Installation de NodeJs et NPM

Pour obtenir cette version, vous pouvez utiliser le gestionnaire de package `apt`. Rafraîchissez d'abord votre index local de paquets en tapant :

```
sudo apt update
sudo apt install nodejs
nodejs -v

Output
v10.19.0

sudo apt install npm
```

Installation de Git avec les paquets par défaut

L'option d'installer avec les paquets par défaut est meilleure si vous souhaitez être rapidement opérationnel avec Git, si vous préférez une version stable largement utilisée ou si vous ne cherchez pas les plus récentes fonctionnalités disponibles. Si vous recherchez la version la plus récente, vous devriez passer à la section sur l'installation à partir de la source.

Git est probablement déjà installé dans votre serveur Ubuntu 20.04. Vous pouvez confirmer que c'est le cas sur votre serveur avec la commande suivante :

```
git --version
##############################################################################
##Si vous recevez une sortie similaire à la suivante, Git est déjà installé.##
##############################################################################
Output
git version 2.25.1

```

Si c'est votre cas, vous pouvez passer à la configuration de Git, ou vous pouvez lire la section suivante sur la façon d'installer à partir des sources si vous avez besoin d'une version plus récente.

Cependant, si vous n'avez pas obtenu de sortie d'un numéro de version Git, vous pouvez l'installer avec l'APT du gestionnaire de paquets par défaut d'Ubuntu.

Tout d'abord, utilisez les outils de gestion de paquets apt pour mettre à jour l'index de vos paquets locaux.

Configuration de Git

Une fois que vous êtes satisfait de votre version de Git, vous devez configurer Git de manière à ce que les messages de validation générés contiennent vos informations correctes et vous aident à créer votre projet logiciel.

La configuration peut être réalisée à l'aide de la commande. Plus précisément, nous devons fournir notre nom et notre adresse e-mail, car Git intègre ces informations dans chaque commit que nous faisons. Nous pouvons ajouter ces informations en tapant : `git config`

```
$ git config --global user.name "Your Name"
$ git config --global user.email "youremail@domain.com"
```

```
$ git config --list
```

```
Output
user.name=Your Name
user.email=youremail@domain.com
...
```

Les informations que vous entrez sont stockées dans votre fichier de configuration Git, que vous pouvez éventuellement modifier à la main avec un éditeur de texte de votre choix comme ceci (nous utiliserons nano) :

```
$ nano ~/.gitconfig
```

~/.gitconfig contenu

```
[user]
  name = Your Name
  email = youremail@domain.com
```

Appuyez sur et , puis sur pour quitter l'éditeur de texte. `CTRLXYENTER`.

Installation de Maria-db serveur

Avant d'installer MariaDB, mettez à jour l'index des paquets sur votre serveur avec `apt` :

```
$ sudo apt update
```

Ensuite, installez le paquet :

```
$ sudo apt install mariadb-server
```

Lorsqu'il est installé à partir des dépôts par défaut, MariaDB se lancera automatiquement. Pour le tester, vérifiez son état.

```
$ sudo systemctl status mariadb
```

Vous obtiendrez un résultat similaire à celui qui suit :

```
Output
● mariadb.service - MariaDB 10.3.22 database server
     Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2020-05-12 13:38:18 UTC; 3min 55s ago
       Docs: man:mysqld(8)
             https://mariadb.com/kb/en/library/systemd/
   Main PID: 25914 (mysqld)
     Status: "Taking your SQL requests now..."
      Tasks: 31 (limit: 2345)
     Memory: 65.6M
     CGroup: /system.slice/mariadb.service
             └─25914 /usr/sbin/mysqld
. . .
```

Si MariaDB ne fonctionne pas, vous pouvez le démarrer avec la commande `sudo systemctl start mariadb`.

Configuration de Maria-db serveur

Exécutez le script de sécurité qui a été installé avec MariaDB. Vous serez alors guidé à travers une série d'invites où vous pourrez apporter quelques modifications aux options de sécurité de votre installation MariaDB :

```
$ sudo mysql_secure_installation
```

La première invite vous demandera de saisir le mot de passe actuel de l'utilisateur **root** de la base de données. Comme vous n'en avez pas encore créé un, appuyez sur `ENTER` pour indiquer « none ».

```
Output
. . .
Enter current password for root (enter for none):
```

L'invite suivante vous demande si vous souhaitez créer un mot de passe pour l'utilisateur **root** de la base de données. Sur Ubuntu, le compte **root** (racine) MariaDB est étroitement lié à la maintenance du système automatisé, nous ne devrions donc pas modifier les méthodes d'authentification configurées pour ce compte. Tapez `N` et appuyez ensuite sur `ENTER`.

```
Output
. . .

Set root password? [Y/n] N
```

À partir de là, vous pouvez appuyer sur `Y` puis sur `ENTER` pour accepter les valeurs par défaut pour toutes les questions suivantes. Cela supprimera certains utilisateurs anonymes et la base de données de test, désactivera les connexions **root** à distance, puis lancera ces nouvelles règles.

Étape 3 - (Facultatif) Création d'un utilisateur administratif qui utilise l'authentification par mot de passe

Sur les systèmes Ubuntu fonctionnant avec MariaDB 10.3, l'utilisateur **root** de MariaDB est configuré pour s'authentifier en utilisant le plugin `unix_socket` par défaut plutôt qu'avec un mot de passe. Parce que le serveur utilise la **root** pour des tâches telles que la rotation du journal et le démarrage et l'arrêt du serveur, il est préférable de ne pas modifier les détails d'authentification du compte **root**. Les responsables des paquets recommandent plutôt de créer un compte administratif séparé pour l'accès par mot de passe.

```
$ sudo mariadb
```

Créez ensuite un nouvel utilisateur avec des privilèges de **root** et un accès par mot de passe. Veillez à modifier le nom d'utilisateur et le mot de passe en fonction de vos préférences :

```
1 GRANT ALL ON *.* TO 'admin'@'localhost' IDENTIFIED BY 'password' WITH GRANT OPTION;
```

Purgez les privilèges pour vous assurer qu'ils sont enregistrés et disponibles dans la session en cours :

```
MariaDB&amp;nbsp;[(none)]> FLUSH PRIVILEGES;
```

Ensuite, quittez le shell MariaDB :

```
MariaDB&amp;nbsp;[(none)]> exit
```

Vous pouvez tester ce nouvel utilisateur avec l'outil `mysqladmin`, un client qui vous permet d'exécuter des commandes administratives. La commande `mysqladmin` suivante se connecte à MariaDB en tant qu'**admin** user et renvoie le numéro de version après avoir demandé le mot de passe de l'utilisateur :

```
$ mysqladmin -u admin -p version
```

Vous recevrez un résultat similaire à celui-ci :

```
Output
mysqladmin  Ver 9.1 Distrib 10.3.22-MariaDB, for debian-linux-gnu on x86_64
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Server version          10.3.22-MariaDB-1ubuntu1
Protocol version        10
Connection              Localhost via UNIX socket
UNIX socket             /var/run/mysqld/mysqld.sock
Uptime:                 4 min 49 sec

Threads: 7  Questions: 467  Slow queries: 0  Opens: 177  Flush tables: 1  Open tables: 31  Qu
```

Suivez le reste des étapes

Conclusion

Dans ce guide, vous avez installé Git, NodeJs, NPM ainsi que le système de gestion de base de données relationnelle MariaDB, et l'avez sécurisé à l'aide du script d'installation `mysql_secure_installation` fourni. Vous aviez également la possibilité de créer un nouvel utilisateur administratif qui utilise l'authentification par mot de passe.