***Node KASPA sur pc dédié***

1. **Installation Ubuntu 20.04 LTS**

// Installer Ubuntu 20.04 LTS sur le pc à l'aide d'une clef bootable monté grâce à Rufus

* Rufus :

<https://github.com/pbatard/rufus/releases/download/v4.2/rufus-4.2.exe>

* Image Ubuntu 20.04 LTS :

<https://releases.ubuntu.com/focal/ubuntu-20.04.6-desktop-amd64.iso>

* Installer Ubuntu :

<https://www.youtube.com/watch?v=POf5mCs5YgI&ab_channel=TechWhite>

// Laisser Ubuntu tout en anglais, nous allons mettre le clavier en azerty par la suite

* Mettre le clavier AZERTY :

<https://www.clubic.com/tutoriels/article-862595-1-comment-clavier-qwerty-azerty-ubuntu.html>

// Une fois sur le Bureau il faut définir un IP fixe en iPv4 et désactivé iPv6

* Cliquer sur l’onglet Réseaux en haut à droite
* Aller dans « Wired Connected »
* Puis « Wired Setting »
* Dans la partie « Wired » cliquer sur l’engrenage
* Onglet « IPv4 » :
  + Sélectionner « Manual »
  + Onglet « Adress » par exemple : 192.168.1.100
  + Onglet « Netmask » : 255.255.255.0
  + Onglet « Gateway » (adresse de la box internet) : 192.168.1.1 pour Orange par ex.
  + Onglet « DNS » : 1.1.1.1
* Onglet « IPv6 »
  + Sélectionner « Disable »
* Appuyer sur « Apply » en haut à droite

// Il nous reste à mettre à jour et installer SSH pour avoir accès à distance pour la suite

* Ouvrir un Terminal et taper les commandes :
  + sudo apt-get update
  + sudo apt-get upgrade -y
  + sudo apt-get install ssh

1. **Installation Nœud KASPA**

**// Sur votre pc de bureau avec Windows nous allons faire la suite grâce au Shell**

* **Ouvrir l’invité de commande dans Windows**
* **Taper : ssh 192.168.1.100 -l nomutilisateur**
* **Puis écrire « yes » et taper sur entrer, voilà vous être connecté à distance à votre Ubuntu**

**// Mettre tout à jour avant de commencer : ligne par ligne**

**sudo apt-get update**

**sudo apt-get upgrade -y**

**// Exécutez la commande suivante pour désinstaller tous les packages en conflit pour Docker :**

**for pkg in docker.io docker-doc docker-compose podman-docker containerd runc; do sudo apt-get remove $pkg; done**

**// Mettez à jour l’Apt index des packages et installer les packages pour permettre Apt l'utilisation d'un référentiel via HTTPS : ligne par ligne**

**sudo apt-get update**

**sudo apt-get install ca-certificates curl gnupg**

**// Ajoutez la clé GPG officielle de Docker : ligne par ligne**

**sudo install -m 0755 -d /etc/apt/keyrings**

**curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg**

**sudo chmod a+r /etc/apt/keyrings/docker.gpg**

**// Utilisez la commande suivante pour configurer le référentiel tout copier en une fois :**

**echo \**

**"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \**

**"$(. /etc/os-release && echo "$VERSION\_CODENAME")" stable" | \**

**sudo tee /etc/apt/sources.list.d/docker.list > /dev/null**

**// Mettez à jour l’Apt index du package :**

**sudo apt-get update**

**// Pour installer la dernière version, exécutez :**

**sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin**

**// installation de go et le dépôt git ligne par ligne**

**sudo apt remove --autoremove golang**

**sudo rm -rf /usr/local/go**

**sudo apt install git**

**sudo apt install snapd**

**sudo snap install go --classic**

**go version**

**// Nous pouvons maintenant configurer notre nœud Kaspa sur cette machine. Utilisez simplement cette commande pour faire tourner le conteneur Docker :**

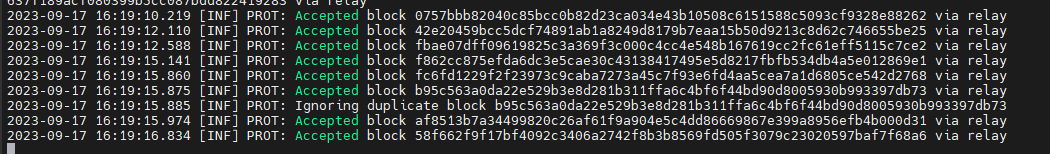
**sudo docker run --pull always -d --restart unless-stopped -p 16110:16110 -p 16111:16111 --name kaspad supertypo/kaspad:latest**

**// Si vous souhaitez vérifier l'état actuel de ce nœud, utilisez cette commande :**

**sudo docker logs -n 100 -f kaspad**

**// Si vous souhaitez quitter ce journal, utilisez simplement CTRL + c**

**// Cela vous montrera quelle quantité de blocs actuels a été traitée, par exemple téléchargée. S’il atteint 100 %, nous pouvons passer à l’étape suivante. Cela peut prendre deux heures si connexion bas débit, des « via relay » seront dans les lignes a la fin de la synchronisation**

****

1. **Installation Bridge pour SOLO mine sur son Node**

**// Après cela, nous pouvons télécharger le dépôt Git pour le Bridge : ligne par ligne**

**curl -OL https://github.com/rdugan/kaspa-stratum-bridge/archive/refs/tags/v1.2.1.tar.gz**

**tar -xvf** **v1.2.1.tar.gz**

**rm -r v1.2.1.tar.gz**

**// Nous devons maintenant accéder au répertoire de ce dépôt téléchargé en utilisant : ligne par ligne**

**cd kaspa-stratum-bridge-1.2.1/**

**cd cmd/kaspabridge/**

**// modifié le document text config.yaml**

**nano config.yaml**

**Utiliser les flèches du clavier pour se déplacer dans l’éditeur de texte**

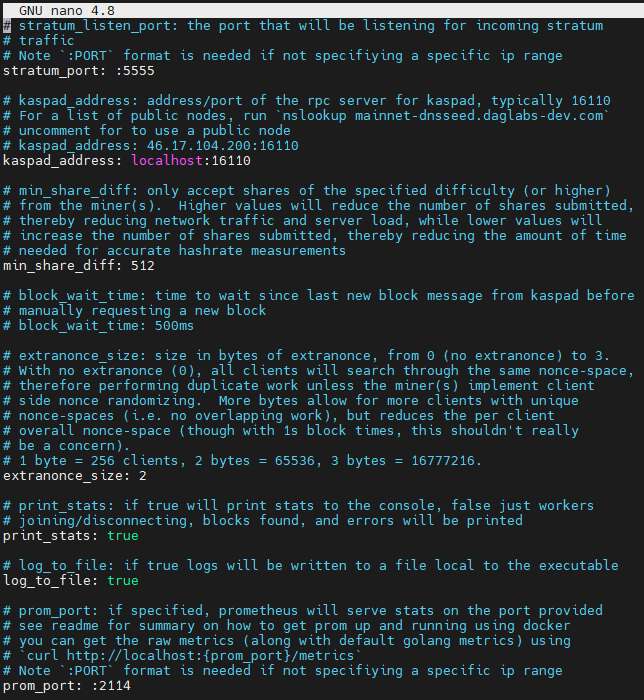
**- min\_share\_diff: 64**

**- extranonce\_size: 2**

**- pow2\_clamp : true**

**- var\_diff: true**

**Appuyer sur Ctrl+X, ensuite « y » , puis « ENTR »**

****

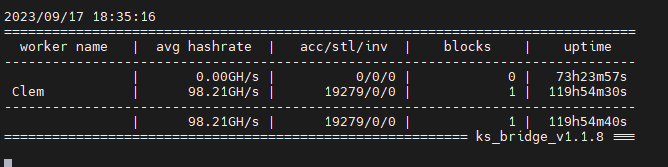
**go build . // pour construire l’exectucable kaspabridge**

**// Installer screen , pour pouvoir ouvrir un screen et lancer kaspabridge dedans : ligne par ligne**

**sudo apt install screen**

**screen -S Bridge**

**./kaspabridge**

****

**// quitter le screen CTRL+a+d**

**// retourné dans le répertoire cd kaspa-stratum-bridge-1.2.1/**

**cd ..**

**cd ..**

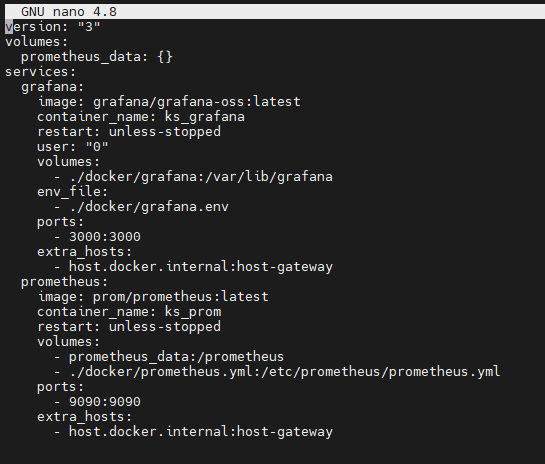
**// modifié le document texte docker-compose-monitoring.yml**

**nano** **docker-compose-monitoring.yml**

**Utiliser les flèches du clavier pour se déplacer dans l’éditeur de texte**

**- ajouter une ligne en haut : version: "3"**



****

**// Installer Docker et Docker-compose ligne par ligne**

**sudo apt install docker**

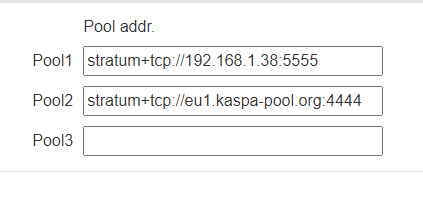
**sudo apt install docker-compose**

**sudo docker-compose -f docker-compose-monitoring.yml up -d**

**// Tout est FINI**

**Interface Web IceRiver :**

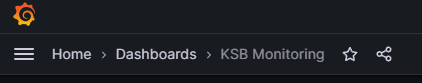
**Adresse du node de l’exemple pour solo mine : stratum+tcp://192.168.1.100:5555**

****

**Interface Web Node Kaspa :**

**Interface https : 192.168.1.100 :3000**

* **Id : admin**
* **Mdp : admin**

****

****