# Troubleshooting et notes à propos du cluster local



## 1 Registre d'images local

Un oubli est survenu lors de la rédaction de l'énoncé du TP. Cela affecte principalement les accès au registre docker local qui devrait vous être exposé sur l'adresse `localregistry.lc:30500`.

Si vous n'avez pas configuré votre résolution DNS (ce qui aurait dû, mais ne se trouve pas dans l'énoncé du TP) vous avez surement rencontré une erreur de connexion. De plus, si vous essayez de push des images à partir de votre hôte vers le registre en *http* vous avez surement aussi eu un problème de certificat.

### 1.1 Fixer la résolution DNS

Pour cela, c'est assez simple, il vous suffit, tel que montré dans les laboratoires, de mettre à jours le fichier `/etc/hosts` de votre machine. Il vous faudra rajouter l'entrée `10.0.0.30 localregistry.lc`.

Votre configuration devrait ressembler à ça:

Listing 1: /etc/hosts

```
1  ##
2  # Host Database
3  #
4  # localhost is used to configure the loopback interface
5  # when the system is booting.  Do not change this entry.
6  ##
7  127.0.0.1   localhost
8  255.255.255.255 broadcasthost
9  ::1             localhost
10 # Added by Docker Desktop
11 # To allow the same kube context to work on the host and the container↩
       :
12 127.0.0.1 kubernetes.docker.internal
13 # End of section
14
15 10.0.0.30 localregistry.lc
```

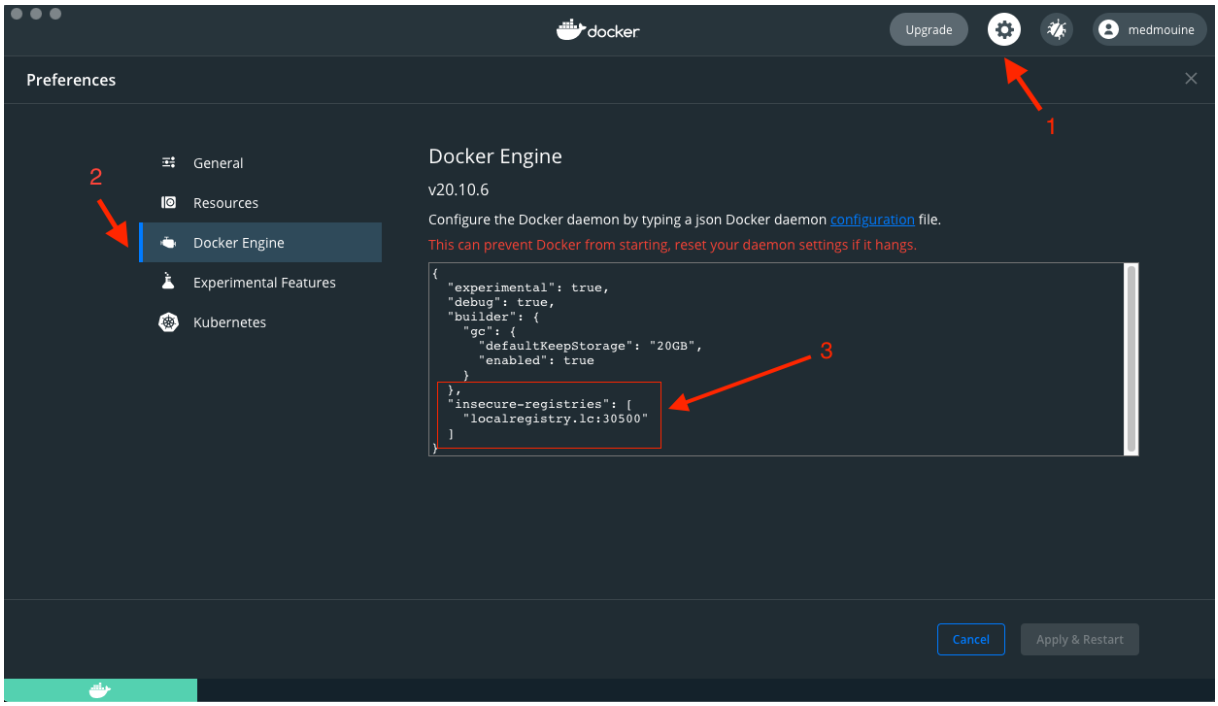
Notez que la dernière ligne est la seule qui importe pour ce fixe. De plus, il est important de noter que cette configuration est protégée et que vous aurez donc besoin des accès `root` de votre

machine pour la modifier. Vous pouvez configurer le tout rapidement grâce aux commandes suivantes:

```
$ sudo -i
<ENTREZ VOTRE MOT DE PASSE ROOT>
$ echo "10.0.0.30 localregistry.lc" >> /etc/hosts
```

## 1.2 Fixer les erreurs de certificats/HTTPS de docker

Pour fixer ce problème, vous pouvez passer directement pas l'interface de docker si vous avez celle-ci. Il vous suffit de rajouter l'entrée suivante dans la configuration du daemon pour que Docker accepte une connexion HTTP non cryptée avec le registre.




Si vous n'avez pas accès à l'interface de Docker, cette configuration se trouve normalement dans le fichier `~/.docker/daemon.json`. Vous pouvez donc modifier ce fichier en rajoutant l'entrée `"insecure-registries" : ["localregistry.lc:30500"]`.

Votre configuration devrait ressembler à ça:

Listing 2: /etc/hosts

```
{
    "experimental": true,
    "debug": true,
    "builder": {
        "gc": {
            "defaultKeepStorage": "20GB",
            "enabled": true
        }
    },
    "insecure-registries": [
        "localregistry.lc:30500"
    ]
}
```

## 2 Erreur de communication lors de la configuration des Nodes

Si vous workers plantent lors de l'initialisation avec Vagrant voici une liste des solutions possible à explorer et des outils de troubleshooting à votre disposition:

### 2.1 Kubenode1: Token not found

Si vos Nodes vous retournent une erreur mentionnant l'absence d'un fichier token quelconque lors de l'initialisation, cela est surement dû à un mal-fonctionnement de votre LoadBalancer Traefik.

Un fixe rapide et simple pour ce problème est de remplacer toutes les références à `VIP` dans le script `provision.sh` par l'adresse d'un de vos MasterNodes. Par exemple MasterNode1, adresse = `10.0.0.11` par défaut. Le plus important est la ligne 43 `--server-ip $vip` qui devra être changée pour pointer vers `10.0.0.11` donc

```
....
38    k3sup join --print-command \
39        --ip $ip \
40        --k3s-version $version \
41        --ssh-key .ssh/id_rsa \
42        --user root \
43        --server-ip 10.0.0.11 \   ############
44        --server-user root \
....
```

## 2.2 Méthodes de troubleshooting

Si vous rencontrez quelconque autre problème lors de l'initialisation du cluster, vous pouvez vous connecter sur les instances directement par SSH au travers de Vagrant en utilisant la commande `vagrant ssh NOM_INSTANCE` par exemple `vagrant ssh kubemaster1`.

Vous pouvez ensuite consulter les logs de K3s qui se trouvent dans le fichier `/var/log/k3s.log` donc en utilisant la commande `sudo cat /var/log/k3s.log`.

Notez que le répertoire `var` est protégé d'où le `sudo`.

Les instances roulant Alpine Linux, vous pouvez gérer vos services au travers des commandes `rc-status` et `rc-service`. Par exemple, pour redémarrer le service de K3s, vous utiliserez la commande `rc-service k3s restart`.

Pour consulter les logs de vos containers, vous pouvez utiliser la commande `kubectl logs <NOM_DU_POD>` cela retournera les logs du container *"principal"* du Pod en question. Vous pouvez aussi rajouter le nom du container à la fin de cette même commande pour spécifier quels logs vous souhaitez afficher.

Vous pouvez vous connecter directement à vos containers en utilisant la commande `kubectl -it exec <NON_DU_POD> -- sh`. Vous pourrez ensuite tester vos applications directement à partir du container les contenant. Il est important de noter que la majorité des images utilisées ne contiennent quasiment aucune application, cela dans le but de les alléger. Si vous souhaitez exécuter des requêtes HTTP, vous pouvez installer `curl` sur Alpine grâce à la commande `apk add curl` ou toute autre gestionnaire de package différent dépendamment de la distribution utilisée ( `apt` pour Ubuntu).

## 2.3 Aide et ressources

Pour finir, je vous rappelle une fois de plus l'existence d'un forum Stackoverflow privé et dédié au cours. Nous vous invitons à poser vos questions sur celui-ci pour avoir une réponse le plus rapidement possible.

Vous pouvez aussi vous joindre à la séance de laboratoire le vendredi 5pm pour avoir de l'aide directement de la part de l'auxiliaire.