Conteneurisation : (8)

o Définition :

Il s’agit d’un type de virtualisation utilisé au niveau des applications. Elle consiste à rassembler le code du logiciel et tous ses composants (bibliothèques, Framework et autres dépendances) de manière à les isoler dans leur propre « conteneur ». Le logiciel ou l'application dans le conteneur peut ainsi être déplacé et exécuté de façon cohérente dans tous les environnements et sur toutes les infrastructures, indépendamment de leur système d'exploitation.

Le conteneur fonctionne comme une sorte de bulle, soit un environnement de calcul qui enveloppe l'application et l'isole de son entourage. C'est en fait un environnement de calcul portable complet.

o Principe de la conteneurisation :

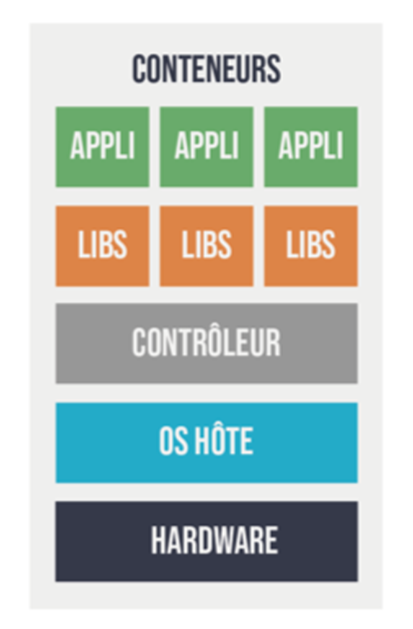
Le principe de la conteneurisation repose sur la création de plusieurs espaces utilisateurs isolés les uns des autres sur un noyau commun. Elle permet aux développeurs de créer et de déployer des applications plus rapidement et de manière plus sécurisée.

Avec les méthodes traditionnelles, le code est développé dans un environnement informatique spécifique. Lorsqu’il est transféré vers un nouvel emplacement, cet environnement entraîne souvent des bogues et des erreurs.

Par exemple, lorsqu’un développeur transfère du code d’un ordinateur de bureau vers une machine virtuelle (VM) ou d’un système d’exploitation Linux vers un système d’exploitation Windows.

La conteneurisation élimine ce problème en regroupant le code de l’application avec les fichiers de configuration, les bibliothèques et les dépendances associés nécessaires à son exécution. Ce package unique de logiciel ou « conteneur » est extrait du système d’exploitation hébergeur.

Par conséquent, il est autonome et devient portable – capable de fonctionner sur n’importe quelle plateforme ou cloud, sans problèmes

source 1:<https://www.selceon.com/glossary/conteneurisation-cloud-computing> 

source 1:https://www.selceon.com/glossary/conteneurisation-cloud-computing

o Inconvénients :

* **Incompatibilité avec certaines tâches**

Thomas Bittman fait remarquer que les conteneurs, bien que polyvalents, sont loin de pouvoir remplacer tous les déploiements de machines virtuelles (VM) existants. En effet, tout comme d'anciennes applications se prêtaient mieux à des déploiements physiques aux premiers temps de la virtualisation, certaines applications ne conviennent pas du tout à une virtualisation en conteneurs.

* **Problème des dépendances**

Les VM classiques sont extrêmement autonomes, chacune comprenant un système d'exploitation (OS) à part entière, des pilotes et des composants d'application. Elles peuvent également migrer vers n'importe quel autre système du moment qu'un hyperviseur approprié est disponible. De leur côté, les conteneurs s'exécutent sur un OS et partagent la majeure partie du noyau sous-jacent ainsi qu'un grand nombre de fichiers binaires et de bibliothèques. D'après Bittman, les dépendances imposées aux conteneurs peuvent limiter la portabilité entre serveurs.

* **Faiblesse relative de l'isolement**

Les VM reposant sur un hyperviseur offrent un degré élevé d'isolement les unes des autres, car les ressources matérielles du système sont toutes virtualisées et présentées aux VM par le biais de l'hyperviseur. Autrement dit, un bug, un virus ou une intrusion peut porter atteinte à une VM sans se propager aux autres.

* **Risque de prolifération**

Si la gestion du cycle de vie des VM est importante dans les environnements basés sur un hyperviseur, elle s'avère absolument essentielle pour les conteneurs. En effet, ces derniers offrent l'avantage non négligeable de pouvoir être mis en service et dupliqués à la vitesse de l'éclair.

Le revers de la médaille est qu'il est également possible de consommer une grande quantité de ressources informatiques sans vraiment s'en rendre compte.

Ce n'est pas très grave si les conteneurs qui composent l'application sont arrêtés ou supprimés dès lors qu'ils ne sont plus nécessaires. Mais en cas d'oubli, la montée en charge d'une application conteneurisée peut se traduire par des coûts de Cloud Computing tout aussi importants qu'inutiles pour l'entreprise.

o Avantages :

· **Portabilité** : un conteneur créé un package exécutable de logiciels extrait du système d’exploitation hébergeur (non lié ou dépendant de celui-ci). Par conséquent, ce package est portable et capable de fonctionner de manière uniforme et cohérente sur n’importe quelle plateforme ou cloud.

· **Agilité** : Le Docker Engine open source qui exécute les conteneurs a lancé la norme de l’industrie pour les conteneurs avec des outils de développement simples et une approche de packaging universelle qui fonctionne à la fois sur les systèmes d’exploitation Linux et Windows.

· **Vitesse** : les conteneurs sont souvent qualifiés de « légers ». Cela signifie qu’ils partagent le noyau du système d’exploitation (OS) de la machine et ne sont pas embourbés par ces frais supplémentaires.

Non seulement cela améliore l’efficacité des serveurs, mais cela réduit également les coûts de matériel et de licence. De plus, les temps de démarrage sont accélérés étant donné qu’il n’y a pas de système d’exploitation à lancer.

· **Isolation des erreurs** : chaque application conteneurisée est isolée et fonctionne indépendamment des autres. La défaillance d’un conteneur n’affecte pas le fonctionnement continu des autres conteneurs. Les équipes de développement peuvent identifier et corriger tout problème technique dans un conteneur sans aucun temps d’arrêt de ceux en cours de fonctionnement. En outre, le moteur de conteneur peut tirer partie de toutes les techniques d’isolation et de sécurité du système d’exploitation, telles que le contrôle d’accès SELinux, pour isoler les erreurs dans les conteneurs.

· **Efficacité** : les logiciels exécutés dans des environnements conteneurisés partagent le noyau du système d’exploitation de la machine. De plus, les couches d’application d’un conteneur peuvent être partagées avec les autres. Ainsi, les conteneurs ont une capacité intrinsèquement plus petite qu’une machine virtuelle et nécessitent moins de temps de démarrage. Cela permet à beaucoup plus de conteneurs de s’exécuter avec la même capacité de calcul d’une seule machine virtuelle. Ce qui améliore l’efficacité des serveurs, réduisant les coûts de matériel et de licence.

· **Facilité de gestion** : une plateforme d’orchestration de conteneurs automatise l’installation, la mise à l’échelle et la gestion des charges de travail et des services conteneurisés. Les plateformes d’orchestration de conteneurs peuvent faciliter les tâches de gestion telles que la mise à l’échelle des applications conteneurisées, le déploiement de nouvelles versions d’applications et la surveillance, la journalisation et le débogage, entre autres fonctions. Kubernetes est une technologie open source (à l’origine open-source par Google, basée sur leur projet interne appelé Borg). Il se classe parmi les systèmes d’orchestration de conteneurs les plus populaires disponibles. A l’origine, il automatise les fonctions des conteneurs Linux. Kubernetes fonctionne avec de nombreux moteurs de conteneur, tels que Docker, mais aussi avec tout système de conteneur conforme aux normes Open Container Initiative (OCI) pour les formats d’image de conteneur et les environnements d’exécution.

· **Sécurité** : l’isolement des applications en tant que conteneurs empêche par nature l’invasion de codes malveillants les empêchant d’affecter d’autres conteneurs ou le système hôte. De plus, des autorisations de sécurité peuvent être définies pour empêcher automatiquement les composants indésirables d’entrer dans des conteneurs ou limiter les communications avec des ressources inutiles.