INTRODUCTION DU COURS POUR APPRENDRE L'ORCHESTRATEUR KUBERNETES (K8S)

Introduction

Hello world, Vous allez bien ? Prêt pour attaquer un nouveau cours ? C'mon here we go !

Chose promise chose due, Je m'étais engagé dans le cours complet sur la technologie Docker, plus précisément dans le chapitre sur Docker Swarm, à faire un cours sur un autre orchestrateur autre que Docker Swarm. Et voilà, nous y sommes, ça sera donc un **cours destiner complètement à l'orchestrateur de conteneur Kubernetes**, commençons d'abord par découvrir son histoire.

Histoire de Kubernetes

Lorsque nous examinons l'histoire de Kubernetes , il est facile de comprendre la motivation qui a poussé les inventeurs de l'outil, à créer ce système d'orchestration de conteneurs, atteignant aujourd'hui un seuil de **maturité** assez exceptionnel.

Pourquoi cet intérêt pour les conteneurs ?

L'histoire de Kubernetes a commencé chez Google, la société avait besoin d'une infrastructure énorme afin de mettre à la portée de tous, son moteur de recherche et les publicités associées. La croissance prévue est juste astronomique, pour lequel diverses idées ont été soulevées.

Nous avions déjà abordé dans cet article les différences entre la virtualisation et la conteneurisation , où j'avais expliqué entre autres, que les machines virtuelles intègrent elles-mêmes un OS pouvant aller jusqu'à des Giga-octets, cependant, ce n'est pas le cas du conteneur. Le conteneur appelle directement l'OS pour réaliser ses appels système et exécuter ses applications. Il est par conséquent beaucoup **moins gourmand en ressources**.

Et c'est exactement le même problème qu'avait rencontré Google à l'époque. En effet, malgré la mise en place d'un système de virtualisation, le potentiel des ressources n'était pas encore pleinement exploité de manière appropriée, ainsi **les coûts informatiques n'étaient pas totalement maîtrisés**, étant donné que le géant du web **payait pour des ressources qu'il ne consommait pas**.

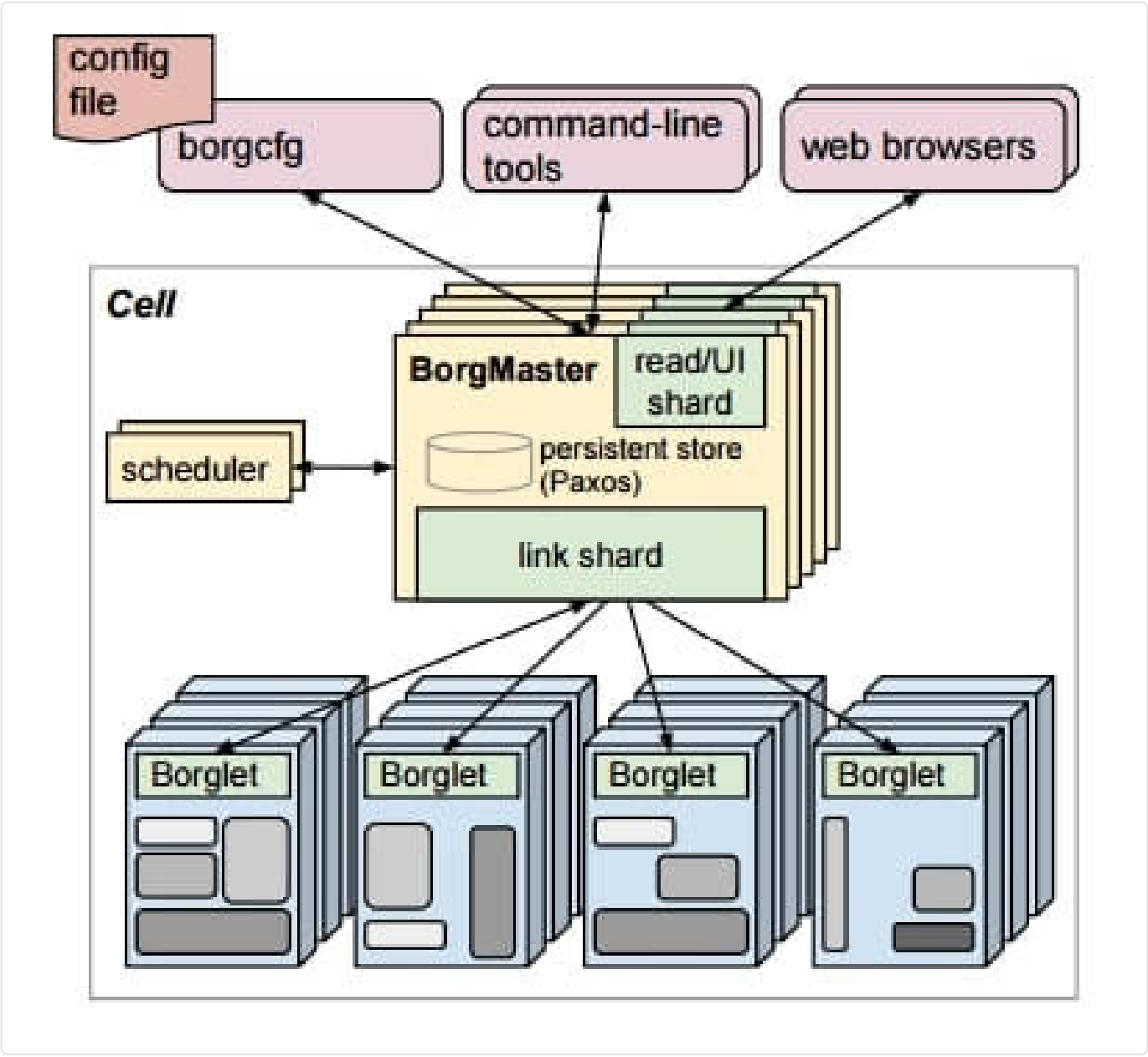
Clairement, la solution était les conteneurs. Bien que l'intérêt général pour les conteneurs de logiciels soit un phénomène relativement récent, Google gère depuis plus de dix ans les conteneurs Linux à grande échelle et a construit trois systèmes

de gestion de conteneurs.

Création de la première solution, le projet Borg

Il avait notamment commencé par construire un système de gestion de cluster appelé **Borg**. Voici une **explication simple du mode de fonctionnement du système Borg de Google** :

Lorsque vous avez une machine standard et que vous souhaitez exécuter plusieurs processus dessus, c'est le système d'exploitation qui gère l'ordonnancement. C'est lui qui détermine quel processus a accès à quel processeur, quel processus a accès à quelles parties mémoire, quels sont les processus qui doivent être gelés ou disparaître dans les cas de pénurie de ressources. Véritablement Borg fait la même chose mais à l'échelle d'un cluster, c'est en quelque sorte, un OS pour cluster. Lorsque vous désirez exécuter un job, vous le faite depuis Borg, vous dites simplement au planificateur de cluster Borg ce que vous voulez, comme par exemple *"lance-moi 300 répliques de mon application web"*. Ensuite le planificateur Borg identifie les machines du cluster qui disposeront de la capacité disponible pour exécuter votre service et envoie une demande d'exécution du service aux nœuds (machines qui constituent votre Cluster).



Google a initialement conçu Borg pour **répondre à ses propres besoins internes**, Borg restait le principal système de gestion des conteneurs au sein de Google en raison de son ampleur et de l'étendue de ses fonctionnalités. Par la suite Google développe Omega, qui est une progéniture de Borg, afin d'**améliorer l'ingénierie logicielle de l'écosystème Borg**.

L'arrivé du projet Kubernetes

Arrive alors en 2014 Kubernetes souvent abrégé k8s, "k + 8 caractères + s", écrit en Go (j'ai d'ailleurs fait un cours complet sur le langage Go, ici) et né à partir du projet Borg et Omega. Ces deux derniers ont été développés en tant que systèmes purement internes à Google, à l'inverse de Kubernetes, qui est quant à lui maintenant devenu une version **opensource** (code source disponible ici) de ses prédécesseurs avec des fonctionnalités améliorées permettant de résoudre les anciens problèmes de gestion des clusters, ainsi qu'un support accru de la part d'IBM, Cisco et Redhat.

De nos jours le projet n'appartient plus vraiment à Google, car Google a fait don du projet Kubernetes en 2015 à la toute récente Cloud Native Computing Foundation.

Public visé

Ce tutoriel est conçu pour les débutants ayant besoin de **comprendre la technologie Kubernetes à partir de zéro**. Ce tutoriel vous donnera une compréhension suffisante de la technologie, qui vous permettra plus tard d'atteindre des niveaux d'expertise beaucoup plus élevés.

Prérequis

Avant de poursuivre ce cours, vous devez au minimum avoir une compréhension de base sur les commandes Linux et sur la technologie Docker, si vous n'avez jamais touché à Docker de votre vie, alors je vous conseille grandement dire lire mon cours complet sur Docker. Si vous maîtrisez Docker, plus précisément Docker Swarm, alors il vous sera très facile de comprendre les concepts de Kubernetes et d'avancer rapidement sur la piste d'apprentissage, mais ce n'est pas non plus indispensable.

Êtes-vous prêt pour découvrir des nouveaux horizons ? alors c'est parti !