Engineer

Noyau

Cœur d'un système d'exploitation

Ce que c’est

Un noyau est le programme au centre d’un système d’exploitation qui s’exécute avec tous les privilèges du processeur et contrôle l’accès au matériel. Il décide du prochain programme à exécuter, distribue la mémoire et maintient celle de chaque programme séparée de celle des autres, lit et écrit sur les disques par les systèmes de fichiers, fait circuler les paquets dans la pile réseau et fait respecter les droits d’accès aux fichiers, ports et périphériques. Tout le reste fonctionne dans l’espace utilisateur avec moins de privilèges et demande au noyau d’agir pour lui.

Ces demandes sont les appels système, que le manuel Linux appelle « l’interface fondamentale entre une application et le noyau Linux ». Les programmes les effectuent rarement à la main : ils appellent des fonctions d’encapsulation dans une bibliothèque C comme glibc, qui transmettent les arguments et transforment la réponse du noyau en valeur de retour et en code d’erreur. Lorsqu’un programme s’arrête avec « Permission denied », ce libellé est généralement la réponse du noyau à l’un de ces appels, traduite par la bibliothèque.

Les noyaux diffèrent par ce qu’ils gardent à l’intérieur. Linux conserve les pilotes de périphériques, les systèmes de fichiers et la pile réseau dans le noyau lui-même, et peut en charger davantage sous forme de modules pendant son exécution : une conception généralement appelée monolithique. Chaque pilote s’exécute alors avec tous les privilèges du noyau. Un micronoyau ne garde que le minimum. seL4 fournit des mécanismes contrôlant l’accès à l’espace d’adressage physique, aux interruptions et au temps processeur, et exécute les pilotes, les systèmes de fichiers et les piles réseau comme des programmes ordinaires en mode utilisateur ; sa version vérifiée RISC-V sur 64 bits compte environ 10 000

lignes de code, avec une preuve formelle que l'implémentation C fait ce que sa spécification indique. Le noyau de macOS, XNU, est un hybride qui combine le micronoyau Mach à des parties de FreeBSD.

Une mise à jour du noyau ne prend normalement effet que lorsque la machine démarre dessus, et le numéro de version du noyau, par exemple 6.12, est distinct du numéro de version de la distribution qui l'entoure.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce qu'un noyau ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. L'entretien quotidien des serveurs, de la messagerie, des comptes et des mises à jour, pour que les outils indispensables restent sans histoire.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. 10 serveurs PostgreSQL et MS SQL sur Ubuntu — 10 serveurs de bases de données déployés et maintenus

10 serveurs PostgreSQL et MS SQL sur Ubuntu Contactez-nous →

Pourquoi c'est important

Pour une entreprise, le noyau tranche trois questions : l'isolation réelle de ses charges de travail, le coût d'un correctif de sécurité en indisponibilité et l'endroit où se trouve la vérité sur une machine.

D'abord, l'isolation. Un conteneur est un ensemble de processus ordinaires que le noyau a placés dans des espaces de noms, qui donnent à chaque groupe sa propre vue des identifiants de processus, interfaces réseau, montages et utilisateurs, et dans des groupes de contrôle, qui répartissent le temps processeur, la mémoire et les entrées-sorties disque. Tous les conteneurs d'un hôte utilisent son unique noyau : une faille de ce noyau affecte donc leur socle commun à tous, et une image qui apporte ses bibliothèques dépend encore des fonctions du noyau disponibles sur l'hôte. Une machine virtuelle démarre son propre noyau. Pour les services d'une équipe, les conteneurs sont une frontière raisonnable ; pour des clients qui ne se font pas confiance, attribuez à chaque client son propre noyau.

Ensuite, les correctifs. Un paquet de noyau corrigé sur disque ne change rien tant que la machine ne démarre pas dessus, et sur un serveur, ce redémarrage constitue le coût. L'application de correctifs à chaud existe parce que, selon la documentation du noyau, les utilisateurs sont « réticents à redémarrer » : elle redirige certaines fonctions vers des versions corrigées pendant que le système

tourne. Seules les fonctions que le noyau peut tracer peuvent être corrigées ainsi : les correctifs à chaud donnent donc du temps pour une urgence au lieu de remplacer les redémarrages. Un plan prévoyant un redémarrage régulier dont la procédure a été répétée coûte moins qu'un plan supposant qu'il n'arrivera jamais.

Enfin, le noyau détient l'état réel en exécution. Un fichier de pare-feu décrit des règles ; celles du filtre de paquets du noyau décident du sort d'un paquet. Un fichier dans `/etc/sysctl.d` décrit un réglage ; la valeur dans `/proc/sys` est celle en vigueur. Un service peut se déclarer sain sans que son action ait jamais atteint le noyau. Les contrôles qui lisent l'état propre du noyau détectent des échecs qu'une revue des fichiers de configuration laisserait passer.

À quoi cela ressemble en pratique

Une revue de durcissement d'un hôte de production, en août 2026, a demandé si ses protections contre la force brute SSH fonctionnaient. Chacune était configurée et figurait dans tous les rapports : la revue les a donc confrontées à ce que le noyau fait réellement d'un paquet, plutôt qu'aux fichiers décrivant ce qui aurait dû se passer. Vu du noyau, aucune des trois n'avait agi une seule fois depuis la création de l'hôte. Les bannissements du jail visaient le port 22, où aucun socket n'écoutait depuis le déplacement de SSH sur le port 1986. La limitation de débit atteignait bien le filtre de paquets, qui évalue les règles de haut en bas et s'arrête à la première correspondance : une règle plus large au-dessus absorbait donc tous les paquets avant que la limite soit consultée. L'action de bannissement lançait un programme de filtrage de paquets que son paquet déclare comme dépendance recommandée plutôt qu'obligatoire ; sur un hôte qui en était dépourvu, le jail démarrait, comptait et décidait, puis échouait au seul moment où il devait transmettre une règle au noyau. Deux règles en sont ressorties : un contrôle de sécurité se prouve par une assertion, ce qu'un commentaire ne peut pas faire, et un pare-feu se vérifie par l'ordre des règles dans le noyau, non par la simple existence d'une règle.

La procédure de bannissement a ensuite été couverte par un test exécuté à chaque application de la configuration. Chaque exécution bannit une adresse d'un bloc réservé à la documentation, relit la règle résultante dans le filtre de paquets du noyau, puis lève le bannissement dans une étape de nettoyage qui s'exécute que le test réussisse ou échoue. C'est la relecture qui en fait un test : l'état du service de bannissement lui-même annoncerait un succès pour un bannissement n'ayant produit aucune règle. Une adresse réservée n'appartient à

aucun réseau réel : un bannissement oublié après un échec ne bloque donc personne. Le coût : quelques secondes par exécution, et une règle de pare-feu écrite en production puis retirée à chaque fois.

Un hôte de 464 Mo sans place pour une pile de métriques a reçu sept rôles en lecture seule qui décrivent en Markdown l'état réel de la machine ; son relevé de données calcule l'utilisation du processeur à partir de deux mesures du compteur propre au noyau. Le rapport de sécurité affiche les règles du pare-feu dans l'ordre de correspondance, à côté des règles de bannissement actives, et chacun des sept rôles ne produit que les chiffres lus pendant sa propre exécution, dont les compteurs du noyau, jamais une estimation pour combler un manque. Ces rapports ont révélé deux contrôles de sécurité inactifs et un port à l'écoute que personne n'avait déclaré.

Où les problèmes surviennent

Un réglage du noyau modifié à la main revient à sa valeur précédente au prochain redémarrage. `sysctl -w` n'écrit la valeur que dans le noyau en cours d'exécution : elle ne dure que jusqu'au redémarrage. L'erreur inverse : un fichier sous `/etc/sysctl.d` est lu au démarrage, et la modification attend donc, sans l'annoncer, un redémarrage ultérieur. Écrivez le réglage dans un fichier, chargez-le maintenant et relisez la valeur active :

```
echo 'vm.swappiness = 10' | sudo tee /etc/sysctl.d/90-local.conf
sudo sysctl --system
sysctl vm.swappiness
```

Une courbe CPU reste plate pendant un pic de charge. La ligne `cpu` de `/proc/stat` compte le temps depuis le démarrage en unités USER_HZ — des centièmes de seconde sur la plupart des architectures. Une lecture unique donne donc la moyenne depuis le démarrage, qui bouge à peine sur une machine active depuis des mois. Lisez-la deux fois et divisez les différences :

```
head -1 /proc/stat; sleep 5; head -1 /proc/stat
```

Le temps occupé est la somme des variations de user, nice, system, irq et softirq ; le temps inactif est la somme des variations de idle et iowait ; l'utilisation est le temps occupé divisé par la somme des temps occupé et inactif. Le manuel

qualifie iowait de peu fiable : traitez-le comme un indice plutôt que comme une alerte.

Un rapport de plantage revient avec une demande de reproduction sur un noyau non contaminé. Le noyau se marque comme contaminé lorsqu'un événement peut compter pour une enquête ultérieure, par exemple le chargement d'un module propriétaire, extérieur à l'arbre du noyau ou non signé ; sa documentation prévient que les développeurs ignorent souvent les rapports de bogues de noyaux contaminés. La marque reste jusqu'au redémarrage, même après déchargement du module. Lisez-la avant d'envoyer le rapport :

```
cat /proc/sys/kernel/tainted
```

Zéro signifie non contaminé ; tout autre nombre est une somme d'indicateurs, dont `1` pour un module propriétaire, `4096` pour un module extérieur à l'arbre et `8192` pour un module non signé. Reproduisez le défaut après avoir démarré sans le module qui a posé l'indicateur.

Une nouvelle carte réseau apparaît dans la liste du matériel, mais le système n'offre aucune interface pour elle. Les pilotes font partie du noyau, et un noyau plus ancien que le matériel peut en manquer. `lspci -k` affiche le pilote utilisé par chaque périphérique et les modules capables de le gérer ; un périphérique sans ligne de pilote utilisé n'en a aucun associé. Installez un noyau d'une série plus récente fournie par la distribution plutôt que de compiler à la main le module du fabricant : un module extérieur à l'arbre des sources doit être recompilé pour chaque nouveau noyau, et son chargement contamine le noyau.

Pour aller plus loin

- Guide de l'utilisateur et de l'administrateur du noyau Linux — le manuel du projet du noyau lui-même pour exploiter Linux.
- syscalls(2) — chaque appel système Linux, avec la version du noyau qui l'a introduit.
- namespaces(7) — types d'espaces de noms et construction des conteneurs à partir d'eux.
- proc_stat(5) — chaque champ de `/proc/stat`, y compris l'avertissement sur iowait.
- Noyaux contaminés — chaque indicateur de contamination et la manière de décoder le nombre.

- FAQ seL4 — ce qu'un micronoyau laisse de côté et les hypothèses de la preuve de seL4.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le parcours de déploiement, les risques opérationnels et les personnes d'astreinte. Nous pouvons définir un changement avec des vérifications, une procédure de reprise et une responsabilité claire.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce qu'un noyau ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/kernel/