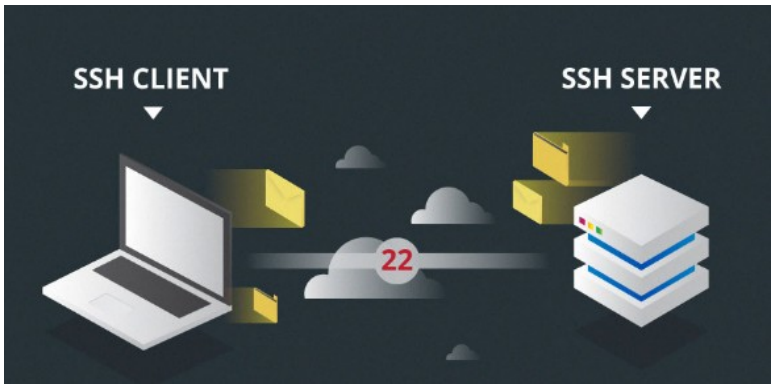
MODULE 4

Serveur ssh



Objectifs de ce module :

- ✓ ***Installation du serveur ssh (openssh)***
- ✓ ***Configuration des fichiers reliés au serveur***
- ✓ ***Activer le serveur***
- ✓ ***Utiliser des utilitaires reliés à ssh***
- ✓ ***Autorisations par clés***

Table des matières

Introduction

Qu'est-ce que SSH

SSH, ou Secure Shell, est un protocole d'administration à distance qui permet aux utilisateurs de contrôler et de modifier leurs serveurs distants via Internet. Le service a été créé en remplacement sécurisé du Telnet non chiffré et utilise des techniques cryptographiques pour garantir que toutes les communications vers et depuis le serveur distant se déroulent de manière chiffrée. Il fournit un mécanisme pour authentifier un utilisateur distant, transférer les entrées du client vers l'hôte et relayer la sortie vers le client.

Quelques concepts clés

L'avantage significatif offert par SSH par rapport à ses prédécesseurs est l'utilisation du cryptage pour assurer un transfert sécurisé des informations entre l'hôte et le client. **L'hôte** fait référence au serveur distant auquel vous essayez d'accéder, tandis que le **client** est l'ordinateur que vous utilisez pour accéder à l'hôte. SSH utilise surtout des algorithmes de chiffrement qui se catégorisent en 2 groupes :

- Chiffrements asymétriques
- Chiffrements symétriques.

Chiffrements Symétriques

C’est le mode de chiffrement qui est utilisé par SSH durant la connexion normale entre 2 ordinateurs. Avec cette méthode, une clé est utilisée pour chiffrer un message depuis l’ordinateur client. Le message encrypté est ensuite envoyé à l’autre ordinateur qui décode ce message encrypté avec la même clé.

Il faut donc que la personne qui chiffre et celle qui déchiffre connaissent toutes deux cette clé qui sert à chiffrer et déchiffrer. Pour cela, il faut initialement que le client envoi la clé qui sera utilisée à l’autre ordinateur. Mais comment faire pour que la clé initiale qui est envoyée ne soit pas interceptée par une tierce partie ?

Chiffrements Asymétriques

De façon à ce que la clé initiale ne soit pas intercepté en clair, celle-ci est envoyée de façon encryptée. C'est là qu'intervient le chiffrement asymétrique. Le chiffrement asymétrique, lui, utilise une clé pour chiffrer, et une autre pour déchiffrer.

Il y a donc deux clés :

- une clé dite « **publique** » qui sert à **chiffrer** ;
- une clé dite « **privée** » qui sert à **déchiffrer**.

La clé publique ne sert qu'à chiffrer. Avec ce type d'algorithme, on ne peut déchiffrer un message que si l'on connaît la clé privée.
On demande à l'ordinateur de générer une paire de clés : une privée et une publique. Elles vont ensemble.

Installation du serveur

Sous Linux Mint : 

```
apt  install  openssh-server
```

Note :

- Par défaut, le serveur ssh n'est pas installé sur linux Mint. Il faut donc procéder à l'installation tel que montré dans l'encadré ci-dessus.

Démarrage du serveur

Il s'agit de démarrer le serveur ssh avec la commande systemctl.

```
sudo systemctl  start  ssh
```

Si vous voulez que le serveur parte à chaque démarrage de l'ordinateur, on oubliera pas de faire aussi :

```
sudo systemctl  enable ssh
```

Utilisation du client

Le client ssh s'utilise avec la commande ssh. Celle-ci est déjà installé par défaut dans la plupart des distributions.

Connexion à un serveur ssh

```
ssh    usager@adresse_IP_Autre_Ordinateur
```

Avec cette commande, vous vous connecterai en utilisant le compte « usager » de l'ordinateur distant. Si vous ne mettez pas l'usager devant l'adresse IP, le compte utilisé sera celui avec lequel vous effectuez la commande.
Vous devez donc vous assurer, à ce moent-là, que le compte existe aussi sur la machine distante.

Option -p

Cette option permet aussi de spécifier le port à utiliser pour ce connecter à la machine distante.

```
ssh    -p  22122 usager@adresse_IP_Autre_Ordinateur
```

Ainsi, dans la commande précédente, le port utilisé sera le 22122 au lieu du 22 comme par défaut.

Configuration du serveur

Emplacement des fichiers de configuration sous Mint

Une fois l'installation du paquet terminé, les fichiers de configuration du serveur se trouve dans le dossier « **/etc/ssh**».

Les fichiers présents sont les suivants :

Fichier	Signification
sshd_config	C'est le fichier principal de configuration du serveur ssh. Il contient les instructions qui mentionnent comment le serveur doit se comporter pour répondre à certains critères.
ssh_config	C'est le fichier de configuration mais pour les client. C'est-à-dire, ceux qui vont utiliser votre serveur ssh. Habituellement, ce fichier se retrouve dans le dossier des usagers et ceux-ci peuvent le configurer selon leur besoin. Ce fichier est uniquement présent pour les cas par défaut.

Fichier de configuration sshd_config

Le fichier de configuration se trouve dans le répertoire « **/etc/ssh** » et porte le nom **« sshd_config** ».

Tableau des principales directives à modifier le cas échéant :

Directive du fichier	Valeur par défaut sous Ubuntu	Valeur possible	Effet de la valeur choisie
Port	22	Tous les ports non utilisés de 1024 à 65535	Permet d'éviter des désagréments avec les robots qui scannent Internet, notamment les ports par défaut
PermitRootLogin	without-password / prohibit-password	yes no without-password forced-commands-only prohibit-password	Yes : le compte root pourra se brancher No : Pas de branchement root
PubkeyAuthentication	yes	no	Laisser yes si vous voulez établir l'authentification par clé.
PasswordAuthentication	yes	no	On peut parfaitement conserver l'authentification par clé pour certains utilisateurs avec celle par mot de passe pour d'autres utilisateurs. Conserver cette valeur à yes tant que l'authentification par clé n'est pas pleinement fonctionnelle.

Directive du fichier	Valeur par défaut sous Ubuntu	Valeur possible	Effet de la valeur choisie
Port	22	Tous les ports non utilisés de 1024 à 65535	Permet d'éviter des désagréments avec les robots qui scannent Internet, notamment les ports par défaut
PermitRootLogin	without-password / prohibit-password	yes no without-password forced-commands-only prohibit-password	Yes : le compte root pourra se brancher No : Pas de branchement root
PubkeyAuthentication	yes	no	Laisser yes si vous voulez établir l'authentification par clé.
PasswordAuthentication	yes	no	On peut parfaitement conserver l'authentification par clé pour certains utilisateurs avec celle par mot de passe pour d'autres utilisateurs. Conserver cette valeur à yes tant que l'authentification par clé n'est pas pleinement fonctionnelle.
X11Forwarding	yes	no	Laisser yes pour faire de l'affichage graphique déporté
#Banner /etc/issue.net	Ligne commentée donc inactive	Décommenter	Lorsque vous essayez de vous connecter à votre serveur par SSH, le fichier /etc/issue.net est affiché (à vous de le personnaliser pour dire bonjour ou mettre un avertissement, un guide, etc.)

Directive du fichier	Valeur par défaut sous Ubuntu	Valeur possible	Effet de la valeur choisie
UsePAM	yes	no	Mettre à no pour ne plus avoir à saisir un mot de passe avec l'usage des clés. Va de pair avec PubkeyAuthentication
AllowUsers	Ligne absente (autorisé à tous)	ajouter la ligne avec valeur(s) : AllowUsers Alice Bob	Spécifie les *logins* des seuls utilisateurs autorisés à se connecter. *Idéal pour ouvrir un compte à un ami tout en restreignant l'accès au shell via SSH*.
DenyUsers	Ligne absente (interdit à personne)	Ajouter la ligne avec valeur(s)	Interdit l'accès à SSH aux utilisateurs listés
AllowGroups	Ligne absente (autorisé à tous les groupes)	ajouter la ligne avec valeur(s) : AllowGroups groupname1 groupname2	L'authentification via SSH ne sera possible que par des utilisateurs des groupes désigné par leur nom (pas par GID)
DenyGroups	Ligne absente (interdit à aucun groupe)	Ajouter la ligne avec valeur(s)	Interdit l'accès à SSH aux utilisateurs des groupes listés
ClientAliveInterval	Ligne absente	Ajouter la ligne avec valeur en secondes : ClientAliveInterval 300	Permet dans certains cas de maintenir une connexion sans coupures

Sécuriser l’accès au serveur ssh

Comment se connecter à SSH avec des clés

S'il est utile de pouvoir se connecter à un système à distance à l'aide de mots de passe, il est bien plus judicieux de mettre en place une authentification par clé.
Vous aurez ainsi le choix d’accepter des connexions juste avec les clés, ou juste avec les mots de passe ou les deux.

Comment fonctionne l'authentification par clé ?

L'authentification par clé fonctionne en créant une paire de clés : une clé privée et une clé publique.

La clé privée se trouve sur la machine du client et est sécurisée et gardée secrète.

La clé publique peut être donnée à n'importe qui ou placée sur n'importe quel serveur auquel vous souhaitez accéder.

Comment créer des clés SSH

Les clés SSH doivent être générées sur l'ordinateur duquel vous souhaitez vous connecter. Il s'agit généralement de votre machine locale.

Étapes 1 : On génère les clés

```
ssh-keygen -t rsa
```

- On vous demande d’entrer l’emplacement pour sauvegarder le fichier, on accepte généralement l’emplacement par défaut (/home/schasse/.ssh/id_rsa)
- On vous demande ensuite une « passphrase ». C’est un espèce de mot de passe qui protège la clé privée. C’est conseillé d’en mettre une mais pour l’instant nous allons laisser la « passphrase » vide. On vous le demandera une autre fois.
- La clé privée et publique est générée de même que le random art qui en découle.

Déplacez-vous dans le dossier : cd .ssh

Lister les fichier : ls -al

Vous verrez les fichiers générés.

Voici un exemple :

```
drwx------  2 schasse schasse 4096 janv. 26 21:58 .
drwxr-xr-x 16 schasse schasse 4096 janv. 26 21:28 ..
-rw-------  1 schasse schasse 2602 janv. 26 21:58 id_rsa
-rw-r--r--  1 schasse schasse  572 janv. 26 21:58 id_rsa.pub
-rw-r--r--  1 schasse schasse  222 janv. 26 13:48 known_hosts
```

Étape 2 : Transférer la clé publique sur le serveur

```
ssh-copy-id   usager@adresse_IP_Ordinateur_Distant
```

Évidemment, la clé publique sera copiée dans le dossier .ssh de l'usager que vous utiliserez dans la commande. Si vous ne mettez pas d'usager, la clé sera copiée dans le compte qui porte le même nom que l'usager courant de votre machine locale.

Exemple avec user1 :

```
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
user1@192.168.0.1's password:
```

Entrez le mot de passe de l'usager en question.

Vous recevrez alors ceci :

```
Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'user1@192.168.0.1'"
and check to make sure that only the key(s) you wanted were added.
```

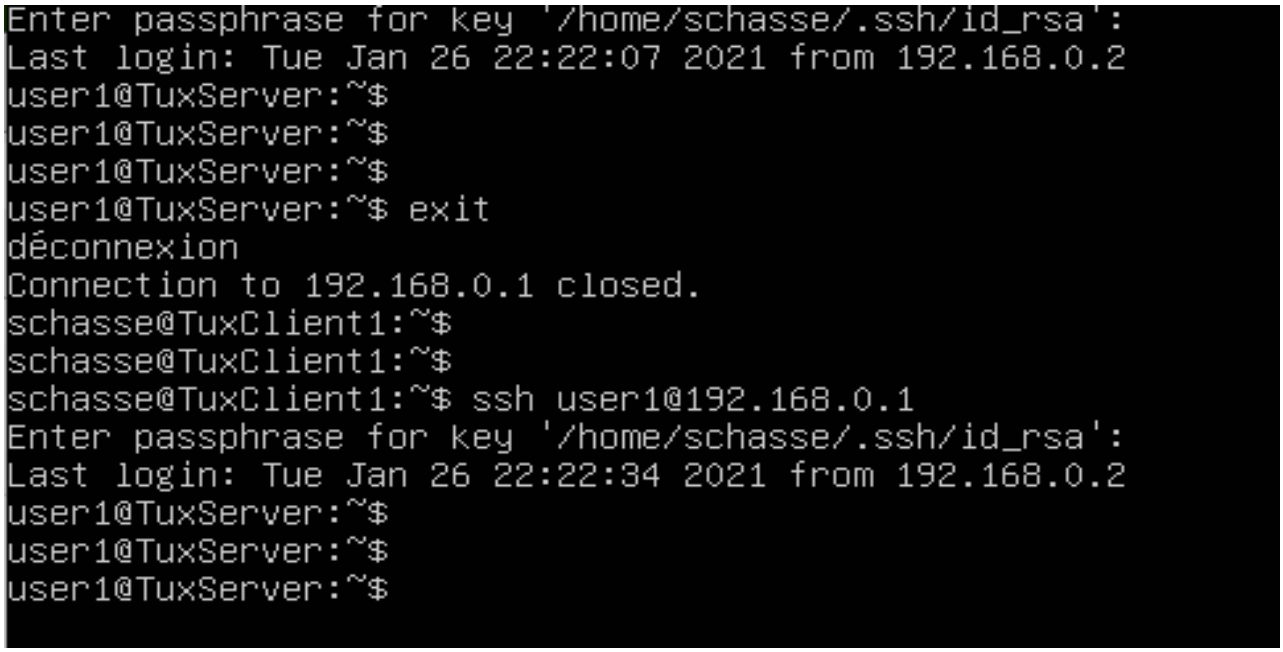
C'est la confirmation que la clé publique a bien été copié sur le serveur dans le compte voulu.

Essayons avec le compte de user1 : ssh user1@192.168.0.1

Nous entrons directement sans avoir besoin d'entrer de mot de passe comme on le faisait avant.

Exemple avec cas d'un « passphrase »

Dans le cas que vous générez vos clés avec une passphrase. Voici ce qui se passera lorsque vous tenterez de vous connecter à votre serveur par ssh en console uniquement (sans interface graphique):



La « passphrase » sera demandée à chaque tentative de connexion, et ce, même si on authentifie l'usager par clé. Ceci va un peu à l'encontre de ce qu'on voulait réaliser en évitant d'entrer un mot de passe!!

Pour palier à ceci, voici l'agent ssh

Agent ssh

Pour permettre de conserver la « passphrase » dans l'utilitaire lorsque vous la taperez la première fois.

Pour ce faire :

Étape 1 : En étant connecté avec le compte avec lequel vous voulez aller porter les informations pour l'agent ssh, réalisez la commande suivante :

```
eval  $(ssh-agent)
```

Vous recevrez le numéro du PID.

Étape 2 : Aller porter la « passphrase » dans l'agent ssh.

```
ssh-add
```

Évidemment, on vous demandera la « passphrase » au moins la première fois.

À partir de maintenant plus aucun mot de passe ni de passphrase seront nécessaire.

Options de configuration côté serveur

Cette section contient quelques options de configuration courantes côté serveur qui peuvent façonner la manière dont votre serveur répond et les types de connexions autorisés.

Désactivation de l'authentification par mot de passe

Si des clés SSH sont configurées, testées et fonctionnent correctement, il est probablement judicieux de désactiver l'authentification par mot de passe. Cela empêchera tout utilisateur de se connecter avec SSH à l'aide d'un mot de passe.
Pour ce faire, connectez-vous à votre serveur distant et ouvrez le fichier `/etc/ssh/sshd_config`fichier avec les privilèges root ou sudo.

À l'intérieur du fichier, recherchez l'option `PasswordAuthentication`directive. S'il est commenté, décommentez-le. Mettes-le à no.

```
PasswordAuthentication no
```

N'oubliez pas après chaque changement au fichier de configuration du serveur, de repartir le service ssh.

```
systemctl restart ssh
```

Modification du port sur lequel le démon SSH s'exécute

Certains administrateurs vous suggèrent de modifier le port par défaut sur lequel SSH s'exécute. Cela peut aider à réduire le nombre de tentatives d'authentification auxquelles votre serveur est soumis à partir de robots automatisés.

Exemple : Dans le fichier sshd_config

```
Port 4444
```

Permettra de faire répondre votre serveur sur le port 4444 au lieu du port 22 par défaut.

Limitation des utilisateurs pouvant se connecter via SSH

Pour limiter explicitement les comptes d'utilisateurs qui peuvent se connecter via SSH, vous pouvez adopter plusieurs approches différentes, chacune impliquant la modification du fichier de configuration du démon SSH.

Par usager (AllowUsers)

La première méthode pour spécifier les comptes autorisés à se connecter consiste à utiliser la directive « AllowUsers ». Recherchez la directive « AllowUsers » dans le fichier. S'il n'en existe pas, créez-le n'importe où (habituellement à la fin).

Exemple :

```
AllowUsers  user1  user2
```

Seul les utilisateurs « user1 » et « user2 » pourront se connecter à votre serveur ssh.

Par groupe (AllowGroups)

Si vous êtes plus à l'aise avec la gestion de groupe, vous pouvez utiliser la directive AllowGroups à la place. Si tel est le cas, ajoutez simplement un seul groupe qui devrait être autorisé à accéder SSH.

Exemple :

```
AllowGroups  sshmembers
```

Désactivation de la connexion racine

Il est souvent conseillé de désactiver complètement la connexion root via SSH après avoir configuré un compte utilisateur SSH disposant de sudoprivilèges.

Exemple :

```
PermitRootLogin   no
```

Autoriser l'accès root pour des commandes spécifiques

Dans certains cas, vous souhaiterez peut-être désactiver l'accès root en général, mais l'activer afin de permettre à certaines applications de fonctionner correctement. Un exemple de ceci pourrait être une routine de sauvegarde.

Ceci peut-être accompli par le fichier authorized_keys dans le compte de « root ».

Étape 1 : Générer les clés pour l'utilisateur « root » à partir de la machine locale.
Suivez les mêmes étapes que pour la création des clés que nous avons fait précédemment mais pour « root » cette fois-ci.

Étape 2 : Copiez la clé sur l'ordinateur distant dans le compte de root.

```
ssh-copy-id  root@IP_Distant
```

Étape 3 : Connectez-vous au serveur avec un usager ayant un accès ssh.

Étape 4 : Éditez le fichier authorized_keys dans le dossier /root/.ssh

Étape 5 : Ajoutez au début de la ligne avec la clé que vous avez téléchargé, la commande avec le chemin complet que vous permettez à l'utilisateur « root » de faire.

```
command="/bin/ls  arg1  arg2…"  ssh-rsa ...
```

Étape 6 : Recherchez la directive « PermitRootLogin » et modifiez-là comme ci-dessous :

```
PermitRootLogin  forced-commands-only
```

Root pourra alors lancer, à partir de la machine locale, la commande suivante :

```
ssh  root@adresse_IP_Distant  ls -al
```

La commande « ls -al » sera exécutée sur la machine distante en étant « root » mais sans avoir à se connecter avec le compte « root ».

Configuration côté client

Pour aller plus vite au lieu de taper :

ssh user1@192.168.0.1

On peut créer un fichier qui comporte l'information voulue et ensuite utiliser ce raccourci pour ssh.

Voyons un exemple avec l'usager « user1 » :

Disons que cet usager se connecte fréquemment au serveur 192.168.0.1 alors il pourra ajouter l'information dans le fichier ssh_config (attention, à ne pas confondre avec le fichier du serveur sshd_config)

Étape 1 : Éditer le fichier ssh_config situé dans le dossier .ssh de l'utilisateur en question

Étape 2 : Entrez les informations suivantes :

```
Host serveur1
  HostName 192.168.0.1
  Port 22
  User user1
```

L'utilisateur « user1 » pourra maintenant taper la commande : ssh serveur1

pour se connecter.

On peut avoir plusieurs « host » dans le même fichier.

Autres usages de ssh

Copie de fichiers à distance

Avec la commande « scp », on peut copier des fichiers à distance.

scp <fichier> <username>@<ipaddressDistant>:<DestinationDirectory>

Exemples

Pour un fichier:

```
scp fichier.txt user1@192.168.0.1:/home/user1
```

Pour un répertoire:

```
scp -r répertoire user1@192.168.0.1:/home/user1
```

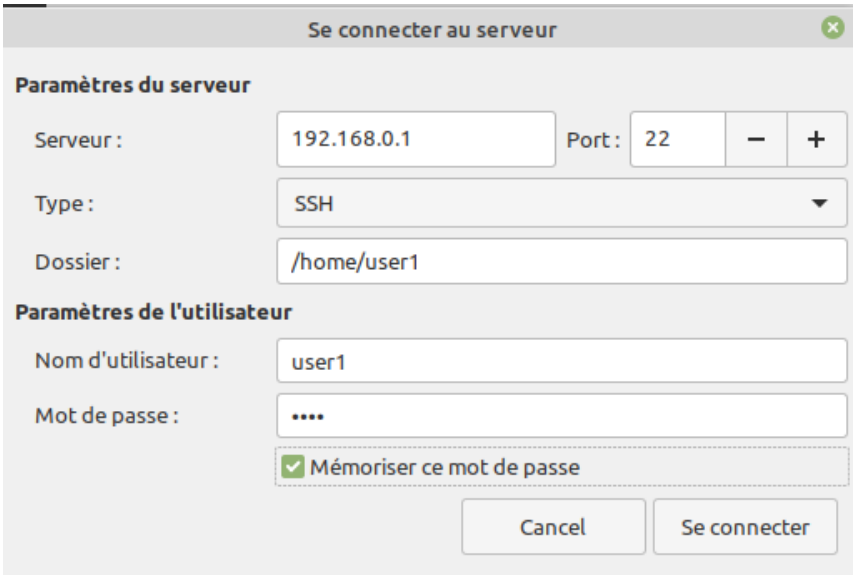
Vous pouvez aussi bien copier des fichiers à partir des ordinateurs à distance sur votre disque local :

```
scp  user1@192.168.0.1:/home/hornbeck/fichier_important.txt   .
```

Monter un répertoire distant, navigation via SFTP

Dans l'explorateur de ficher, sélectionnez fichier → « Se connecter au serveur »

Entrez les informations semblables à celles ci-dessous et reliées à votre cas.
Voici un exemple :



Monter un répertoire distant de manière automatique

Utilisation en mode console

Si votre serveur ssh est installé, rien de plus simple :

Faites-vous un dossier dans votre compte et exécutez la commande sshfs.

Voici un exemple :

```
mkdir  documents_sur_serveur1

sshfs  user1@192.168.0.1:/home/user1/Documents   documents_sur_serveur1
```

Vous devrez démonter le dossier lorsque vous en avez plus besoin :

```
   umount  documents_sur_serveur1
```

Montage automatique d’un dossier via sshfs

si vous montez souvent les mêmes répertoires, il peut être intéressant de les entrer dans le fichier fstab.

D'abord, créez un répertoire, dans cet exemple sous /mnt, puis assignez-lui les droits :

```
sudo mkdir /mnt/mon_rep

sudo chown root:users /mnt/mon_rep

sudo chmod 770 /mnt/mon_rep
```

Ensuite modifiez le fichier **/etc/fstab**. Ajouter cette ligne à la fin du fichier.

```
user@machine:/répertoire/distant    /mnt/mon_rep          fuse.sshfs      port=22,user,auto,noatime    0   0
```

Le dossier « /répertoire/distant » de « machine » sera automatiquement accessible après le démarrage de la machine dans le dossier « /mnt/mon_rep » dans la machine locale.

Exemple :

user1@192.168.0.1:/home/user1/Documents /mnt/serveur1/User1/Documents fuse.sshfs port=22,user,auto,noatime 0 0