Formation Linux N2

2H - Florent Guillemette - Session 1/3

Qu'attendez-vous de cette formation ?

Sommaire 1/2

- Exécuter des commandes de manière plus efficace en utilisant les fonctionnalités avancées du shell Bash, des scripts shell et divers utilitaires fournis par Red Hat Enterprise Linux.
- Configurer un service de ligne de commande sécurisé sur des systèmes distants en utilisant OpenSSH.
- Archiver et copier des fichiers d'un système à un autre.
- Contrôler les connexions réseau vers les services en utilisant le pare-feu système et les règles SELinux.
- Programmer l'exécution de commandes dans le futur, soit une fois ou de manière récurrente.

Sommaire 2/2

- Configurer les interfaces réseau et les paramètres des serveurs Red Hat Enterprise Linux.
- Accéder, inspecter et utiliser des systèmes de fichiers existants sur le stockage attaché à un serveur Linux.
- Améliorer les performances du système en définissant des paramètres de réglage et en ajustant la priorité de planification des processus.

OpenSSH > Présentation

sshd est le serveur (daemon)

ssh est le client

D'autres utilitaires sont présents aussi.

OpenSSH > Fonctions

Permet la connexion à distance sur une machine (similaire à RDP)

Permet l'envoi et le téléchargement de fichiers.

OpenSSH > Installation 1/2

Téléchargement et installation

```
dnf install openssh-server
```

Démarrage du service

```
systemctl start sshd
systemctl enable sshd
```

Controle du service

```
systemctl status sshd
```

OpenSSH > Installation 2/2

Configuration du parefeux

```
# firewall-cmd --zone=public --permanent --add-service=ssh
# firewall-cmd --reload
```

OpenSSH > Configuration 1/2

Le fichier de configuration

```
/etc/ssh/sshd_config
```

OpenSSH > Configuration 2/2

Par défaut, l'utilisateur root n'est pas autorisé à se connecter

Pour autoriser la connexion de root:

```
PermitRootLogin yes
```

OpenSSH > Authentification par clé 1/3

Vous devez posséder une paire de clés.

Cette paire est constituée d'une clé privée et d'une clé publique.

La clé privée doit rester secrète.

La clé publique est "ajoutée" sur les machines cibles

Pour générer une clé sur la machine client;

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

OpenSSH > Authentification par clé 2/3

Pour copier la clé publique sur le serveur

```
ssh-copy-id user@remote_host
```

OpenSSH > Authentification par clé 3/3

Pour tester la connexion

```
ssh user@remote_host
```

SCP > Présentation

SCP (Secure Copy) est une commande utilisée pour transférer des fichiers entre des systèmes Linux distants de manière sécurisée.

```
scp [OPTIONS] SOURCE DESTINATION
```

SCP > Exemples d'utilisation

- Copier un fichier local vers un système distant :

```
scp fichier.txt utilisateur@serveur distant:/chemin/destination
```

- Copier un fichier distant vers un système local :

```
scp utilisateur@serveur distant:/chemin/source/fichier.txt /chemin/destination
```

SCP > Options couramment utilisées

- `-P port` : Spécifier un port personnalisé pour la connexion SSH.
- `-r` : Copier récursivement un répertoire et son contenu.
- `-v` : Afficher la sortie détaillée pour le débogage.
- `-p` : Préserver les attributs d'origine du fichier (horodatage, permissions, etc.).
- `-C` : Activer la compression lors de la transmission des données.

SCP > Authentification

SCP utilise SSH pour l'authentification et le transfert sécurisé des fichiers.

- Vous pouvez utiliser des clés SSH pour éviter de saisir le mot de passe à chaque transfert.
- Exemple : `scp -i chemin/vers/cle_privee fichier.txt utilisateur@serveur:/chemin/destination`

firewalld > Objectifs

- Gestion des zones de pare-feu basées sur les emplacements physiques ou les interfaces réseau.
- Configuration dynamique des règles sans interrompre les connexions établies.
- Prise en charge des services, des ports, des applications et des protocoles.
- Intégration avec NetworkManager pour une gestion cohérente des connexions réseau.

firewalld > Terminologie

- **Zone** : Un ensemble de règles de pare-feu appliquées à un emplacement ou une interface réseau spécifique.
- **Service** : Une définition de service prédéfini qui regroupe un ensemble de règles spécifiques pour une application ou un protocole.
- **Port** : Un numéro de port TCP ou UDP spécifique à autoriser ou à bloquer.
- **Source** : Les adresses IP ou les plages d'adresses autorisées à accéder au système.
- **Masquerade** : Une technique de traduction d'adresse (NAT) pour masquer l'adresse source des paquets sortants.

firewalld > Commandes utiles

- `firewall-cmd --get-default-zone` : Affiche la zone par défaut.
- `firewall-cmd --get-active-zones` : Affiche les zones actives.
- `firewall-cmd --list-all` : Affiche toutes les règles et les zones configurées.
- `firewall-cmd --zone=zone_name --add-service=service_name` : Ajoute un service à une zone spécifique.
- `firewall-cmd --zone=zone_name --add-port=port_number/protocol` : Ajoute un port à une zone spécifique.

firewalld > Gestion des zones

- `public` : Utilisée pour les connexions non fiables depuis Internet.
- `trusted` : Utilisée pour les connexions fiables en réseau local.
- `internal` : Utilisée pour les réseaux internes plus sécurisés.
- `work` : Utilisée pour les connexions professionnelles.
- `home` : Utilisée pour les connexions domestiques.
- `dmz` : Utilisée pour les systèmes exposés dans la zone démilitarisée (DMZ).

firewalld > Configuration persistante

- firewalld stocke les règles dans des fichiers XML dans le répertoire `/etc/firewalld`.
- Les modifications apportées via `firewall-cmd` sont appliquées immédiatement mais ne persistent pas après le redémarrage.
- Pour rendre les modifications persistantes, utilisez `firewall-cmd --runtime-to-permanent`.

SELinux - Security-Enhanced Linux > Présentation

SELinux (Security-Enhanced Linux) est un module de sécurité intégré dans le noyau Linux qui renforce les politiques de contrôle d'accès pour renforcer la sécurité du système.

SELinux > Objectifs

- Contrôle fin des autorisations d'accès aux fichiers, processus, ports réseau, etc.
- Isolation des processus pour limiter les effets des failles de sécurité.
- Protection contre les attaques de type "escalade de privilèges".
- Audits et logs pour une meilleure visibilité des activités système.

SELinux > Modes de fonctionnement

- **Enforcing** : Mode par défaut. SELinux applique strictement les politiques de sécurité et bloque les actions non autorisées.
- **Permissive** : SELinux génère des avertissements sans bloquer les actions non autorisées. Utile pour le débogage des politiques de sécurité.
- **Disabled** : SELinux est désactivé. Les politiques de sécurité ne sont pas appliquées.

SELinux > Contextes

- Chaque fichier, processus, port réseau, etc. a un contexte SELinux.
- Les contextes sont composés de :
  - **User** : Identifie l'utilisateur ou le rôle.
  - **Role** : Indique le rôle associé à l'utilisateur.
  - **Type** : Définit le type d'objet.
  - **MLS/MCS Level** : Niveau de sécurité multilabel (optionnel).

SELinux > Commandes utiles

- `sestatus` : Affiche l'état actuel de SELinux et le mode d'application.
- `getenforce` : Vérifie si SELinux est en mode "Enforcing" ou "Permissive".
- `setenforce` : Change le mode d'application SELinux.
- `semanage` : Gère les politiques SELinux, notamment les ports, les modules, les utilisateurs, etc.
- `ls -Z` : Affiche les contextes SELinux des fichiers et répertoires.

SELinux > Politiques SELinux

- Les politiques SELinux définissent les règles de contrôle d'accès.
- Les modules de politique sont utilisés pour personnaliser les politiques SELinux.
- Exemple : `semodule -i mon_module.pp` pour installer un module de politique personnalisé.

Cron & At > Présentation

- **cron** : Planificateur de tâches basé sur le temps.
- **at** : Planificateur de tâches ponctuelles.
- **systemd timers** : Planificateur de tâches intégré à systemd.

Cron & At > Utilisation de cron

- `crontab -e` : Édite la table des tâches cron de l'utilisateur courant.
- `crontab -l` : Affiche la table des tâches cron de l'utilisateur courant.
- `crontab -r` : Supprime la table des tâches cron de l'utilisateur courant.
- Syntaxe de la ligne de commande cron : `* * * * * commande`
  - Les cinq astérisques correspondent respectivement à la minute, à l'heure, au jour du mois, au mois et au jour de la semaine.
  - Utilisez des nombres ou des plages pour spécifier des valeurs spécifiques.
  - Exemple : `0 0 * * * sauvegarde.sh` exécutera le script "sauvegarde.sh" tous les jours à minuit.

Cron & At > Utilisation de at

- `at now + 5 minutes` : Planifie l'exécution d'une commande dans 5 minutes.
- `at 2:00 PM` : Planifie l'exécution d'une commande à une heure spécifique.
- `atq` : Affiche la file d'attente des tâches planifiées avec at.
- `atrm job_number` : Supprime une tâche planifiée spécifique de la file d'attente.
- `at -l` : Affiche les tâches planifiées de l'utilisateur courant.

Utilisation de systemd timers

- Créez un fichier de service unité avec une extension `.service` dans `/etc/systemd/system/`.
- Créez un fichier de timer unité avec une extension `.timer` dans `/etc/systemd/system/`.
- Exemple de fichier `.service` :

```
[Unit]
Description=Ma tâche planifiée

[Service]
ExecStart=/chemin/vers/commande

[Install]
WantedBy=multi-user.target
```

- Exemple de fichier `.timer` :

```
[Unit]
Description=Mon timer

[Timer]
OnCalendar=*-*-* 00:00:00
Unit=mon-service.service

[Install]
WantedBy=timers.target
```

- Activez et démarrez le timer avec `systemctl start mon-timer.timer`.