Déploiement d'un nœud de smart contrat

Nous allons voir comment déployer un nœud capable de lancer des smart contrats. Dans le cadre de ce tuto, nous utilisons une blockchain de test, dont la liste des master nodes est : 167.86.103.31:8080,5.189.168.49:8080,173.212.229.88:8080,62.171.153.36:8080,167.86.124.188:8080

Les administrateurs de cette blockchain sont des membres de l'équipe Kalima Systems. Pour toute manipulation d'administration, comme pour autoriser votre nœud par exemple, veuillez-vous adressez à : jerome.delaire@kalima.io

Développer le nœud

On considère dans ce tutoriel que le smart contrat node est déjà développé. Si ce n'est pas le cas, vous pouvez le développer en suivant le tutoriel SmartContractNode.

Installer le nœud

Pour fonctionner, le nœud doit être installé sur une machine linux possédant une adresse ip publique, car les contrats seront déployés sur cette machine par une chaine devOps via SSH. Si vous souhaitez installer un smart contrat node dans une machine qui ne dispose pas d'une adresse ip publique, nous sommes en mesure de proposer une solution alternative.

Dans le cadre de ce tutoriel, nous allons expliquer comment procéder pour déployer un smart contrat node sur un serveur. Le nœud qui sera déployé est issue du code d'exemple SmartContractNode, mais vous pouvez bien sûr faire de même avec votre propre nœud.

Tout d'abord, il nous faut exporter le code implémentant le smart contract node au format .jar

Dans eclipse, ceci se fait simplement en cliquant sur fichier → export → Java → runnable jar. Vous choisissez ensuite la bonne configuration et l'emplacement de votre jar sur votre machine locale. Dans le cadre de notre exemple SmartContractNode, il faudra passer deux paramètres lors du lancement du nœud : Le fichier de configuration du nœud Kalima, ainsi que le nom du répertoire git sur lequel les contrats seront stockés. Le jar pourra par exemple être lancé via la commande suivante :

```
java -jar SmartContractNode.jar node.config ContractsTest
```

Ensuite, il faut vous connecter sur votre machine linux (ici, un vps). Il est important de rappeler que cette dernière doit posséder une ip publique pour permettre l'accès extérieur à la chaîne DevOps en SSH.

Une fois cela fais, il vous faut tout simplement intégrer votre jar sur votre machine linux (par ligne de commandes par exemple ; sur un vps, on utilise la commande ***ssh nom_d'utilisateur@adresse*** pour se connecter en ssh sur l'utilisateur spécifié et à l'adresse spécifiée, puis on utilise la commande ***scp nomDuFichierAEnvoyer nomUtilisateur@AdresseServeur:~/RenommerLeFichier*** pour transmettre des fichiers).

On notera qu'en plus du jar, il ne faut pas oublier de transmettre le fichier node.config à votre machine.

Créer un répertoire git pour les contrats

Les contrats sont stockés sur un répertoire git, puis les contrats cryptés sont déployés vers les smart contract nodes. Vous devez donc créer un répertoire git pour vos smarts contrats. Ce répertoire git devra être accessible en lecture par la chaîne devOps.

Dans le cadre de ce tutorial, vous pouvez créer un répertoire sur notre serveur git : https://git.kalimadb.com/

Il faut ensuite donner l'accès en lecture à l'utilisateur gocd, et nous indiquer le lien du répertoire.

Rappelons qu'en argument on doit **uniquement** indiquer le nom du répertoire. Pour plus d'informations sur le déclenchement des smartcontracts, voir la partie « from scratch » de la documentation SmartContractNode.

Copier la clé publique SSH

Pour que les contrats puissent être déployés sur votre machine, il faut que la chaîne devOps puisse y accéder. Pour cela, il faut copier la clé publique SSH de la chaîne devOps sur votre machine. Dans le cadre de ce tutoriel, la clé vous a été fournie par Kalima Systems, si ce n'est pas le cas, veuillez contacter jerome.delaire@kalima.io

Se placer dans le dossier contenant la clé, puis utiliser cette commande pour copier la clé :

ssh-copy-id -f -i id_rsa.pub <user>@<ip_cible>

Avec :

- user ➔ L'utilisateur de votre machine (par exemple, rcs)
- ip_cible ➔ L'adresse IP de votre machine

Connecter le nœud à la blockchain Kalima

Pour fonctionner, le nœud doit être autorisé sur la blockchain sur une liste d'adresse de notre choix avec à minima une autorisation sur /Kalima_Scripts (pour récupérer les infos de cryptage et les signatures des contrats).

Cette étape d'autorisation est faite par un admin, on autorise le nœud via un serialId temporaire (valable 5 minutes). Pour qu'un administrateur de la blockchain puisse autoriser votre smart contrat node, vous devez lui fournir :

- L'adresse IP de la machine qui va héberger le nœud
- L'utilisateur de la machine pour lequel les contrats seront déployés. Si l'utilisateur est rcs par exemple, les contrats seront déployés dans /home/rcs/contracts/
- L'url du repository git
- La liste des smarts contrats que votre nœud pourra lancer

Une fois toutes les étapes effectuées, on peut lancer notre jar avec la commande : ***java -jar nomdufichier.jar chemin_du_fichier_node_config nom_du_répertoire_git***