

## La plateforme Docker Enterprise Edition



Fig. 1: Projixi-Europe

### Version

Auteur	Modification	Version	Date
Alexandre Fernandes	Initialisation du mémo	1.0	12/02/2019
—	—	—	—

### Liens

[Webinar](#)

[Site officiel](#)

[Docker Enterprise hosted trial](#)

### Résumé

Docker Enterprise Edition (Docker EE) est une solution de conteneurisation destinée aux entreprises, elle dispose de fonctionnalités étendues par rapport au - bien connu - Docker Engine et déploie une plateforme unifiée de gestion de clusters de conteneurs. Son but est d'accompagner les organisations afin de moderniser leurs infrastructures applicatives en intégrant des fonctionnalités critiques pour les entreprises telle que notamment la sécurité et la gouvernance.

**Respect de 3 piliers :**

- **choix** : liberté opérationnelle des outils (CLI/Web UI), des orchestrateurs (Swarm/Kubernetes), des images (OS, applications), du type de cloud (public, privé, hybrid)
- **sécurité** : authentification, rôles, encryption, sécurité des images via des signatures et des scans de vulnérabilités
- **agilité** : interopérabilité, automatisation, accélération des déploiements d'applications et de leur mise à jour dans un cycle continu DevOps

**Tarification :**

Facturation au nombre de noeuds du cluster, disponible en 3 versions : - basique / standard / advanced

2 niveaux de support : - en jours ouvrés / en 24/24h

**Certification / Formation :**

- formations gratuites en ligne: playwithdocker, playwithkubernetes
- formations en classe orientées fondamentaux, operations, développeurs, sécurité et update
- certifications diplômantes en ligne :
  - niveau 1: Docker Certified Associate
  - niveau 2 : Docker Certified Professional

**Outils :**

- **Universal Control Plane (UCP)**: web console unifiée, permet de manager des clusters qui s'exécutent sur Swarm ou sur Kubernetes
- **Docker Trusted Registry (DTR)** : DockerHub privatif sécurisé
- **CLI** : le **Client Bundle** permet d'instancier un environnement pour les outils standard docker et kubectl
- **Docker Bench for Security** : outil d'audit de vulnérabilité du Docker Engine
- **Docker Application Converter**: DAC est une application qui va détecter les applications d'une machine monolithique virtuelle ou baremetal pour les “Dockériser”
- **Docker Application Designer**: permet créer des containers sans écrire une seule ligne de code de manière totalement transparente sans besoin de formation à Docker

## Démo en ligne

- 12H de démo : Docker Enterprise hosted trial.

Une fois le tutorial Kubernetes fini, vous pouvez avoir accès à la plateforme en mode test pendant 12H.

Par exemple, pour créer un service Jenkins avec Swarm , il faut:

1. Dans l'UCP, cliquer dans le menu gauche sur Swarm > Services
2. Puis, cliquer sur le bouton Create
3. Configurer l'onglet Details du service

Choisir un nom : Jenkins

Choisir une image : jenkins:latest

4. Configurer l'onglet Network du service

Cliquer sur Add Port +

Target port : 8080

Published port : 8080

Cliquer sur Confirm

5. Créer le service en cliquant sur Create en bas à droite

6. Une fois que le service est créé et que tous les voyants sont au vert, cliquer sur le service et récupérer son IP dans le paragraphe:

Endpoints

<http://a.b.c.d:8080>

7. On peut alors accéder à ce service et l'explorer dans UCP, le reconfigurer, le scaler à la volée, etc

Pour télécharger le client bundle, il faut:

1. Dans l'UCP, cliquer dans le menu gauche sur admin > My profile
2. Cliquer sur New Client Bundle > Generate Client Bundle
3. Un fichier zip est alors téléchargé, il faut le dézipper puis sourcer le fichier env.sh

```
```shell
# unzip du client bundle
unzip ucp-bundle-admin.zip
# chargement de l'environnement Docker EE
source env.sh
# interrogation du démon docker déployé via Docker EE
docker ps
```
```

---

## Épisode 1 : vue d'ensemble

### Container :

Dans Docker Enterprise, il est bien de revenir sur la définition du container qui n'est pas une grosse brique logicielle difficile à manipuler (image des containers sur les cargos) mais plutôt un paquet portable, léger et flexible. En anglais : *a lightweight portable package that isolates software component*

### Les différentes versions de Docker :

- moby project : framework opensource pour les contributeurs
- docker desktop & engine : outil gratuit de distribution de containers pour les développeurs.
- docker enterprise : plateforme d'exécution et de production de containers pour les entreprises.

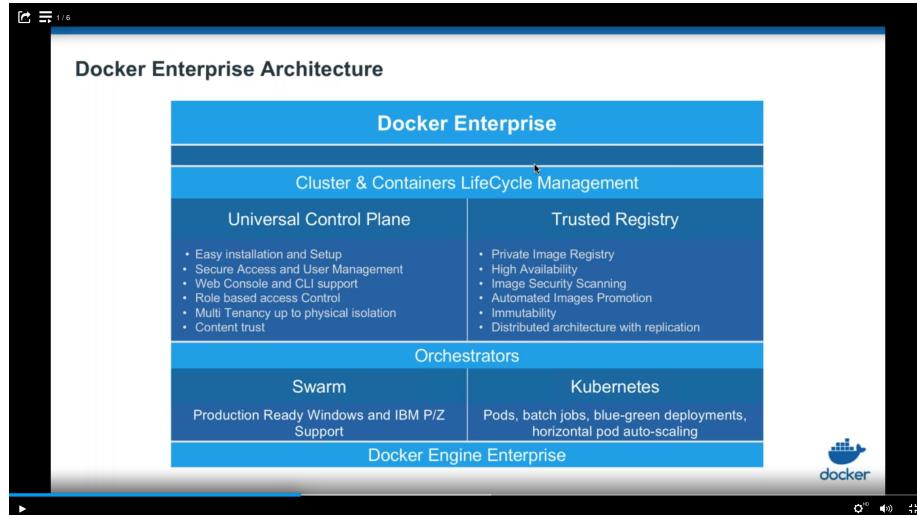
### Fonctionnalités spécifiques :

Docker Enterprise est une plateforme adaptée à la production et au déploiement de containers en clusters, pour cela des fonctionnalités spécifiques au domaine des opérations ont été ajoutées: - **sécurité** : contrôle de la qualité, contrôle des vulnérabilités des images, encryptions, certificats, 3rd parties contrôlées - **gouvernance** : règles et droits d'accès et d'administration, responsabilités limitées sur les clusters de production, ... - **automatisation** : orchestration, intégration avec le reste de la production, automatisation des logs, ... - **support et certification** : support d'entreprise dédié, plug-ins et infrastructures certifiées

### Les 4 piliers de mise en production de Docker Enterprise :

- **gouvernance** : bonnes pratiques d'organisation autour de la mise en production, formation des développeurs, des opérateurs aux outils de conteneurisation
- **plateforme** : architecture des applications conteneurisées
- **pipeline** : méthodologie DevOps, optimisation des chaînes d'intégrations et de déploiements continus
- **applications**

## Docker Enterprise architecture



2 outils complémentaires pour gérer la plateforme et les orchestrateurs:

- **Universal Control Plane (UCP)**: manager de clusters docker qui s'exécute sur Swarm ou sur Kubernetes en apportant d'autres fonctionnalités que n'ont pas ces outils opensource :

  - une installation facilitée
  - une console d'administration
  - de la sécurité
  - du contrôle d'accès basé sur des rôles
  - l'isolation physique des applications
  - l'intégration avec des annuaires LDAP/ActiveDirectory
  - l'intégration avec le Data Center : logs, utilisateurs, etc ...

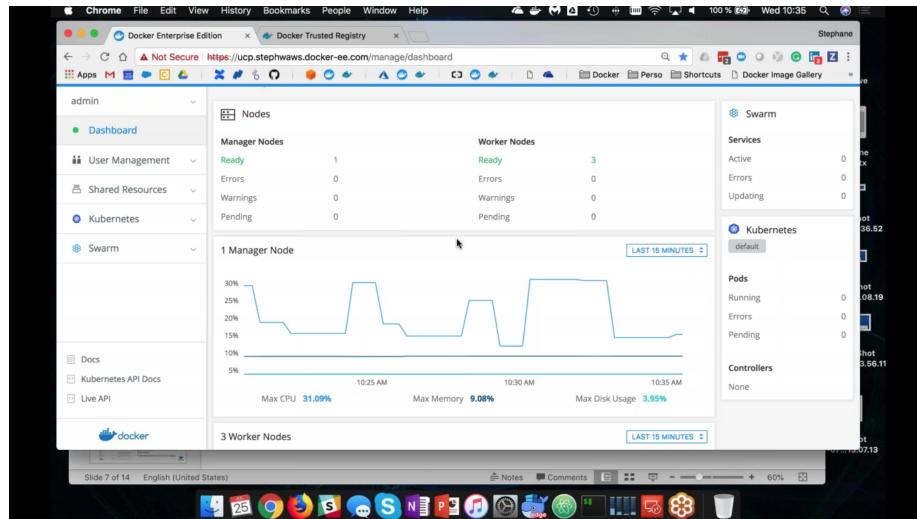


Fig. 2: Screenshot Dashboard

- **Docker Trusted Registry (DTR)** : cycle de vie des images Docker, DockerHub privatif sécurisé
  - interface graphique d'administration
  - redondance des images
  - sécurité des images : scanner de vulnérabilité, protection des images de production
  - règles de promotion des images : nombre de vulnérabilités critiques
  - usage du repository:  
`docker login dtr.xxx.docker-ee.com`  
`docker tag nginx dtr.xxx.docker-ee.com/dev/portal:dev`  
`docker push dtr.xxx.docker-ee.com/dev/portal:dev`

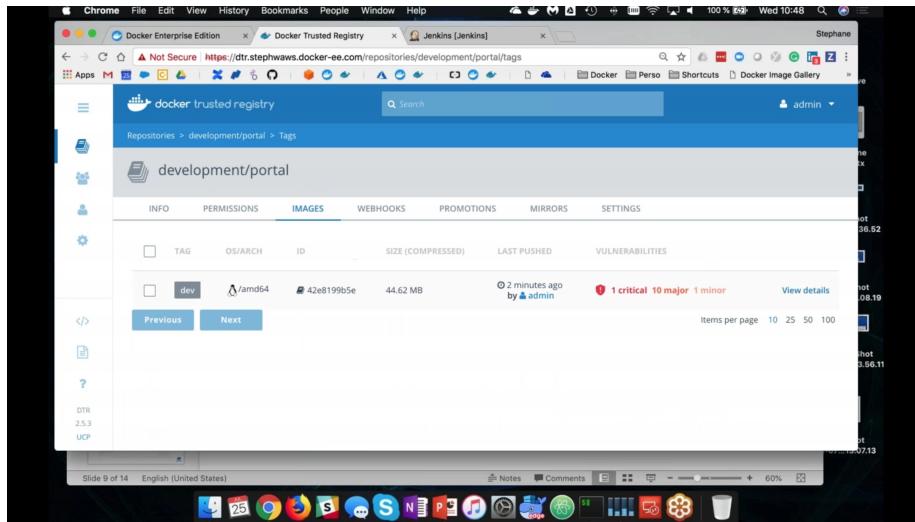


Fig. 3: Screenshot WebDTR

### Sécurité de la chaîne de déploiement

3 catégories de sécurisation : - secure platform : certificats électroniques, encryption - secure content : qualité des images (scanner de vulnérabilité, signature des images) - secure access : authentification et sécurisation de la connexion des administrateurs

### Épisode 3 : La sécurité

Docker Enterprise offre une sécurité intégrée des conteneurs à chaque étape du cycle de vie des applications, sans impact sur les performances et sans coûts additionnels.

## Principaux composants de sécurité de Docker Enterprise

### Secure platform:

- sécurité de l'engine Docker : sécuriser les containers, les isoler les uns des autres
- identité cryptographique validée par un certificat, toutes les communications entre les éléments Swarm ou Kubernetes sont cryptées en TLS
- propre identité de certification de Docker mais possibilité d'utiliser les siens
- rotation automatique des certificats
- il est prouvé que les containers sont une technologie sécurisée : isolation avec NameSpaces, limitation des resources avec CGroups, administration des droits d'accès avec LibCap, protection du kernel avec AppArmor, SELinux ou Seccomp, etc
- les images sont en read-only donc ne peuvent pas être compromises, seul le container est writable donc la surface d'attaque est limitée
- **Docker Bench for Security** : outils d'audit de vulnérabilité du Docker Engine (disponible pour Docker CE également sans contrôle d'accès, en version mono utilisateur)

Utilisation :

```
docker run -it --net host --pid host --userns host --cap-add audit_control \
-e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
-v /var/lib:/var/lib \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /usr/lib/systemd:/usr/lib/systemd \
-v /etc:/etc --label docker_bench_security \
docker/docker-bench-security
...
Status: Downloaded newer image for docker/docker-bench-security:latest
# -----
# Docker Bench for Security v1.3.4
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers
# in production.
# Inspired by the CIS Docker Community Edition Benchmark v1.1.0.
# -----
```

Initializing Wed Feb 13 09:32:12 UTC 2019

```
[INFO] 1 - Host Configuration
[WARN] 1.1 - Ensure a separate partition for containers has been created
[NOTE] 1.2 - Ensure the container host has been Hardened
[INFO] 1.3 - Ensure Docker is up to date
[INFO]     * Using 18.06.1, verify is it up to date as deemed necessary
[INFO]     * Your operating system vendor may provide support and security
      maintenance for Docker
[INFO] 1.4 - Ensure only trusted users are allowed to control Docker daemon
...
[INFO] 2 - Docker daemon configuration
...
[INFO] 3 - Docker daemon configuration files
...
[INFO] 4 - Container Images and Build File
[INFO] 4.1 - Ensure a user for the container has been created
[INFO]     * No containers running
[NOTE] 4.2 - Ensure that containers use trusted base images
[NOTE] 4.3 - Ensure unnecessary packages are not installed in the container
[NOTE] 4.4 - Ensure images are scanned and rebuilt to include security patches
[WARN] 4.5 - Ensure Content trust for Docker is Enabled
[WARN] 4.6 - Ensure HEALTHCHECK instructions have been added to the container image
[WARN]     * No Healthcheck found: [maven-repository:latest]
[WARN]     * No Healthcheck found: [maven-repository:latest]
[WARN]     * No Healthcheck found: [maven-repository:latest]
[PASS] 4.7 - Ensure update instructions are not use alone in the Dockerfile
[NOTE] 4.8 - Ensure setuid and setgid permissions are removed in the images
[INFO] 4.9 - Ensure COPY is used instead of ADD in Dockerfile
[INFO]     * ADD in image history: [docker/docker-bench-security:latest]
[INFO]     * ADD in image history: [maven-repository:v_1 username/maven:11]
[INFO]     * ADD in image history: [maven-repository:v_1 username/maven:11]
[INFO]     * ADD in image history: [maven-repository:v_1 username/maven:11]
[NOTE] 4.10 - Ensure secrets are not stored in Dockerfiles
[NOTE] 4.11 - Ensure verified packages are only Installed

[INFO] 5 - Container Runtime
[INFO]     * No containers running, skipping Section 5

[INFO] 6 - Docker Security Operations
[INFO] 6.1 - Avoid image sprawl
[INFO]     * There are currently: 3 images
[INFO] 6.2 - Avoid container sprawl
[INFO]     * There are currently a total of 3 containers,
      with 1 of them currently running
```

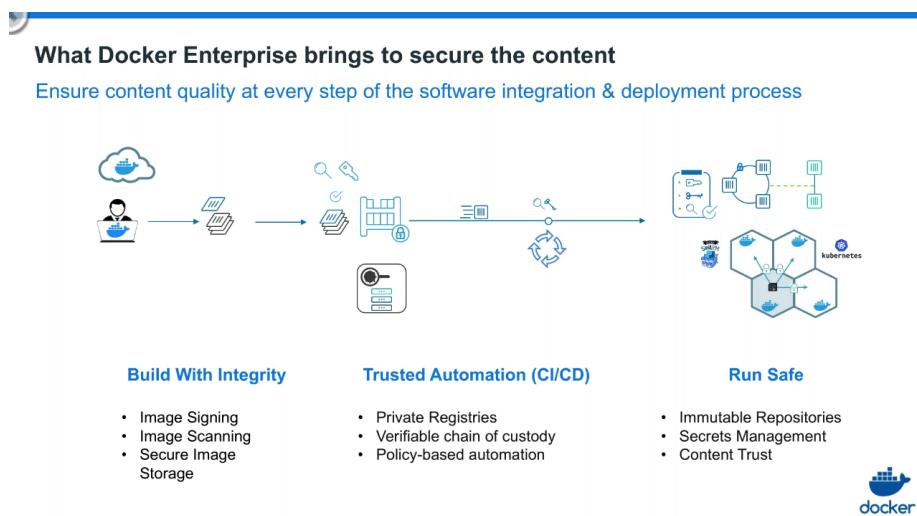
[INFO] 7 - Docker Swarm Configuration

...

[INFO] Checks: 74

[INFO] Score: 8

## Secure content



En résumé: - repositories immutables dans DTR: read-only avec règles de promotion des images de développement - scan de vulnérabilité des images avant exécution, après le push sur le repository - politique de promotion d'images selon des règles (par ex. 0 vulnérabilité critique) - les vulnérabilités des images sont décrites par couches pour permettre aux développeurs de les corriger - détection des vulnérabilités par hash des binaires présents dans les couches des images (DTR met à jour sa base de signature régulièrement ou manuellement)

## Secure operations

Mécanisme : - authentification des administrateurs - intégration with AD/LDAP : Admin Settings > Authentication & Authorization - contrôle d'accès avancé aux clusters : isolation de noeuds (ex: isolation d'une application sur des noeuds donnés), à la fois en Swarm et en Kubernetes - contrôle d'accès basé sur des rôles (RBAC): - rôles : ensemble de droit d'accès - collection : ensemble de ressources sur lequel on peut donner des droits d'accès, hiérarchisée avec des sous collections - sujets : individus ou des équipes d'administration - grant : association d'un rôle à des sujets

Exemple pour la création d'un service et des ses droits d'accès : - choix de l'image et configuration du service - association à une collection - définition des droits d'accès en tant qu'administrateur docker : - création de rôles: par exemple: start-restart-role, avec des droits d'accès bien particuliers sur des actions sur les containers (logs, start, stop, view), services (log, view) - création du grant : on affecte à un sujet donné un role avec des droits donnés sur une collection donnée, par exemple : l'utilisateur operator n'aura le droit que de redémarrer un service faisant partie de la collection portail-de-production

---

## Épisode 4 : L'agilité

La plateforme Docker Enterprise offre une totale agilité opérationnelle, qui permet l'accélération des déploiements d'applications et de leur mise à jour dans un cycle continu, celle de l'interopérabilité et enfin celle de l'automatisation. Docker effectue la transition vers DevOps en connectant développeurs et opérations de façon concrète.

### Les atouts du container dans l'IT

- **Cloud Computing** : cloud migration public/hybrid/privé, réversibilité
- **Modernisation des applications** : Docker EE réduit le time to market,
- **DevOps** : le container simplifie la mise en production, Docker EE optimise - l'intégration continuer
- **Réduction des coûts** : moins de machines virtuelles pour des services et des fonctionnalités identiques, le nombre de cores nécessaires restent approximativement le même mais l'utilisation des CPU est moindre de par le fonctionnement en containers
- **Multiple possibilités de déploiement** : Docker EE permet différents scénarios :
  - sur différents continents : déploiement offshore
  - mirroring de registries
  - caching sécurisé des images de développement en local
- **UCP**: un outil unique pour gérer tous ses clusters
- **Outils de migration** des applications d'un environnement monolithique vers une architecture de containers :
  - une version opensource : non disponible pour test sans passer par les sales Docker :(
  - une version propriétaire **Docker Application Converter**: DAC est une application qui va détecter les applications d'une machine monolithique virtuelle ou baremetal pour les "Dockériser"
- **Paradigme de l'universalité du packaging Docker** : la complexité des chaînes de déploiement logicielles est grandement simplifiée grâce aux containers, Build&Ship anywhere

- en cours de développement chez Docker : **Docker Application Designer** vise à apporter plus de facilité dans l'usage des containers : créer des containers sans écrire une seule ligne de code de manière totalement transparente sans besoin de formation à Docker
    - exemple de création d'une application .net avec 2 containers : ASP .net et mssql sans aucune ligne de code. L'application est prête à l'emploi pour les développeurs
    - exemple de création d'une application Ruby on Rails avec 2 containers : rails et postgres
- 

## Épisode 5 : Intégration de Kubernetes

L'orchestrateur Kubernetes est maintenant intégré à la plateforme Docker Enterprise. Cela signifie que les architectures Kubernetes bénéficient des différents modules de Docker Enterprise notamment les modules de sécurité. Cela permet notamment de répondre au mieux aux enjeux de l'IA et du serverless qui représentent actuellement deux des tendances majeures du cloud (tant public que privé).

### Swarm & Kubernetes

Swarm et Kubernetes sont deux orchestrateurs qui fonctionnent en parallèle par défaut dans Docker EE : l'installation de Kubernetes est automatique.

- Les nœuds schedulers sont soit Swarm soit Kubernetes
- Les nœuds managers sont à la fois Swarm et Kubernetes
- Les nœuds workers sont soit Swarm, soit Kubernetes, soit mixte (non recommandé en prod)

Swarm : développé par Docker pour aider les devs à se servir des images disponibles sur le Hub, permet un assemblage rapide et simple d'applications

Kubernetes : développé par Google pour des déploiements complexes, des applications plus lourdes, intégré dans Docker EE ou Docker desktop sur Mac/Windows (pas sur Linux :( )

### Gestion simplifiée du cluster Kubernetes grâce à Swarm :

La commande de création à exécuter avec des tokens d'identification est :

```
docker swarm join [OPTIONS] HOST:PORT
```

## Réseau

Dans Docker EE la pile réseau CNI Calico est installée automatiquement

## Contrôle d'accès

Les rôles Kubernetes sont disponibles dans Docker EE, mais Docker EE va au delà notamment au niveau de l'isolation physique au niveau des namespaces Kubernetes Rôles prédéfinis et possibilité de créer de nouveaux rôles custom avec des règles d'opérations spécifiques aux containers, aux services, à Kubernetes et à Swarm Namespaces Kubernetes et collections Docker EE peuvent être liés et être affectés à des noeuds particuliers.

## Console web et ligne de commande

Docker EE utilise les API et la CLI Kubernetes standard On peut déployer des fichiers Docker Compose en ressources Kubernetes, à la place des YAML: le langage des Docker compose est plus simple que le YAML Kubernetes, la migration de Swarm vers Kubernetes est simplifiée

Sur la console Web, à la création du service on choisi le docker compose voulu, et on choisi le déploiement de type Kubernetes

En ligne de commande, on utilise le client bundle puis la commande `kubectl` :

```
source env.sh
kubectl get nodes
kubectl get all
# déploiement de pods & services
kubectl apply -f conf.yml
# suppression des ressources
kubectl delete
```

---

## Épisode 6: Méthodologie de déploiement de Docker Enterprise en production

Docker est très largement utilisé en développement et de plus en plus déployé en production dans les plus grandes entreprises dans le monde. Cela permet ainsi de faciliter et sécuriser les déploiements tout en permettant la montée en charge de vos applications.

Etapes dans l'adoption des containers:

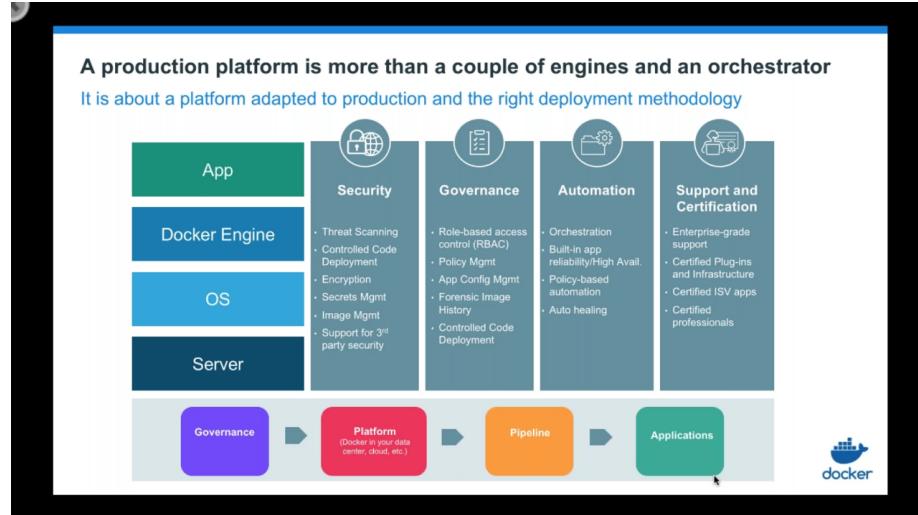


Fig. 4: Diagramme

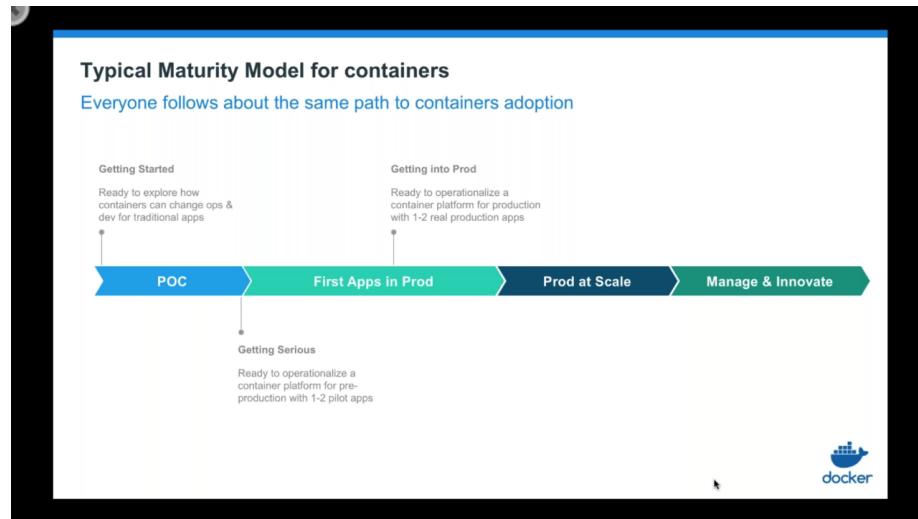
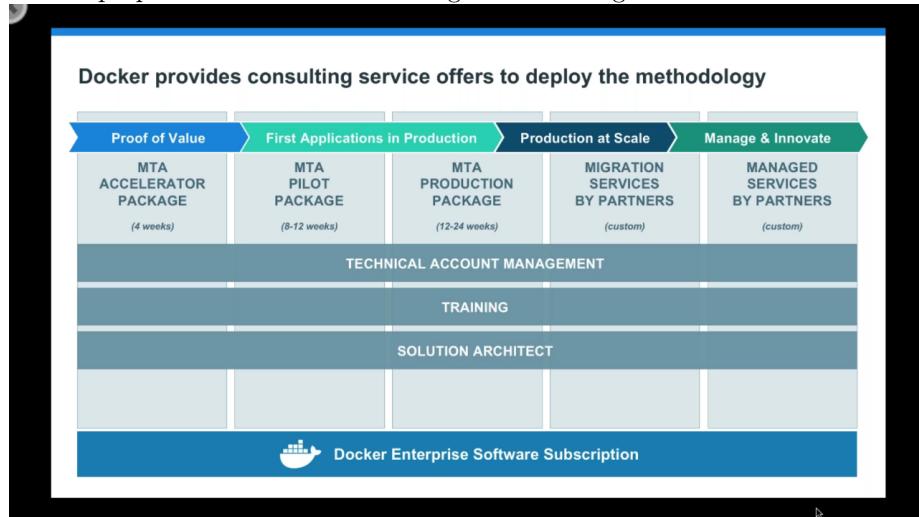


Fig. 5: Etapes

Méthodologie : - PoC - Assesment / Analyse de l'existant : évaluer les compétences des équipes de déploiement existantes, trier les outils et choisir une application pilote qui va servir de prototype. - puis 4 workstreams en parallèle pour pouvoir montrer des résultats rapides et réguliers : - gouvernance: en particulier du marketing interne, de la promotion de la technologie des containers, - construction de la plateforme : architecture adaptée au besoin - dev des applications à conteneuriser - pipeline automatisée d'intégration continue et de DevOps - GoLive - Prod

Les taches à réaliser doivent être lister par étapes avec des liens vers des guides techniques, des référence d'architectures, des outils, des runbooks, des templates, etc

Docker propose des offres de consulting et de training avec différentes formules :



Exemples d'utilisateurs de Docker EE en Europe : - Société Générale, déploiement de SaaS conteneurisés - Intesa, déploiement applicatif intelligent sur de multiples DataCenter pour garantir une haute disponibilité - Bosh, multiples plateformes pour lesquelles les images se répliquent depuis le centre IT (mirroring de DTR) - Finnish Rails, déploiement d'applications internet mobiles et interne sur le Cloud avec de la gestion de montée en charge