**Engineer**

# Durcissement

Supprimer ce qu'un attaquant pourrait exploiter

## Ce que c’est

Le durcissement consiste à configurer un système pour qu’il n’expose et ne permette que ce dont sa mission a besoin : moins de services en cours d’exécution, moins de ports en écoute, moins de privilèges par processus et des réglages prédéfinis plus stricts pour tout ce qui reste. Le logiciel est le même avant et après. Ce qui change, c’est ce qu’un intrus peut atteindre et jusqu’où un premier point d’appui lui permet d’aller. On parle aussi de durcissement du système, du serveur, du système d’exploitation ou de SSH, selon la couche renforcée.

Une intervention typique sur un serveur Linux touche plusieurs couches à la fois. La connexion distante accepte les clés et refuse les mots de passe. Le compte root ne peut pas se connecter directement. Les paquets inutilisés sont supprimés, plutôt que simplement arrêtés. Chaque service de longue durée reçoit son propre utilisateur sans privilèges, une vue en lecture seule du système de fichiers et une liste des appels système qu’il peut effectuer. Le pare-feu admet les quelques ports nécessaires aux services de l’hôte. Rien de tout cela n’est exotique ; la discipline consiste à le faire entièrement, sur chaque hôte, puis à maintenir ces conditions.

Il est utile de situer le durcissement parmi les pratiques avec lesquelles on le confond souvent. Les correctifs réparent une faille précise du code lorsqu’une correction existe. La supervision et la détection d’intrusions repèrent une attaque en cours. Un pare-feu décide qui peut atteindre un port. Le durcissement décide de ce que le programme derrière ce port pourra faire une fois atteint, et si le port devrait être ouvert. Les référentiels publiés, tels que les CIS Benchmarks, en font des listes de contrôle par système d’exploitation et par produit : utiles comme point de départ minimal, trompeuses comme ligne d’arrivée.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce que le durcissement ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Des comptes, mots de passe et droits attribués selon les besoins exacts de chaque poste, puis retirés le jour du départ.

**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. 1 000 règles RBAC pour sécuriser Linux — 1,000 règles d'accès écrites pour combler une faille de sécurité

**1 000 règles RBAC pour sécuriser Linux** Contactez-nous →

## Pourquoi c’est important

La vraie décision est de savoir à quoi consacrer un budget de sécurité limité. Le durcissement a cette particularité de protéger contre des vulnérabilités qui ne seront découvertes que l’année suivante. Un correctif protège contre une vulnérabilité connue ; un service durci limite ce qu’un processus compromis peut faire. Par exemple, lui interdire d’écrire dans une mémoire exécutable, d’ouvrir des sockets brutes ou d’accéder aux répertoires personnels vise à limiter les conséquences d’une faille d’exécution de code à distance. Les correctifs ferment les brèches ; le durcissement réduit ce que vaut chacune. Une exploitation sérieuse a besoin des deux, et le durcissement fait gagner du temps entre la divulgation et l’arrivée du correctif.

Le coût est surtout du temps d’ingénierie et un risque de panne, plutôt que des licences. Chaque restriction repose sur le pari que le service n’a jamais besoin de ce qu’on lui retire ; certains paris sont perdus : une sauvegarde exécutée chaque nuit, une rotation des journaux qui envoie un signal, un plugin qui lance une commande shell. C’est pourquoi le durcissement manuel d’un serveur en service tend à s’arrêter. Une restriction casse quelque chose à trois heures du matin, et la personne échaudée cesse de renforcer les protections. Décrit dans du code, avec des restrictions relues, versionnées et appliquées à l’identique à chaque reconstruction, le même travail devient peu coûteux à répéter, avec un retour sûr à la version précédente.

Pour un acheteur, les questions pratiques sont de savoir quand le faire et comment vérifier que cela a été fait. La première réponse est lors de la construction de l’hôte, dans l’automatisation qui le crée : une liste appliquée une seule fois à une machine en service ne correspond plus à la réalité quelques mois plus tard. La seconde est d’apporter des preuves issues du système en cours d’exécution, plutôt que du dépôt de configuration. Les auditeurs demandent de plus en plus les deux : un socle documenté et la preuve que la production y correspond.

# À quoi cela ressemble en pratique

Sur l’hôte de production de l’entreprise, SSH est la seule voie d’accès interactive, et l’automatisation écrit sa configuration. En septembre 2026, un rapport d’accès a relu les réglages effectivement retenus par le démon et trouvé deux directives prévues supplantées : les valeurs prédéfinies du système d’exploitation étaient chargées avant le fichier complémentaire de l’entreprise, et la première valeur l’emporte pour ces réglages. Un préfixe de nom de fichier classé plus tôt a corrigé l’ordre. Le changement durable a été la validation en trois étapes de 24 directives SSH à chaque convergence : chaque proposition doit être syntaxiquement valide avant de remplacer le fichier actif, la configuration assemblée doit passer un contrôle une fois installée, puis les réglages réellement retenus par le démon sont relus et vérifiés pour les 24 directives. Deux autres directives ont été volontairement exclues, car le démon ne les implémente plus.

L’automatisation installe aussi 13 fichiers d’unité sur ce même hôte. Les unités correspondantes fonctionnaient jusque-là avec les privilèges initiaux du gestionnaire de services. Chaque modèle d’unité a reçu un plafond de privilèges, un filtre d’appels système, des familles d’adresses restreintes et une mémoire qui peut être inscriptible ou exécutable, mais jamais les deux. Selon les scores du superviseur lui-même, l’isolation a fait passer un service de veille à signal de vie de 9,6 à 1,5, la forge Git exposée à Internet de 8,3 à 1,5, et les deux unités de sauvegarde de 9,6 à 2,3 et 2,5. Pendant leur rédaction, une directive s’est révélée mal orthographiée dans trois modèles ; le superviseur journalise une clé inconnue et démarre quand même l’unité. Chaque unité passe désormais par le vérificateur du superviseur pendant la convergence.

Une revue de durcissement en août 2026 a cherché à savoir si les protections SSH contre la force brute fonctionnaient, en lisant le système en cours d’exécution. Les trois protections étaient inertes depuis la création de l’hôte : le mécanisme de bannissement (jail) surveillait le port 22 alors que le démon écoutait sur le 1986, une limite de débit du pare-feu figurait sous une règle plus large qui s’appliquait d’abord, et l’action de bannissement appelait un binaire de filtrage de paquets simplement recommandé par le gestionnaire de paquets et qui était absent. Deux règles en sont issues et régissent toujours le dépôt : un contrôle de sécurité reçoit une assertion, pas un commentaire, et un pare-feu se vérifie par la position de ses règles, pas par leur présence.

# Où les problèmes surviennent

**Le réglage est dans le fichier, le service l'ignore.** De nombreux démons lisent plusieurs fichiers et conservent la première ou la dernière valeur rencontrée pour chaque clé. Une directive de durcissement dans un fichier complémentaire peut céder devant une valeur du fournisseur chargée plus tôt. Demandez au programme ce qu'il fera réellement, avec `sshd -T` pour SSH, et comparez sa sortie aux valeurs prévues à chaque exécution.

**Une faute de frappe transforme un contrôle en commentaire.** Les gestionnaires de services et de nombreux démons journalisent une clé inconnue et poursuivent. L'unité démarre, le tableau de bord est vert et la restriction est absente. Exécutez le vérificateur de l'analyseur pendant le déploiement pour qu'une directive inconnue arrête la mise en production.

```
# What the SSH daemon will do once every Include is resolved
sshd -T | grep -E '^(passwordauthentication|permitrootlogin) '
# How much a unit's sandbox still exposes, scored 0.0 to 10.0
systemd-analyze security --no-pager example.service
# Unknown directives and other unit-file errors
systemd-analyze verify /etc/systemd/system/example.service
```

**Une protection garde la mauvaise porte.** Un port change, mais la règle de bannissement continue de surveiller l'ancien. Une limite de débit est ajoutée sous une autorisation large déjà appliquée. Le contrôle se dit sain parce qu'il s'exécute, ce qui ne prouve pas qu'il se déclenche. Testez chaque contrôle par son effet : déclenchez-le volontairement depuis l'extérieur et vérifiez que le paquet est abandonné.

**L'isolation casse le service pendant la nuit.** Un filtre d'appels système ou un chemin en lecture seule casse un chemin de code rare, tel que l'export hebdomadaire ou le hook de sauvegarde. La panne apparaît des heures après le changement, à l'exécution plutôt qu'à l'analyse syntaxique. Renforcez les restrictions par étapes, testez chaque tâche planifiée après chaque étape et surveillez dans le journal les processus tués par le filtre.

**Le score du référentiel devient l'objectif.** Un hôte peut satisfaire chaque ligne d'une liste générique alors que son unique application exposée à Internet s'exécute en root et peut écrire dans son propre code. Partez de ce qui est

accessible depuis le réseau et de ce que chaque processus accessible pourrait atteindre, puis utilisez le référentiel pour trouver ce que cette revue a manqué.

## Pour aller plus loin

- Durcissement, dans le glossaire du NIST — La définition du NIST et la publication dont elle est tirée.
- sshd_config(5) — Chaque directive du serveur SSH et la règle selon laquelle la première valeur obtenue pour un mot-clé est utilisée.
- systemd.exec(5) — Directives d'isolation des services : filtres d'appels système, familles d'adresses et mémoire inscriptible ou exécutable.
- systemd-analyze(1) — Score d'exposition de la commande `security` et commande `verify`, qui détecte les directives inconnues.
- CIS Benchmarks — Socles de configuration consensuels pour systèmes d'exploitation, plateformes cloud et logiciels serveur.



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 que le durcissement ? » 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/hardening/**