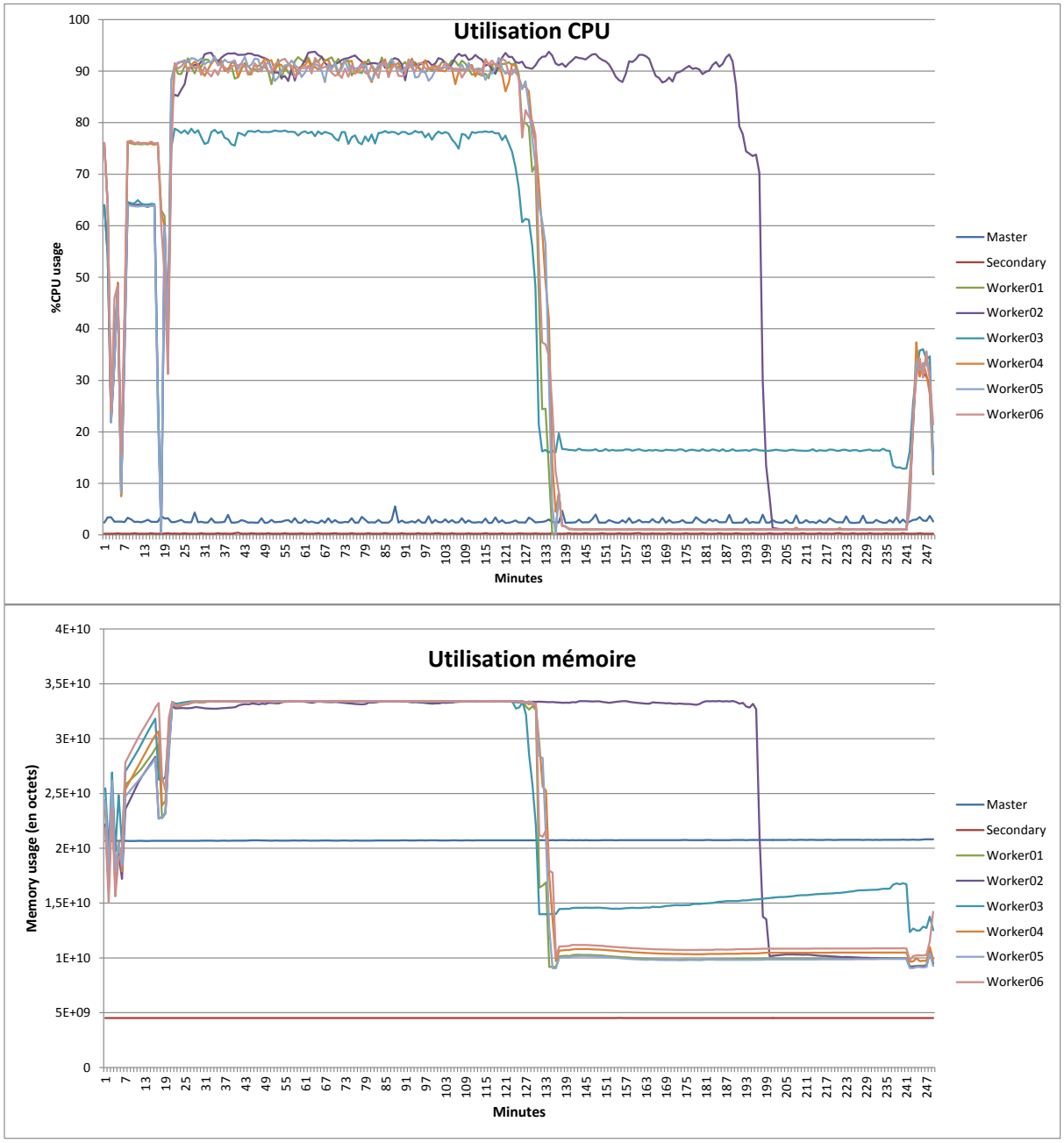


Figure 4.10 – Activité CPU et mémoire des noeuds durant la création du modèle (6 workers)

Grâce au système de fichier distribué HDFS et le Framework de calcul distribué MapReduce, nous pouvons réduire de moitié le temps de calcul en multipliant le nombre de workers par 2. Pour cela, nous avons expérimenté le même calcul

précédent avec 6 workers de capacité identique. La figure 4.10 montre la variation du CPU et de la mémoire utilisés pendant toute la période de calcul. Le gain lors de la première phase de calcul de la matrice est très limité, puisque c'est un calcul rapide et le temps de préparation d'un traitement MapReduce est relativement long quelque soit le nombre de workers. Par contre, sur la deuxième phase de calcul de la matrice moyenne, le temps de calcul a été quasiment divisé par deux, nous passons alors de 22 à 11 minutes.

Nous avons optimisé la configuration du cluster avec 6 workers pour accélérer les calculs MapReduce : Le nombre de tâches Map a été limité à 6, et le nombre de tâches Reduce à 5 par noeud.

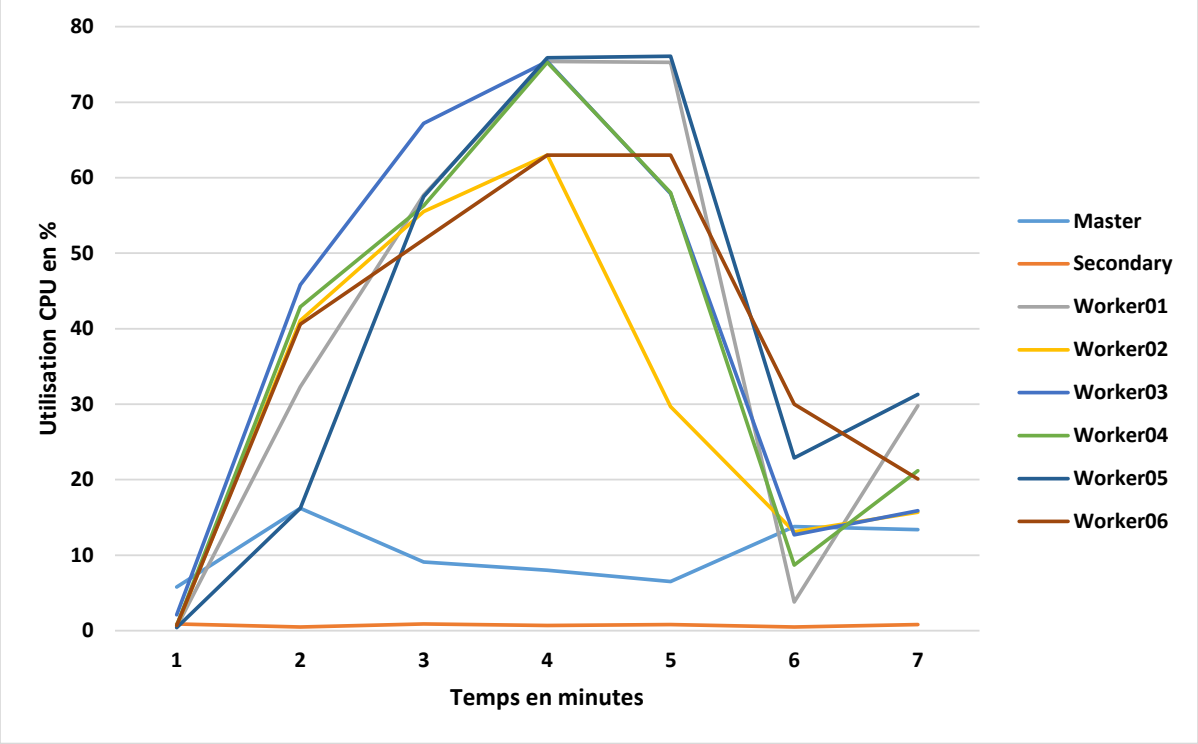


Figure 4.11 – Activité CPU lors du calcul de la covariance avec Spark

Le calcul de la covariance, qui consiste à multiplier deux énormes matrices, est le traitement qui consomme le plus de temps dans la chaîne de création du modèle. Le temps de calcul augmente exponentiellement avec l'augmentation du nombre de clés, et le calcul peut durer donc plusieurs jours sur notre plateforme. Nous avons travaillé sur une nouvelle implémentation avec Spark en utilisant la librairie MLlib. Le gain est énorme puisqu'on passe de plusieurs heures à quelques

minutes. Le calcul de covariance sur une matrice de données de 5Go dure moins de 6 minutes sachant qu'avec MapReduce ce calcul peut durer plusieurs jours. La figure 4.11 montre la variation de l'utilisation du CPU sur l'ensemble des machines du cluster Hadoop pendant le calcul du modèle.

A partir d'un mois d'historique de journaux de pare-feux, soit 150Go de données, nous obtenons un modèle ACP de moins de 10Mo que nous pouvons utiliser ensuite dans la topologie Storm pour filtrer le trafic en en temps réel. Nous pouvons calculer le modèle à partir d'un historique très long (plusieurs mois), ce qui permet de prendre en compte les variations sur le long terme. Notre modèle peut être généré à partir des données ayant une granularité ou une périodicité non standard.

4.6.3.2 Mesures de performance de la topologie Storm

Au contraire de la couche de traitement par lots, le filtrage des données de trafic qui arrivent au fil de l'eau doit être traité en temps réel.

Le tableau 4.1 présente les valeurs de latence d'exécution moyen mesurés par Storm, suite à plusieurs tests avec des débits de messages allant de 2000 à 20000 messages/s. Le Spout, qui représente le point d'entrée de la topologie, récupère les données sans réaliser de traitement, ce qui explique la latence nulle. Si le Spout est chargé, moins de messages sont alors consommés depuis la file d'attente RabbitMQ et les autres Bolts sont donc moins chargés. Pour cela, la file d'attente du serveur RabbitMQ doit rester vide, c'est à dire que les nouveaux flux sont consommés en temps réel sans latence.

Les Bolts "Parseur" et "Agrégation" traitent autant de messages que les flux reçus par le Spout, ils sont dons sollicités en permanence. Le Bot "Assembleur" reçoit un message par type clé à la fin de chaque période et il envoi ensuite un seul message qui concaténé l'ensemble des clés au Bolt "Détection". Ce dernier lance un seul traitement à la fin de chaque période, il est donc très peu sollicité, mais les calculs qu'il réalise sont très complexes d'où la latence élevée.

Table 4.1 – Mesure des latences d'exécution des noeuds de la topologie Storm

Spout	Parseur	Agrégation	Assembleur	Détection
0 ms	0.232 ms	0.044 ms	8.5 ms	785 ms

Nos critères de performance de la plateforme Storm sont l'utilisation mémoire et CPU de l'ensemble des machines. Pour éviter les latences, aucune surcharge ne doit être détectée, même en cas de forte d'affluence de messages lors d'une attaque. Suite à plusieurs tests, la topologie que nous avons mis en place peut traiter jusqu'à 20000 messages par seconde sans latences. Le trafic moyen sur une journée étant de l'ordre de 1500 flux par seconde,

La période d'échantillonnage peut être paramétrée lors de la création de la topologie. On peut intégrer un mode très haute granularité des périodes avec une granularité fine allant jusqu'à 5 secondes, selon la granularité de l'échantillonnage des logs et le modèle choisis. Ce mode peut être activé en cas de forts risques d'attaques.

4.6.3.3 Détection des attaques

Eolas ne disposant pas de solution de mitigation DDoS auparavant, la liste des historiques d'attaques fournis par Eolas représente des attaques DDoS de grande ampleur, faciles à détecter et visibles facilement même sur une courbe de trafic non décomposé en trafic normal et anormal. Parmi la liste des grandes attaques fournis par Eolas (8 attaques), nous avons identifié 100% des attaques avec notre modèle.

La Figure 4.12 montre l'amplitude d'une attaque DDoS, qui peut saturer les liens réseaux du datacenter d'Eolas . Sans solution de détection d'attaque temps réel, l'attaque peut avoir un impact sur la disponibilité de l'ensemble des applications. Notre modèle a été calculé avec une période d'échantillonnage de 60 secondes et nous détecté l'attaque 5 minutes plus tôt que l'outil d'alerte de saturation du lien, ce qui permet de mettre en place des solutions de contournement avant qu'il ne soit trop tard.

La solution Wanguard [82] a été mise en place récemment pour la protection du datacenter d'Eolas cotre les attaques de type DDoS en temps réel. Nous avons utilisé les résultats de cette solution pour étudier notre modèle et vérifier le taux de détection et de faux positifs générés par la méthode ACP. Nous avons récupéré la liste des anomalies détectées par la solution Wanguard sur une période de 4 jours et comparons ensuite les résultats par rapport au modèle ACP. Le modèle que nous utilisons a été généré sur une période de 1minute avec 5 composantes principales. La figure 4.13 montre la courbe de trafic anormal calculé à partir

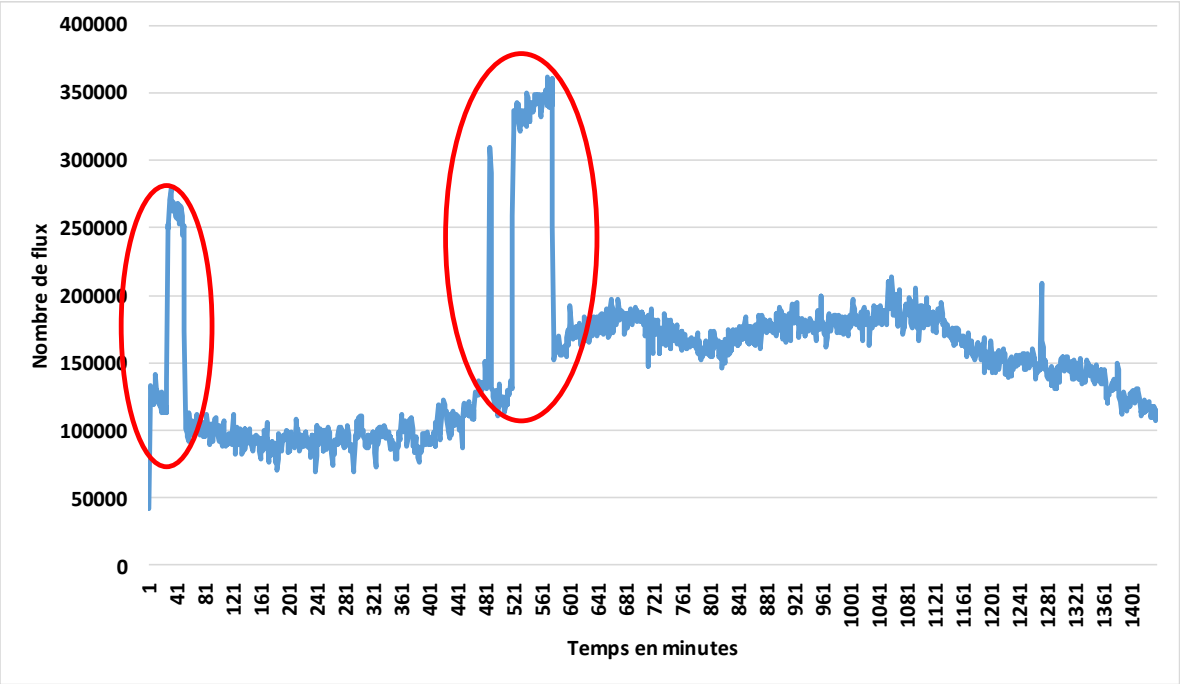


Figure 4.12 – Évolution du trafic lors des grosses attaques DDoS

du modèle. Nous avons définis plusieurs seuils de détection des anomalies pour comprendre l'impact du seuil sur le taux de détection et le taux de fausses alertes.

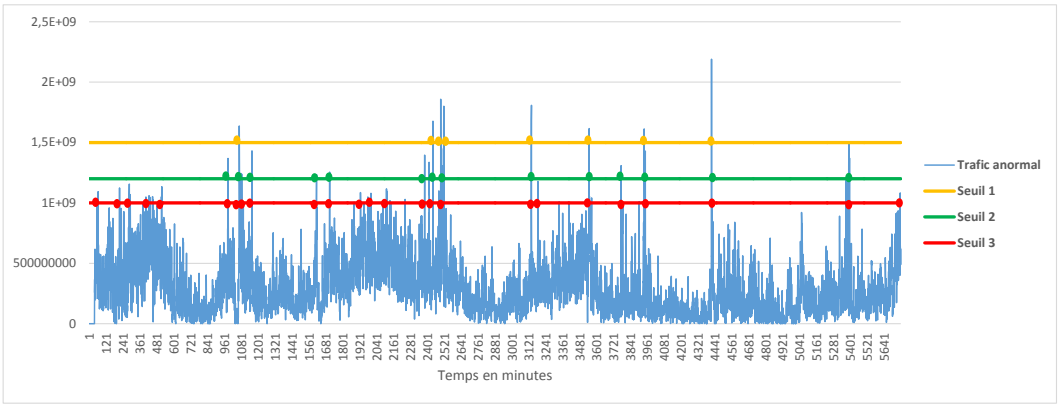


Figure 4.13 – Détection du dépassement de seuil du trafic anormal

Dans la table 4.2, nous pouvons remarquer que le taux de détection augmente en fonction de la choix du seuil. Ce seuil peut évoluer avec l'évolution du datacenter et l'hébergement de nouveaux clients. Plus le nombre de paquets d'une attaque

est important, plus l'amplitude du trafic anormal est importante. Pour détecter les petites attaques, nous devrons définir un seuil plus bas, au risque d'augmenter le taux de faux positifs.

Table 4.2 – Taux de détection du modèle ACP

Seuil	Wanguard	ACP	Taux de détection	Taux faux positif
Seuil 1	28	8	28.57%	0%
Seuil 2	28	15	53.57%	$< 13.3\%$
Seuil 3	28	26	92.86%	$< 23.07\%$

La solution Wanguard a détecté 28 attaques sur la période des 4 jours. Avec le seuil "Seuil 3", nous avons détecté 20 attaques communes avec celles détectés par Wanguard. Par contre, Wanguard a détecté 8 alertes ayant une durée inférieure à 5s que notre modèle n'a pas pu détecter. Notre modèle est calculé sur une période de 1 minute, l'anomalie est donc indétectable. En effet, la solution Wanguard est alimentée par une sonde dédiée en temps réel, tandis que le système de journaux "Syslog" installé sur les pare-feux trace les flux transitant sur le réseau toutes les 30 secondes, et donc notre modèle ne peut pas être calculé sur des périodes inférieures à 30 secondes. Avec le modèle ACP, nous avons détectés 6 attaques qui ne sont pas détectés par la solution Wanguard : nous n'avons pas pu déterminer si ces attaques correspondent à des faux positifs. La vérification des attaques complexes à la main est impossible dans l'environnement très hétérogène Eolas.

4.7 Conclusion

Dans ce chapitre nous avons présenté l'architecture distribuée et scalable pour la détection des anomalies réseaux et notamment les attaques de type DDoS. Le but du DDoS est de surcharger la victime pour bloquer ses communications et ses transactions. L'attaque peut être sophistiquée en utilisant des hôtes zombie innocents contrôlés par les hackers et des IP sources modifiés. Les filtres basés sur les règles ne fonctionnent pas correctement quand le hacker envoie des paquets, analyse les réponses et la dégradation du réseau de la victime, pour détecter le profil du trafic nominal et faire des attaques ayant le même profil que le trafic normal. Nous avons utilisé un modèle statistique difficile à contourner par des attaquants, pour la détection des attaques. Le modèle ACP permet de différencier

un trafic légitime d'une attaque en comparant le volume de trafic actuel d'un attribut par rapport au trafic nominal.

Nous avons présenté notre architecture basée sur des solutions Big Data pour le traitement par lots et le traitement en temps réel des flux de données. Nous avons démontré aussi l'efficacité de l'algorithme pour la détection des attaques avec un faible taux de faux positifs, sur des données réelles provenant d'un datacenter en exploitation. Le temps de calcul du modèle dépend du nombre de variables et très peu du nombre de périodes. On peut donc inclure un très grand historique de données.

L'architecture générique que nous proposons peut être utilisée pour d'autres cas d'usage de détection d'anomalies en modifiant le modèle et les sources de données. La scalabilité des solutions utilisées permet d'adapter l'architecture pour des infrastructures de taille plus grande.

5

Conclusion

Contents

5.1 Rappel du contexte 107

5.2 Conclusion 108

5.3 Perspectives 110

5.3.1 Perspectives court terme 110

5.3.2 Perspectives long terme 111

5.1 Rappel du contexte

Avec le développement du Cloud Computing et la virtualisation des ressources informatiques, les gestionnaires des datacenters font face à de nouveaux défis de taille : fournir un plus haut niveau de service SLA (Disponibilité, sécurité, performance...). Des systèmes redondés de refroidissement et de distribution électrique sont mis en place pour garantir un meilleur SLA, ce qui fait grimper la consommation électrique. Avec l'augmentation de la concentration des équipements informatiques dans les datacenters et l'augmentation du prix de l'électricité, de plus en plus d'hébergeurs s'intéressent au "Green Computing" pour réduire leur consommation et améliorer le rendement énergétique de leur datacenter. L'indicateur PUE d'efficience énergétique ainsi que d'autres indicateurs tel quel le CUE ou WUE servent à mesurer la performance énergétique et écologique des

datacenters, présentent un argument de différenciation et sont des outils permettant de mesurer le progrès de chaque datacenter. L'optimisation de l'efficience énergétique d'un datacenter peut être faite au niveau logiciel avec des systèmes de consolidations ou au niveau matériel avec des outils d'optimisation des "Facilités" (distribution électrique et refroidissement). Les automates industriels implémentés dans la majorité des datacenter sont très fiables et peuvent offrir un niveau de disponibilité élevé en cas de panne de l'un des systèmes de refroidissement. Cependant, chaque automate contrôle un ou plusieurs systèmes de refroidissement sans avoir une vue globale sur les autres équipements ou une connaissance des données provenant de sources externes, ce qui implique des décisions non optimisées et parfois contradictoires.

D'autre part, la diversité des applications, la complexité d'administration et la sensibilité des données hébergées, posent de nouveaux problèmes de sécurité et de performance. Il est primordial de détecter les dysfonctionnements le plus rapidement possible pour agir vite. Les outils de monitoring standards permettent de détecter les anomalies simples sur les différentes couches du datacenter (FaaS, IaaS, PaaS et SaaS), mais certains comportements plus complexes sont impossibles à identifier. La collecte des métriques depuis les milliers de capteurs déployés dans le datacenter produit de très gros volumes de données à analyser en temps réel, on parle alors de données Big Data. Le traitement Big Data permet d'analyser de très gros volumes de données (plusieurs dizaines de To) afin d'en sortir des indicateurs utiles et détecter des comportements anormaux difficiles à identifier avec les outils de supervision classiques.

5.2 Conclusion

Dans cette thèse, nous nous sommes intéressés aux problématiques majeures de gestion des datacenters Green dans un contexte de Cloud Computing : offrir une meilleure qualité de service au client tout en réduisant les coûts de l'énergie et l'empreinte écologique du datacenter. Dans les travaux réalisés nous proposons deux contributions visant à réduire la consommation d'un datacenter Green et détecter en temps réel des comportements anormaux de l'exploitation du datacenter.

Dans la première contribution, nous avons proposé un algorithme pour l'op-

timisation du refroidissement dans un datacenter Green, basé sur la prédiction de la charge et la corrélation avec des sources de données externes. Nous avons présenté l'architecture de refroidissement du datacenter Eolas sur lequel nous avons réalisés nos expérimentations. Ce datacenter en Tier III+ a été conçu avec des systèmes de refroidissement et de distribution électrique redondés pour une meilleure disponibilité tout en conservant un PUE de moins de 1,35. Dans notre implémentation, nous utilisons des sources de données externes pour enrichir les bases de données du datacenter, et pouvoir améliorer le rendement du système de refroidissement. Les prévisions météorologiques permettent de réduire le nombre de transitions entre les différents systèmes de refroidissement quand la température extérieure est très variable. De plus, en se basant sur l'historique de l'activité des équipements informatiques du datacenter, nous utilisons le modèle SARIMA pour la prédiction de la charge sur des périodes futures et ainsi prévoir les pics de charge conséquentes et de les gérer au mieux. Ces nouvelles sources de données permettent de proposer un plan d'action pour planifier l'utilisation des différents systèmes de refroidissement sur une période future, tout en réduisant le cycles de démarrage/arrêt et en réduisant le coût du refroidissement. Le but final est d'optimiser le PUE du datacenter tout en garantissant une meilleure qualité de service SLA. Nous avons réalisé des expérimentations sur un vrai datacenter en exploitation pour évaluer notre algorithme et mesurer les gains.

Dans la deuxième contribution nous proposons une architecture distribuée et scalable, pour la détection des anomalies dans un datacenter en temps réel, en utilisant des données Big Data et des flux de messages en temps réel. Les technologies Big Data sont devenues les outils incontournables pour l'apprentissage (Machine Learning), surtout pour les volumes de données conséquents provenant de l'exploitation d'un datacenter. Dans notre architecture nous disposons d'un module de calcul Big Data utilisant le Framework Hadoop pour la création d'un modèle statistique réduit. Ce modèle issu d'une analyse complexe des données, permet d'identifier une empreinte des données pour détecter facilement les comportements normaux et anormaux qu'il est difficile de détecter avec des outils de monitoring classiques. Ce modèle sera utilisé ensuite par une topologie Storm alimentée en permanence avec les données du datacenter, pour détecter les anomalies de fonctionnement en temps réel. L'architecture que nous proposons peut être adaptée pour la détection de tout type d'anomalie de fonctionnement dans un datacenter,

en temps réel et dans cette thèse nous avons démontré son efficacité pour la détection en temps réel des attaques réseau de type Déni de Service Distribué. La détection rapide des attaques permet aux administrateurs d'agir au plus tôt afin d'éviter tout endommagement de l'infrastructure du datacenter ou une dégradation du SLA. Notre architecture a été évaluée avec des données de pare-feu provenant d'un datacenter en exploitation et hébergeant des applications réelles.

5.3 Perspectives

Avec l'avancement des travaux de cette thèse, nous avons identifié des axes de prolongement de nos travaux, notamment avec les nouveautés dans le domaine du Cloud Computing et du Big Data. Ces axes concernent à la fois l'optimisation de la consommation des datacenters et la détection des anomalies de fonctionnement d'un datacenter, dans le but d'améliorer encore les SLA. Nous pouvons diviser les travaux futurs dans deux catégories : les travaux réalisables sur le court terme, et les travaux plus conséquents sur le long terme.

5.3.1 Perspectives court terme

Actuellement, le temps de calcul de la prédiction avec SARIMA est dépendant de la taille de l'historique et peut être parfois très long. Afin d'améliorer la précision de l'algorithme de prédiction de charge et prévoir les événements occasionnels, nous prévoyons de mettre en place une implémentation Big Data de l'algorithme de prédiction SARIMA, afin d'inclure un historique plus long avec une granularité de mesures plus fine. Cela permettrait de réaliser encore des économies en estimant la consommation des systèmes de refroidissement sur des périodes plus petites.

D'autres améliorations sont envisagées pour l'algorithme d'optimisation du refroidissement du datacenter. A l'heure où les pics de consommation électriques battent des records et surtout en hiver, les fournisseurs électricité prévoient de faire payer aux entreprises les pics de consommation qu'ils génèrent. Une évolution de notre algorithme d'optimisation permettrait de prévoir ces pics de consommations électriques dues à une augmentation de l'activité des serveurs ou à l'augmentation de la consommation du refroidissement, pour lisser la consommation électrique et éviter donc les pénalités.

Une autre validation de notre architecture de détection des anomalies en temps réel peut consister à adapter notre modèle statistique pour détecter les comportements anormaux du système de refroidissement ou de la distribution électrique. Ce travail non étudié dans cette thèse consiste à détecter des comportements à risque dans le fonctionnement des "Facilités" d'un datacenter pour agir proactivement. A partir de l'historique de fonctionnement des équipements et des pannes détectées, nous serons capables de prédire les pannes et les dysfonctionnements à l'avance et éviter une dégradation, même limitée, du niveau de service.

5.3.2 Perspectives long terme

Dans notre architecture de détection des anomalies, la plateforme Storm de traitement des données en temps réel a été optimisée pour absorber les pics de flux de données qui peuvent être dus, dans le cas du démonstrateur de détection des attaques, à une attaque de type déni de service distribué qui génère d'énormes trafics. Ce dimensionnement statique des ressources est inutile en cas de périodes de trafic normal, ce qui entraîne un gaspillage des ressources informatiques, ou au contraire un sous-dimensionnement qui peut impacter la performance de la plateforme en période de fortes charges. Nous envisageons d'étudier la mise en place d'une plateforme Storm élastique qui s'adapte à la variation de la charge pour assurer des économies de ressources tout en assurant une meilleure qualité de service. Ce travail est très prometteur surtout avec l'utilisation de conteneurs "Docker", très rapides à déployer pour une meilleure réactivité.

Glossaire

CAPEX Dépenses d'investissement de Capital. 7, 20, 21

CMDB Configuration Management Database. 45, 87

CUE Carbon Usage Effectiveness. 2, 12, 17, 107

DCaaS DataCenter-as-a-Service. 3

DDoS Distributed Denial of Service. 7, 8, 35–38, 40, 78, 79, 81–84, 86, 87, 92, 94, 95, 103, 105

EER Energy Efficiency Ratio. 62, 64

FaaS Facilities-as-a-Service. 3, 8, 13, 46, 75, 97, 108

GTI Garantie de Temps d'Intervention. 11

GTR Garantie de Temps de Réparation. 11

HDFS Hadoop Distributed File System. 24, 25, 29, 84, 86, 90, 98, 100

HTA Haute Tension A. 17, 49

HTB Haute Tension B. 49

IaaS Infrastructure-as-a-Service. 2–4, 10, 13, 46, 75, 88, 97, 108

IDS Intrusion Detection System. 35

OPEX Dépenses d'exploitation. 7

PaaS Platform-as-a-Service. 2, 3, 10, 13, 46, 75, 88, 97, 108

PCA Principal Component Analysis. 38, 81

PUE Power Usage Effectiveness. 1, 2, 4, 5, 7, 12, 17, 18, 20, 39, 43, 63, 65, 72, 75, 107, 109

SaaS Software-as-a-Service. 2, 3, 10, 88, 97, 108

SARIMA Seasonal Autoregressive Integrated Moving Average. 22, 37, 60, 109, 110

SLA Service Level Agreement. iii, 2–8, 10–15, 19, 21, 27, 35, 43, 44, 46, 47, 65, 75, 80, 82, 107, 109, 110

SLO Service Level Objectives. 12–14

SVD Singular Value Decomposition. 92

TGBT Tableau Electrique Basse Tension. 50

VM Virtual Machine. 13, 15, 18, 19, 46

WUE Water Usage Effectiveness. 2, 12, 17, 51, 55, 107

YARN Yet Another Resource Negotiator. 27–29

Bibliographie

[1] Don Atwood and John Miner, *Reducing data center cost with an air economizer*, Intel Information Technology, 2008.

[2] Dan Azevedo, Christian Belady, and Jack Pouchet, *Water usage effectiveness (wue) : a green grid data center sustainability metric*, 2011.

[3] Dan Azevedo, Michael Patterson, Jack Pouchet, and Roger Tipley, *Carbon usage effectiveness (cue) : a green grid data center sustainability metric*, 2010.

[4] Christian Belady, Andy Rawson, John Pfleuger, and Tahir Cader, *Green grid data center power efficiency metrics : Pue and dcie*, Tech. report, Green Grid, 2008.

[5] Anton Beloglazov and Rajkumar Buyya, *Energy efficient allocation of virtual machines in cloud data centers*, Cluster, Cloud and Grid Computing (CCGrid), IEEE, 2010, pp. 577–578.

[6] Siddharth Bhopte, Dereje Agonafer, Roger Schmidt, and Bahgat Sammakia, *Optimization of data center room layout to minimize rack inlet air temperature*, Journal of electronic packaging (2006), 380–387.

[7] Norman Bobroff, Andrzej Kochut, and Kirk Beaty, *Dynamic placement of virtual machines for managing sla violations*, Integrated Network Management. 10th IFIP/IEEE International Symposium, May 2007, pp. 119–128.

[8] Frédéric Bordage, *Data center : comment allier efficience énergétique et environnement ?*, `http://www.greenit.fr/article/acteurs/hebergeur/data-center-comment-allier-efficience-energetique-et-enviro_nnement-4771`.

[9] Frederic Bordage, *Le water cooling à nouveau à la mode*, http://www.greenit.fr/article/bonnes-pratiques/le-water-cooling-a-nouveau-a-la-mode-4491.

[10] Rory Bray, Daniel Cid, and Andrew Hay, *Ossec host-based intrusion detection guide*, Syngress, 2008.

[11] David Breitgand and Amir Epstein, *Sla-aware placement of multi-virtual machine elastic services in compute clouds*, Integrated Network Management (IM), IEEE, 2011, pp. 161–168.

[12] Emmanuel Cecchet, Rahul Singh, Upendra Sharma, and Prashant Shenoy, *Dolly : virtualization-driven database provisioning for the cloud*, ACM SIGPLAN Notices, vol. 46, ACM, 2011, pp. 51–62.

[13] Shaoming Chen, Yue Hu, and Lu Peng, *Optimization of electricity and server maintenance costs in hybrid cooling data centers*, IEEE Sixth International Conference on Cloud Computing (2013), 526–533.

[14] *Cloudera : The platform for big data and the leading solution for apache hadoop in the entreprise*, http://www.cloudera.com/.

[15] European Commission, *Cloud service level agreement standardisation guidelines*, June 2014.

[16] *Ctrlgreen project*, http://www.ctrlgreen.org/.

[17] Par Dalin, *Free cooling/natural cooling is crucial for a successful district cooling development*, April 2012.

[18] *Datalyse project*, http://www.datalyse.fr/.

[19] Groupement Romand de l'Informatique (GRI), *Sla – service level agreement : Marche à suivre*, Tech. report, GRI, Avril 2012.

[20] Jeffrey Dean and Sanjay Ghemawat, *Mapreduce : Simplified data processing on large clusters*, OSDI (2004), 107–113.

[21] Vincent Debusschere and Seddik Bacha, *Hourly server workload forecasting up to 168 hours ahead using seasonal arima model*, IEEE International Conference on Industrial Technology, 2012.

[22] Business Decision, *La greenethiquette*, http://www.greenethiquette.fr/.

[23] *Docker*, https://www.docker.com/.

[24] Consortium EnergeTIC, *Efficience des data centers, les retombées du projet energetic*, Tech. report, Grenoble, 2013.

[25] Prolexic Security Engineering and Research Team, *Q4 2014 state of the internet security report*, http://www.stateoftheinternet.com/resources-web-security-2014-q4-internet-security-report.html, Jan. 2015.

[26] *Eolas*, http://www.businessdecision-eolas.com/544-green-it.htm.

[27] Jeremy Geelan et al., *Twenty-one experts define cloud computing*, Cloud Computing Journal, vol. 4, 2009, pp. 1–5.

[28] Lars George, *Hbase : The definitive guide*, O'REILLY Media, United States of America, September 2011.

[29] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, *The google file system*, SIGOPS Oper. Syst. Rev. **37** (2003), no. 5, 29–43.

[30] *Google : Raise your data center temperature*, http://www.datacenterknowledge.com/archives/2008/10/14/google-raise-your-data-center-temperature/.

[31] The Green Grid, *Recommendations for measuring and reporting overall data center efficiency*, Tech. report, The Green Grid, July 2010.

[32] Ajay Gulati, Anne Holler, Minwen Ji, Ganesha Shanmuganathan, Carl Waldspurger, and Xiaoyun Zhu, *Vmware distributed resource management : Design, implementation, and lessons learned*, http://waldspurger.org/carl/papers/drs-vmtj-mar12.pdf.

[33] *Hadoop : Open-source software for reliable, scalable, distributed computing*, http://hadoop.apache.org.

[34] Herodotos Herodotou and Shivnath Babu, *Profiling, what-if analysis, and cost-based optimization of mapreduce programs*, Proc. of the VLDB Endowment **4** (2011), no. 11, 1111–1122.

[35] *Horthonworks*, http://hortonworks.com/.

[36] Peter Huber, *Robust statistics.*, John Wiley and Sons, 2004.

[37] *Hadoop : un marche de 813 m en 2016*, http://www.lemagit.fr/technologie/gestion-des-donnees/2012/05/09/hadoop-march-eacute-813-2016/.

[38] Waheed Iqbal, Matthew N Dailey, and David Carrera, *Sla-driven dynamic resource management for multi-tier web applications in a cloud*, Cluster, Cloud and Grid Computing (CCGrid), IEEE, 2010, pp. 832–837.

[39] *Apache jmeter*, http://jmeter.apache.org/.

[40] M Tim Jones, *Process real-time big data with twitter storm*, IBM Technical Library (2013), 1–10.

[41] John Judge, Jack Pouchet, Anand Ekbote, and Sachin Dixit, *Reducing data center energy consumption*, ASHRAE J., Nov (2008), 14–26.

[42] Yoshiki Kanda, Kensuke Fukuda, and Toshiharu Sugawara, *Evaluation of anomaly detection based on sketch and pca*, Proceedings of the IEEE Global Telecommunications Conference (2010), 1–5.

[43] Attila Kertesz, Gabor Kecskemeti, and Ivona Brandic, *Autonomic sla-aware service virtualization for distributed systems*, Parallel, Distributed and Network-Based Processing (PDP), IEEE, 2011, pp. 503–510.

[44] Jungsoo Kim, Martino Ruggiero, and David Atienza, *Free cooling-aware dynamic power management for green datacenters*, International Conference on High Performance Computing and Simulation (HPCS) (2012), 140–146.

[45] Jonathan Koomey, *Growth in data center electricity use 2005 to 2010*, Analytics Press, october, 2011.

[46] Jay Kyathsandra and Chuck Rego, *The efficient datacenter*, Intel Information Technology, 2014.

[47] Anukool Lakhina, Mark Crovella, and Christophe Diot, *Diagnosing network-wide traffic anomalies*, In Proceedings of SIGCOMM (2004), 219–230.

[48] *Lambda architecture*, http://lambda-architecture.net/.

[49] Byung-Yun Lee and Gil-Haeng Lee, *Service oriented architecture for sla management system*, The 9th International Conference on Advanced Communication Technology, vol. 2, IEEE, 2007, pp. 1415–1418.

[50] Yeonhee Lee, Wonchul Kang, and Youngseok Lee, *A hadoop-based packet trace processing tool*, TMA (2011), 51–63.

[51] Yeonhee Lee and Youngseok Lee, *Detecting ddos attacks with hadoop*, ACM CoNEXT Student Workshop (2011), 1–7.

[52] Guangdeng Liao, Kushal Datta, and Theodore L Willke, *Gunther : search-based auto-tuning of mapreduce*, Euro-Par Parallel Processing, Springer, 2013, pp. 406–419.

[53] Paul Lin, Victor Avelar, and John Niemann, *Implementing hot and cold air containment in existing data centers*, Tech. report, APC, 2013.

[54] Scott Lowe, *Mastering vmware vsphere 5*, John Wiley & Sons, 2011.

[55] *Apache mahout*, `http://mahout.apache.org/`.

[56] *Mapr : Apache hadoop distribution*, `https://www.mapr.com/`.

[57] Le monde, *La facture d'électricité des français augmenterait de 50% d'ici à 2020 - 19.07.2012*, `http://www.lemonde.fr/economie/article/2012/07/19/la-facture-d-electricite-des-francais-doublera-d-ici-a-2020_1735483_3234.html`.

[58] *Neoload : Logiciel de test en charge*, `http://www.neotys.fr/product/overview-neoload.html`.

[59] *Netflow analyser*, `https://www.manageengine.com/products/netflow/`.

[60] Chandrakant D Patel, Ratnesh Sharma, Cullen E Bash, and Abdlmonem Beitelmal, *Thermal considerations in cooling large scale high compute density data centers*, Inter Society Conference on Thermal Phenomena (2002), 767–776.

[61] Michael K Patterson, Don Atwood, and John G Miner, *Evaluation of air-side economizer use in a compute-intensive data center*, ASME InterPACK Conference, Vol. 2 (July 2009), 1009–1014.

[62] Steven Pelley, David Meisner, Thomas F Wenisch, and James W VanGilder, *Understanding and abstracting total data center power*, Workshop on Energy-Efficient Design, 2009.

[63] Martin Preene, *Sustainable groundwater-source cooling systems for buildings*, Proceedings of the ICE-Engineering Sustainability **161** (2008), no. 2, 123–133.

[64] *Rabbitmq - messaging that just works*, `https://www.rabbitmq.com/`.

[65] OB Remi-Omosowon, *Statistical analysis of snort alerts*, Advances in Communications, Computing, Networks and Security Volume 8 (2011), 207–215.

[66] Ahmed El Rheddane, Noël De Palma, Fabienne Boyer, Frédéric Dumont, Jean-Marc Menaud, and Alain Tchana, *Dynamic scalability of a consolidation service*, IEEE Sixth International Conference on Cloud Computing (CLOUD), IEEE, 2013, pp. 748–754.

[67] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot, *Sensitivity of pca for traffic anomaly detection*, In Proceedings of SIGMETRICS - San Diego, California, USA (2007), 109–120.

[68] Benjamin Rubinstein, Blaine Nelson, Ling Huang, Anthony Joseph, and Shinghon Lau et al, *Understanding and defending against the poisoning of anomaly detectors*, In Proceedings of IMC (2009), 1–14.

[69] Ibrahim Safieddine and Noël De Palma, *Efficient management of cooling systems in green datacenters*, ICAS (2015), 73–76.

[70] Ibrahim Safieddine, Noël De Palma, and Gérald Dulac, *Adviz.eolas : Restitution et pilotage de process temps réel data-driven*, BDA (2015), 1–6.

[71] SENAT - France, *La commission d'enquête sur le coût réel de l'eĺectricité*, July 2012.

[72] Ratnesh K Sharma, Cullen E Bash, Chandrakant D Patel, Richard J Friedrich, and Jeffrey S Chase, *Balance of power : Dynamic thermal management for internet data centers*, Internet Computing (Vol.9 , Iss. 1) (2005), 42–49.

[73] *Snort intrusion prevention system*, `https://snort.org/`.

[74] Barrie Sosinsky, *Cloud computing bible*, vol. 762, John Wiley & Sons, 2010.

[75] *Spark : Lightning-fast cluster computing*, `http://spark.apache.org/`.

[76] *Storm : Distributed and fault-tolerant realtime computation*, `https://storm.apache.org/`.

[77] Huizhong Sun, Yan Zhaung, and H. Jonathan Chao, *A principal components analysis-based robust ddos defense system*, IEEE Communications Society, ICC proceedings (2008), 1663–1669.

[78] Alain Tchana, Noel De Palma, Ahmed El-Rheddane, Bruno Dillenseger, Xavier Etchevers, and Ibrahim Safieddine, *A Scalable Benchmark as a Service Platform*, DAIS - International Conference on Distributed Applications and Interoperable Systems (Florence, Italy), vol. 7891, Springer, June 2013, pp. 113–126.

[79] Alain Tchana, Noel De Palma, Ibrahim Safieddine, Daniel Hagimont, Bruno Diot, and Nicolas Vuillerme, *Software consolidation as an efficient energy and cost saving solution for a saas/paas cloud model*, Euro-Par : Parallel Processing, Springer Berlin Heidelberg, 2015, pp. 305–316.

[80] Uniflair, *New design strategies for energy saving - intelligent free-cooling*, March 2007.

[81] Uniflair, *Engineering data manual - aquaflair*, October 2011.

[82] *Andrisoft wanguard ddos detection software*, `https://www.andrisoft.com/`.

[83] Tom White, *Hadoop : The definitive guide*, O'REILLY, 2012.

[84] *Hadoop yarn*, `http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html`.

[85] Jianqi Zhu, Feng Fu, Kexin Yin, Haizhen Li, and Yanheng Liu, *A novel ddos attack detection method*, Journal of Computational Information Systems **9** (2013), no. 2, 549–556.

Table des figures

1.1 Gestion des plateformes dans le Cloud (http ://www.cloud-serveur.fr/fr/le-cloud/cloud-kesako) 3

2.1 Architecture du système de fichiers HDFS 25
2.2 Architecture Hadoop MapReduce 26
2.3 Fonctionnement de MapReduce 27
2.4 Architecture Yarn (MRv2) - (http ://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html) 28
2.5 Composants Apache Spark 30
2.6 Exemple de topologie Storm 33
2.7 Architecture d'un cluster Storm 34

3.1 Système d'optimisation du refroidissement 47
3.2 La distribution électrique du datacenter Eolas 49
3.3 Vue détaillé des systèmes de refroidissement redondés électriquement 52
3.4 Circuits de refroidissement du datacenter Eolas 53
3.5 Permutation de mode de refroidissement en fonction de la température externe 54
3.6 Un cube informatique avec confinement d'allée chaude 56
3.7 Vue temps réel de l'état et des métriques du système de refroidissement 58
3.8 Variation de l'activité du datacenter sur une journée 59
3.9 Variation de l'activité du datacenter sur 14 jours 60
3.10 Prédiction de l'activité du datacenter sur une semaine avec SARIMA. 61
3.11 Choix du système de refroidissement en fonction de la température extérieure 62

3.12 Limitation des transitions des systèmes de refroidissement en fonction de la température extérieure 63
3.13 Consommation informatique globale du datacenter d'Eolas sur une journée 68
3.14 Variation de la consommation d'un serveur équipé d'un processeur Intel Haswell en fonction de la charge CPU 69
3.15 Variation de la charge CPU en % dans un cluster en production . 70
3.16 Fonctionnement du datacenter en mode "Natural Cooling" 71
3.17 Fonctionnement du datacenter en mode "FreeCooling" 72
3.18 Optimisation du refroidissement avec une température stable . . . 73
3.19 Exécution de l'algorithme sans limitation des transitions 73
3.20 Exécution de l'algorithme avec limitation du nombre de transitions 74

4.1 Architecture générale de la plateforme de détection 81
4.2 Décomposition du trafic réel en trafic normal et anormal (Données du datacenter Eolas) 83
4.3 Architecture globale de l'architecture Lambda 85
4.4 Chaine de traitement globale données de pare-feux 88
4.5 La topologie Storm de détection des attaques en temps réel 93
4.6 La plateforme Hadoop d'évaluation 95
4.7 La plateforme Apache Storm d'évaluation 96
4.8 Architecture réseau des datacenters Eolas 97
4.9 Activité CPU des noeuds durant la création du modèle (3 workers) 99
4.10 Activité CPU et mémoire des noeuds durant la création du modèle (6 workers) 100
4.11 Activité CPU lors du calcul de la covariance avec Spark 101
4.12 Évolution du trafic lors des grosses attaques DDoS 104
4.13 Détection du dépassement de seuil du trafic anormal 104

Liste des tableaux

1.1 Récapitulatif des niveaux de Tier dans un datacenter 6

3.1 Réglementation pour l'utilisation de l'eau de la nappe phréatique 57

4.1 Mesure des latences d'exécution des noeuds de la topologie Storm 102
4.2 Taux de détection du modèle ACP 105