Etude et mise en place d'Edge Node sur la base d'OpenEdgeComputing

HOANG Tuan Dung, KAF Merwan, LE CORRE Pierre

***Résumé*—Avec l'arrivée massive des objets connectés, la bande passante du réseau internet risque d'être saturé par la trop grande demande de ressources faites aux serveurs situés dans le Cloud. Afin de palier à ce problème, Orange Labs nous a demandé d'étudier une des solutions existantes : Open Edge Computing. Cette solution consiste à se servir de la puissance de calculs de machines situés en périphérie du réseau (mini datacenter, livebox, ...) afin de permettre aux objets connectés d'accéder à de la puissance de calculs sans remonter jusqu'au serveurs situés dans le cloud. Il nous a ensuite été demandé de nous pencher sur l'implémentation d'une telle solution et d'en estimer la viabilité.**

***Mots clés*—Cloudlet, Edge Node, Open Edge Computing, Open Stack, Cloud, Virtualisation, Zeroconf**

I. Introduction

Le projet que nous a confié la société Orange s'inscrit dans l'anticipation de la multiplication future des objets connectés à internet. Ces derniers utilisent la puissance de calculs de serveurs distants très haut dans le réseau - que l'on appelle plus communément Cloud computing - afin d'effectuer divers traitement. Cependant, Orange estime que leurs nombres augmentera de façon tellement importante d'ici 5 à 10 ans que ces objets engorgeront le réseau ce qui altérera l'expérience des utilisateurs. Afin de palier à ce problème futur, Orange réfléchis à divers manières d'éviter aux objets connectés d'effectuer leurs traitements très haut dans le réseau. (cf. figure 1)

Nous avons donc travaillé sur le concept de Edge Computing permettant de fournir de la puissance de calculs beaucoup plus bas dans le réseau et au plus proche des utilisateurs. L'intérêt est donc double, à la fois désengorger le réseau mais aussi diminuer la latence. Dans ce concept notre but est d'évaluer une solution open source : OpenEdgeComputing et de la mettre en oeuvre en proposant des outils spécifiques au besoin de notre client.

Notre critère d'évaluation se pose sur les rapports d'état de l'art de notre recherche. En effet, ayant partis dans plusieurs références bibliographiques repositories de code, nous devons former notre opinion sur l'état de l'art de chaque concept étudié, ainsi que leur faisabilité en prenant compte des ressources de logiciel open source à notre disposition.

II. Contexte technique

Dans le cadre de ce sujet, il peut-être utile de définir ou de rappeler certaines notions indispensables à la compréhension du projet :

FIGURE 1. Que préférer ? Le gout, la couleur ou les deux ?

- Cloud computing Ici la notion principale du cloud computing qui nous intéresse est l'IAAS, traduction de "infrastructure as a service", elle consiste
- Edge computing
  - Cloudlet
  - OpenEdgeComputing
- OpenStack

III. Méthode

Notre projet s'est déroulé en 3 phases distinctes :

- Recherche sur edge computing et plus spécifiquement sur le projet open source OpenEdgeComputing ;
- Mise en place du projet OpenEdgeComputing ;
- Mise en place de notre propre solution.

A. Recherche sur OpenEdgeComputing

La première partie de notre projet consistait à l'étude des différents projets EdgeComputing, et plus précisément celui de OpenEdgeComputing. Nous avons étudié l'architecture général des différents projets EdgeComputing (cf. figure représentant l'architecture).

B. Mise en place de OpenEdgeComputing

Nous avons mis en place le projet qui se base en grande partie sur le projet OpenStack qui permet de gérer un ensemble de serveur afin de constituer un cloud utilisable et manageable. Le projet OpenEdgeComputing est en quelque sorte une surcouche d'Openstack simplifiant son utilisation avec des cloudlets en apportant quelques nouvelles fonctionnalitées : La synthèse de machines virtuelles (VM Synthesis) Une gestion de la continuité de certaines machines virtuelles (VM handoff) Cartographie des Cloudlets pour sélectionner le plus proche (Discovery)

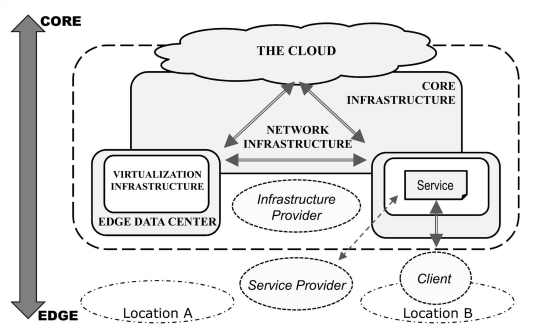

FIGURE 2. Que préférer ? Le gout, la couleur ou les deux ?

La première partie consiste à installer Openstack Kilo, ensuite le projet OpenEdgeComputing possède plusieurs parties qui sont liées aux différentes fonctionnalitées ajoutées. La première à installer est la synthèse de machines virtuelle, elle consiste en l'installation d'une version modifiée de qemu ainsi que d'outils permettant la gestion de la synthèse de machines virtuelles. La deuxième partie consiste en une modification de l'interface d'openstack, qui inclut désormais la gestion des cloudlets.

C. Mise en place de notre solution

Nous avons finalement mis en place notre solution. Nous avons discuté de l'architecture général de notre application avec M. LE TOQUIN. (cf figure de notre architecture).

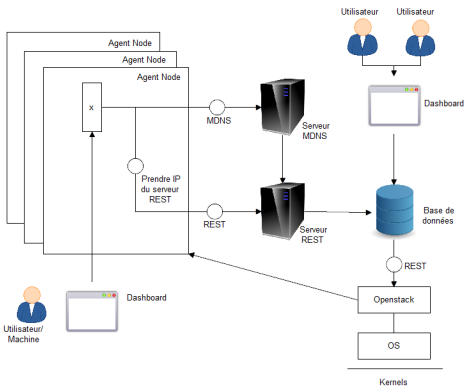

FIGURE 3. Que préférer ? Le gout, la couleur ou les deux ?

Nous pouvons distinguer deux parties dans cette architecture. La partie OpenStack La partie applicative

1) Langage : Python: Nous avons choisi d'utiliser le langage Python pour sa simplicité et sa mise en oeuvre rapide et efficace pour une architecture légère. Ses performances sont suffisantes pour l'application déployé.

2) Gestion du cloud : Agent OpenStack: L'agent OpenStack permet une connection à une infrastructure Openstack et une gestion de certains processus de virtualisation afin de pouvoir utiliser la puissance des équipements sur lesquels cet agent est installé, comme un noeud nova-compute standard. Un noeud nova-compute permet d'exécuter des machines virtuelles sur un ordinateur, sous la supervision d'OpenStack. Cet agent est basé sur docker ainsi que sur ubuntu 10.04 : Ce choix est basé sur deux choses :

— Docker permet un déploiement rapide et sans dépendance logicielle autre que lui-même
— Ubuntu 14.04 est la version imposé par le projet OpenEdgeComputing, car celui-ci ne fonctionne que sur cette distribution linux.

Le fonctionnement de cet agent est le suivant :

— Il va dans un premier temps récupérer l'adresse ip du serveur OpenStack qui lui correspond via l'agent node.
— Puis il va s'y connecter pour pouvoir offrir sa puissance de calcul à l'infrastructure OpenStack.

3) Interface utilisateur : Agent Node: L'interface utilisateur appelé Agent Node permet à un utilisateur de notre application de pouvoir venir mettre des ressources matérielles à disposition. Pour cela une interface listant tous les objets compatibles avec le cloud et OpenStack est à sa disposition. (cf figure interface) En venant cliquer sur le bouton "NOM DU BOUTON", l'utilisateur enregistre directement son appareil dans la base de donnée où se trouve toutes les ressources disponibles sur le cloud d'Orange. Pour cela nous avons utilisé un serveur REST ainsi qu'un serveur mDNS détaillé ci-dessous.

4) Processus de découverte : mDNS: Lorsqu'un utilisateur se connecte à notre interface, il doit pouvoir mettre à disposition les ressources qu'il souhaite. Pour ce faire, la première étape consiste en la découverte du serveur REST ainsi qu'en la découverte du serveur OpenStack. Le protocole mDNS-SD signifie multicast Domain Name Service - Service Discovery. Il s'agit une mécanisme de adresser les messages dans le réseau sans savoir à quel périphérique étranger qu'on communique (mDNS), ainsi que la découverte de services (Service Discovery). En prenant compte des conseils de M. Letoquin, nous avons utilisé la bibliothèque zeroconf écrite en Python. La solution zeroconf permet de découvrir de manière simple les services se situant sur le même réseau. Nous avons donc implémenté un serveur mDNS permettant de publier le service du serveur REST, transmettre les informations de ce serveur et du serveur OpenStack sur le réseau. Un simple script permet ensuite à notre Agent Node la découverte de ces serveurs et de communiquer avec.

5) Serveur REST: Une fois la découverte effectué, afin que l'utilisateur puisse mettre à disposition ses appareils, il doit pouvoir enregistrer son appareil dans la base de donnée regroupant toutes les ressources disponibles sur le réseau d'Orange. Pour ce faire, nous avons implémenté un serveur REST (representational state transfer). Un serveur REST per-

met de comprendre les requêtes envoyés par un client via un format défini au préalable et ensuite enregistré les données reçues dans une base de donnée. Nous avons utilisé le format Json, très utilisé et simple d'utilisation. (cf. Exemple de json)

FIGURE 4. Exemple de JSon

6) Affichage de la base de donnée: Afin de pouvoir bien s'en rendre compte du bon enregistrement des appareils dans la base de donnée, nous avons créé une interface affichant la base de donnée. (cf. image dashboard)

FIGURE 5. Représentation de notre Dashboard

IV. RÉSULTATS

On dit un truc du genre on va diviser notre truc en 3 partie

A. Recherche sur OpenEdgeComputing

Les recherches effectués effectués sur OpenEdgeComputing n'ont pas été particulièrement convaincantes. Le projet OpenEdgeComputing est un projet universitaire assez bien documenté, cependant ce projet n'est plus maintenu depuis plus d'un an maintenant (3ans pour le code source). En plus, ce projet OpenEdgeComputing a été partiellement privatisé par les entreprises. Cela explique pourquoi nous avons une diffculté de progresser dans notre recherche. Le deuxième projet traitant de la même problématique est OpenFog cependant ce projet n'en est encore qu'à ses balbutiements, et le premier forum public de ce consortium ne se tiendra qu'en mars.

On parle un peu de Openstack/Devstack ici ? ? ?

B. Mise en place de OpenEdgeComputing

La mise en place d'OpenEdgeComputing n'a pas été concluante en effet le manque de maintien du projet fait que de nombreux liens notamment au niveau des VMs ne sont plus disponibles. Le projet est à l'heure actuelle très difficilement exploitable et absolument pas prêt pour une quelconque mise en production.

C. Mise en place de notre solution

La solution que nous avons mise en oeuvre est concluante et fonctionne bien d'un point de vue logique à petite échelle. Il faudrait cependant déployer notre application à grande échelle afin de voir si celle-ci tient la scalabilité.

V. ANALYSE

Une importante partie de notre travail a été consacré à la découverte de Edge Computing et de l'état de l'art des existants. Cependant peu de chose ont été faites dans ce domaine et il nous a fallu repartir de 0 et implémenter notre solution. Celle-ci n'est probablement pas exploitable en l'état à grande échelle mais permet de poser les bases sur lesquelles s'appuyer pour un développement futur plus important.

VI. DISCUSSION

Le Edge Computing n'en est qu'à ses balbutiements, il n'y a pas encore de standard de disponible sur le marché que cela vienne d'un acteur privé ou du monde de l'open source. Cependant certains paradigmes semble revenir comme le montre l'article (scientifique traité par Merwan a retrovué et a cité). En effet, l'usage du protocole mDNS-SD est très connu dans le monde professionnel, par contre, aucun essai a été mené pour améliorer la solution cloud. Notre solution est basé sur ces paradigmes mais ne présente qu'un début de solution. Cette solution nécessite une recherche plus coûteuse en demandant d'accès à des logiciels à propriété privée.

VII. RÉFÉRENCES BIBLIOGRAPHIQUES