

## Introduction à Docker

<b>1. Docker comme alternative légère à la virtualisation</b>	<b>13</b>
1.1 L'approche par virtualisation	14
1.2 Docker pour optimiser l'usage des ressources	15
1.3 Positionnement de Docker par rapport à la virtualisation	18
<b>2. Principe des conteneurs</b>	<b>21</b>
2.1 Principe des conteneurs industriels	22
2.2 Docker et l'approche normalisée	23
<b>3. Les fondements de Docker</b>	<b>24</b>
3.1 Namespaces	24
3.2 Cgroups	26
3.3 LXC	26
3.4 Libcontainer	27
3.5 Système de fichiers en couches	29
3.5.1 Principe d'isolation des fichiers	29
3.5.2 Approche par virtualisation	29
3.5.3 Utilité des systèmes de fichiers en couches	30
3.5.4 Gestion des modifications de fichiers	32

3.5.5 Dernière couche en écriture	34
3.5.6 Technologies utilisées	35
<b>4. Les plus de Docker</b>	<b>36</b>
<b>5. L'écosystème Docker</b>	<b>37</b>
<b>6. Architectures de services</b>	<b>38</b>
6.1 Historique des architectures de services	39
6.1.1 Principes	39
6.1.2 Approche EAI	39
6.1.3 Approche SOA	40
6.1.4 Microservices	40
6.1.5 Lien à l'urbanisation des SI	41
6.2 Architecture de microservices	41
6.2.1 Principe	41
6.2.2 Avantages	43
6.2.3 Inconvénients	46
6.3 Apport de Docker	47
6.4 Fil conducteur	48

Premiers pas

<b>1. Installation de Docker</b>	<b>49</b>
1.1 Utiliser des machines préconfigurées	50
1.2 Installation de Docker sur Linux	62
1.2.1 Prérequis système	62
1.2.2 Installation par gestionnaire de paquets	62
1.2.3 Installation par script	67
1.3 Installation de Docker sous Windows	68
1.3.1 Un paradoxe résolu	68
1.3.2 Installation sur Windows 10	70
1.3.3 Installation sur Windows Server 2016	74
1.3.4 Machine virtuelle	76
<b>2. Hello World, Docker</b>	<b>76</b>
2.1 Démarrage d'un conteneur simple	76
2.2 Que s'est-il passé ?	78
2.2.1 Récupération de l'image	78
2.2.2 Anatomie de l'image obtenue	79
2.2.3 Lancement du processus	82
2.2.4 Arrêt du conteneur	84
<b>3. Utiliser des images Docker préexistantes</b>	<b>86</b>

3.1 Le Docker Store	86
3.1.1 Le principe	86
3.1.2 Recherche et qualification d'images	88
3.1.3 Exemple de recherche	90
3.1.4 Cas des images communautaires	92
3.1.5 Compléments sur les images officielles	94
3.1.6 Lien avec le registre Docker Hub	96
3.1.7 Recherche par la ligne de commande	97
3.1.8 Précautions sur une image non officielle	100
3.2 Gestion du compte DockerHub et dépôts privés	106
3.2.1 Création d'un compte	107
3.2.2 Caractéristiques du compte	108
3.2.3 Automated build et compte GitHub	110
3.2.4 Connexion au compte en ligne de commande	117
3.2.5 Webhook sur évènement de push dans Docker Hub	118
3.2.6 Déconnexion des comptes Docker Hub et Github	120
<b>4. Un second conteneur</b>	<b>122</b>
4.1 Récupération de l'image	122
4.2 Explication des tags	123
4.3 Premier lancement	125

4.4 Lancement en mode interactif	128
4.5 Persistance des modifications sous forme d'une image	130
4.6 Prise en main du client Docker	133
4.6.1 Ménage dans les conteneurs	133
4.6.2 Ménage dans les images	134
4.6.3 Le grand ménage	134
4.6.4 Suppression automatique à la sortie	135
4.6.5 Affectation d'un nom de conteneur	136
4.6.6 Modification du point d'entrée par défaut	137
4.6.7 Envoi de variables d'environnement	138
4.6.8 Modification du hostname	140
4.7 Manipulation des conteneurs	141
4.7.1 Lancement en mode bloquant	141
4.7.2 Lancement en arrière-plan	143
4.7.3 Gestion correcte du cycle de vie des conteneurs	148
4.7.4 Exposition de fichiers	151
4.7.5 Supervision des conteneurs	153
<b>5. Retours sur les premiers pas</b>	<b>155</b>

## Création de vos propres images

<b>1. Création manuelle d'une nouvelle image</b>	<b>157</b>
1.1 Installation d'un logiciel dans un conteneur	157
1.2 Persistance de l'image pour une utilisation future	160
1.3 Utilisation de l'image créée	161
1.4 Connexion depuis la machine hôte	163
1.5 Suite des opérations	165
<b>2. Utilisation d'un Dockerfile</b>	<b>166</b>
2.1 Intérêt des fichiers Dockerfile	166
2.2 Utilisation d'un fichier Dockerfile	168
2.3 Résultats de l'utilisation d'un Dockerfile complet	170
2.4 Anatomie d'un fichier Dockerfile	171
2.4.1 FROM	172
2.4.2 RUN	172
2.4.3 ENV	173
2.4.4 VOLUME	175
2.4.5 COPY	179
2.4.6 ENTRYPOINT	180
2.4.7 EXPOSE	182
2.4.8 CMD	182
2.5 Notre premier Dockerfile	182

2.5.1 Création et test du script	183
2.5.2 Création du Dockerfile	183
2.5.3 Génération de l'image	185
2.5.4 Lancement du conteneur	187
2.5.5 Arrêt et relance du conteneur	188
2.5.6 Gestion des paramètres	189
2.5.7 Reconstruction de l'image et cache	191
2.6 Commandes additionnelles	192
2.6.1 Gestion des fichiers	194
2.6.2 Notion de contexte	194
2.6.3 Retours sur l'affectation du processus à démarrer	195
2.6.4 Remarque sur le format ligne de commande ou exécution	196
2.6.5 Commandes diverses	199
<b>3. Partage et réutilisation simple des images</b>	<b>201</b>
3.1 Envoi sur votre compte Docker Hub	205
3.2 Export et import sous forme de fichiers	210
<b>4. Bonnes pratiques</b>	<b>211</b>
4.1 Principe du cache sur les images	211
4.2 Principe du cache à la compilation	216

4.2.1 Retour sur les images intermédiaires	216
4.2.2 Anatomie d'une compilation d'image	220
4.2.3 Analyse d'une modification du Dockerfile	222
4.2.4 Gestion correcte des étiquettes	225
4.2.5 Invalidation du cache par modification de l'image de base	229
4.2.6 Invalidation du cache par modification du contexte	234
4.3 Conséquences sur l'écriture des Dockerfile	236
4.3.1 Le problème sur les opérations non idempotentes	236
4.3.2 Contournement du problème de cache	239
4.3.3 Effets bénéfiques sur le nombre et la taille des images	243
4.3.4 Ordonnancement des commandes dans le Dockerfile	245
4.4 Conséquences sur le choix des images de base	246
4.4.1 La bonne image de base	246
4.4.2 Votre propre image de base	248
4.5 Arborescence recommandée	251
4.5.1 Avantages d'une arborescence type	251
4.5.2 Intégration des fichiers	252
4.5.3 Limitation du contexte	254
4.6 La question du processus unique	255
4.6.1 Principe général	255
4.6.2 Exception au principe général avec Supervisor	256



4.6.3 Critique	257
4.6.4 Approche intermédiaire	258
<b>5. Pour aller plus loin</b>	<b>260</b>
Installation d'un registre privé	
<b>1. Premiers pas pour un registre privé</b>	<b>261</b>
1.1 Remarque préliminaire importante	262
1.2 Avertissement sur l'ancien registre	262
1.3 Image Docker en local	263
1.4 Pointer sur un registre donné	265
1.5 Registre sur un réseau public	268
1.5.1 Scénario et préparation des machines	269
1.5.2 Démarrage du registre	270
1.5.3 Dépôt de l'image depuis une autre machine	273
1.5.4 Utilisation de l'image depuis une troisième machine	276
1.5.5 Suppression de l'image sur la machine source	277
<b>2. Un registre plus professionnel</b>	<b>279</b>
2.1 Gestion de la persistance	279
2.1.1 Gestion locale par volume	279

2.1.2 Remarque sur SELinux	279
2.1.3 Un stockage plus sécurisé	281
2.2 Sécurisation du registre	282
2.2.1 Préparation des clés pour le canal crypté	284
2.2.2 Mise en place du frontal Nginx	284
2.2.3 Accès par le client Docker	287
2.2.4 Outils additionnels de diagnostic	293
2.2.5 Génération du fichier des utilisateurs autorisés	295
2.2.6 Ajout de l'authentification dans Nginx	297
2.2.7 Correction sur les headers	298
2.2.8 Gestion de la connexion	300
2.3 Un vrai registre en production	301
<b>3. Utilisation d'un service de registre dans le cloud</b>	<b>302</b>
<b>3.1 Principe</b>	<b>304</b>
3.2 Offre payante de Docker Hub	304
3.3 Azure Container Registry	306
3.3.1 Préparation	309
3.3.2 Création du registre	309
3.3.3 Paramétrage	310
	313

<b>4. Approches complémentaires</b>	<b>315</b>
4.1 L'API du registre	315
4.2 Mise en place d'un miroir	315
4.3 Mise en place d'un cache pour les paquetages	319
4.3.1 Contournement possible	319
4.3.2 Mise en œuvre d'un cache de paquetages	321
 Mise en œuvre d'une architecture logicielle	
<b>1. Présentation de l'application exemple</b>	<b>325</b>
1.1 Architecture	325
1.2 Installation	327
1.3 Utilisation	330
1.4 Utilité	332
1.5 Principes à l'œuvre	333
1.5.1 Un mot sur les architectures de microservices	333
1.5.2 Lien avec la programmation SOLID	334
 <b>2. Création de l'architecture exemple</b>	 <b>335</b>
2.1 Principes de construction	335
2.2 Détails du service optimizer	336

2.2.1 Fonctionnement	336
2.2.2 Intégration dans Docker	341
2.2.3 Tests	344
2.2.4 Remarques	346
2.3 Détails du service calculator	347
2.3.1 Fonctionnement	347
2.3.2 Intégration dans Docker	349
2.3.3 Tests	351
2.4 Mise en place de liens entre conteneurs	354
2.5 Détails du service reporting	359
2.5.1 Fonctionnement	359
2.5.2 Dockerisation	361
2.5.3 Tests	362
2.6 Détails du service notifier	364
2.6.1 Fonctionnement	364
2.6.2 Dockerisation	367
2.6.3 Tests	368
2.7 Détails du service de persistance	372
2.8 Mise en œuvre éventuelle d'une image de base	373
2.9 Détails du service portal	376
2.9.1 Fonctionnement	376

2.9.2 Dockerisation	379
2.9.3 Tests	380
2.10 État atteint	381
<b>3. Redéployer automatiquement avec Docker Compose</b>	<b>382</b>
3.1 Principe de Docker Compose	382
3.2 Écriture du fichier docker-compose.yml	383
3.3 Mise en œuvre	386
3.3.1 Préparation	386
3.3.2 Lancement des conteneurs	387
3.3.3 Gestion des conteneurs	389
3.4 Parallélisation des traitements	390
3.5 Limites	392
<b>4. Exploitation d'une infrastructure Docker</b>	<b>393</b>
4.1 Le réseau dans Docker	393
4.1.1 Mode de fonctionnement standard (bridge)	393
4.1.2 Modes de fonctionnement alternatifs	396
4.1.3 Support des liens entre conteneurs	399
4.1.4 Autres options	401
4.1.5 Limites de la couche réseau existante	401
4.2 Les volumes Docker	

4.2.1 Le problème de la persistance	402
4.2.2 Les volumes comme solution simple	402
4.2.3 Lien direct sur un répertoire local	403
4.2.4 Partage de volumes	405
4.2.5 Gestion des volumes orphelins	406
4.2.6 Sophistication de l'approche	407
4.2.7 Application à la gestion des logs	408
4.2.8 Plus loin avec les volumes	409
	410
<b>Déploiement dans un cluster</b>	
<b>1. Description globale de l'approche</b>	
	<b>411</b>
1.1 Objectif	411
1.2 État des lieux	412
1.3 Déroulement de l'exemple	413
1.4 Avertissement	413
<b>2. Montage d'un cluster Swarm</b>	
	<b>414</b>
2.1 Fonctionnement résumé de Swarm	414
2.2 Description des machines utilisées	415
2.3 Initialisation du cluster	419

2.4	Attachement des autres machines	419
2.5	Prise en main à distance	420
<b>3.</b>	<b>Déploiement sur le cluster Swarm</b>	<b>421</b>
3.1	Principes des services	421
3.2	Envoi des images sur le registre	423
3.3	Lancement d'une stack	425
3.4	Passage à l'échelle	429
<b>4.</b>	<b>Quelques informations supplémentaires sur Swarm</b>	<b>432</b>
4.1	Ce qui s'est passé sous le capot	432
4.1.1	Commandes de diagnostic	432
4.1.2	Gestion du réseau overlay et des ports	435
4.2	Pour aller plus loin	436
4.2.1	Arrêt de la plateforme	436
4.2.2	Branchement sur un registre privé	437
4.2.3	Gestion des contraintes	438
4.2.4	Déploiement incrémental (rolling update)	439

Encore plus loin avec Docker

## 1. Docker dans votre usine logicielle

	<b>443</b>
1.1 Docker à tous les étages	443
1.2 Produire des images en sortie de build	445
1.2.1 Positionnement	445
1.2.2 Utilité	445
1.2.3 Conseils de mise en place	446
1.3 Utiliser des conteneurs pour l'usine logicielle	447
1.3.1 Positionnement	447
1.3.2 Utilité	447
1.3.3 Conseils de mise en place	448
1.4 Utiliser des conteneurs pour les tests	449
1.4.1 Positionnement	449
1.4.2 Utilité	449
1.4.3 Conseils de mise en place	450
1.4.4 Variante associée	450
1.5 Retour sur le registre	452
<b>2. Avant de partir en production avec Docker</b>	<b>453</b>
2.1 Sécurité	453
2.2 Restriction sur les ressources	454
<b>3. Docker et Windows</b>	<b>455</b>



<b>4. Conclusion</b>	<b>455</b>
----------------------	------------

<b>Index</b>	<b>459</b>
--------------	------------