DÉPLOIEMENT DE SERVICES AVEC DOCKER ET DOCKER COMPOSE

DESCRIPTION DU THÈME

Propriétés	Description
Intitulé long	Déploiement de services avec Docker et Docker-compose
Formation(s) concernée(s)	☐ Classes de première Sciences et technologies du management et de la gestion (STMG) ☐ Terminale STMG Système d'information de gestion (SIG) ☒ BTS Services Informatiques aux Organisations
Matière(s)	☐ Sciences de gestion ☐ SIG ☒ Bloc 1 – Support et mise à disposition de services informatiques ☒ Bloc 2 SISR – Administration des systèmes et des réseaux ☐ Bloc 3 SLAM – Cybersécurité des services informatiques
Présentation	L'objectif de ce « Labo » est de comprendre l'intérêt de l'outil Docker et d'en tester les principaux usages (déployer rapidement plusieurs instances d'un service pour les SISR, développer et tester une application pour les SLAM). Seul Docker sur Linux est étudié ici mais les principes restent les mêmes sur Windows. Ce « Labo » comporte **5 activités** qui peuvent être réalisées en bloc1 pour tous les étudiants ou en bloc 2 SISR (voir compétences associées) : • **Activité 1** : installation et première exploitation de Docker • **Activité 2** : déploiement d'un service avec une gestion des données persistantes • **Activité 3** : construction d'une image avec un dockerfile et publication sur Docker Hub (le registre public de Docker) • **Activité 4** : déploiement de la distribution Kali-Linux avec gestion du réseau • **Activité 5** : déploiement d'un service avec Docker Compose

Savoirs	Savoir-faire : • Justifier le choix d'une solution de mise en production d'un service et d'un système • Automatiser l'installation d'un service • Valider et documenter la mise en exploitation d'un service Savoirs associés : • Mise en production, méthodes, technologies, techniques, normes et standards associés • Technologies et techniques associées à l'installation des services • Test d'intégration d'un service
Compétences	**Bloc 1** : • Mettre à disposition des utilisateurs un service informatique **Bloc 2** : • Déployer une solution d'infrastructure
Transversalité	
Prérequis	Commandes de base d'administration d'un système Linux. Notions de virtualisation.
Outils	Un serveur physique ou virtuel avec une distribution Linux 64 bits (ici Debian 12 – version stable actuelle) sur lequel Docker et Docker Compose seront installés. Sites officiels : https://www.docker.com/, https://hub.docker.com et https://docs.docker.com/
Mots-clés	Docker compose virtualisation container conteneur services micro-services déploiement
Durée	12 heures
Auteur·e·s	Apollonie Raffalli
Version	v1
Date de publication	Septembre 2024

DERNIÈRES RÉVISIONS

Ce tableau contient les modifications apportées au document après sa publication uniquement.

Date	Auteur·e	Description

Indications aux professeurs :

Il est possible, avant de commencer les activités, de :

- visualiser une ou plusieurs vidéos présentant Docker et/ou rappelant les techniques de virtualisation. Les vidéos qui suivent sont un peu anciennes mais toujours d'actualité :
  - cookie connecté - Docker : l'essentiel en 7 minutes :

    https://www.youtube.com/watch?v=caXHwYC3tq8
  - bien comprendre et maîtriser Docker : https://www.youtube.com/watch?v=jkRGw3zgHuQ
- lire ce qui suit dans la présentation ;
- vérifier ensuite avec éventuellement un QCM les acquis des étudiants concernant les notions de virtualisation et les différences entre la virtualisation classique et les conteneurs.

CONTEXTE

Née en décembre 2010, la société CUB est une entreprise spécialisée dans l'incubation de startups partageant les mêmes valeurs de solidarité et de développement durable. Au travers de sa plate-forme web, CUB permet à des professionnels d'accéder à des espaces de travail dédiés : salles de réunion, de formation ou de séminaire.

Le concept novateur de CUB repose sur une démarche collaborative de type « BtoB », en effet CUB propose aux entreprises qui disposent d'espaces inoccupés de les louer à l'heure ou à la journée.

Forte d'une croissance très rapide, CUB fait une levée de fonds de 1,2 millions d'euros en 2016 et développe son activité pour atteindre sa dimension actuelle d'incubateur.

À la différence d'une pépinière d'entreprises classique, CUB s'adresse à des sociétés très jeunes, ou encore en création, pour leur apporter un appui lors des premières étapes de la vie de l'entreprise. Elle met à disposition de ses clients un ensemble de solutions techniques d'accès dans un millier de salles de réunion situées dans une quarantaine de villes différentes. Les ressources et outils du Web 2.0 qui permettent aux entreprises de gérer leurs contenus et leurs connaissances de manière sécurisée sont accessibles indépendamment via des prestataires de type informatique dans les nuages (cloud computing) : partage de fichiers, gestion de projet, réseau social d'entreprise, wiki d'entreprise, etc.

La direction des systèmes d'information (DSI), située au siège à Paris, participe étroitement aux choix stratégiques de CUB, elle a pour mission de définir et mettre en œuvre la politique informatique en accord avec la stratégie générale et ses objectifs de performance.

CUB offre de nombreux services applicatifs Web, par exemple :

- un site Web basé sur le CMS WordPress ;
- l'application phpIPAM ;
- des applications développées en interne.

La DSI a décidé de :

- **migrer l'ensemble de ces applications, actuellement en production sur des machines virtuelles, dans des conteneurs Docker ;**
- **mettre à la disposition de ses développeurs et de ses administrateurs réseaux des distributions personnalisées sous forme de conteneurs docker.**

Les trois premières activités vous permettront de vous familiariser avec la technologie Docker avant de commencer à procéder aux déploiements demandés dans les deux dernières activités.

QU'EST-CE QUE DOCKER ?

Docker est une plateforme open source qui automatise le déploiement, la gestion et l'exécution d'applications dans des conteneurs légers. Un conteneur est une unité logicielle qui encapsule une application et ses dépendances, garantissant ainsi sa portabilité et son exécution cohérente quel que soit l'environnement.

Docker a été initialement développé par Solomon Hykes pour un projet interne à la société de *plateforme en tant que service* (platform as a service – PaaS[1]) qu'il a créée en 2008, *dotCloud*. Docker a été distribué en tant que projet open source en 2013, projet qui a rapidement été très actif avec de nombreux contributeurs.

Le nom Docker recouvre le logiciel lui-même, la technologie et les outils qu'il embarque, le projet d'une communauté Open Source (communauté Open Source Docker), la société créée par Solomon Hykes (*Docker inc*) en charge du développement qui s'appuie sur le travail de la communauté et le nom de la plateforme officielle en ligne. À noter que Salomon Hykes a quitté la société en mars 2018.

Docker est une **technologie de conteneurisation** reposant sur le noyau Linux et ses fonctionnalités de virtualisation par conteneurs (LXC pour Linux Containers), notamment :

- le composant *cgroups* pour contrôler et limiter l'utilisation des ressources pour un processus ou un groupe de processus (utilisation de la RAM, CPU entre autres) associé au système d'initialisation *systemd* qui permet de définir l'espace utilisateur et de gérer les processus associés ;
- les *espaces de noms* ou *namespaces* qui permettent de créer des environnements sécurisés de manière à isoler les conteneurs et empêcher par exemple qu'un groupe puisse « voir » les ressources des autres groupes.

Docker offre des outils pour utilise ces fonctionnalités de manière simplifiée pour permettre, entre autres :

- la duplication et la suppression des conteneurs ;
- l'accessibilité des conteneurs à travers la gestion des API et CLI ;
- la migration (à froid ou à chaud) de conteneurs.

Un conteneur Docker se construit à partir d'une image. Une image Docker est un package léger, autonome et exécutable d'un logiciel qui inclut tout ce qui est nécessaire pour l'exécuter : code de l'application, environnement d'exécution (runtime), outils système et librairies, etc.

De nombreuses images comme *Nextcloud,* Debian sont disponibles sur :

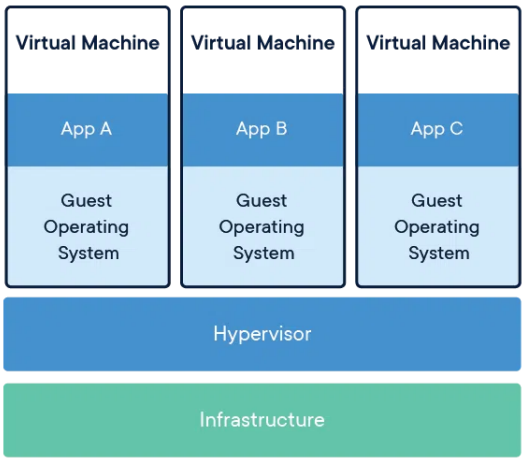
- le registre officiel (appelé *hub*): https://hub.docker.com ;
- de nombreux dépôts initiés par de « simples » utilisateurs.

Il est bien sûr possible de proposer des images, d'en modifier d'autres et de déposer la modification sur le dépôt officiel.

1 ***PaaS*** *est un des types de cloud computing où le fournisseur met à disposition des entreprises un environnement d'exécution rapidement disponible (infrastructure, serveurs, logiciels de base, etc) en leur laissant la maîtrise des applications qu'elles peuvent installer, configurer et utiliser elles-mêmes.*

LES FORCES DE DOCKER FACE À UNE VIRTUALISATION CLASSIQUE

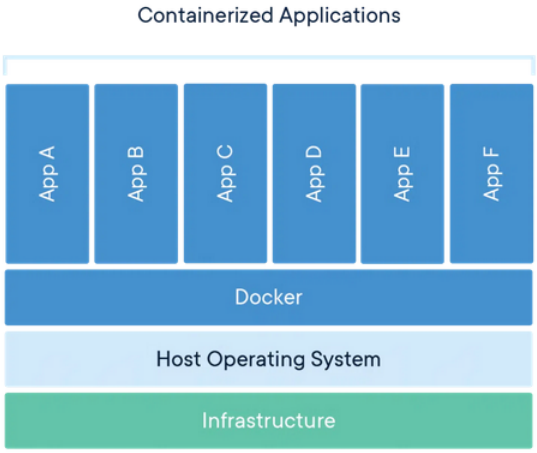
https://www.docker.com/what-container



La virtualisation classique permet, via un hyperviseur, de simuler une ou plusieurs machines physiques, et de les exécuter sur le serveur hôte sous forme de machines virtuelles (VM). Ces VM intègrent elles-mêmes un système d'exploitation complet sur lequel les applications qu'elles contiennent sont exécutées.

L'hyperviseur est donc responsable de tous les échanges de données. Exécuter plusieurs machines virtuelles sur un même serveur demande de grosses performances et un nombre suffisant de ressources pour assumer plusieurs machines virtuelles. Le démarrage d'une machine virtuelle peut être plus ou moins long en fonction aussi de la technique utilisée (virtualisation complète ou paravirtualisation).

https://www.docker.com/what-container



Les conteneurs Docker s'exécutent sur une machine hôte, partagent le noyau du système d'exploitation de cette machine et font directement appel à celui-ci pour exécuter les applications. Les conteneurs démarrent ainsi très rapidement et utilisent peu de ressources (processeur, mémoire vive, etc.).

Du fait que les conteneurs n'embarquent pas de système d'exploitation et qu'ils peuvent partager des fichiers communs, ils sont très légers : cela réduit l'utilisation du disque, ils migrent plus facilement d'une machine physique à une autre et les téléchargements d'images sont beaucoup plus rapides.

Par ailleurs, la technologie mise en œuvre isole les applications les unes des autres et de l'infrastructure sous-jacente.

Pour terminer cette rapide présentation, il est utile de préciser que les conteneurs Docker sont actuellement pris en charge par les principaux acteurs du cloud. Ainsi, un conteneur Docker peut se déployer en de multiples instances sur toutes les principales distributions Linux, Microsoft Windows, et sur toutes les infrastructures, y compris les machines virtuelles, bare-metal mais aussi dans le cloud.

DOCKER POUR QUOI FAIRE ?

Docker est une plateforme de conteneurisation révolutionnaire qui a changé la manière dont les applications sont développées, déployées et gérées. Depuis son introduction, Docker a eu un impact significatif sur la manière dont les équipes de développement et d'exploitation collaborent, accélérant le cycle de vie des applications et améliorant la flexibilité des déploiements.

Vous trouverez ici des statistiques pertinentes concernant Docker en 2024.

Pourquoi tant d'engouement ? Quelles sont les utilisations possibles, efficaces et pertinentes de Docker ?

- **Déployer rapidement un service** lorsque l'on a besoin de le déployer plusieurs fois : cette reproductibilité est la base de docker, c'est typiquement l'utilisation que peut en faire un fournisseur de cloud.
- **Distribuer une application** : Docker en tant que « système de distribution d'une application » est de plus en plus utilisé par les développeurs qui créent rapidement un serveur de développement et n'ont plus à packager une application sous différents systèmes (deb, rpm, etc) avant de la distribuer. Par ailleurs, plus de 75 % des organisations qui adoptent des pipelines CI/CD utilisent Docker pour la conteneurisation de leurs applications.

  Quant à l'utilisateur, il utilise une image docker (en général très légère) pour tester puis garder/migrer ou finalement jeter une application (plutôt que de compiler un tarball). Souvent quelques commandes suffisent pour avoir une solution comme Nextcloud ou Ansible opérationnelle.
- **Développer et tester une application** car Docker permet :
  - de concevoir une architecture de test plus agile, chaque conteneur de test pouvant par exemple intégrer une brique de l'application (base de données, langages, composants, etc.), le développeur pourra tester sur la même machine plusieurs versions d'un même logiciel en inter changeant le conteneur correspondant. Par exemple pour une application PHP, on pourrait facilement tester plusieurs versions de PHP comme plusieurs versions d'Apache ou de Nginx ;
  - de développer une application selon le concept d'architecture de micro-services avec pour chaque couche des conteneurs isolant les composants de l'application ;
  - de faciliter le process de mise à jour de l'application ; les images Docker sont versionnées et permettent une mise à jour simplifiée et maîtrisée. Le process de *rollback* est aussi simplifié : on redéploie la version précédente de l'image.
  - d'avoir un environnement de développement identique à l'environnement de production d'autant plus que du fait de la disparition du système d'exploitation intermédiaire classique des machines virtuelles, les développeurs bénéficient d'une pile applicative plus proche de celle de l'environnement de production ce qui engendre mécaniquement moins de mauvaises surprises lors des passages en production.

LES LIMITES DES CONTENEURS DOCKER

Les conteneurs Docker souffrent tout de même de quelques limites :

- **sécurité :** les conteneurs peuvent être plus vulnérables, car ils partagent un noyau et des composants systèmes et leur fonctionnement exige déjà un niveau d'autorisation élevé (généralement l'accès root dans les environnements Linux) : si toute l'architecture est basée sur Docker et si le système hôte est attaqué tous les services seront « accessibles » et exposés plus facilement et rapidement aux attaques. À noter que les plateformes de conteneurs évoluent dans le sens d'une plus grande sécurisation en matière d'isolement et de séparation des droits des OS ;
- **complexité : la multiplication facile des conteneurs** rend possible une consommation d'une grande quantité de ressources sans s'en rendre compte. Par ailleurs, la gestion d'un grand nombre de conteneurs peut devenir complexe. L'orchestration avec des outils tels que Kubernetes est souvent nécessaire pour simplifier le déploiement et la gestion en production ;
- **non idéal pour toutes les applications :** certaines applications avec des exigences spécifiques peuvent ne pas être adaptées aux conteneurs. Il est important de comprendre les besoins de l'application et du contexte avant de choisir Docker.