

Administration et Optimisation des Performances

CentOS / Ubuntu Systems

Guide complet pour ingénieurs système

Table des Matières

1. Surveillance des Ressources Système
2. Analyse des Logs
3. Optimisation des Performances
4. Monitoring Continu
5. Optimisations Spécifiques
6. Planification et Automatisation
7. Benchmarking
8. Gestion des Ports Réseau

1. Surveillance des Ressources Système

Un ingénieur système doit constamment surveiller les ressources pour maintenir des performances optimales.

1.1 Surveillance de la Mémoire (RAM)

Commandes essentielles pour surveiller l'utilisation de la mémoire :

```
free -h          # Utilisation mémoire en format lisible
vmstat 1        # Statistiques mémoire en temps réel
cat /proc/meminfo # Informations détaillées sur la mémoire
top -o %MEM     # Trier par utilisation mémoire
```

1.2 Surveillance du CPU

Outils pour monitorer l'utilisation du processeur :

```
top             # Vue en temps réel des processus
htop            # Version améliorée et interactive de top
mpstat -P ALL 1 # Statistiques par cœur CPU
lscpu           # Informations détaillées sur le CPU
uptime          # Charge moyenne du système
```

1.3 Surveillance du Disque

Commandes pour surveiller l'espace disque et les performances I/O :

```
df -h           # Espace disque utilisé par partition
du -sh /var/log/* # Taille des dossiers
iostop          # Activité I/O en temps réel (root requis)
iostat -x 1     # Statistiques I/O détaillées
lsblk           # Liste des périphériques de bloc
```

1.4 Surveillance du Réseau

Outils de monitoring réseau :

```
iftop           # Bande passante en temps réel
nethogs         # Utilisation réseau par processus
ss -tunap       # Connexions réseau actives
ip -s link      # Statistiques des interfaces réseau
nload           # Graphique de la bande passante
```

2. Analyse des Logs

Les logs système sont essentiels pour diagnostiquer les problèmes et surveiller l'activité du système.

2.1 Commandes de Base

```
# Logs système (systemd)
journalctl -xe          # Logs récents avec détails
journalctl -f            # Suivre les logs en temps réel
journalctl -u nginx      # Logs d'un service spécifique
journalctl --since "1 hour ago" # Logs de la dernière heure

# Logs traditionnels
tail -f /var/log/syslog    # Ubuntu/Debian
tail -f /var/log/messages   # CentOS/RHEL
dmesg | tail                # Messages du noyau
cat /var/log/auth.log       # Logs d'authentification
```

2.2 Analyse Avancée

```
# Rechercher des erreurs
journalctl -p err -b      # Erreurs depuis le dernier boot
grep -i error /var/log/syslog # Recherche dans les logs

# Rotation des logs
logrotate -f /etc/logrotate.conf # Forcer la rotation
```

3. Optimisation des Performances

3.1 Gestion des Services

```
# Lister les services actifs
systemctl list-units --type=service --state=running

# Désactiver les services inutiles
systemctl disable service_name
systemctl stop service_name

# Analyser le temps de boot
systemd-analyze blame
systemd-analyze critical-chain
```

3.2 Optimisation des Processus

```
# Identifier les processus gourmands
ps aux --sort=-%mem | head -10    # Top 10 RAM
ps aux --sort=-%cpu | head -10    # Top 10 CPU

# Gérer les priorités
nice -n 10 command                # Lancer avec priorité basse
renice -n -5 -p PID                # Augmenter la priorité
```

3.3 Paramètres Kernel (sysctl)

Optimisation des paramètres système :

```
# Voir tous les paramètres
sysctl -a

# Paramètres courants à optimiser
# /etc/sysctl.conf

# Augmenter la taille du cache
vm.swappiness=10                  # Réduire l'utilisation du swap
vm.vfs_cache_pressure=50          # Conserver le cache plus longtemps

# Optimisation réseau
net.ipv4.tcp_fin_timeout=30
net.ipv4.tcp_keepalive_time=300
net.core.netdev_max_backlog=5000

# Appliquer les changements
sysctl -p
```

3.4 Gestion des Packages

```
# Ubuntu/Debian
apt update && apt upgrade -y
apt autoremove
apt autoclean

# CentOS/RHEL
yum update -y
yum clean all
yum autoremove
```

4. Monitoring Continu

4.1 Outils de Monitoring

Solutions professionnelles :

Outil	Type	Usage
Nagios	Complet	Surveillance infrastructure
Zabbix	Complet	Monitoring avancé avec alertes
Prometheus	Métriques	Collecte de métriques
Grafana	Visualisation	Dashboards et graphiques
Netdata	Temps réel	Monitoring instantané
Glances	Système	Vue d'ensemble en terminal

4.2 Script de Surveillance

```
#!/bin/bash
# check_system.sh - Script de vérification quotidien

echo "===="
echo "Rapport Système - $(date)"
echo "===="

echo -e "\n==== Charge CPU ==="
uptime

echo -e "\n==== Utilisation Mémoire ==="
free -h

echo -e "\n==== Espace Disque ==="
df -h | grep -v tmpfs

echo -e "\n==== Top 5 Processus CPU ==="
ps aux --sort=-%cpu | head -6

echo -e "\n==== Top 5 Processus RAM ==="
ps aux --sort=-%mem | head -6

echo -e "\n==== Connexions Réseau ==="
ss -s

# Sauvegarder dans un fichier
# Utilisation: ./check_system.sh > /var/log/system_report_$(date +%Y%m%d).log
```

5. Optimisations Spécifiques

5.1 Base de Données

Optimisation MySQL/MariaDB :

```
# Configuration /etc/mysql/my.cnf
[mysqld]
innodb_buffer_pool_size = 2G          # 70% de la RAM disponible
max_connections = 200
query_cache_size = 64M
tmp_table_size = 64M
max_heap_table_size = 64M

# Analyser les requêtes lentes
slow_query_log = 1
long_query_time = 2

# Redémarrer MySQL
systemctl restart mysql
```

5.2 Serveur Web

Optimisation Nginx :

```
# /etc/nginx/nginx.conf
worker_processes auto;
worker_connections 2048;

# Compression
gzip on;
gzip_comp_level 5;
gzip_types text/plain text/css application/json application/javascript;

# Cache
proxy_cache_path /var/cache/nginx levels=1:2 keys_zone=my_cache:10m;

# Buffers
client_body_buffer_size 10K;
client_header_buffer_size 1k;
large_client_header_buffers 2 1k;
```

5.3 Sécurité

```
# Fail2ban - Protection contre brute-force
fail2ban-client status
fail2ban-client status sshd

# Firewall (UFW - Ubuntu)
ufw status
ufw allow 22/tcp
ufw allow 80/tcp
ufw allow 443/tcp
ufw enable

# Firewall (firewalld - CentOS)
firewall-cmd --list-all
firewall-cmd --add-service=http --permanent
firewall-cmd --reload

# Audit de sécurité
lynis audit system
```

6. Planification et Automatisation

6.1 Cron Jobs

Automatiser les tâches répétitives :

```
# Éditer le crontab  
crontab -e  
  
# Exemples de tâches planifiées  
  
# Sauvegarde quotidienne à 2h du matin  
0 2 * * * /usr/local/bin/backup.sh  
  
# Nettoyage des fichiers temporaires (tous les jours)  
0 3 * * * /usr/bin/find /tmp -type f -atime +7 -delete  
  
# Vérification système hebdomadaire (dimanche à 1h)  
0 1 * * 0 /usr/local/bin/check_system.sh > /var/log/weekly_check.log  
  
# Mise à jour des packages (1er du mois à 4h)  
0 4 1 * * apt update && apt upgrade -y  
  
# Rotation des logs personnalisés  
0 0 * * * /usr/sbin/logrotate /etc/logrotate.conf
```

6.2 Systemd Timers

Alternative moderne aux cron jobs :

```
# Lister les timers actifs  
systemctl list-timers  
  
# Créer un timer personnalisé  
# /etc/systemd/system/backup.timer  
[Unit]  
Description=Backup quotidien  
  
[Timer]  
OnCalendar=daily  
Persistent=true  
  
[Install]  
WantedBy=timers.target  
  
# Activer le timer  
systemctl enable backup.timer  
systemctl start backup.timer
```

7. Benchmarking

Tests de performance pour évaluer les capacités du système.

7.1 Tests CPU

```
# Sysbench - Test CPU
sysbench cpu --cpu-max-prime=20000 run

# Stress test
stress --cpu 4 --timeout 60s
```

7.2 Tests Disque

```
# Test d'écriture
dd if=/dev/zero of=/tmp/test.img bs=1G count=1 oflag=direct

# Test de lecture
dd if=/tmp/test.img of=/dev/null bs=1G count=1 iflag=direct

# FIO - Test avancé
fio --name=random-write --ioengine=posixaio --rw=randwrite --bs=4k \
--size=1g --numjobs=1 --runtime=60 --time_based
```

7.3 Tests Réseau

```
# iperf3 - Test de bande passante
# Serveur
iperf3 -s

# Client
iperf3 -c server_ip -t 30

# Ping et latence
ping -c 100 server_ip
mtr server_ip           # Traceroute amélioré
```

8. Gestion des Ports Réseau

Contrôler et gérer les ports utilisés sur le système.

8.1 Commandes de Vérification

```
# ss (moderne, remplace netstat)
ss -tuln          # Tous les ports en écoute
ss -tulnp         # Avec les processus (root requis)
ss -tan           # Toutes les connexions TCP
ss -tan | grep :443    # Vérifier un port spécifique

# netstat (ancienne méthode)
netstat -tuln      # Tous les ports en écoute
netstat -tulnp     # Avec les processus
netstat -an | grep LISTEN # Seulement les ports en écoute

# lsof (list open files)
lsof -i             # Toutes les connexions réseau
lsof -i :80          # Qui utilise le port 80
lsof -i TCP:22       # Port SSH spécifique
```

8.2 Scanner de Ports

```
# nmap - Scanner de ports
nmap localhost      # Scanner les ports locaux
nmap -p 1-65535 localhost # Tous les ports
nmap -p 80,443,22 localhost # Ports spécifiques

# netcat - Test de connexion
nc -zv localhost 80      # Tester si le port est ouvert
nc -zv google.com 443     # Test externe
```

8.3 Identification du Processus

```
# Trouver quel processus utilise un port
sudo lsof -i :3306
sudo ss -tulnp | grep :3306
sudo netstat -tulnp | grep :3306

# Avec fuser
sudo fuser 3306/tcp      # Affiche le PID

# Tuer un processus sur un port
sudo kill $(sudo lsof -t -i:8080)
sudo fuser -k 8080/tcp     # Force kill
```

8.4 Ports Standards

Port	Service	Description
22	SSH	Connexion sécurisée
80	HTTP	Web non-sécurisé
443	HTTPS	Web sécurisé (SSL/TLS)
3306	MySQL	Base de données MySQL
5432	PostgreSQL	Base de données PostgreSQL
6379	Redis	Cache en mémoire
27017	MongoDB	Base de données NoSQL

8080	HTTP Alt	Serveur web alternatif
9090	Prometheus	Monitoring
3000	Grafana	Visualisation

8.5 Gestion du Firewall

```
# Ubuntu (ufw)
sudo ufw status
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw deny 23/tcp      # Bloquer telnet
sudo ufw enable

# CentOS (firewalld)
sudo firewall-cmd --list-ports
sudo firewall-cmd --add-port=80/tcp --permanent
sudo firewall-cmd --add-service=http --permanent
sudo firewall-cmd --reload

# iptables
sudo iptables -L -n
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

Conclusion

Un administrateur système efficace combine tous ces outils et techniques pour maintenir un système Linux performant, stable et sécurisé. La surveillance proactive, l'optimisation continue et l'automatisation des tâches répétitives sont les clés du succès. Points clés à retenir :

- Surveillez régulièrement les ressources système (CPU, RAM, disque, réseau)
- Analysez les logs pour anticiper les problèmes
- Optimisez les services et les paramètres kernel selon vos besoins
- Mettez en place un système de monitoring continu
- Automatisez les tâches répétitives avec cron ou systemd timers
- Sécurisez votre système avec un firewall et fail2ban
- Effectuez des benchmarks pour évaluer les performances
- Documentez vos configurations et procédures

La maîtrise de ces compétences vous permettra de gérer efficacement des infrastructures Linux, qu'elles soient simples ou complexes, et d'assurer leur disponibilité et leurs performances optimales.

Ressources Additionnelles

- Documentation officielle : Ubuntu (help.ubuntu.com), CentOS (docs.centos.org) • Man pages : man command_name
- Communautés : Stack Overflow, Server Fault, Reddit ([r/linux](https://www.reddit.com/r/linux), [r/linuxadmin](https://www.reddit.com/r/linuxadmin))
- Livres recommandés : "UNIX and Linux System Administration Handbook", "Linux Performance Tools"
- Certifications : LPIC, RHCSA, RHCE