RECOMMANDATIONS DE CONFIGURATION D'UN SYSTÈME GNU/LINUX

GUIDE ANSSI

ANSSI-BP-028
03/10/2022

PUBLIC VISÉ :

Développeur **Administrateur** **RSSI** DSI Utilisateur

Informations

Attention

Ce document rédigé par l'ANSSI présente les **« Recommandations de configuration d'un système GNU/Linux »**. Il est téléchargeable sur le site `www.ssi.gouv.fr`.
Il constitue une production originale de l'ANSSI placée sous le régime de la « Licence Ouverte v2.0 » publiée par la mission Etalab [15].
Conformément à la Licence Ouverte v2.0, le guide peut être réutilisé librement, sous réserve de mentionner sa paternité (source et date de la dernière mise à jour). La réutilisation s'entend du droit de communiquer, diffuser, redistribuer, publier, transmettre, reproduire, copier, adapter, modifier, extraire, transformer et exploiter, y compris à des fins commerciales.
Sauf disposition réglementaire contraire, ces recommandations n'ont pas de caractère normatif ; elles sont livrées en l'état et adaptées aux menaces au jour de leur publication. Au regard de la diversité des systèmes d'information, l'ANSSI ne peut garantir que ces informations puissent être reprises sans adaptation sur les systèmes d'information cibles. Dans tous les cas, la pertinence de l'implémentation des éléments proposés par l'ANSSI doit être soumise, au préalable, à la validation de l'administrateur du système et/ou des personnes en charge de la sécurité des systèmes d'information.

Évolutions du document :

VERSION	DATE	NATURE DES MODIFICATIONS
1.0	08/10/2015	Version initiale
1.1	12/08/2016	Révision mineure
1.2	22/02/2019	Révision mineure et passage au nouveau modèle ANSSI
2.0	03/10/2022	Intégration des éléments de durcissement du noyau Linux et réorganisation des chapitres

Table des matières

1 Préambule **4**
- 1.1 Avant-propos . . . 4
- 1.2 Niveau de durcissement . . . 4

2 Menaces et objectifs des attaquants **6**

3 Principes généraux de sécurité et de durcissement **8**
- 3.1 Principe de minimisation . . . 8
- 3.2 Principe de moindre privilège . . . 8
- 3.3 Principe de défense en profondeur . . . 9

4 Configuration matérielle **10**
- 4.1 Support matériel . . . 10
- 4.2 BIOS/UEFI . . . 10
- 4.3 Démarrage sécurisé UEFI . . . 11
  - 4.3.1 Clés préchargées . . . 11
  - 4.3.2 Nouvelles clés . . . 12
- 4.4 Démarrage de confiance . . . 13

5 Configuration du noyau Linux **14**
- 5.1 Chargeur de démarrage . . . 14
- 5.2 Configuration dynamique . . . 15
  - 5.2.1 Configuration de la mémoire . . . 16
  - 5.2.2 Configuration du noyau . . . 19
  - 5.2.3 Configuration des processus . . . 20
  - 5.2.4 Configuration du réseau . . . 21
  - 5.2.5 Configuration des systèmes de fichiers . . . 23
- 5.3 Configuration statique . . . 23
  - 5.3.1 Configuration indépendante de la cible matérielle . . . 24
  - 5.3.2 Configuration spécifique aux architectures matérielles . . . 30

6 Configuration système **33**
- 6.1 Partitionnement . . . 33
- 6.2 Comptes d'accès . . . 35
  - 6.2.1 Comptes utilisateur . . . 35
  - 6.2.2 Comptes administrateur . . . 36
  - 6.2.3 Comptes de service . . . 38
- 6.3 Contrôle d'accès . . . 38
  - 6.3.1 Modèle traditionnel Unix . . . 39
  - 6.3.2 `AppArmor` . . . 44
  - 6.3.3 `SELinux` . . . 45
- 6.4 Fichiers et répertoires . . . 49
  - 6.4.1 Fichiers et répertoires sensibles . . . 49
  - 6.4.2 Fichiers IPC nommés, *sockets* ou *pipes* . . . 50

6.4.3 Droits d'accès 51
6.5 Gestion des paquets 54
6.6 Veille et maintenance 55

7 Configuration des services **57**
7.1 Cloisonnement 59
7.2 Services système 60
7.2.1 `Pluggable Authentication Module` ou module d'authentification enfichable 61
7.2.2 `Name Service Switch` ou service de gestion de noms 63
7.2.3 Journalisation 64
7.2.4 Messagerie 67
7.2.5 Surveillance du système de fichiers 68
7.3 Services réseau 69

Liste des recommandations **71**

Annexe A Patchs au noyau Linux **74**

Annexe B Conformité de la configuration **77**

Annexe C Évolutions du guide **78**
C.1 Nouvelles recommandations 78
C.2 Mises à jour entre les versions 1.2 et 2.0 78
C.3 Matrice de rétrocompatibilité 79

Bibliographie **82**

1
Préambule

1.1 Avant-propos

Aujourd'hui les systèmes d'exploitation Unix et dérivés, et notamment les distributions GNU/Linux, jouent un rôle important dans l'écosystème des équipements, systèmes, réseaux et télécommunications. Ils sont en effet souvent déployés dans de nombreux produits (commutateurs, routeurs, téléviseurs, véhicules…)

Leur diversité et leur composition font qu'ils sont utilisés suivant un grand nombre de combinaisons possibles. Il n'est pas utile ici d'aborder dans le détail chacun des différents cas d'usage. Des règles de configuration permettent cependant d'obtenir des systèmes raisonnablement sûrs du moment que certains principes fondamentaux sont respectés, et de vérifier méthodologiquement qu'elles sont correctement appliquées par exemple à l'aide d'une liste de vérification.

Le présent guide s'adresse aux administrateurs de système et concepteurs de produits reposant sur Linux souhaitant renforcer la sécurité de leurs systèmes. Il se concentre principalement sur des directives de configuration système génériques et des principes de bon sens qu'il convient d'appliquer lors du déploiement de services sur un système GNU/Linux. La configuration de ces services eux-mêmes peut faire l'objet d'une note technique dédiée, par exemple les *Recommandations pour la sécurisation des sites* `Web` [5] ou les *Recommandations pour un usage sécurisé d'OpenSSH* [3] sur le site de l'ANSSI. Pour étudier de manière plus approfondie le cloisonnement, abordé ici de manière superficielle, le lecteur pourra aussi se référer au guide *Recommandations pour la mise en place de cloisonnement système* [8].

Il convient d'étudier l'applicabilité et la maintenabilité de chaque recommandation au cas d'usage considéré. Il est par ailleurs vivement conseillé d'avoir recours aux compétences d'un expert en système GNU/Linux pour la mise en œuvre de ces bonnes pratiques.

1.2 Niveau de durcissement

Les distributions GNU/Linux étant très hétérogènes, la maîtrise du socle système est une tâche complexe ; une expertise devient réellement nécessaire au fur et à mesure que le nombre de services et de serveurs augmente. Cependant certaines mesures de durcissement peuvent être mises en place en fonction du niveau de sécurité attendu, qui va dépendre de la sensibilité des données manipulées ou hébergées par le système et de l'exposition dudit système.

Les recommandations de ce guide sont données en fonction d'un niveau de durcissement relatif. Ce niveau ne dépend pas forcément de la difficulté et du temps requis pour déployer une recomman-

dation donnée, choses qui ne peuvent être appréciées qu'après un audit dans un contexte donné. Ces niveaux doivent être vus comme des fils conducteurs pour aider à l'administration système.

Sauf particularité, tous les systèmes doivent implémenter les recommandations de niveau minimal. L'application des recommandations de niveau intermédiaire doit ensuite être recherchée. Les recommandations de niveau renforcé et élevé ne sont quant à elles pas à appliquer systématiquement. Celles-ci peuvent n'être rentables que pour des systèmes à besoin de sécurité plus important. De plus, celles-ci peuvent nécessiter un personnel qualifié et un effort important pour leur mise en œuvre correcte et leur maintien dans le temps, sous peine de devenir contre productives. En revanche, l'application des recommandations de niveau élevé et renforcé pourra apporter un gain important de sécurité et sera nécessaire dans certains contextes.

À titre d'exemple, le durcissement du noyau sera très bénéfique pour un système hébergeant des conteneurs car ceux-ci se reposent sur les fonctionnalités du noyau pour leurs primitives de cloisonnement. Par contre, un noyau non mis à jour régulièrement par manque de ressources dégradera considérablement le niveau de sécurité du système, l'exposant aux nombreuses vulnérabilités trouvées et dévoilées chaque mois.

Niveau	Description
	Recommandation de niveau **minimal**. À mettre en œuvre systématiquement sur tout système.
	Recommandation de niveau **intermédiaire**. A mettre en œuvre dès que possible sur la plupart des systèmes dès que les recommandations de niveau minimal sont appliquées.
	Recommandation de niveau **renforcé**. A mettre en œuvre sur des systèmes ayant un fort besoin de sécurité ou ayant plusieurs applications à isoler les unes des autres sur le même système.
	Recommandation de niveau **élevé**. A ne mettre en œuvre que si les ressources internes ont les compétences et le temps nécessaire pour assurer leur maintien régulier sous peine de dégrader la sécurité du système. Celles-ci peuvent cependant apporter un gain de sécurité important.

Table 1 – Grille de lecture des niveaux de durcissement

En fonction du niveau de durcissement retenu, les recommandations s'appliquent du niveau minimal jusqu'au niveau envisagé.

2
Menaces et objectifs des attaquants

Au premier abord, un système d'exploitation tel que Linux ne paraît pas être directement exposé aux menaces de la même manière que, par exemple, une application web puisse l'être. Cependant, tous les services, aussi exposés soient-ils, sont dépendants du système sous-jacent pour effectuer certaines opérations telles que la lecture et écriture sur le disque ou l'utilisation des périphériques réseau…). Les services exposent donc indirectement le système d'exploitation à un attaquant via leur fonctionnement usuel ou via une vulnérabilité.

Les systèmes d'exploitation ont donc développé différents mécanismes pour se protéger eux-mêmes contre des applications ou utilisateurs malveillants, mais également pour séparer les ressources des différentes applications et limiter l'impact d'une vulnérabilité.

Les menaces étant extrêmement variées, nous les classerons ici par objectif.

- **La disponibilité**, ou plutôt le fait de rendre indisponible le système ciblé est l'objectif le plus simple à atteindre pour un attaquant. La surcharge d'une des ressources (réseau, système de fichier, mémoire…) peut impacter la totalité du système. L'exploitation de certaines vulnérabilités peut également paralyser une ressource, par exemple en forçant une application à redémarrer en permanence.

  L'objectif d'un attaquant ici peut être d'empêcher l'usage d'une application métier, mais également de gêner le bon fonctionnement des fonctions de sécurité d'un système d'information. Un déni de service du serveur d'agrégation de log empêchera par exemple la réception des logs des machines du parc informatique, cachant l'activité malveillante d'un attaquant et prévenant toute remédiation.

- **La confidentialité** ou le vol de données est un autre objectif des attaquants. Le vol de données métier (bases de données, fichiers…) ou l'usurpation d'identité peut avoir un intérêt lucratif direct. Enfin, le vol de comptes sur les machines et l'élévation de privilèges peut permettre à un attaquant de se latéraliser et d'accéder à de nouvelles ressources. La machine initialement corrompue sert alors de passerelle pour atteindre des systèmes non exposés et plus critiques.

- **L'intégrité** du système est le dernier objectif des attaquants. Son altération permet à un attaquant de réutiliser les ressources à son propre profit. Il peut ainsi défigurer des applications exposées pour faire passer un message, utiliser les ressources de calcul pour du minage de cryptomonnaies, ou encore s'en servir comme passerelle pour mener des attaques de manière anonyme.

  La modification du système implique également souvent la mise en place de moyens de persistance. Ceux-ci permettent à l'attaquant de revenir sur le système pour modifier leur malware en fonction de leur besoin ou de le redéposer si celui-ci a été détecté et supprimé. Ces moyens de persistance peuvent être mis en place à tout niveau (applicatif, noyau, chaîne de démarrage…).

Exemple

Certaines menaces combinent plusieurs des objectifs décrits ci-dessus. On prendra pour exemple les rançongiciels. Ceux-ci peuvent exfiltrer les données des machines d'un parc informatique (vol de données), puis ils chiffreront les disques des machines (disponibilité).

La protection contre ces menaces passe à la fois par le durcissement des applications et du système d'exploitation, par leur maintien en condition de sécurité, par la mise en place de mécanismes permettant de détecter les tentatives d'attaques et par l'organisation régulière de tests d'intrusion et d'audits de sécurité.

3
Principes généraux de sécurité et de durcissement

Objectif

Guider l'installation et la configuration d'un système d'exploitation au travers de différents principes essentiels.

3.1 Principe de minimisation

Ce principe indique que les systèmes conçus et installés doivent éviter autant que possible toute complexité inutile en vue de :

- réduire la surface d'attaque au strict minimum ;
- permettre une mise à jour et un suivi des systèmes efficace ;
- rendre l'activité de surveillance des systèmes plus accessible, dans la mesure où le nombre de composants à surveiller est réduit.

La mise en œuvre de ce principe est parfois délicate car il peut se retrouver rapidement en contradiction avec d'autres, tout aussi importants. Seule une étude de cas avec l'aide d'une expertise système et sécurité permettra de faire des choix raisonnables. Les chapitres suivants donneront des recommandations ciblées suivant les parties envisagées du système.

3.2 Principe de moindre privilège

Ce principe définit que tout objet ou entité gérée par un système ne dispose que des droits strictement nécessaires à son exécution, et rien de plus.

L'objectif est à la fois un gain en sécurité et sûreté :

- les conséquences de dysfonctionnements ou vulnérabilités sont limitées aux privilèges octroyés ;
- l'altération ou la compromission du système nécessitent une escalade de privilèges, moins triviale et discrète à réaliser dans les cas où plusieurs couches de protection sont mises en place.

3.3 Principe de défense en profondeur

Ce principe impose la conception de plusieurs couches de sécurité indépendantes et complémentaires en vue de retarder un attaquant dont l'objectif est la compromission du système.

Chaque couche de sécurité est donc un point de résistance supplémentaire que l'attaquant doit franchir. La mise en défaut d'une couche s'accompagne de signaux, d'alarmes ou d'évènements journalisés permettant de détecter une activité suspecte et de pouvoir y réagir. L'étape de remédiation se trouve aussi facilitée grâce aux informations supplémentaires agrégées sur le contexte de la compromission.

Ce principe a donc un réel avantage : détection, facilité de remédiation, et amélioration de la sécurité.

Sous les systèmes d'exploitation Unix et dérivés, la défense en profondeur doit reposer sur une combinaison de barrières qu'il faut garder indépendantes les unes des autres.

Exemple

- authentification nécessaire avant d'effectuer des opérations, notamment quand elles sont privilégiées ;
- journalisation centralisée d'évènements au niveau système et service ;
- utilisation préférentielle de services qui implémentent des mécanismes de cloisonnement ou de séparation de privilèges.

4
Configuration matérielle

Objectif

Paramétrer certaines options matérielles afin de durcir le socle qui va servir à l'installation. Ce paramétrage doit être fait de préférence avant l'installation pour que le système soit capable d'en tenir compte le plus tôt possible.

4.1 Support matériel

Les recommandations de la note technique « *Recommandations de configuration matérielle de postes clients et serveurs x86* » [2] s'appliquent aux architectures x86. L'approche reste cependant applicable à d'autres architectures à ceci près que les mécanismes et directives de configuration seront susceptibles d'être différents.

R1 M I R

Choisir et configurer son matériel

Il est conseillé d'appliquer les recommandations du support matériel mentionnées dans la note technique « *Recommandations de configuration matérielle de postes clients et serveurs x86* » [2].

4.2 BIOS/UEFI

Le BIOS (`Basic Input/Output System` ou système entrée/sortie basique) ou son pendant moderne l'UEFI (`Unified Extensible Firmware Interface` ou interface micrologicielle extensible unifiée) est l'interface principale de configuration matérielle du système. Cette interface n'est souvent accessible que lors des premiers instants du démarrage de la machine au travers de la frappe d'une combinaison de touches.

La configuration matérielle de la machine dépend plus de l'usage qui en sera fait que du système d'exploitation installé. Il est cependant nécessaire de préciser que la désactivation ou l'activation de fonctionnalités au niveau du BIOS ou UEFI peut bloquer l'utilisation de celles-ci par le système. La note technique « *Recommandations de configuration matérielle de postes clients et serveurs x86* » [2] aborde les différentes options que l'on peut trouver sur une machine x86 contemporaine.

R2

Configurer le BIOS/UEFI

Il est conseillé d'appliquer les recommandations de la configuration du BIOS/UEFI mentionnées dans la note technique « *Recommandations de configuration matérielle de postes clients et serveurs x86* » [2].

4.3 Démarrage sécurisé UEFI

Le démarrage sécurisé UEFI (ou `Secure Boot` UEFI) est un mécanisme de vérification du code chargé par l'UEFI. Il est conçu pour empêcher l'exécution de code non signé par la machine.

Les codes chargés par l'UEFI peuvent être :

- des chargeurs de démarrage,
- des binaires de mises à jour du micrologiciel UEFI ou
- une image d'un noyau Linux au format `EFI` en utilisant l'`EFI Boot Stub` [1].

Le fonctionnement du démarrage sécurisé UEFI repose sur un mécanisme de signatures et d'empreintes : un code dont la signature ou l'empreinte n'est pas reconnue par l'UEFI ou une des entités de vérification de signature de la chaîne de démarrage ne peut pas être chargé.

Information

Le mécanisme de vérification du démarrage sécurisé UEFI (`UEFI Specification Version 2.3.1`) repose sur les 4 types de variables suivants :

- une base de données **db** (`Authorized Signature database`) qui contient la liste des signatures, des clés publiques ou des empreintes des codes autorisés à être chargés,
- une base de données **dbx** (`Revoked Signature database`) qui contient la liste des signatures, des clés publiques ou des empreintes révoqués des codes non autorisés,
- une ou plusieurs clés **KEK** (`Key Exchange Key` ou `Key Enrollment Key`) qui permettent de vérifier la signature des bases de données `db` et `dbx`,
- une clé **PK** (`Platform Key`) qui verrouille et sécurise le micrologiciel UEFI contre les modifications non autorisées. Cette clé gère notamment les mises à jour des clés `KEK` et elle est généralement fournie par le fabricant de la machine.

4.3.1 Clés préchargées

Dans le cadre du démarrage sécurisé, un `SHIM` est une application EFI triviale qui, quand exécutée, va charger et exécuter une autre application. Il agit en amont du chargeur de démarrage sur les systèmes UEFI. Sa signature est vérifiée avec les clés chargées dans l'UEFI. Un `SHIM` permet d'éviter de faire signer individuellement chaque code chargé.

1. https://www.kernel.org/doc/html/latest/admin-guide/efi-stub.html

Information

La plupart des machines x86 sont préchargées avec des clés `KEK Microsoft`. L'UEFI de ces machines autorise donc le chargement de codes signés par le service d'homologation UEFI hébergé par `Microsoft`. Un `SHIM` signé par ce service d'homologation est développé par les mainteneurs de certaines distributions, par exemple les mainteneurs des distributions dérivées de `Red Hat` [2] ou de `Debian` [3] qui sont en charge ensuite de signer chaque noyau Linux de la distribution et le chargeur de démarrage utilisé.

Une fois chargé, un `SHIM` vérifie les codes chargés non plus avec les clés préchargées mais avec les clés intégrées au `SHIM` au moment de sa compilation, par exemple, la clé publique de la CA (`Certificate Authority` ou autorité de certification) de clés de la distribution.

R3

Activer le démarrage sécurisé UEFI

Il est recommandé d'activer la configuration du démarrage sécurisé UEFI associée à la distribution.

4.3.2 Nouvelles clés

En l'absence de mise à jour des clés préchargées, toute vulnérabilité d'une version signée d'un `SHIM` ou d'un chargeur de démarrage peut permettre à un attaquant de contourner le démarrage sécurisé UEFI. On peut citer par exemple la vulnérabilité `BootHole`, CVE-2020-10713, d'un chargeur de démarrage signé par les mainteneurs de la distribution.

Les clés préchargées peuvent être remplacées par de nouvelles clés. Dans ce cas, l'utilisation d'un `SHIM` n'est plus nécessaire. Il est alors possible, au choix :

- de signer séparément le chargeur de démarrage et le noyau Linux ;
- de générer et signer une image du noyau Linux au format `EFI` qui sera vérifiée et chargée directement par l'UEFI sans passer par un chargeur de démarrage.

R4

Remplacer les clés préchargées

Il est recommandé de remplacer les clés préchargées dans l'UEFI par de nouvelles clés avec lesquelles seront signés :

- le chargeur de démarrage et le noyau Linux ou
- l'image du noyau Linux au format `EFI`.

Information

Pour protéger les paramètres de la ligne de commande du noyau détaillés au paragraphe 5.2 ou pour charger un noyau Linux reconstruit tel que détaillé au paragraphe 5.3, il est nécessaire de remplacer les clés préchargées par de nouvelles.

2. https://github.com/rhboot/shim
3. https://packages.debian.org/source/stable/shim-signed

Il est alors important de mettre en place une IGC (Infrastructure de gestion de clés) ou PKI (`Public Key Infrastructure`).

4.4 Démarrage de confiance

Le démarrage de confiance, ou `Measured Boot`, est une technique complémentaire du démarrage sécurisé `Secure Boot`. En effet, ce dernier assure que les binaires UEFI de la chaîne de démarrage sont tous signés, mais permet donc de démarrer plusieurs versions du système (ancienne version, version parallèle signée…). Par contraste, le démarrage de confiance s'appuie sur plus de paramètres et permet de s'assurer que le système actuellement démarré est dans un état précis attendu.

Pour cela, le démarrage de confiance s'appuie sur le TPM (`Trusted Platform Module`) et ses PCR (`Platform Configuration Registers` ou registres de configuration de plateforme). Lors du démarrage, les différents éléments constituant la chaîne de démarrage (firmware, binaires UEFI, état du démarrage sécurisé…) sont mesurés, et leurs hachés sont stockés dans les PCR du TPM. Cet ensemble de valeurs représente donc de façon unique l'état du système actuellement démarré. Un secret peut ensuite être scellé dans le TPM avec une politique de descellement basée sur un ensemble de PCR et leurs valeurs. Au prochain redémarrage, le secret sera descellé par le TPM à la seule condition que les valeurs des PCR du système démarré correspondent aux valeurs décrites dans la politique associée au secret. Autrement dit, il faut que l'état du système soit le même que celui choisi lors du scellement, assurant que le démarrage n'a pas été compromis.

Ce secret peut être utilisé, par exemple, comme clef de chiffrement d'un disque. En cas de démarrage compromis, le démarrage de confiance n'interrompra pas le démarrage, mais le TPM ne descellera pas le secret, empêchant le déchiffrement du disque.

Information

Chaque PCR permet la mesure d'un ou plusieurs composants lors du démarrage. Les huit premiers PCR (`PCR{0..7}`) sont utilisés lors de la chaîne de démarrage. La définition des mesures effectuées pour chaque PCR est définie par le `Trusted Computing Group`[4], et certaines mesures diffèrent en fonction de l'architecture matérielle.

Attention

La mise à jour d'un élément de la chaîne de démarrage qui fait l'objet d'une mesure altère la valeur de son haché. Au prochain redémarrage, les valeurs mesurées ne correspondront donc plus aux valeurs de la politique de descellement du secret. Il convient donc de s'assurer que les valeurs de la politique sont également mises à jour. Cependant, certains de ces changements ne peuvent n'être observés qu'au redémarrage de la machine et changeront donc l'état du système. Par exemple, la mise à jour du firmware d'un périphérique ne sera appliquée qu'au prochain redémarrage, rendant la mise à jour de la politique délicate.

4. https://trustedcomputinggroup.org/work-groups/pc-client

5
Configuration du noyau Linux

Le noyau Linux est un noyau riche, en charge de fournir à la fois l'ensemble des pilotes logiciels de la plateforme[5]*, un sous-ensemble de l'interface* `POSIX` *(complétée par la GNU* `libc`*) et tout un ensemble de mécanismes de sécurité pour se protéger et pour protéger le système.*

Objectif

Le durcissement du noyau Linux consiste à accroître les mécanismes de protection du système d'exploitation et à valider la présence des mécanismes d'auto-protection du noyau, en suivant le principe de défense en profondeur. On peut à la fois apporter des protections ou des contre-mesures à des vulnérabilités logicielles ou matérielles potentielles de la plateforme et réduire dans le même temps la surface d'attaque du noyau, initialement très grande.

5.1 Chargeur de démarrage

Pour des raisons équivalentes à la configuration du BIOS/UEFI, le chargeur de démarrage communément appelé `bootloader` est un élément important de la chaîne de démarrage. Ceux d'aujourd'hui (`GNU GRUB 2`[6], `systemd-boot`[7] …) sont fonctionnellement riches : ils permettent d'accéder aux systèmes de fichiers, de modifier éventuellement les données, de démarrer sur des périphériques externes ou de changer les options de démarrage du noyau sélectionné.

R5

Configurer un mot de passe pour le chargeur de démarrage

Un chargeur de démarrage permettant de protéger son démarrage par mot de passe est à privilégier. Ce mot de passe doit empêcher un utilisateur quelconque de modifier ses options de configuration.
Quand le chargeur de démarrage n'offre pas la possibilité de lui adjoindre un mot de passe, une autre mesure technique ou organisationnelle doit être mise en place afin de bloquer toutes tentatives de modification des options de configuration par un utilisateur non-autorisé.

5. à quelques exceptions près, comme par exemple la `libusb` en espace utilisateur.
6. https://www.gnu.org/software/grub
7. https://www.freedesktop.org/software/systemd/man/systemd-boot.html

Information

`GNU GRUB 2` (chargeur de démarrage pour architecture x86) offre la possibilité d'ajouter un mot de passe de déverrouillage.
Consultez la documentation [16] pour de plus amples informations.

Attention

Un changement de configuration du chargeur de démarrage peut nécessiter l'exécution d'un logiciel tiers pour être pris en compte dans la configuration (`update-grub` par exemple) ainsi que le redémarrage complet du système.

5.2 Configuration dynamique

Cette section traite des configurations permettant de durcir l'exécution d'un noyau Linux déjà compilé. Elle s'applique à la fois aux noyaux Linux compilés par des tiers (distributions…) et aux noyaux Linux pleinement maîtrisés et recompilés.

Les paramètres de ligne de commande permettent de modifier un certain nombre d'options de configuration du noyau Linux lors du démarrage. Ils sont souvent utilisés pour écraser les valeurs par défaut ou pour spécifier des directives de configuration.

Information

Avec `GNU GRUB 2`, l'option `GRUB_CMDLINE_LINUX` du fichier de configuration `/etc/default/grub` permet d'ajouter ou de modifier les paramètres de la ligne de commande du noyau.

Le démarrage sécurisé UEFI détaillé au paragraphe 4.3 permet d'assurer l'intégrité et l'authenticité des binaires UEFI lors du démarrage (chargeur de démarrage, noyau Linux…). Cependant, les paramètres de ligne de commande du noyau et l'`initramfs` (`INITial RAM filesystem`) n'étant pas des binaires UEFI, ils ne sont pas protégés par le démarrage sécurisé. Il convient alors de créer une `Unified kernel image` qui regroupe le noyau, l'`initramfs` et les paramètres de ligne de commande du noyau dans un seul binaire UEFI. La signature de ce dernier sera ensuite vérifiée au démarrage du système, assurant l'intégrité et l'authenticité de tous ces composants.

R6

M I R E Protéger les paramètres de ligne de commande du noyau et l'`initramfs`

Il est recommandé de protéger les paramètres de ligne de commande du noyau Linux et l'`initramfs` pour qu'ils soient vérifiés lors du démarrage sécurisé UEFI.

Un certain nombre d'options de configuration du noyau Linux peuvent être modifiées dynamiquement grâce à l'utilitaire `sysctl` [8]

8. https://man7.org/linux/man-pages/man8/sysctl.8.html

Information

Le fichier de configuration `sysctl.conf` [9] permet de modifier ces options de configuration à chaque démarrage du système grâce au service `systemd-sysctl.service` [10] du gestionnaire de système et de services `systemd` [11].

Ces options de configuration peuvent agir à tout niveau :

- **mémoire :** configuration de la pagination, des allocateurs, des propriétés des mappings…
- **noyau :** contrôle des caches, `swap`, `scheduling`…
- **processus :** ressources allouées, restrictions d'exécution, cloisonnement…
- **réseau :** tailles et caractéristiques des tampons, choix d'algorithmes…
- **systèmes de fichiers :** `setuid dumps`, `hard` et `soft links`, quotas…

Ces options s'enrichissent au gré des évolutions et des améliorations apportées au noyau Linux. Il est donc nécessaire de se rapporter à la documentation (généralement sous `Documentation/` dans les sources ou à la documentation officielle en ligne [1] disponible pour les dernières versions du noyau Linux) pour obtenir une explication détaillée. Parmi ces options, certaines ont un impact critique en termes de sécurité, et doivent donc être minutieusement étudiées.

Attention

La portée, l'usage et le paramétrage des options de configuration du noyau Linux accessibles par paramètres de ligne de commande ou dynamiquement grâce à l'utilitaire `sysctl` doivent faire l'objet d'une veille régulière si un administrateur souhaite profiter au mieux de leurs fonctionnalités.

5.2.1 Configuration de la mémoire

L'activation du service d'IOMMU (`Input/Output Memory Management Unit` ou unité de gestion de mémoire d'entrée/sortie) permet de protéger la mémoire du système vis-à-vis d'accès arbitraires réalisés par des périphériques. L'efficacité de cette protection dépend grandement de la façon dont le système est construit. Même si sa conception est imparfaite (voir l'article [14]), l'IOMMU contribue à réduire les risques d'accès arbitraire à la mémoire par les périphériques.

L'activation de la fonctionnalité dépend de la configuration matérielle détaillée au paragraphe 4.1 et du réglage du système d'exploitation. Suivant la situation (présence ou absence d'une IOMMU fonctionnelle, quantité de mémoire présente sur le système…), le noyau Linux peut décider de ne pas initialiser l'IOMMU. Il faut donc lui indiquer d'en forcer l'usage au travers d'une directive de configuration passée en paramètre lors du démarrage.

9. https://man7.org/linux/man-pages/man5/sysctl.conf.5.html
10. https://man7.org/linux/man-pages/man8/systemd-sysctl.service.8.html
11. https://systemd.io

Attention

L'activation de l'IOMMU sur un système peut engendrer des instabilités matérielles, il convient donc de s'assurer du bon fonctionnement de celui-ci avant de déployer une telle mesure en production.

R7

M I R Activer l'IOMMU

Il est recommandé d'activer l'IOMMU en ajoutant la directive `iommu=force` à la liste des paramètres du noyau lors du démarrage en plus de celles déjà présentes dans les fichiers de configuration du chargeur de démarrage.

La configuration de la mémoire est relativement générique et doit pouvoir s'appliquer indépendamment du cas d'usage de la cible.

R8

M I Paramétrer les options de configuration de la mémoire

Les listes ci-dessous présentent les options de configuration de la mémoire recommandées.

Listes des options de configuration de la mémoire recommandées

Les options de configuration de la mémoire détaillées dans cette liste sont à ajouter à la liste des paramètres du noyau lors du démarrage en plus de celles déjà présentes dans le fichier de configuration du chargeur de démarrage :

- *l1tf=full,force* : active sans possibilité de désactivation ultérieure toutes les contre-mesures pour la vulnérabilité `L1 Terminal Fault` (L1TF) présente sur la plupart des processeurs Intel (en 2018 tout du moins). À noter que cela désactive `Symmetric MultiThreading` (SMT) et peut donc avoir un impact fort sur les performances du système. Cette option n'est toutefois nécessaire que lorsque le système est susceptible d'être utilisé comme hyperviseur. Si les machines virtuelles sont de confiance, c'est-à-dire avec un système d'exploitation invité à la fois de confiance et protégé contre la vulnérabilité L1TF, cette option n'est pas nécessaire et peut même être remplacée par l1tf=off pour maximiser les performances ;
- *page_poison=on* : active le `poisoning` des pages de l'allocateur de pages (`buddy allocator`). Cette fonctionnalité permet de remplir les pages libérées avec des motifs lors de leur libération et de vérifier les motifs lors de leur allocation. Ce remplissage permet de réduire le risque de fuites d'informations à partir des données libérées ;
- *pti=on* : force l'utilisation de `Page Table Isolation` (PTI) y compris sur les processeurs se prétendant non impactés par la vulnérabilité `Meltdown` ;
- *slab_nomerge=yes* (équivalent à `CONFIG_SLAB_MERGE_DEFAULT=n`) : désactive la fusion de caches slabs (allocations mémoire dynamiques) de taille identique. Cette fonctionnalité permet de différentier les allocations entre les différents

caches slabs, et complique fortement les méthodologies de pétrissage du tas (`heap massaging`) en cas de `heap overflow` ;

- *slub_debug=FZP* : active certaines options de vérification des caches slabs (allocations mémoire dynamiques) :
  - F active les tests de cohérence des métadonnées des caches slabs,
  - Z active le `Red Zoning` ; dans un cache slab, ajoute une zone rouge après chaque objet afin de détecter des écritures après celui-ci. Il est important de noter que la valeur utilisée pour la zone rouge n'est pas aléatoire et qu'il s'agit donc d'un durcissement bien plus faible que l'utilisation de véritables canaris,
  - P active le poisoning des objets et du padding, c'est-à-dire provoque une erreur lors de l'accès aux zones empoisonnées ;
- *spec_store_bypass_disable=seccomp* : force le système à utiliser la contre-mesure par défaut (sur un système x86 supportant `seccomp`) pour la vulnérabilité Spectre v4 (`Speculative Store Bypass`) ;
- *spectre_v2=on* : force le système à utiliser une contre-mesure pour la vulnérabilité Spectre v2 (`Branch Target Injection`). Cette option active *spectre_v2_user=on* qui évite les attaques `Single Threaded Indirect Branch Predictors` (STIBP) et `Indirect Branch Prediction Barrier` [12] (IBPB) ;
- `mds=full,nosmt` : force le système à utiliser `Microarchitectural Data Sampling` (MDS) pour atténuer les vulnérabilités des processeurs Intel. L'option `mds=full`, qui laisse `Symmetric MultiThreading` (SMT) activé, n'est donc pas une atténuation complète. Cette atténuation nécessite une mise à jour du microcode Intel et atténue également la vulnérabilité `TSX Asynchronous Abort` (TAA) du processeur Intel sur les systèmes affectés par MDS ;
- `mce=0` : force un `kernel panic` sur les erreurs non corrigées signalées par le support `Machine Check`. Sinon, certaines d'entre elles provoquent uniquement l'envoi d'un `SIGBUS`, permettant potentiellement à un processus malveillant de continuer à essayer d'exploiter une vulnérabilité par exemple `Rowhammer` ;
- `page_alloc.shuffle=1` (équivalent à `CONFIG_SHUFFLE_PAGE_ALLOCATOR=y`) : active le `Page allocator randomization` qui améliore les performances pour l'utilisation du `direct-mapped memory-side-cache` mais réduit la prévisibilité des allocations de page et complète ainsi `SLAB_FREELIST_RANDOM` ;
- `rng_core.default_quality=500` : augmente la confiance dans `HWRNG` du TPM pour une initialisation robuste et rapide du `CSPRNG` de Linux en créditant la moitié de l'entropie qu'il fournit.

12. https://www.kernel.org/doc/html/latest/admin-guide/hw-vuln/spectre.html

5.2.2 Configuration du noyau

La configuration du noyau est relativement générique et doit pouvoir s'appliquer indépendamment du cas d'usage de la cible.

R9

Paramétrer les options de configuration du noyau

La liste ci-dessous présente les options de configuration du noyau recommandées.

Liste des options de configuration du noyau recommandées

Les options de configuration du noyau détaillées dans cette liste sont présentées telles que rencontrées dans le fichier de configuration `sysctl.conf` :

```
# Restreint l'accès au buffer dmesg (équivalent à
# CONFIG_SECURITY_DMESG_RESTRICT=y)
kernel.dmesg_restrict=1
# Cache les adresses noyau dans /proc et les différentes autres interfaces,
# y compris aux utilisateurs privilégiés
kernel.kptr_restrict=2
# Spécifie explicitement l'espace d'identifiants de processus supporté par le
# noyau, 65 536 étant une valeur donnée à titre d'exemple
kernel.pid_max=65536
# Restreint l'utilisation du sous-système perf
kernel.perf_cpu_time_max_percent=1
kernel.perf_event_max_sample_rate=1
# Interdit l'accès non privilégié à l'appel système perf_event_open(). Avec une
# valeur plus grande que 2, on impose la possession de CAP_SYS_ADMIN, pour pouvoir
# recueillir les évènements perf.
kernel.perf_event_paranoid=2
# Active l'ASLR
kernel.randomize_va_space=2
# Désactive les combinaisons de touches magiques (Magic System Request Key)
kernel.sysrq=0
# Restreint l'usage du BPF noyau aux utilisateurs privilégiés
kernel.unprivileged_bpf_disabled=1
# Arrête complètement le système en cas de comportement inattendu du noyau Linux
kernel.panic_on_oops=1
```

Il est possible d'interdire le chargement de nouveaux modules (y compris par `root`) avec l'option de configuration du noyau Linux `kernel.modules_disabled`. Un module noyau peut être chargé explicitement par un utilisateur privilégié, mais également par un utilisateur non-privilégié en utilisant le chargement à la demande. Ce mécanisme propose de ne charger un module que lorsqu'une action le nécessite. Par exemple, la création d'une socket DCCP (`Datagram Congestion Control Protocol`) va entraîner le chargement du module `dccp` par le noyau si celui-ci n'est pas encore chargé. L'option de configuration `kernel.modules_disabled` nous protège donc du chargement d'un module tiers malveillant, mais également du chargement d'un module vulnérable. Un utilisateur non-privilégié peut en effet déclencher le chargement d'un tel module et ensuite en exploiter la vulnérabilité pour accroître ses privilèges.

Cette mesure peut avoir des conséquences non triviales sur le fonctionnement du reste du système. Il faut donc s'assurer que son activation n'entraîne pas d'impacts fonctionnels significatifs.

Notamment, le chargement de modules n'étant plus possible une fois l'option activée, il est nécessaire que tous soient chargés avant. Pour ce faire, il est possible de déclarer les modules à charger

lors du démarrage avant que l'option ne s'applique.

R10

Désactiver le chargement des modules noyau

Il est recommandé de bloquer le chargement des modules noyau par l'activation de l'option de configuration du noyau `kernel.modules_disabled`.

Option de configuration de désactivation du chargement des modules noyau

L'option de configuration du noyau permettant de bloquer le chargement des modules noyau est présentée telle que rencontrée dans le fichier de configuration `/etc/sysctl.conf` :

```
# Interdit le chargement des modules noyau (sauf ceux déjà chargés à ce point)
kernel.modules_disabled=1
```

Liste des modules noyau chargés par défaut au démarrage

Des modules noyau peuvent être définis pour être chargés au démarrage, avant que l'option de configuration du noyau Linux `kernel.modules_disabled` ne soit active. La liste est définie dans `/etc/modules`.
La liste des modules chargés sur le système à un moment donné peut être trouvée avec la commande `lsmod`.

Attention

Une fois cette option activée, celle-ci n'est plus désactivable sans redémarrer le système. Le chargement d'un nouveau module noyau nécessitera donc également un redémarrage.

5.2.3 Configuration des processus

La configuration des processus est relativement générique et doit pouvoir s'appliquer indépendamment du cas d'usage de la cible.

Le module de sécurité `Yama`, permet d'ajouter l'option de configuration des processus `kernel.yama.ptrace_scope` qui contrôle les droits d'accès à l'appel système `ptrace`. Cet appel système est particulièrement sensible car il permet de tracer le fonctionnement d'un processus. Par défaut, tout utilisateur peut déboguer tous les processus lui appartenant, ce qui inclut les processus stockant des éléments sensibles dans leur mémoire comme des clés ou des mots de passe (navigateur Internet, `ssh-agent`…). La compromission d'un processus de l'utilisateur permettrait de récupérer ces éléments sensibles.

L'option `kernel.yama.ptrace_scope` permet de restreindre l'usage de l'appel système `ptrace` :

- =0, tous les processus peuvent être débogués pourvu qu'ils partagent le même UID (`User IDentifier` ou identifiant utilisateur)

- =1, un processus non privilégié (ne possédant pas `CAP_SYS_PTRACE`) ne peut déboguer que les processus descendants, sauf si le processus cible l'a explicitement autorisé avec la commande `prctl(PR_SET_PTRACER, pid, ...)`
- =2, seul un processus privilégié (possédant `CAP_SYS_PTRACE`) peut utiliser `ptrace`
- =3, aucun processus ne peut utiliser `ptrace`

R11

Activer et configurer le LSM `Yama`

Il est recommandé de charger le module de sécurité `Yama` lors du démarrage, par exemple en passant la directive `security=yama` au noyau, et d'affecter à l'option de configuration du noyau `kernel.yama.ptrace_scope` une valeur au moins égale à 1.

5.2.4 Configuration du réseau

La configuration du réseau varie selon le cas d'usage (équipement routeur, équipement terminal, usage ou non d'un système de redondance réseau). Elle doit donc être considérée en lien avec les contraintes fournies par les spécifications réseau de l'équipement.

R12

Paramétrer les options de configuration du réseau IPv4

La liste ci-dessous présente les options de configuration du réseau *IPv4* pour un hôte « serveur » typique qui n'effectue pas de routage et ayant une configuration *IPv4* minimaliste (adressage statique).

Liste des options de configuration du réseau IPv4 recommandées

Les options de configuration du réseau *IPv4* détaillées dans cette liste sont recommandées pour un hôte de type « serveur » n'effectuant pas de routage et ayant une configuration *IPv4* minimaliste. Elles sont présentées telles que rencontrées dans le fichier de configuration `sysctl.conf` :

```
# Atténuation de l'effet de dispersion du JIT noyau au coût d'un compromis sur
# les performances associées.
net.core.bpf_jit_harden=2
# Pas de routage entre les interfaces. Cette option est spéciale et peut
# entrainer des modifications d'autres options. En plaçant cette option au plus
# tôt, on s'assure que la configuration des options suivantes ne change pas.
net.ipv4.ip_forward=0
# Considère comme invalides les paquets reçus de l'extérieur ayant comme source
# le réseau 127/8.
net.ipv4.conf.all.accept_local=0
# Refuse la réception de paquet ICMP redirect. Le paramétrage suggéré de cette
# option est à considérer fortement dans le cas de routeurs qui ne doivent pas
# dépendre d'un élément extérieur pour déterminer le calcul d'une route. Même
# pour le cas de machines non-routeurs, ce paramétrage prémunit contre les
# détournements de trafic avec des paquets de type ICMP redirect.
net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.default.accept_redirects=0
net.ipv4.conf.all.secure_redirects=0
net.ipv4.conf.default.secure_redirects=0
net.ipv4.conf.all.shared_media=0
net.ipv4.conf.default.shared_media=0
```

```
# Refuse les informations d'en-têtes de source routing fournies par le paquet
# pour déterminer sa route.
net.ipv4.conf.all.accept_source_route=0
net.ipv4.conf.default.accept_source_route=0
# Empêche le noyau Linux de gérer la table ARP globalement. Par défaut, il peut
# répondre à une requête ARP d'une interface X avec les informations d'une
# interface Y. Ce comportement est problématique pour les routeurs et les
# équipements d'un système en haute disponibilité (VRRP...).
net.ipv4.conf.all.arp_filter=1
# Ne répond aux sollicitations ARP que si l'adresse source et destination sont sur
# le même réseau et sur l'interface sur laquelle le paquet a été reçu. Il est à
# noter que la configuration de cette option est à étudier selon le cas d'usage.
net.ipv4.conf.all.arp_ignore=2
# Refuse le routage de paquet dont l'adresse source ou destination est celle de la
# boucle locale. Cela interdit l'émission de paquet ayant comme source 127/8.
net.ipv4.conf.all.route_localnet=0
# Ignore les sollicitations de type gratuitous ARP. Cette configuration est
# efficace contre les attaques de type ARP poisoning mais ne s'applique qu'en
# association avec un ou plusieurs proxy ARP maîtrisés. Elle peut également être
# problématique sur un réseau avec des équipements en haute disponibilité (VRRP...)
net.ipv4.conf.all.drop_gratuitous_arp=1
# Vérifie que l'adresse source des paquets reçus sur une interface donnée aurait
# bien été contactée via cette même interface. À défaut, le paquet est ignoré.
# Selon l'usage, la valeur 1 peut  permettre d'accroître la vérification à
# l'ensemble des interfaces, lorsque l'équipement est un routeur dont le calcul de
# routes est dynamique. Le lecteur intéressé est renvoyé à la RFC3704 pour tout
# complément concernant cette fonctionnalité.
net.ipv4.conf.default.rp_filter=1
net.ipv4.conf.all.rp_filter=1
# Cette option ne doit être mise à 1 que dans le cas d'un routeur, car pour ces
# équipements l'envoi de ICMP redirect est un comportement normal. Un équipement
# terminal n'a pas de raison de recevoir un flux dont il n'est pas destinataire et
# donc d'émettre un paquet ICMP redirect.
net.ipv4.conf.default.send_redirects=0
net.ipv4.conf.all.send_redirects=0
# Ignorer les réponses non conformes à la RFC 1122
net.ipv4.icmp_ignore_bogus_error_responses=1
# Augmenter la plage pour les ports éphémères
net.ipv4.ip_local_port_range=32768 65535
# RFC 1337
net.ipv4.tcp_rfc1337=1
# Utilise les SYN cookies. Cette option permet la prévention d'attaque de
# type SYN flood.
net.ipv4.tcp_syncookies=1
```

La plupart du temps, les systèmes d'information utilisent des réseaux *IPv4*. Il est donc essentiel, dans ce cas, de ne pas conserver inutilement actif un plan *IPv6*.

R13

Désactiver le plan IPv6

Quand *IPv6* n'est pas utilisé, il est recommandé de désactiver la pile `IPv6` en :

- ajoutant la directive `ipv6.disable=1` à la liste des paramètres du noyau choisi lors du démarrage en plus de celles déjà présentes dans les fichiers de configuration du chargeur de démarrage,
- activant les options de configuration du réseau présentées dans la liste ci-dessous.

Liste des options de configuration du réseau pour désactiver le plan IPv6

Les options de configuration du réseau détaillées dans cette liste désactivent le plan *IPv6*. Elles sont présentées telles que rencontrées dans le fichier de configuration

`sysctl.conf` :

```
net.ipv6.conf.default.disable_ipv6=1
net.ipv6.conf.all.disable_ipv6=1
```

Information

Avec `GNU GRUB 2`, la désactivation du plan `IPv6` est possible en ajoutant la directive `ipv6.disable=1` à l'option suivante dans le fichier de configuration `/etc/default/grub` :

```
GRUB_CMDLINE_LINUX=" ipv6.disable=1"
```

À défaut, le plan *IPv6* doit, tout comme le plan *IPv4*, être sécurisé.

5.2.5 Configuration des systèmes de fichiers

La configuration des systèmes de fichiers est relativement générique et doit pouvoir s'appliquer indépendamment du cas d'usage de la cible.

R14

Paramétrer les options de configuration des systèmes de fichiers

La liste ci-dessous présente les options de configuration des systèmes de fichiers recommandées au format du ficher de configuration `sysctl.conf`.

Liste des options de configuration des systèmes de fichiers recommandées

Les options de configuration des systèmes de fichiers détaillées dans cette liste sont présentées telles que rencontrées dans le fichier de configuration `sysctl.conf` :

```
# Désactive la création de coredump pour les exécutables setuid
# Notez qu'il est possible de désactiver tous les coredumps avec la
# configuration CONFIG_COREDUMP=n
fs.suid_dumpable = 0
# Disponible à partir de la version 4.19 du noyau Linux, permet d'interdire
# l'ouverture des FIFOS et des fichiers "réguliers" qui ne sont pas la propriété
# de l'utilisateur dans les dossiers sticky en écriture pour tout le monde.
fs.protected_fifos=2
fs.protected_regular=2
# Restreint la création de liens symboliques à  des fichiers dont l'utilisateur
# est propriétaire. Cette option fait partie des mécanismes de prévention contre
# les vulnérabilités de la famille Time of Check - Time of Use (Time of Check -
# Time of Use)
fs.protected_symlinks=1
# Restreint la création de liens durs à des fichiers dont l'utilisateur est
# propriétaire. Ce sysctl fait partie des mécanismes de prévention contre les
# vulnérabilités Time of Check - Time of Use, mais aussi contre la possibilité de
# conserver des accès à des fichiers obsolètes
fs.protected_hardinks=1
```

5.3 Configuration statique

Le noyau Linux peut être recompilé à partir du code source. Dans ce cas, il est fortement recommandé de prendre en compte au moment de la configuration des sources du noyau les considéra-

tions de sécurité décrites dans cette section.

Ces dernières apportent un grand nombre d'éléments de protection au noyau mais également aux applications de l'équipement (principe de minimisation, principe de défense en profondeur…).

La configuration de sécurité du noyau est séparée en deux parties :

- Les éléments de configuration indépendants de la cible matérielle
- Les éléments de configuration spécifiques à une cible matérielle donnée

Attention

La recompilation du noyau Linux peut apporter un réel gain en sécurité. Il faut cependant prendre en compte les problématiques de MCO (`Maintien en Condition Opérationelle`) et MCS (`Maintien en Condition de Sécurité`) que cela apporte. Le déploiement des paquets du noyau seront en effet plus complexe. Il faudra mettre en place un répertoire de paquets interne ainsi qu'une PKI pour signer les paquets. Il sera également nécessaire d'investir du temps pour suivre les évolutions des options de configuration. Enfin, il est primordial que les recompilations soient fréquentes afin de s'assurer d'avoir des noyaux déployés à jour. Il serait contre-productif d'avoir un noyau durci mais vieux et vulnérable.

5.3.1 Configuration indépendante de la cible matérielle

Les tables de pages hébergent les informations de configuration mémoire de chaque processus, que la MMU (`Memory Management Unit` ou unité de gestion de mémoire) va utiliser pour gérer les droits d'accès de ce dernier à son espace d'adressage. Ces dernières doivent être efficacement protégées. De même, la mémoire du noyau doit être gérée de manière sécurisée. Il existe pour cela divers mécanismes, comme KASLR (`Kernel Address Space Layout Randomisation` ou distribution aléatoire de l'espace d'adressage du noyau) ou des canaris. Ces mécanismes de protection, tout comme pour les processus applicatifs, complexifient l'exploitation d'une vulnérabilité noyau.

R15

M I R E Paramétrer les options de compilation pour la gestion de la mémoire

La liste ci-dessous présente les options de compilation du noyau recommandées pour durcir la gestion de la mémoire.

Liste des options de compilation du noyau recommandées pour durcir la gestion de la mémoire

```
# Cette option a remplacé CONFIG_DEBUG_RODATA dans le noyau (>=4.11)
# qui était une dépendance de CONFIG_DEBUG_KERNEL
CONFIG_STRICT_KERNEL_RWX=y
# CONFIG_ARCH_OPTIONAL_KERNEL_RWX et CONFIG_ARCH_HAS_STRICT_KERNEL_RWX sont
# des dépendances de CONFIG_STRICT_KERNEL_RWX
CONFIG_ARCH_OPTIONAL_KERNEL_RWX=y
CONFIG_ARCH_HAS_STRICT_KERNEL_RWX=y
# Vérifie et rapporte les permissions de mapping mémoire dangereuses, par
# exemple, lorsque les pages noyau sont en écriture et sont exécutables.
```

```
# Cette option n'est pas disponible sur l'ensemble des architectures
# matérielles (x86 >=4.4, arm64 >=4.10, etc.).
CONFIG_DEBUG_WX=y
# Désactive le système de fichiers utilisé uniquement dans le cadre de
# débogage. Protéger ce système de fichiers nécessite un travail
# supplémentaire.
CONFIG_DEBUG_FS=n
# A partir de la version 4.18 du noyau, ces options ajoutent
# -fstack-protector-strong à la compilation afin de renforcer le stack canary,
# il est nécessaire d'avoir une version de GCC au moins égale à 4.9.
# Avant la version 4.18 du noyau linux, il faut utiliser les options
# CONFIG_CC_STACKPROTECTOR et CONFIG_CC_STACKPROTECTOR_STRONG
CONFIG_STACKPROTECTOR=y
CONFIG_STACKPROTECTOR_STRONG=y
# Interdit l'accès direct à la mémoire physique.
# En cas de besoin et uniquement dans ce cas, il est possible d'activer un
# accès strict à la mémoire, limitant ainsi son accès, avec les options
# CONFIG_STRICT_DEVMEM=y et CONFIG_IO_STRICT_DEVMEM=y
#CONFIG_DEVMEM is not set
# Détecte la corruption de la pile pendant l'appel à l'ordonnanceur
CONFIG_SCHED_STACK_END_CHECK=y
# Impose une vérification des limites du mapping mémoire du processus au
# moment des copies de données.
CONFIG_HARDENED_USERCOPY=y
# Interdit de revenir à une vérification du mapping auprès de l'allocateur si
# l'option précédente a échoué (<=5.15)
#CONFIG_HARDENED_USERCOPY_FALLBACK is not set
# Ajout de pages de garde entre les piles noyau. Cela protège contre les effets
# de bord des débordements de pile (cette option n'est pas supportée sur toutes
# les architectures).
CONFIG_VMAP_STACK=y
# Vérifications exhaustives sur les compteurs de référence du noyau (<=5.4)
CONFIG_REFCOUNT_FULL=y
# Vérifie les actions de recopie mémoire qui pourraient provoquer une corruption
# de structure dans les fonctions noyau str*() et mem*(). Cette vérification est
# effectuée à la compilation et à l'exécution.
CONFIG_FORTIFY_SOURCE=y
# Désactive l'usage dangereux de l'ACPI, pouvant entrainer une écriture directe
# en mémoire physique.
#CONFIG_ACPI_CUSTOM_METHOD is not set
# Interdit tout accès direct à la mémoire du noyau à partir du userspace (<=5.12)
#CONFIG_DEVKMEM is not set
# Interdit la fourniture de la disposition de l'image noyau
#CONFIG_PROC_KCORE is not set
# Désactive la rétrocompatibilité VDSO, qui rend impossible l'usage de l'ASLR
#CONFIG_COMPAT_VDSO is not set
# Empêche les utilisateurs non privilégiés de récupérer des adresses noyau
# avec dmesg(8)
CONFIG_SECURITY_DMESG_RESTRICT=y
# Active les retpoline qui sont nécessaires pour se protéger de Spectre v2
# GCC >= 7.3.0 est requis.
CONFIG_RETPOLINE=y
# Désactive la table vsyscall. Elle n'est plus requise par la libc et est une
# source potentielle de ROP gadgets.
CONFIG_LEGACY_VSYSCALL_NONE=y
CONFIG_LEGACY_VSYSCALL_EMULATE=n
CONFIG_LEGACY_VSYSCALL_XONLY=n
CONFIG_X86_VSYSCALL_EMULATION=n
```

Apporter des mécanismes de protection (tels que KASLR ou des canaris) peut être insuffisant. Un certain nombre de structures mémoire du noyau sont souvent la cible d'attaques et des mécanismes existent pour en contrôler l'intégrité. Avec ces mécanismes, il est possible de détecter leur corruption et de prendre des mesures pour réagir, en général en stoppant le processus à l'origine de la corruption ou en arrêtant le système en cas d'erreur non récupérable. Ces mesures sont utiles car une attaque entraînant la corruption de la mémoire du noyau est suffisamment avancée pour potentiellement permettre l'exécution contrôlée de code en mode superviseur sur l'équipement ou

la désactivation de mécanismes de protection génériques.

R16

M I R E Paramétrer les options de compilation pour les structures de données du noyau

La liste ci-dessous présente les options de compilation du noyau recommandées pour protéger les structures de données du noyau.

Liste des options de compilation du noyau recommandées pour protéger les structures de données du noyau

```
# Vérifie les structures de gestion des autorisations.
CONFIG_DEBUG_CREDENTIALS=y
# Vérifie les structures de notification.
CONFIG_DEBUG_NOTIFIERS=y
# Vérifie les listes noyau.
CONFIG_DEBUG_LIST=y
# Vérifie les tables de Scatter-Gather du noyau.
CONFIG_DEBUG_SG=y
# Génère un appel à BUG() en cas de détection de corruption.
CONFIG_BUG_ON_DATA_CORRUPTION=y
```

Il existe un certain nombre de structures sensibles dans le noyau, hébergeant des données impactantes pour la sécurité. Ces structures sont celles qui sont classiquement visées en cas d'exploitation d'une vulnérabilité noyau. Il est possible d'activer des mécanismes de validation supplémentaires de ces structures pour aider à détecter leur corruption.

R17

M I R E Paramétrer les options de compilation pour l'allocateur mémoire

La liste ci-dessous présente les options de compilation du noyau recommandées pour durcir l'allocateur mémoire.

Liste des options de compilation du noyau recommandées pour durcir l'allocateur mémoire

```
# Positionne les informations de chaînage des blocs libres de l'allocateur de
# manière aléatoire.
CONFIG_SLAB_FREELIST_RANDOM=y
# CONFI_SLAB est une dépendance de CONFIG_SLAB_FREELIST_RANDOM
CONFIG_SLUB=y
# Protège les métadonnées de l'allocateur en intégrité.
CONFIG_SLAB_FREELIST_HARDENED=y
# A partir de la version 4.13 du noyau, cette option désactive le merge des
# caches SLAB
CONFIG_SLAB_MERGE_DEFAULT=n
# Active la vérification des structures de l'allocateur mémoire et réinitialise
# à zéro les zones allouées à leur libération (nécessite l'activation de
# l'option de configuration de la mémoire page_poison=on ajoutée à la liste des
# paramètres du noyau lors du démarrage). Il est préférable d'utiliser
# l'option de configuration slub_debug de la mémoire ajoutée à la liste des
# paramètres du noyau lors du démarrage car elle permet une gestion plus fine
# du slub debug.
CONFIG_SLUB_DEBUG=y
```

```
# Nettoie les pages mémoire au moment de leur libération.
CONFIG_PAGE_POISONING=y
# Nettoyage en profondeur désactivé. Cette option a un coût important;
# cependant si l'impact en performance est compatible avec le besoin
# opérationnel de l'équipement il est recommandé de l'activer (<=5.10)
CONFIG_PAGE_POISONING_NO_SANITY=y
# Le nettoyage des pages mémoire est effectué avec une réécriture de zéros en
# lieu et place des données (<=5.10)
CONFIG_PAGE_POISONING_ZERO=y
# Désactive la rétrocompatibilité avec brk() qui rend impossible une
# implémentation de brk() avec l'ASLR.
#CONFIG_COMPAT_BRK is not set
```

Il peut être nécessaire d'utiliser des modules. En effet, la présence d'un parc hétérogène peut nécessiter d'activer certaines fonctionnalités et pilotes sous forme de modules afin de limiter le nombre d'images noyau à gérer. Néanmoins, l'usage de modules implique des considérations supplémentaires sur la configuration. Ces derniers doivent être signés à l'aide d'une clef générée au moment de la compilation du noyau et leur signature vérifiée au moment du chargement. Ce mécanisme est supporté nativement dans le noyau Linux.

R18

M I R E Paramétrer les options de compilation pour la gestion des modules noyau

La liste ci-dessous présente les options de compilation du noyau recommandées pour durcir la gestion des modules noyau.

Liste des options de compilation du noyau recommandées pour durcir la gestion des modules noyau

```
# Activation du support des modules
CONFIG_MODULES=y
# Cette option a remplacé CONFIG_DEBUG_SET_MODULE_RONX dans le noyau (>=4.11)
CONFIG_STRICT_MODULE_RWX=y
# Les modules doivent être signés. Attention, les modules ne doivent pas être
# strippés après signature.
CONFIG_MODULE_SIG=y
# Empêche le chargement des modules non signés ou signés avec une clé qui ne
# nous appartient pas.
CONFIG_MODULE_SIG_FORCE=y
# Activer CONFIG_MODULE_SIG_ALL permet de signer tous les modules
# automatiquement pendant l'étape "make modules_install", sans cette option
# les modules doivent être signés manuellement en utilisant le script
# scripts/sign-file. L'option CONFIG_MODULE_SIG_ALL dépend de CONFIG_MODULE_SIG
# et CONFIG_MODULE_SIG_FORCE, elles doivent donc être activées.
CONFIG_MODULE_SIG_ALL=y
# La signature des modules utilise SHA512 comme fonction de hash
CONFIG_MODULE_SIG_SHA512=y
CONFIG_MODULE_SIG_HASH="sha512"
# Indique le chemin vers le fichier contenant à la fois la clé privée et son
# certificat X.509 au format PEM utilisé pour la signature des modules,
# relatif à la racine du noyau.
CONFIG_MODULE_SIG_KEY="certs/signing_key.pem"
```

Lors d'une compromission, l'exploitation d'une vulnérabilité passe souvent par l'exécution de code arbitraire par un processus donné. Cette exécution peut provoquer un comportement incorrect dû à la corruption de certaines structures de données applicatives. En cas d'erreur critique, le

noyau génère un appel à la macro `BUG()`, dont le but est d'afficher un évènement journalisé et d'arrêter immédiatement le composant fautif du système. Compiler le noyau Linux avec l'option `CONFIG_PANIC_ON_OOPS` aura pour conséquence l'arrêt complet du système en cas de comportement inattendu du noyau.

R19

M I R E Paramétrer les options de compilation pour les évènements anormaux

La liste ci-dessous présente les options de compilation du noyau recommandées pour réagir aux évènements anormaux.

Liste des options de compilation du noyau recommandées pour réagir aux évènements anormaux

```
# Émet un rapport sur les conditions d'appel à la macro noyau BUG()
# et tue le processus à l'origine de l'appel. Ne pas positionner cette variable
# peut passer sous silence un certain nombre d'erreurs critiques.
CONFIG_BUG=y
# Arrête le système en cas d'erreur critique pour couper court à tout comportement
# erroné.
CONFIG_PANIC_ON_OOPS=y
# Empêche le redémarrage de la machine suite à un panic ce qui coupe court à
# toute tentative d'attaque par force brute. Ceci a évidemment un impact fort
# sur des serveurs en production.
CONFIG_PANIC_TIMEOUT=-1
```

Le noyau Linux fournit un grand nombre de mécanismes de sécurité visant à protéger les processus applicatifs. Ces derniers sont principalement implémentés sous forme de LSM (`Linux Security Module` ou module de sécurité Linux) qui peuvent pour certains être exécutés séquentiellement pour autoriser un accès. Certains LSM nécessitent des configurations centralisées dans l'équipement, comme `AppArmor` détaillé au paragraphe 6.3.2 ou `SELinux` détaillé au paragraphe 6.3.3. D'autres correspondent à un durcissement du noyau sans configuration spécifique comme `Yama`. Le noyau offre également la possibilité de mettre en place des mécanismes de sécurité propres à chaque processus tels que `Seccomp` ou les `Namespaces`.

R20

M I R E Paramétrer les options de compilation pour les primitives de sécurité du noyau

La liste ci-dessous présente une liste des options de compilation du noyau recommandées pour configurer les primitives de sécurité du noyau.

Liste des options de compilation du noyau recommandées pour configurer les primitives de sécurité du noyau

```
# Active la possibilité de filtrer les appels système faits par une
# application.
CONFIG_SECCOMP=y
# Active la possibilité d'utiliser des script BPF (Berkeley Packet Filter).
CONFIG_SECCOMP_FILTER=y
# Active les primitives de sécurité du noyau Linux, nécessaire pour les LSM.
```

```
CONFIG_SECURITY=y
# Active Yama, qui permet de limiter simplement l'usage de l'appel système
# ptrace(). Une fois les modules de sécurité utilisés par le système
# sélectionné, il convient de désactiver le support des autres modules de
# sécurité dans le noyau afin de réduire la surface d'attaque.
CONFIG_SECURITY_YAMA=y
# Assure que les structures noyau associées aux LSM sont toujours mappées en
# lecture seule après le démarrage du système. Dans le cas où SELinux est
# utilisé, il faut s'assurer que CONFIG_SECURITY_SELINUX_DISABLE is not set,
# pour que cette option soit disponible. Il n'est alors plus possible de
# désactiver un LSM une fois le noyau démarré. Il est en revanche encore
# possible de le faire en modifiant la ligne de commande du noyau. Pour le
# noyau 4.18 les LSM présents sont: AppArmor, LoadPin, SELinux, Smack, TOMOYO
# et Yama.
#CONFIG_SECURITY_WRITABLE_HOOKS is not set
```

Depuis `GCC 4.5`, il est possible d'intégrer des `plugins` au compilateur pour apporter des mécanismes de durcissement complémentaires à la compilation des sources. Cette technique a été, par exemple, utilisée par le patch PaX [13]. Il existe aujourd'hui un certain nombre de `plugins` présents dans les sources du noyau Linux afin d'ajouter des propriétés de sécurité lors de la compilation.

R21

M I R E Paramétrer les options de compilation pour les `plugins` du compilateur

La liste ci-dessous présente les options de compilation du noyau recommandées pour configurer les `plugins` du compilateur.

Liste des options de compilation du noyau recommandées pour configurer les `plugins` du compilateur

```
# Active le support des plugins du compilateur (implique l'usage de GCC).
CONFIG_GCC_PLUGINS=y
# Amplifie la génération d'entropie au démarrage de l'équipement pour ceux
# ayant des sources d'entropie inappropriées
CONFIG_GCC_PLUGIN_LATENT_ENTROPY=y
# Nettoie le contenu de la pile au moment de l'appel système exit.
CONFIG_GCC_PLUGIN_STACKLEAK=y
# Impose l'initialisation des structures en mémoire pour éviter la fuite de
# données par superposition avec une ancienne structure.
CONFIG_GCC_PLUGIN_STRUCTLEAK=y
# Globalisation de l'option précédente au cas du passage de structure par
# référence entre fonction si celles-ci ont des champs non initialisés.
CONFIG_GCC_PLUGIN_STRUCTLEAK_BYREF_ALL=y
# Construit un plan mémoire aléatoire pour les structures système du noyau.
# Cette option a un impact fort sur les performances. L'option
# CONFIG_GCC_PLUGIN_RANDSTRUCT_PERFORMANCE=y est à utiliser en substitution si
# cet impact n'est pas acceptable.
CONFIG_GCC_PLUGIN_RANDSTRUCT=y
```

La plupart du temps, les systèmes d'information utilisent des réseaux *IPv4*. Il est possible, dans ce cas, de compiler le noyau Linux sans le plan *IPv6*.

13. https://pax.grsecurity.net

R22

M I R E Paramétrer les options de compilation pour la pile réseau

La liste ci-dessous présente les options de compilation du noyau recommandées pour configurer la pile réseau.

Liste des options de compilation du noyau recommandées pour configurer la pile réseau

```
# Désactive le plan IPv6
#CONFIG_IPV6 option set.
# Permet de prévenir les attaques de type SYN flood.
CONFIG_SYN_COOKIES=y
```

R23

M I R E Paramétrer les options de compilation pour divers comportements du noyau

La liste ci-dessous présente les options de compilation du noyau recommandées pour configurer divers comportements du noyau.

Certains comportements du noyau Linux peuvent augmenter inutilement la surface d'attaque du noyau. Si ceux-ci ne sont pas absolument nécessaires, ils ne doivent pas être activés.

Liste des options de compilation du noyau recommandées pour configurer divers comportements du noyau

```
# Interdit l'exécution d'une nouvelle image noyau à chaud.
#CONFIG_KEXEC is not set
# Interdit le passage en mode hibernation, qui permet de substituer l'image
# noyau à son insu.
#CONFIG_HIBERNATION is not set
# Désactive le support de format binaire arbitraire.
#CONFIG_BINFMT_MISC is not set
# Impose l'utilisation des ptys modernes (devpts) dont on peut contrôler le
# nombre et l'usage.
#CONFIG_LEGACY_PTYS is not set
# Si le support des modules n'est pas absolument nécessaire celui-ci doit être
# désactivé.
#CONFIG_MODULES is not set
```

5.3.2 Configuration spécifique aux architectures matérielles

Le noyau Linux supporte une multitude d'architectures (x86, ARM, MIPS, PowerPC…) de 32 ou 64 bits. Il existe des configurations de sécurité spécifiques à ces architectures. Cette section traite des deux principales architectures, x86 et ARM, en 32 et 64 bits.

R24

M I R E Paramétrer les options de compilation spécifiques aux architectures 32 bits

La liste ci-dessous présente les options de compilation du noyau recommandées pour les architectures de type x86 32 bits. Elles ne sont pas toutes pertinentes sur des architectures 64 bits.

Liste des options de compilation du noyau recommandées pour les architectures 32 bits

```
# Active le support des extensions d'adresse physique, ce qui est un prérequis
# pour le support du bit de permission NX dans la table des pages qui permet de
# marquer certaines pages comme non exécutables.
CONFIG_HIGHMEM64G=y
CONFIG_X86_PAE=y
# Interdit l'utilisation d'adresses mémoire en dessous d'une certaine valeur
# (contre-mesure contre les null pointer dereference).
CONFIG_DEFAULT_MMAP_MIN_ADDR=65536
# Rend non prédictible l'adresse de base du noyau. Cette option complexifie la
# tâche d'un attaquant.
CONFIG_RANDOMIZE_BASE=y
```

R25

M I R E Paramétrer les options de compilation spécifiques aux architectures x86_64 bits

La liste ci-dessous présente les options de compilation du noyau recommandées pour les architectures de type x86_64 bits. Elles ne sont pas toutes pertinentes sur des architectures 32 bits.

Liste des options de compilation du noyau recommandées pour les architectures x86_64 bits

```
# Activate le mode complet 64 bits.
CONFIG_X86_64=y
# Interdit l'utilisation d'adresses mémoire en dessous d'une certaine valeur
# (contre-mesure contre les null pointer dereference).
CONFIG_DEFAULT_MMAP_MIN_ADDR=65536
# Rend non prédictible l'adresse de base du noyau, cette option complexifie
# la tâche d'un attaquant.
CONFIG_RANDOMIZE_BASE=y
# Rend non prédictible l'adresse à laquelle les composants du noyau sont
# exposés dans l'espace d'adressage des processus.
CONFIG_RANDOMIZE_MEMORY=y
# Contre-mesure à l'attaque Meltdown.
CONFIG_PAGE_TABLE_ISOLATION=y
# Désactive la rétro-compatibilité 32 bits, ce qui permet de réduire la
# surface d'attaque mais empêche d'exécuter des binaires 32 bits.
#CONFIG_IA32_EMULATION is not set
# Interdit la surcharge par processus de la Local Descriptor Table
# (mécanisme lié à l'usage de la segmentation).
#CONFIG_MODIFY_LDT_SYSCALL is not set
```

R26

M I R E Paramétrer les options de compilation spécifiques aux architectures ARM

La liste ci-dessous présente les options de compilation du noyau recommandées pour les architectures de type ARM.

Liste des options de compilation du noyau recommandées pour les architectures ARM

```
# Interdit l'utilisation d'adresses mémoire en dessous d'une certaine valeur
# (contre-mesure contre les null pointer dereference).
CONFIG_DEFAULT_MMAP_MIN_ADDR=32768
# Maximise la taille de la mémoire virtuelle des processus (et de l'ASLR
# afférent).
CONFIG_VMSPLIT_3G=y
# Interdit les mappings mémoire RWX.
CONFIG_STRICT_MEMORY_RWX=y
# Interdit les accès par le noyau à la mémoire utilisateur (mécanisme
# équivalent sur ARM à SMAP sur x86_64).
CONFIG_CPU_SW_DOMAIN_PAN=y
# Cette option de compatibilité avec l'ancienne ABI ARM est dangereuse et
# ouvre la voie à diverses attaques.
#CONFIG_OABI_COMPAT is unset
```

R27

M I R E Paramétrer les options de compilation spécifiques aux architectures ARM 64 bits

La liste ci-dessous présente les options de compilation du noyau recommandées pour les architectures de type ARM 64 bits.

Liste des options de compilation du noyau recommandées pour les architectures ARM 64 bits

```
# Interdit l'utilisation d'adresses mémoire en dessous d'une certaine valeur
# (contre-mesure contre les null pointer dereference).
CONFIG_DEFAULT_MMAP_MIN_ADDR=32768
# Rend non prédictible l'adresse de base du noyau, cette option complexifie
# la tâche d'un attaquant. L'entropie nécessaire à la génération de l'aléa
# doit être fournie par l'UEFI ou, à défaut, par le chargeur de démarrage.
CONFIG_RANDOMIZE_BASE=y
# Interdit les accès par le noyau à la mémoire utilisateur (mécanisme
# équivalent sur ARM à SMAP sur 86_64).
CONFIG_ARM64_SW_TTBR0_PAN=y
# Contre mesure à l'attaque Meltdown.
CONFIG_UNMAP_KERNEL_AT_EL0=y
```

6
Configuration système

Ce chapitre traite des recommandations à suivre lors de la première installation du système. Chaque système d'exploitation et distribution GNU/Linux adopte un cheminement qui lui est propre pendant cette étape. Quelques éléments de configuration vont cependant s'appliquer de manière quasi-universelle.

Objectif

Appliquer des paramètres de sécurité lors de la première installation d'un système GNU/Linux.

6.1 Partitionnement

Il est usuel de réserver des partitions dédiées aux services pouvant générer beaucoup de volumétrie afin d'éviter de saturer les partitions système. L'espace à réserver pour chaque partition dépend des cas d'usage : un serveur de fichiers aura besoin d'une volumétrie importante pour `/srv` ou `/var/ftp/`, tandis qu'un serveur de journaux sera plutôt concerné par la volumétrie utilisable pour `/var/log`.

Le partitionnement doit ainsi permettre de protéger et isoler les différents composants du système de fichiers. Il est par défaut souvent insatisfaisant : il ne tient pas suffisamment compte des options `nosuid` (ignore les bits `setuid`/`setgid`), `nodev` (ignore les fichiers spéciaux caractère ou bloc), et `noexec` (ignore les droits d'exécution).

R28

Partitionnement type

Le partitionnement type recommandé est le suivant :

Point de montage	Options	Description
/	<sans option>	Partition racine, contient le reste de l'arborescence
/boot	nosuid,nodev,noexec (noauto optionnel)	Contient le noyau et le chargeur de démarrage. Pas d'accès nécessaire une fois le boot terminé (sauf mise à jour)
/opt	nosuid,nodev (ro optionnel)	Packages additionnels au système. Montage en lecture seule si non utilisé
/tmp	nosuid,nodev,noexec	Fichiers temporaires. Ne doit contenir que des éléments non exécutables. Nettoyé après redémarrage ou préférablement de type tmpfs
/srv	nosuid,nodev (noexec,ro optionnels)	Contient des fichiers servis par un service type Web, FTP…
/home	nosuid,nodev,noexec	Contient les HOME utilisateurs.
/proc	hidepid=2 [14]	Contient des informations sur les processus et le système
/usr	nodev	Contient la majorité des utilitaires et fichiers système
/var	nosuid,nodev,noexec	Partition contenant des fichiers variables pendant la vie du système (mails, fichiers PID, bases de données d'un service)
/var/log [15]	nosuid,nodev,noexec	Contient les logs du système
/var/tmp	nosuid,nodev,noexec	Fichiers temporaires conservés après extinction

Information

Il faut noter que suivant les systèmes et distributions, certaines options de montage ne seront pas applicables de manière transitoire ; par exemple des logiciels, gestionnaires de paquets ou produits requièrent que les fichiers écrits dans /tmp ou /var doivent être exécutables. Dans ces cas, il est nécessaire d'adapter le partitionnement ou de mettre en place des alternatives ; par exemple pour les distributions dérivées de Debian [16]dont le /var/lib/dpkg nécessite des droits d'exécution, une alternative serait d'implémenter une procédure de maintenance durant laquelle les mises à jour sont installées, à l'image de ce que l'on trouve sur d'autres systèmes d'exploitation.

14. Pour les distributions sous systemd, le service systemd-journald du gestionnaire de système et de services systemd permet d'éviter la saturation des partitions avec notamment les options préfixées par « System » (cf. https://www.freedesktop.org/software/systemd/man/journald.conf.html#SystemMaxUse=)

15. Nécessite de configurer le service systemd-logind du gestionnaire de système et de services systemd (cf. https://wiki.debian.org/Hardening#Mounting_.2Fproc_with_hidepid)

16. https://www.debian.org

La partition `/boot` contient le noyau de démarrage ainsi que le ou les fichiers `System.map` contenant la table des symboles utilisée par celui-ci. Ce fichier est souvent parcouru par différents logiciels malveillants afin de construire plus facilement des « exploits » de code noyau. Le partitionnement idéal demande à ce que cette partition ne soit pas montée automatiquement au démarrage (option `noauto`), car son contenu est inutile en phase d'utilisation normale d'un système. En revanche, son montage est indispensable pour réaliser certaines actions d'administration par exemple, la mise à jour du noyau.

R29

M I R Restreindre les accès au dossier `/boot`

Lorsque c'est possible, il est recommandé de ne pas monter automatiquement la partition `/boot`.
Dans tous les cas, l'accès au dossier `/boot` doit être uniquement autorisé pour l'utilisateur `root`.

Attention

La modification du montage de `/boot` demande une adaptation des outils système à cette configuration particulière, notamment pour la mise à jour du noyau et du chargeur de démarrage.

Information

L'utilitaire bas niveau, `dpkg`, qui gère les paquets des distributions dérivées de `Debian` et qui est utilisé par l'utilitaire `apt` sur lequel reposent les gestionnaires de paquets, `aptitude` et `synaptic`, peut être configuré pour réaliser des commandes particulières avant ou après les commandes d'installation de paquets.

6.2 Comptes d'accès

Les systèmes d'exploitation Unix et dérivés, et notamment GNU/Linux, ont une séparation de privilèges qui repose principalement sur la notion de comptes d'accès à deux niveaux : les comptes d'utilisateur « classique » et le compte d'administrateur système (communément appelé `root`).

Si cette séparation manque de finesse, en particulier aujourd'hui, où un utilisateur dispose d'un accès assez large aux primitives système (tout en étant de plus en plus exposé à des attaques : vol de comptes et d'identifiants, fuite d'information…), l'analyse approfondie d'un expert est souvent requise pour mettre en exergue les vulnérabilités résiduelles dans le but d'en minimiser autant que possible les conséquences.

6.2.1 Comptes utilisateur

Une gestion rigoureuse des comptes utilisateur participe à la sécurisation du système au travers des principes essentiels de sécurité.

R30

Désactiver les comptes utilisateur inutilisés

Les comptes utilisateur inutilisés doivent être désactivés au niveau du système.

Le mot de passe des comptes utilisateur doit être choisi avec le plus grand soin et conformément aux recommandations actuelles.

R31

Utiliser des mots de passe robustes

Il est conseillé d'appliquer les recommandations de la note technique « *Recommandations relatives à l'authentification multifacteur et aux mots de passe* » [9].
Ce mot de passe doit être unique et propre à chaque machine.

R32

Expirer les sessions utilisateur locales

Les sessions utilisateur locales (console `TTY`, session graphique) doivent être verrouillées au bout d'un certain délai d'inactivité.

6.2.2 Comptes administrateur

Il est souvent d'usage de définir différents niveaux de privilèges sur le système en fonction des prérogatives des administrateurs. Certains peuvent avoir des responsabilités qui concernent le système, d'autres uniquement le site `Web` ou l'infrastructure de journalisation, ou encore les bases de données.

Le compte d'administrateur système (ou `root`) est impropre à ces usages. Il donne plein pouvoir à la personne qui y a accès. L'usage d'un tel compte générique ne facilite pas l'imputabilité lors d'un incident, et ne favorise pas un modèle fin de séparation des privilèges, par exemple entre différentes équipes d'administration.

Il est indispensable de dissocier les rôles sur le système, en particulier pour un administrateur : simple utilisateur ou administrateur avec des droits privilégiés. De plus, un administrateur peut intervenir sur plusieurs domaines techniques. En conséquence, des comptes distincts doivent être créés et utilisés selon le rôle (utilisateur ou administrateur) ainsi que des comptes d'administration distincts par domaine technique comme décrit dans le guide *Recommandations relatives à l'administration sécurisée des systèmes d'information* [6].

En toute logique, et pour éviter tout rejeu d'un secret potentiellement compromis, les secrets, par exemple un code PIN, un mot de passe, une clé privée… associés doivent être différents pour chaque compte.

R33

Assurer l'imputabilité des actions d'administration

Les actions d'administration nécessitant des privilèges doivent être tracées afin de pouvoir identifier l'administrateur à l'origine d'une activité malveillante.

Plusieurs approches permettent de répondre à cette problématique. Une première, la plus courante, consiste à utiliser des comptes dédiés pour chaque administrateur et à utiliser `sudo` pour toute commande privilégiée. Une seconde, plus puissante, consiste à identifier l'utilisateur de manière unique sur le système, puis à journaliser la création de tout processus avec `auditd`.
Plusieurs approches peuvent également être combinées pour avoir une meilleure traçabilité.

Usage de comptes d'administration dédiés et de `sudo`

Chaque administrateur doit disposer d'un ou plusieurs comptes nominatifs d'administration dédiés (local ou distant) et ne pas utiliser le compte d'administration système (ou `root`) comme compte d'accès pour l'administration. Les secrets d'authentification doivent être différents suivant le compte utilisé.
Le compte d'administration système (ou `root`) doit être désactivé. Les actions nécessitant des droits privilégiés devront donc reposer sur `sudo` qui journalisera les commandes réalisées.
La désactivation d'un compte peut passer par l'invalidation de ce compte au niveau de son mot de passe (suppression du champ `pw_passwd` dans le `shadow` et `shell` de `login` à `/bin/false`). Ces mesures n'empêchent cependant pas un administrateur d'exécuter `sudo bash` et d'obtenir un shell `root`. Elles sont donc à combiner avec une mesure organisationnelle d'interdiction de l'usage du compte `root`.

```
# Verrouillage d'un compte
usermod -L -e 1 <compte>
# Désactivation du shell de login
usermod -s /bin/false <compte>
```

Journalisation de la création de tout processus

La journalisation de la création de tout nouveau processus permet de reconstituer l'enchaînement des opérations effectuées sur le système. Il est donc ensuite possible d'imputer une action à un administrateur sous réserve que celui-ci ait été identifié de manière nominative et unique. Cette identification peut se faire entre autres par l'usage d'un compte dédié ou d'identifiants dédiés (clef SSH utilisée pour la connexion…). Il faudra journaliser cet événement avec les données d'identification et le PID du processus initial.
Il est possible de journaliser la création de processus sur le système avec les règles `auditd` suivantes :

```
-a exit,always -F arch=b64 -S execve,execveat
-a exit,always -F arch=b32 -S execve,execveat
```

Il est important de réaliser que le volume de log généré par cette méthode peut être très important en fonction des tâches effectuées sur le système. Il est donc conseillé d'utiliser cette méthode en combinaison d'un export de logs régulier comme conseillé dans le guide *Recommandations de sécurité pour l'architecture d'un système de journalisation* [7].

6.2.3 Comptes de service

La grande majorité des systèmes d'exploitation Unix et dérivés isolent les applications et services les uns des autres en les dédiant chacun à un compte utilisateur propre. Par exemple un serveur `Web`, une fois démarré, utilise les privilèges d'un utilisateur « `Web` », par exemple `www` ou `www-data`, tandis que le serveur de noms s'exécute sous un compte distinct, par exemple `named`. La majorité des services n'ont pas besoin d'ouvrir de session pour leur bon fonctionnement. Les comptes de service doivent donc être désactivés pour éviter une connexion à ce compte.

R34

Désactiver les comptes de service

Les comptes de service doivent être désactivés.

Il est important de noter que certains services peuvent se déclarer avec le compte `nobody`. Quand ceux-ci sont plusieurs à adopter un tel comportement, ils se retrouvent à partager les mêmes identifiants et donc les mêmes privilèges au niveau du système d'exploitation. Un serveur `Web` et un annuaire utilisant le compte `nobody` peuvent donc se contrôler mutuellement et altérer leur exécution l'un l'autre : modification de configuration, envoi de signaux, privilèges `ptrace`…

R35

Utiliser des comptes de service uniques et exclusifs

Chaque service métier (nginx, php, mysql, …) ajouté par l'administrateur doit posséder son propre compte système dédié.

6.3 Contrôle d'accès

Le contrôle d'accès consiste à s'assurer qu'un compte d'accès a des droits minimaux en vue d'accéder à une ressource donnée. Bien que le contrôle d'accès permette d'effectuer du cloisonnement, l'approche choisie est généralement différente de celle adoptée par les mécanismes de virtualisation. En effet, le contrôle d'accès laisse souvent la référence à un objet système visible à l'application et retourne une erreur en cas de privilège insuffisant pour y accéder, alors qu'un système reposant sur la virtualisation cloisonne l'application par l'absence de référence à cet objet (pointeur, chemin d'accès…).

Le contrôle d'accès sous les systèmes Unix et dérivés repose historiquement sur la notion d'utilisateur et de groupe d'utilisateurs. D'autres mécanismes sont apparus, notamment au travers des LSM dont `SELinux` détaillé au paragraphe 6.3.3 et `AppArmor` détaillé au paragraphe 6.3.2 sont les plus connus et utilisés, en vue de permettre un contrôle supplémentaire plus fin des droits. Mais la séparation des privilèges basée sur l'utilisateur et les groupes d'utilisateurs reste encore la plus usitée. Elle peut s'accompagner de mécanismes de cloisonnement (mécanismes compatibles `UNIX`, conteneur…) ou de filtrage (`seccomp`) en vue de limiter les accès au système d'exploitation sous-jacent.

6.3.1 Modèle traditionnel Unix

Le modèle d'accès historique repose sur la notion d'utilisateur qui est connu par le système au travers d'un identifiant unique, appelé UID. Chaque ressource (processus, fichier, répertoire…) est associée à un UID (son propriétaire), qui couplé à des droits, va autoriser ou refuser un accès à un utilisateur.

Plusieurs UID peuvent être réunis en groupe auquel un identifiant unique peut être associé : un GID (`Group IDentifier` ou identifiant de groupe). Le principe de gestion des accès reste cependant similaire aux UID.

Ce modèle d'accès est un DAC (`Discretionary Access Control` ou contrôle d'accès discrétionnaire). Il est discrétionnaire car l'utilisateur propriétaire d'une ressource spécifie les droits d'accès, avec des droits en lecture, en écriture et d'exécution chacun donné pour :

- l'utilisateur propriétaire de la ressource ;
- le groupe propriétaire de la ressource ;
- le reste du monde.

Lorsqu'un utilisateur crée un fichier ou un répertoire, ses droits d'accès sont définis à partir du UMASK (`User file creation mode MASK` ou masque de création de fichier par l'utilisateur) du processus créateur. Ce UMASK est par défaut à la valeur `0022`, ce qui est très permissif. Ceci assignera en effet les droits d'accès `0644` au fichier créé (lecture par tout le monde) et `0755` au répertoire créé (lecture et exécution par tout le monde). Il est donc recommandé de changer la valeur par défaut du UMASK pour une valeur plus restrictive.

Il n'est pas possible définir cette valeur pour l'ensemble du système mais il est possible de le faire pour chaque composant. Il est important de la modifier pour les shells utilisateurs et au cas par cas pour les services du système.

R36 — 

Modifier la valeur par défaut de UMASK

La valeur par défaut de UMASK des shells doit être positionnée à `0077` pour permettre un accès au fichier ou au répertoire créé en lecture et écriture uniquement à l'utilisateur propriétaire. Celle-ci peut être spécifiée dans le fichier de configuration `/etc/profile` que la plupart des shells (bash, dash, ksh…) utiliseront.

La valeur par défaut de UMASK des services doit être étudiée au cas par cas mais devrait généralement être à `0027` (ou plus restrictif). Ceci permet un accès au fichier ou au répertoire créé en lecture uniquement à l'utilisateur propriétaire et à son groupe et un accès en écriture au propriétaire. Pour les services `systemd`, cette valeur peut être spécifiée dans le fichier de configuration du service avec la directive `UMask=0027`.

Le modèle d'accès DAC est encore majoritaire aujourd'hui et reste appliqué par défaut sous les systèmes Unix et dérivés. Il présente cependant d'importantes limitations :

- les droits sont à la discrétion du propriétaire, ce qui peut ne pas convenir aux environnements contraints par une politique de sécurité ;

- le propriétaire peut être incapable de donner les bons droits sur ses ressources, ce qui offre des possibilités d'accès à des données confidentielles (voire des compromissions) ;
- l'isolation entre les utilisateurs est grossière, les droits par défaut étant souvent laxistes ;
- il n'y a que deux niveaux de privilèges sur le système : les comptes d'utilisateur « classique » et le compte d'administrateur (ou `root`). Le changement d'utilisateur requiert l'usage d'exécutables avec des droits spéciaux, par exemple `setuid root` ;
- il ne permet pas de retirer des privilèges particuliers à un processus suivant le contexte : le navigateur `Web` et le traitement de texte d'un même utilisateur auront autant de privilèges l'un que l'autre sur les ressources ;
- la surface d'attaque est grande, un utilisateur standard a accès à un grand nombre d'informations sur le système d'exploitation sous-jacent.

D'autres modèles d'accès, comme le MAC (`Mandatory Access Control` ou contrôle d'accès obligatoire), offrent plus de possibilités, mais au prix d'un investissement et d'une complexité plus élevés.

R37

Utiliser des fonctionnalités de contrôle d'accès obligatoire MAC

Il est recommandé d'utiliser les fonctionnalités de `Mandatory Access Control` ou contrôle d'accès obligatoire (MAC) en plus du traditionnel modèle utilisateur Unix, `Discretionary Access Control` ou contrôle d'accès discrétionnaire (DAC), voire éventuellement de les combiner avec des mécanismes de cloisonnement.

Les limitations du DAC ont contribué à l'émergence d'outils de gestion de délégation de droit, le plus connu étant `sudo` qui est un utilitaire installé lorsqu'il y a un besoin de déléguer des droits et privilèges à différents utilisateurs. Cette délégation repose sur la possibilité pour un utilisateur donné d'exécuter une commande préalablement définie avec les privilèges d'un autre utilisateur. Afin de pouvoir réaliser cela, `sudo` est un exécutable `setuid root`. Il est donc important de se préoccuper de sa sécurité à deux niveaux :

- au niveau du durcissement, afin d'éviter à un utilisateur malveillant de pouvoir exploiter les vulnérabilités du binaire ;
- au niveau de sa configuration, où le fait de donner le droit à un utilisateur d'exécuter certaines commandes peut lui attribuer plus de privilèges et de prérogatives qu'initialement nécessaires.

Le fonctionnement de `sudo` repose en très grande partie sur le modèle traditionnel Unix pour fonctionner, qu'il enrichit avec une logique de transition plus fine que ne le permet originellement l'utilitaire `su`. Les éléments ci-dessous visent à donner quelques recommandations et pistes à étudier en vue d'utiliser et configurer `sudo` de telle sorte qu'il offre le moins de contournement et d'effets de bords possibles.

Ces recommandations s'appliquent aussi bien pour les cas où `sudo` est utilisé pour l'exécution de commandes privilégiées par un utilisateur « classique », que pour la restriction de privilèges quand un utilisateur privilégié (tel que `root`) souhaite exécuter une commande avec des droits de moindres privilégiés, par précaution.

`sudo` est généralement installé par défaut avec des droits ouverts permettant à n'importe quel utilisateur de l'utiliser (droit d'exécution pour `tout le monde`). `sudo` étant un exécutable privilégié et complexe de par sa configuration, il est préférable de restreindre ses droits d'exécution à un groupe d'utilisateurs dédiés. Cela réduit la surface d'attaque, notamment quand le système et le binaire sont victimes de vulnérabilités exploitables, par exemple CVE-2019-14287 (`potential bypass of Runas user restrictions`[17]) ou CVE-2021-3156 (`buffer overflow in command line unescaping`[18]), par n'importe quel utilisateur.

R38

Créer un groupe dédié à l'usage de `sudo`

Un groupe dédié à l'usage de `sudo` doit être créé. Seuls les utilisateurs membres de ce groupe doivent avoir le droit d'exécuter `sudo`.

Exemple de création d'un groupe `sudo` dédié

Un cas envisageable est de créer un groupe dédié, par exemple `sudogrp`, et de lui attribuer les droits pour exécuter `sudo`.
Seuls les membres de ce groupe pourront ainsi y faire appel :

```
# ls -al /usr/bin/sudo
-rwsr-x---. 2 root sudogrp [...] /usr/bin/sudo
```

Attention

Cette modification peut être écrasée lors d'une mise à jour du paquet de sudo. Il est donc conseillé de vérifier les droits sur le fichier après une mise à jour, et réappliquer les bons droits au besoin. Cette opération peut être automatisée sur la plupart des gestionnaires de paquets[19].

La configuration de `sudo` se fait à l'aide du fichier `/etc/sudoers` accessible à partir de la commande `visudo`. Il contient notamment un ensemble de directives permettant à `sudo` de connaître les commandes qu'un utilisateur aura le droit d'exécuter ou pas, éventuellement en fonction du nom d'hôte de la machine.

Il est particulièrement difficile de fournir un ensemble de recommandations capables de couvrir l'ensemble des situations dans lesquelles `sudo` peut être utilisé. En effet, le nombre de commandes disponibles, l'architecture du système, les services installés et leurs configurations, rendent quasiment impossible l'établissement d'une configuration standard sécurisée.

Cependant quelques règles de bonnes pratiques doivent être respectées afin d'éviter autant que possible des erreurs de configuration pouvant conduire à des contournements de la politique de sécurité. Les valeurs par défaut des directives de configuration `sudo` doivent donc être remplacées par des valeurs plus sûres, quitte à être ensuite surchargées au cas par cas. Les risques associés à cette surcharge doivent être étudiés pour chaque règle. La lecture de la page man de `sudoers` est fortement recommandée, notamment la partie traitant des échappements `shell`.

17. https ://www.sudo.ws/alerts/minus_1_uid.html
18. https ://www.sudo.ws/alerts/unescape_overflow.html
19. https://manpages.debian.org/bullseye/apt/apt.conf.5.fr.html

R39

Modifier les directives de configuration `sudo`

Les directives suivantes doivent être activées par défaut :

noexec	appliquer le tag **NOEXEC** par défaut sur les commandes
requiretty	imposer à l'utilisateur d'avoir un `tty` de login
use_pty	utiliser un `pseudo-tty` lorsqu'une commande est exécutée
umask=0077	forcer `umask` à un masque plus restrictif
ignore_dot	ignorer le « . » dans `$PATH`
env_reset	réinitialiser les variables d'environnement

Directives pour le fichier de configuration `/etc/sudoers`

La liste des directives sont présentées telles que rencontrées dans le fichier de configuration `/etc/sudoers` :

```
Defaults noexec,requiretty,use_pty,umask=0027
Defaults ignore_dot,env_reset
```

Il est ensuite possible de surcharger la valeur par défaut si nécessaire lors de la définition d'un droit (1) ou pour tout un groupe (2).

```
myuser ALL = EXEC: /usr/bin/mycommand    # (1)
Defaults:%admins !noexec                 # (2)
```

Le reste du fichier est ensuite majoritairement constitué de règles permettant de déclarer pour un utilisateur ou un groupe donné la liste des commandes qu'il est en mesure d'exécuter, le tout aidé de spécifications d'alias (`User_Alias`, `Runas_Alias`…) lorsque la politique de gestion de droit devient complexe. La vérification du droit d'exécution ou non d'une commande repose sur une comparaison de chaîne entre la commande souhaitée par l'utilisateur et la spécification présente dans le fichier de configuration `sudoers`. Le modèle le plus simple est le suivant :

```
utilisateur   hostname = ( utilisateur-cible ) commande, [...]
```

Il est courant que la commande à exécuter ne requière pas un niveau de privilèges élevés (édition d'un fichier dont le propriétaire n'est pas `root`, envoi d'un signal à un processus non privilégié…). Afin de limiter toute tentative d'escalade de privilèges au travers d'une commande exécutée, il est préférable que des droits d'utilisateur « classique » lui soient appliqués.

R40

Utiliser des utilisateurs cibles non-privilégiés pour les commandes `sudo`

Les utilisateurs cibles d'une règle doivent autant que possible être des utilisateurs non privilégiés (c'est-à-dire non `root`).

Des commandes fonctionnellement riches peuvent être exécutées au travers de `sudo`, tels des éditeurs de texte (`vi`). Elles peuvent permettre l'exécution de sous commandes dans l'environnement créé par `sudo` et ainsi donner la possibilité à un utilisateur d'exécuter d'autres logiciels avec des privilèges non prévus à l'origine. Cela peut ainsi conduire à des contournements de la politique de sécurité, voire des escalades de privilèges.

À cette fin, `sudo` permet de surcharger les différentes fonctions afin d'exécuter d'autres logiciels. Cela permet, par exemple, de bloquer la création triviale d'un sous `shell` au travers de `vi`. Il est cependant important de noter que cette protection est imparfaite et ne fait que surcharger les fonctions permettant de créer ces nouveaux processus. Le processus reste libre d'exécuter les appels système qu'il souhaite, `execve` notamment, et ainsi contourner ce mécanisme de protection.

R41

Limiter l'utilisation de commandes nécessitant la directive EXEC

Les commandes nécessitant l'exécution de sous-processus, directive `EXEC` doivent être explicitement listées et réduites autant que possible au strict minimum.

Spécifier des droits d'accès par négation, approche de type liste d'interdiction, est inefficace et peut être facilement contourné.

Exemple

On s'attend, par exemple, à ce qu'une spécification du type suivant interdise l'exécution du `shell` :

```
# A éviter absolument, cette règle est trivialement contournable!
user   ALL = ALL,!/bin/sh
```

Il n'en est rien, il suffit de copier le binaire `/bin/sh` sous un autre nom pour que celui-ci devienne exécutable par cette règle grâce à la directive `ALL`.

R42

Bannir les négations dans les spécifications `sudo`

Les spécifications de commande ne doivent pas faire intervenir de négation.

Afin de déterminer si une commande est exécutable par un utilisateur, `sudo` réalise une comparaison de chaîne entre la commande désirée et les spécifications correspondantes dans le `sudoers` suivant les règles du `globbing shell`. Cette évaluation inclut les arguments.

Les arguments permettent de modifier de façon assez importante le comportement d'un logiciel, que cela soit en matière d'opérations réalisées (lecture, écriture, suppression…) ou de ressources accédées (chemin d'accès dans une arborescence). Afin d'éviter toute possibilité de détournement d'une commande par un utilisateur, les ambiguïtés doivent être levées au niveau de sa spécification.

R43

Préciser les arguments dans les spécifications `sudo`

Toutes les commandes du fichier de configuration `sudoers` doivent préciser strictement les arguments autorisés à être utilisés pour un utilisateur donné.
L'usage de `*` (`wildcard`) dans les règles doit être autant que possible évité. L'absence d'arguments auprès d'une commande doit être spécifiée par la présence d'une chaîne vide (`""`).

Exemple

Sur certains systèmes, les évènements noyau ne sont accessibles que par `root`. Si un utilisateur doit néanmoins posséder les privilèges lui permettant de les afficher, il faut restreindre ses arguments afin d'éviter qu'il puisse lire des fichiers arbitraires au travers de l'option `--file` :

```
user  ALL = dmesg ""
```

Attention

Il est important de noter que permettre à un utilisateur non privilégié d'exécuter sous l'identité `root` un logiciel susceptible de réaliser une écriture arbitraire (rendue possible par l'écriture trop laxiste d'une règle) reviendra dans les faits à donner à cet utilisateur les privilèges de `root` car celui-ci pourra, par diverses méthodes (manipulation des fichiers de mots de passe, modification de binaires `setuid root`…), obtenir les privilèges de `root` sans le contrôle d'accès et d'exécution que permet `sudo`.

Il est assez courant de permettre l'édition d'un fichier par un utilisateur au travers de `sudo` en appelant directement un éditeur de texte. Un éditeur de texte est un logiciel fonctionnellement riche, pouvant ouvrir et éditer d'autres fichiers par des commandes internes voire même exécuter un script `shell`. Certains éditeurs peuvent donc offrir un environnement privilégié complet à un utilisateur. L'éditeur de texte `sudoedit` s'affranchit de ces problématiques en laissant le soin à l'éditeur de modifier un fichier identique temporaire avec les droits de l'utilisateur courant, puis de remplacer le fichier pointé par ce même fichier temporaire une fois l'opération terminée. L'éditeur ne s'exécute donc qu'avec les droits de l'utilisateur courant et jamais avec les privilèges de l'utilisateur cible.

R44

Éditer les fichiers de manière sécurisée avec `sudo`

L'édition d'un fichier par `sudo` doit être réalisée au travers de l'éditeur de texte `sudoedit`.

6.3.2 AppArmor

`AppArmor` est un des LSM fréquemment rencontrés sous GNU/Linux, et celui utilisé, par défaut, sur les variantes de SUSE [20] (dont OpenSUSE [21]) et Ubuntu [22]. C'est un modèle d'accès MAC où une autorité décide des accès d'une application sans que celle-ci puisse les altérer.

Sa documentation est directement accessible sur le site du projet [11]. Sa configuration repose sur des spécifications de droit s'appuyant sur des chemins dans l'arborescence du système de fichiers, et appliquées pour chaque exécutable. Il est donc possible de mettre en place des profils `AppArmor` pour des applications spécifiques sans avoir à se soucier des interactions entre le LSM et les autres applications ou le système. Ceci le rend relativement simple à prendre en main et à intégrer à un système.

20. https://www.suse.com
21. https://www.opensuse.org
22. https://ubuntu.com

En plus des spécifications d'accès, AppArmor permet de bloquer l'usage des capacités POSIX, ainsi que de restreindre l'usage réseau (directive network). Ces restrictions restent cependant sommaires et n'ont pas la richesse d'expression d'iptables ou nftables [23] qui sont les utilitaires système permettant de configurer les règles de pare-feu sous GNU/Linux.

Lorsqu'un exécutable sous AppArmor appelle un autre exécutable, son profil de sécurité permet de déclarer la façon dont la transition doit s'effectuer (héritage du profil actuel, activation d'un autre profil de sécurité, sortie du confinement…).

Bien que simple à appréhender pour un administrateur habitué au modèle d'accès traditionnel Unix, AppArmor ne permet pas d'attacher de labels de sécurité à un flux ou à des messages. Il se concentre essentiellement sur la spécification d'accès et de privilèges pour des exécutables, la notion de label de sécurité en est absente.

Une grande partie des distributions GNU/Linux qui l'utilisent fournissent, généralement pour les exécutables les plus sensibles (syslogd, ntpd…) des profils AppArmor par défaut dans le répertoire /etc/apparmor.d/, un fichier par exécutable. Les actions, y compris les refus d'accès, sont enregistrées par auditd détaillé au paragraphe 7.2.3. Les profils peuvent être activés en mode enforce ou en mode complaining. Dans le mode complaining, les actions qui auraient été bloquées sont enregistrées sans être bloquées réellement.

R45

Activer les profils de sécurité AppArmor

Tout profil de sécurité AppArmor présent sur le système doit être activé en mode enforce par défaut quand la distribution en offre le support et qu'elle n'exploite pas de module de sécurité autre que AppArmor.

Information

Sur les distributions GNU/Linux récentes, AppArmor est activé par défaut. Pour s'assurer que AppArmor est bien actif, utiliser aa-status :

```
# aa-status
apparmor module is loaded.
40 profiles are loaded.
23 profiles are in enforce mode.
...
14 processes have profiles defined.
12 processes are in enforce mode.
...
0 processes are unconfined but have a profile defined.
```

Il faut notamment vérifier que les processus en cours d'exécution sur le système et pour lesquels un profil est déclaré soient confinés en mode enforce par AppArmor.

6.3.3 SELinux

Security-Enhanced Linux, abrégé SELinux, est un LSM largement utilisé dans les distributions dérivées de Red Hat.

23. https://www.netfilter.org

SELinux repose sur l'usage de « labels » assignés à des objets (fichiers, processus, *sockets*…) qui permettent, suivant le « domaine » auquel un processus est confiné, d'autoriser ou de refuser un accès à une ressource.

Information

Au niveau du système de fichiers, les `labels` sont stockés comme attributs étendus et peuvent être obtenus avec la commande :

```
# ls -Z
```

Ceux des processus sont visibles avec la commande :

```
# ps -Z
```

L'usage des « labels » permet de déclarer des « domaines » de responsabilité et donc de connaître à tout instant les actions que peut effectuer un processus sur une ressource donnée. SELinux permet aussi de définir les « transitions » d'un domaine à un autre lorsqu'un utilisateur requiert des privilèges d'accès spécifiques. La granularité d'accès est bien plus fine que dans AppArmor, au détriment d'une complexité plus importante.

La documentation de SELinux est accessible sur le page wiki du projet [19].

Cette section se limite à donner quelques vérifications pour que tout administrateur désireux d'utiliser les profils fournis par sa distribution puisse le faire correctement.

SELinux propose les trois modes de fonctionnement suivants :

- le mode `disabled` dans lequel les règles SELinux ne sont pas chargées et aucun message d'accès n'est journalisé ;
- le mode `permissive` dans lequel les règles SELinux sont chargées et interrogées mais aucun accès n'est refusé, et les messages d'erreur d'accès sont journalisés ;
- le mode `enforcing` dans lequel les accès sont conditionnés aux règles SELinux.

Les politiques SELinux disponibles par défaut sur les distributions sont les suivantes :

minimale	(`minimum`) politique minimale qui ne confine qu'un nombre très restreint de services ;
ciblée	(appelée `targeted` ou `default`) politique cloisonnant chaque service système dans un domaine. Les utilisateurs interactifs ne sont pas cloisonnés ;
multi-niveaux	(`mls`) politique incluant la politique ciblée et qui propose en sus l'utilisation de niveaux de classification hiérarchisés (sensibilité).

L'utilisation de la politique `minimale` est déconseillée car les éléments du système ne sont pas confinés dans des domaines distincts. Selon ses créateurs, cette politique doit être utilisée dans les environnements contraints (peu de mémoire) ou pour des tests [18].

La politique `multi-niveaux` a été créée pour traiter des informations de sensibilités différentes sur un même système en assurant le principe de « besoin d'en connaître ». Il est important de

noter qu'elle ne permet pas de traiter des informations d'un niveau de sensibilité particulier sur des systèmes dont le niveau de sensibilité est inférieur ou incompatible (voir le document [10] pour de plus amples informations). Son fonctionnement ne sera pas abordé ici.

La suite de cette section va se concentrer sur la politique par défaut `targeted`. Chaque service système est confiné dans un domaine distinct. Plusieurs instances d'un même service exécuté dans un même domaine peuvent être séparées à l'aide des catégories. Les utilisateurs interactifs sont par défaut associés à un domaine non confiné (`unconfined_t`) qui n'impose pas de restrictions supplémentaires. Le modèle de contrôle d'accès reste dans ce cas très proche du DAC.

R46

M I R E Activer SELinux avec la politique targeted

Il est recommandé d'activer `SELinux` en mode `enforcing` et d'utiliser la politique `targeted` quand la distribution en offre le support et qu'elle n'exploite pas de module de sécurité autre que `SELinux`.

Information

L'activation de `SELinux` avec la politique `targeted` passe généralement par les étapes suivantes :

1. Vérifier la valeur des variables `SELINUX` et `SELINUXTYPE` dans le fichier de configuration `/etc/selinux/config` :

```
# grep ^SELINUX /etc/selinux/config
SELINUX=enforcing
SELINUXTYPE=targeted
```

2. Si la valeur de `SELINUX` n'est pas `enforcing` ou si la valeur de `SELINUXTYPE` n'est pas `targeted` (ou `default`), éditer le fichier pour les corriger.

3. Si `SELinux` n'était pas en mode `enforcing`, il est nécessaire de vérifier que des `labels SELinux` sont correctement attribués aux fichiers sur l'ensemble du système. Cette vérification peut être programmée pour être exécutée de façon automatique au prochain démarrage avec la commande :

```
# fixfiles onboot
```

4. Si la politique utilisée n'était pas la politique `targeted`, recharger la politique utilisée par le noyau avec la commande :

```
# semodule -R
```

5. Il peut s'avérer nécessaire de redémarrer le système pour que le changement de politique ou l'activation de `SELinux` soient correctement pris en compte.

6. Enfin, pour s'assurer que `SELinux` est actif et avec les bons paramètres :

```
# sestatus
SELinux status:                 enabled
...
Loaded policy name:             targeted
Current mode:                   enforcing
...
```

Par défaut, les droits des utilisateurs interactifs ne sont pas restreints dans la politique `targeted`. Mais il est possible de confiner sélectivement les utilisateurs n'ayant pas besoin d'effectuer des actions d'administration sur un système.

Exemple

La commande suivante permet d'associer à un utilisateur interactif à l'utilisateur `SELinux` :

```
# usermod -Z user_u <utilisateur>
```

R47

M I R E Confiner les utilisateurs interactifs non privilégiés

Les utilisateurs interactifs non privilégiés d'un système doivent être confinés en leur associant un utilisateur `SELinux` confiné.

Le comportement d'une politique de sécurité peut être modifié au travers de variables booléennes. Certaines sont utiles d'un point de vue sécurité.

R48

M I R E Paramétrer les variables `SELinux`

Il est recommandé d'appliquer les politiques de sécurité suivantes :

- Interdit aux processus de rendre exécutable leur tas - `heap` -;
- Interdit aux processus d'avoir une zone mémoire avec des droits en écriture - `w` - et en exécution - `x` -;
- Interdit aux processus le droit de rendre leur pile - `stack` - exécutable;
- Interdit le chargement dynamique des modules par n'importe quel processus;
- Interdit les logins SSH de se connecter directement en rôle sysadmin.

Exemple

Sur les distributions dérivées de `Debian`, l'application de ces politiques de sécurité passe par la modification, avec la commande `setsebool`, des valeurs par défaut des variables suivantes :

```
#setsebool -P allow_execheap=off
#setsebool -P allow_execmem=off
#setsebool -P allow_execstack=off
#setsebool -P secure_mode_insmod=on
#setsebool -P ssh_sysadm_login=off
```

Il en existe de nombreuses autres. Les commandes `getsebool -a` ou `semanage boolean --list` permettent d'obtenir des informations sur les variables booléennes définies et leur état actuel dans la politique utilisée sur un système.

Information

Une grande partie de ces variables sont documentées dans le wiki de CentOS [17].

Tout comme `AppArmor`, les actions sont enregistrées par `auditd` détaillé au paragraphe 7.2.3.

L'usage des outils de manipulation et de débogage de politique `SELinux` doit être limité aux machines de développement, sur lesquelles l'interprétation d'une erreur remontée dans les logs d'audit `SELinux` peut être effectuée.

R49 M I R E **Désinstaller les outils de débogage de politique SELinux**

Les outils de manipulation et de débogage de politique `SELinux` ne doivent pas être installés sur une machine en production.

Attention

Le service `setroubleshootd` doit être désactivé et les paquets suivants désinstallés :

- `setroubleshoot`,
- `setroubleshoot-server`,
- `setroubleshoot-plugins`.

6.4 Fichiers et répertoires

Une attention toute particulière doit être portée :

- aux fichiers et aux répertoires contenant des éléments secrets, comme des mots de passe, des empreintes de mot de passe, des clés secrètes ou privées ;
- aux fichiers IPC (`Inter-Process Communication` ou communication inter-processus) nommés, comme les *sockets* ou les *pipes*, qui permettent à différents processus d'établir des canaux de communication entre eux ;
- aux répertoires de stockage temporaires auxquels tout le monde à accès ;
- aux exécutables possédant des droits spéciaux, comme `setuid` (ou `suid` abrégé `Set User ID`) ou `setgid` (ou `sgid` abrégé `Set Group ID`).

Des droits adaptés vont devoir s'appliquer rationnellement à chacun de ces types.

6.4.1 Fichiers et répertoires sensibles

Les fichiers et les répertoires contenant des clés privées ou secrètes, des mots de passe, des empreintes…(voire les journaux) sont considérés comme des fichiers ou répertoires sensibles.

Quand ces fichiers contiennent les mots de passe système ou les empreintes de mots de passe système, ils ne doivent être lisibles que par `root`. En revanche, les fichiers publics qui contiennent la liste des utilisateurs sont lisibles par tout le monde, mais sont modifiables uniquement par `root`.

Fichiers sensibles

Quelques exemples de fichiers contenant des éléments sensibles :

```
-rw-r----- root root  /etc/gshadow
-rw-r----- root root  /etc/shadow
-rw------- foo  users /home/foo/.ssh/id_rsa
...
```

Même lorsque ces fichiers ont leur contenu protégé par des mesures supplémentaires (chiffrement, empreintes pour les mots de passe…), il n'en demeure pas moins nécessaire d'en restreindre l'accès par principe de défense en profondeur.

R50

M I Restreindre les droits d'accès aux fichiers et aux répertoires sensibles

Les fichiers et les répertoires sensibles ne doivent être lisibles que par les utilisateurs ayant le strict besoin d'en connaître.

Information

Le schéma d'analyse des fichiers ou répertoires sensibles est le suivant :

1. les fichiers ou répertoires sensibles système doivent avoir comme propriétaire `root` afin d'éviter tout changement de droit par un utilisateur non privilégié ;
2. les fichiers ou répertoires sensibles accessibles à un utilisateur différent de `root` (par exemple, la base des mots de passe d'un serveur `Web`) doivent avoir comme propriétaire cet utilisateur (par exemple, l'utilisateur associé au serveur `Web`) qui doit être membre d'un groupe dédié (par exemple, le groupe `www-group`) et qui aura un droit d'accès en lecture seule à ce fichier ou répertoire ;
3. le reste des utilisateurs ne doit posséder aucun droit sur les fichiers ou répertoires sensibles.

Tous les fichiers sensibles, générés ou installés lors l'installation du système, et ceux qui concourent aux mécanismes d'authentification (certificats d'autorité racine, clés publiques, certificats…) doivent être mis en place dès l'installation du système.

R51

M I R Changer les secrets et droits d'accès dès l'installation

Tous les fichiers sensibles et ceux qui concourent aux mécanismes d'authentification doivent être mis en place dès l'installation du système. Si des secrets par défaut sont préconfigurés, ils doivent alors être remplacés pendant, ou juste après, la phase d'installation du système.

6.4.2 Fichiers IPC nommés, sockets ou pipes

Les applications peuvent échanger des informations au travers de *socket*. La plupart du temps ces *sockets* sont établies au niveau réseau (`IP`). Quand il s'agit d'établir des connexions entre processus

locaux, des alternatives existent comme les *pipes* ou les *sockets* locales Unix.

Il faut prendre garde au fait que les droits d'accès qui s'appliquent à une *socket* locale peuvent être ceux du répertoire la contenant et non ceux de la *socket* directement. Bien que certains systèmes honoreront tout de même ces permissions, la norme `Portable Operating System Interface` (`POSIX`) ne l'impose pas.

R52

Restreindre les accès aux sockets et aux pipes nommées

Les *sockets* et *pipes* nommées doivent être protégées en accès par un répertoire possédant des droits appropriés. Les droits de la *socket* ou du *pipe* nommé doivent également restreindre les accès.

Information

Plusieurs commandes permettent d'obtenir des informations sur les *sockets* en cours d'utilisation sur le système. Suivant la politique de sécurité appliquée et les modules chargés, le résultat affiché peut ne pas être exhaustif.
Les commandes suivantes listent l'ensemble des *sockets* et les informations des processus associées pour des *sockets* locales :

```
# sockstat
# ss -xp
```

La commande suivante liste les renseignements sur l'utilisation des ressources IPC :

```
# ipcs
```

La commande suivante liste l'ensemble des mémoires virtuelles partagées par les processus et leurs droits d'accès associés :

```
# ls /dev/shm
```

La commande suivante permet d'obtenir des informations détaillées sur les `I/O` et IPC pour les processus du système :

```
# lsof
```

Attention

Les *sockets* locales ne doivent pas être créées à la racine d'un répertoire temporaire accessible en écriture à tous.

6.4.3 Droits d'accès

Les fichiers ou répertoires sans utilisateur ou groupe connu peuvent être incorrectement attribués à un utilisateur ou à un groupe lors de la création d'un nouveau compte utilisateur ou d'un nouveau groupe. Il faut donc s'assurer qu'aucun fichier ou répertoire ne soit dans cette situation sur le système.

R53

M Éviter les fichiers ou répertoires sans utilisateur ou sans groupe connu

Les fichiers ou répertoires sans utilisateur ou groupe identifiable par le système doivent être analysés et éventuellement corrigés afin d'avoir un propriétaire ou un groupe connu du système.

Information

La commande suivante permet de lister l'ensemble des fichiers qui n'ont plus d'utilisateur ou de groupe associé :

```
# find / -type f \( -nouser -o -nogroup \) -ls 2>/dev/null
```

Les répertoires accessibles à tous en écriture sont dans la majorité des cas utilisés comme zones de stockage temporaire, dans lequel une application enregistre des données pour les traiter. De nombreuses applications les utilisent de façon incorrecte. Les fichiers temporaires peuvent alors être détournés de leur fonction première et être exploités comme rebond, par exemple pour une escalade de privilèges.

Un palliatif est alors d'activer le `sticky bit` sur le répertoire afin que seul l'utilisateur ou l'application qui a créé le fichier soit en mesure de le supprimer.

Cela évite ainsi qu'un utilisateur ou une application puisse arbitrairement supprimer ou remplacer les fichiers temporaires d'un autre utilisateur ou d'une autre application.

R54

M Activer le `sticky bit` sur les répertoires inscriptibles

Tous les répertoires accessibles en écriture par tous doivent avoir le `sticky bit` positionné.

Attention

Il faut aussi s'assurer que le propriétaire du répertoire est bien `root`, sans quoi l'utilisateur propriétaire pourra à loisir modifier son contenu et ce malgré le `sticky bit`.

Information

la commande suivante permet de lister l'ensemble des répertoires modifiables par tous et sans `sticky bit` :

```
# find / -type d \( -perm -0002 -a \! -perm -1000 \) -ls 2>/dev/null
```

La commande suivante permet de lister l'ensemble des répertoires modifiables par tous et dont le propriétaire n'est pas `root` :

```
# find / -type d -perm -0002 -a \! -uid 0 -ls 2>/dev/null
```

Le `sticky bit` n'empêche pas les situations de compétition entre deux applications s'exécutant simultanément sous le même compte utilisateur.

R55

Séparer les répertoires temporaires des utilisateurs

Chaque utilisateur ou application doit posséder son propre répertoire temporaire et en disposer exclusivement.

Information

Sur les distributions GNU/Linux récentes, la méthode la plus directe pour créer un répertoire temporaire propre à chaque utilisateur est d'utiliser des modules PAM (`Pluggable Authentication Module` ou module d'authentification enfichable), tels que `pam_mktemp` et `pam_namespace` détaillés au paragraphe 7.2.1.

Quand un fichier doit être modifiable par plusieurs utilisateurs ou applications en même temps, un groupe doit être créé et seul ce groupe devra avoir des droits d'écriture sur ce fichier.

Information

La commande suivante permet de lister l'ensemble des fichiers modifiables par tout le monde :

```
# find / -type f -perm -0002 -ls 2>/dev/null
```

Les exécutables avec les droits spéciaux `setuid` ou `setgid` sont particulièrement sensibles car ils s'exécutent avec les privilèges de l'utilisateur propriétaire ou du groupe auquel appartient l'utilisateur propriétaire et non avec les privilèges de l'utilisateur courant ou du groupe auquel appartient l'utilisateur courant. Ces exécutables doivent être de confiance, par exemple issus de la distribution ou de dépôts officiels de paquets conformément à la recommandation R59.

La présence des droits spéciaux `setuid` ou `setgid` sur un exécutable nécessite un certain nombre de précautions pour se prémunir de vulnérabilités liées au changement d'utilisateur, par exemple, nettoyer son environnement et réinitialiser un certain nombre d'éléments hérités du contexte antérieur (masques de signalisation, descripteurs de fichiers ouverts…). La plupart des exécutables ne mettent pas en œuvre de telles précautions, et leur ajouter les droits spéciaux `setuid` ou `setgid` introduirait par conséquent des possibilités d'escalade de privilèges.

R56

Éviter l'usage d'exécutables avec les droits spéciaux `setuid` et `setgid`

Seuls les logiciels de confiance issus, par exemple, de la distribution ou de dépôts officiels de paquets et spécifiquement conçus pour être utilisés avec les droits spéciaux `setuid` ou `setgid` peuvent avoir ces droits positionnés.

Information

La commande suivante permet de lister l'ensemble des fichiers avec les droits spéciaux `setuid` et `setgid` présents sur le système :

```
# find / -type f -perm /6000 -ls 2>/dev/null
```

Information

La commande suivante permet de retirer les droits spéciaux `setuid` ou `setgid` :

```
# chmod u-s <fichier>  # Retire le droit spécial setuid
# chmod g-s <fichier>  # Retire le droit spécial setgid
```

Le cas le plus courant est l'exécutable dont le propriétaire est `root` avec les droits spéciaux `setuid` (`setuid root`) ou `setgid` (`setgid root`) qui s'exécute avec les privilèges de `root` et non ceux de l'utilisateur appelant. Ces exécutables permettent à un utilisateur non privilégié d'effectuer des actions pour lesquelles il n'a, a priori, pas de droit.

En présence de vulnérabilités, ces exécutables sont exploités en vue de fournir un `shell root` à un utilisateur malveillant, ou tout du moins de détourner l'usage légitime de l'exécutable. Ces exécutables sont donc à étudier au cas par cas.

R57

Éviter l'usage d'exécutables avec les droits spéciaux `setuid root` et `setgid root`

Les exécutables avec les droits spéciaux `setuid root` et `setgid root` doivent être le moins nombreux possible.
Lorsqu'il est attendu que seuls les administrateurs les exécutent, il faut leur retirer ces droits spéciaux (`setuid` ou `setgid`) et leur préférer les commandes comme `su` ou `sudo`, qui peuvent être surveillées.

Attention

Une vérification doit avoir lieu après chaque mise à jour car les droits spéciaux peuvent avoir été restaurés ou des exécutables supplémentaires peuvent aussi avoir été installés lors d'une mise à jour.

6.5 Gestion des paquets

L'installation des paquets est l'étape cruciale qui va déterminer l'ensemble des fichiers qui seront présents sur le système et les services rendus par celui-ci. Les paquets installés devront être maintenus dans le temps.

R58

N'installer que les paquets strictement nécessaires

Le choix des paquets doit conduire à une installation aussi minimale que possible, se limitant à ne sélectionner que ceux qui sont nécessaires au besoin.

Installation minimaliste

Il est parfois plus facile d'obtenir une installation minimaliste en retirant tous les paquets présélectionnés, et de ne choisir que ceux nécessaires au contexte d'utilisation. Par exemple, la mise en œuvre d'un serveur ne requiert pas systématiquement l'installation d'une interface graphique locale (*serveur X*).

Attention

Certaines distributions fournissent des « rôles » préconfigurés. Il est déconseillé de baser son installation sur lesdits rôles étant donné que les choix des mainteneurs de la distribution ne correspondent pas forcément aux besoins propres, ce qui nécessitera l'installation de paquets supplémentaires.

Ces paquets sont généralement issus de serveurs appelés dépôts qui contiennent un ensemble de paquets accessibles et installables à partir d'un gestionnaire de parquets. Chaque distribution dispose de dépôts officiels configurés, par défaut, dans le gestionnaire de paquets lors de l'installation et qui sont gérés par les mainteneurs de la distribution. Ces dépôts fournissent principalement les paquets de « base », leur mise à jour et les mises à jour de sécurité.

R59

Utiliser des dépôts de paquets de confiance

Seuls les dépôts internes à l'organisation ou officiels (de la distribution ou de d'un éditeur) doivent être utilisés.

R60

Utiliser des dépôts de paquets durcis

Lorsque la distribution fournit plusieurs types de dépôts, la préférence doit aller à ceux contenant des paquets faisant l'objet de mesures de durcissement supplémentaires. Entre deux paquets fournissant le même service, ceux faisant l'objet de mesures de durcissement (à la compilation, à l'installation ou dans la configuration par défaut) doivent être privilégiés.

Attention

Dans certains cas, les paquets sont issus de serveurs appelés miroirs de dépôts et configurés dans le gestionnaire de paquets. Ces miroirs doivent être des copies conformes de dépôts officiels à jour.

6.6 Veille et maintenance

Tout système doit être maintenu en conditions de sécurité.

Les correctifs logiciels peuvent apporter de nouvelles fonctionnalités à l'environnement logiciel, contribuant ainsi à accroître son niveau de sécurité. D'autres visent à rectifier des vulnérabilités

présentes sur le système. Une activité de veille sur les mesures correctives à appliquer est essentielle, car elle permet de connaître les vulnérabilités auxquelles celui-ci est ou a été exposé, et donc d'y prêter une attention particulière le temps qu'un correctif soit disponible.

R61

Effectuer des mises à jour régulières

Il est recommandé d'avoir une procédure de mise à jour de sécurité régulière et réactive.

L'activité de veille peut se faire par l'inscription à des listes de diffusion des équipes sécurité des composants et applications installés et de leurs éditeurs, aux flux RSS (`Really Simple Syndication` ou réseau de syndication simple) de CERT (`Computer Emergency Response Team` ou équipe d'intervention d'urgence informatique), par exemple le site du CERT-FR [12].

7
Configuration des services

La configuration d'un système d'exploitation s'accompagne du déploiement de services. Les recommandations suivantes visent à établir quelques bonnes pratiques d'administration et de configuration.

Objectif

Identifier les services essentiels et les mesures de durcissement adaptées pour mettre en œuvre des solutions destinées à retarder la compromission d'un système.

Lors de l'installation d'une distribution, certains services sont installés par défaut selon le choix des mainteneurs de la distribution. Ces services ne correspondent pas forcément aux besoins propres.

R62

Désactiver les services non nécessaires

Seuls les composants strictement nécessaires au service rendu par le système doivent être installés.
Tout service, et particulièrement ceux en écoute active sur le réseau, est un élément sensible. Seuls ceux connus et requis pour le fonctionnement et la maintenance doivent être installés. Il est recommandé de désinstaller les services dont la présence ne peut être justifiée ou de les désactiver si leur désinstallation n'est pas possible.

Information

Pour les distributions sous `systemd`, la commande suivante permet de lister l'ensemble des services installés sur le système :

```
# systemctl list-units --type service
```

Information

Les services couramment installés sur les distributions sont :

- les services d'auto-configuration, tels DHCP[24] ou ZeroConf[25], outre le fait qu'ils peuvent perturber le réseau sur lequel ils sont connectés, reçoivent des éléments dont la légitimité est difficile à assurer. Sauf besoin opérationnel, ceux-ci ne devraient pas s'exécuter sur un serveur ;
- les services de `RPC` (`portmap`, `rpc.statd`, `rpcbind`…) ne sont en pratique utilisés que pour un serveur `NFS` ;

- les services bureautiques comme `dbus`, `hald`, `ConsoleKit`, `CUPS` ou `PolicyKit` ne devraient pas s'exécuter sur un serveur ;
- le service `avahi`, utilisé pour la publication et la découverte automatique de services sur le réseau. Sauf besoin opérationnel, celui-ci ne devrait pas s'exécuter sur un serveur ;
- le `serveur X`, rarement utile sur un serveur.

Les services sont souvent installés avec des configurations par défaut qui activent des fonctionnalités potentiellement problématiques d'un point de vue sécurité, par exemple les redirections de port SSH qui sont souvent acceptées puis utilisées pour contourner des règles de pare-feu, ou un serveur `Web Apache` avec l'indexation des répertoires activée qui permet de naviguer dans l'arborescence du système.

R63

M I Désactiver les fonctionnalités des services non essentielles

Les fonctionnalités configurées au niveau des services démarrés doivent être limitées au strict nécessaire.

Attention

La configuration par défaut des services suivants est souvent incomplète : `SMTP` (`Exim`, `Postfix`), `NTP` (`ntpd`) et `DNS` (`Bind`).

Le noyau Linux subdivise les privilèges traditionnellement associés au compte d'administrateur système (ou `root`) en unités distinctes, appelées `Linux capabilities` [26]. Les `Linux capabilities` peuvent être activées ou désactivées indépendamment et ne se limitent pas aux processus et se placent également sur les fichiers exécutables.

Attention

Le noyau Linux offre 41 `capabilities` à ce jour. Au minimum 19 d'entre elles permettent d'élever trivialement ses droits à celui de l'utilisateur `root` (`UID=0`) (*full root*) [27]. La plupart des autres peuvent également fortement aider à obtenir un accès `root` sur le système.

Information

La commande suivante permet de lister l'ensemble des fichiers exécutables pour lesquels un ou plusieurs `Linux capabilities` ont été activées :

```
# find / -type f -perm /111 -exec getcap {} \; 2>/dev/null
```

24. Dynamic Host Configuration Protocol
25. Zero-configuration networking
26. https://man7.org/linux/man-pages/man7/capabilities.7.html
27. https://forums.grsecurity.net/viewtopic.php?f=7&t=2522#p10271

Dans le cadre du principe de moindre privilège, il est important d'isoler les différents services s'exécutant sur le système.

R64

Configurer les privilèges des services

Tous les services et exécutables disponibles sur le système doivent faire l'objet d'une analyse afin de connaître les privilèges qui leur sont associés, et doivent ensuite être mis en œuvre et configurés en vue d'en utiliser le strict nécessaire.

Information

Certains services ou exécutables nécessitent de posséder initialement ou tout le temps des privilèges pour démarrer ou pour fonctionner et d'accéder aux ressources du système.

7.1 Cloisonnement

Le cloisonnement est une technique qui vise à isoler les processus s'exécutant sur un même système. Historiquement, il était réalisé au travers de l'utilitaire `chroot` qui permet de restreindre la vue du système de fichiers d'un processus à un répertoire donné qui devient sa racine. Une fois confiné ou *chrooté*, le processus ne peut alors plus accéder aux répertoires autres que celui pris comme racine ainsi que ses sous-répertoires. Il n'a donc qu'une vision partielle du système de fichiers.

`chroot` présente de nombreuses faiblesses sous GNU/Linux, parmi lesquelles :

- impossibilité d'interdire à l'utilisateur `root` de sortir de sa cage ;
- impossibilité sur certains systèmes d'exploitation d'interdire à un utilisateur non privilégié de sortir de sa cage avec la complicité d'un processus externe ;
- cloisonnement limité au système de fichiers, les accès aux autres processus, au réseau…ne sont pas bloqués ;
- nécessité de posséder initialement les privilèges de `root` afin de pouvoir être exécuté.

D'autres mécanismes sont apparus au gré des évolutions et des améliorations apportées au noyau Linux pour cloisonner les processus selon les grands axes suivants :

- **confinement** avec les `Linux namespaces` ou « espaces de noms Linux » qui permettent de partitionner les ressources du noyau Linux dans des espaces abstraits appelés « espace de noms » ;
- **filtrage** avec `seccomp` (`SECure COMputing Mode`) ou `seccomp BPF`[28] (`SECure COMPuting with Berkeley Packet Filter`) qui permet de filtrer les appels système des processus ;
- **isolation** avec les `cgroups` (`control groups`) qui permettent d'organiser les processus en système hiérarchique afin de limiter, compter et isoler l'utilisation des ressources (partage du processeur, limite de la mémoire, utilisation disque…) du système par les processus.

28. https://www.kernel.org/doc/html/latest/userspace-api/seccomp_filter.html

Ces mécanismes peuvent être mis en œuvre de façon granulaire à partir des systèmes d'initialisation (ou `init`) des systèmes d'exploitation Unix et dérivés (`systemd`, `OpenRC`[29]…).

R65 

Cloisonner les services

Dans le cadre du principe de minimisation, il est recommandé de cloisonner les services avec les mécanismes de confinement, de filtrage et d'isolation disponibles dans le noyau Linux.

Aujourd'hui d'autres solutions de cloisonnement existent et offrent différents niveaux d'abstraction. Elles se caractérisent généralement par la technique de cloisonnement :

- par conteneurs, où le noyau est capable de gérer plusieurs instances systèmes (`LXC`[30], `Docker`[31], `podman`[32]…);
- par émulation, où un émulateur reproduit une machine physique complète (`QEMU`[33], `VirtualBox`[34]…);
- par hyperviseur léger (ou `bare-metal`) (`Xen`[35]…), où l'hyperviseur va gérer éventuellement avec l'aide d'un système hôte différentes machines virtuelles;
- par hyperviseur noyau (`Linux KVM`[36]…).

Ces solutions ne sont pas exclusives. Certaines solutions (`LXC`, `Docker`, `podman`…) mettent en œuvre les fonctionnalités du noyau Linux introduites précédemment : `Linux namespaces`, `Linux capabilities`, `seccomp` ou `cgroups`. Il doit en être fait un usage raisonnable en gardant à l'esprit que les interfaces qu'elles offrent sont autant de portes par lesquelles une intrusion est possible, par exemple un système à base de conteneurs pour lequel le noyau est compromis verra l'ensemble des conteneurs eux-mêmes compromis. Il en va de même avec les machines virtuelles et leur hyperviseur.

R66 

Durcir les composants de cloisonnement

Tout composant support de cloisonnement doit être durci, notamment par l'application de mesures techniques contrant les tentatives d'exploitation.

7.2 Services système

Plusieurs services et outils système peuvent apporter des éléments d'information sur le système. Une grande partie d'entre eux ne sont pas configurés de façon optimale à l'issue de l'installation du système. Les recommandations suivantes visent à combler cette carence.

29. https://github.com/OpenRC/openrc
30. https://linuxcontainers.org/lxc
31. https://www.docker.com
32. https://podman.io
33. https://www.qemu.org
34. https://www.virtualbox.org
35. https://xenproject.org
36. https://www.linux-kvm.org

7.2.1 `Pluggable Authentication Module` ou module d'authentification enfichable

PAM est un ensemble de modules qui permet de configurer « dynamiquement » les différents mécanismes d'authentification et de gestion des comptes sur un système GNU/Linux.

La documentation de PAM est riche, tout comme l'ensemble des fonctionnalités offertes par ses modules. L'objet de cette note n'est pas d'en expliquer le fonctionnement.

PAM va essentiellement fournir le service de gestion des comptes, c'est-à-dire permettre l'authentification de l'utilisateur, la création de sa session et éventuellement toute opération qui doit se dérouler lors de la tentative d'accès :

- création d'environnement,
- récupération de tickets ou de données,
- droits d'accès,
- changement de mot de passe…

Lorsqu'une application fait appel à PAM afin d'authentifier un utilisateur, l'opération est directement réalisée par un module PAM.

Dans les grandes lignes, l'application soumet aux modules PAM les éléments d'authentification. Suivant la configuration située dans différents fichiers présents dans les répertoires `/etc/` et `/etc/pam.d/`, PAM retourne le résultat, échec ou succès, à l'application qui donne ensuite l'accès ou non au système.

Deux éléments importants doivent être notés :

- les modules PAM sont appelés par l'application. L'application manipule, au moins partiellement, des éléments qui contribuent à authentifier l'utilisateur incluant éventuellement des données sensibles comme des mots de passe ;
- en fonction des modules PAM utilisés, PAM vérifie ces éléments à l'aide de bases de données locales, par exemple `shadow`, ou distantes, par exemple à l'aide de requêtes LDAP (`Lightweight Directory Access Protocol` ou protocole léger d'accès léger à un répertoire), SQL (`Structured Query Language` ou langage de requête structurée) ou du protocole `Kerberos`….

R67

Sécuriser les authentifications distante par PAM

Quand l'authentification se déroule au travers d'une application distante (réseau), le protocole d'authentification utilisé par PAM doit être sécurisé (chiffrement du flux, authentification du serveur distant, mécanismes d'anti-rejeu…).

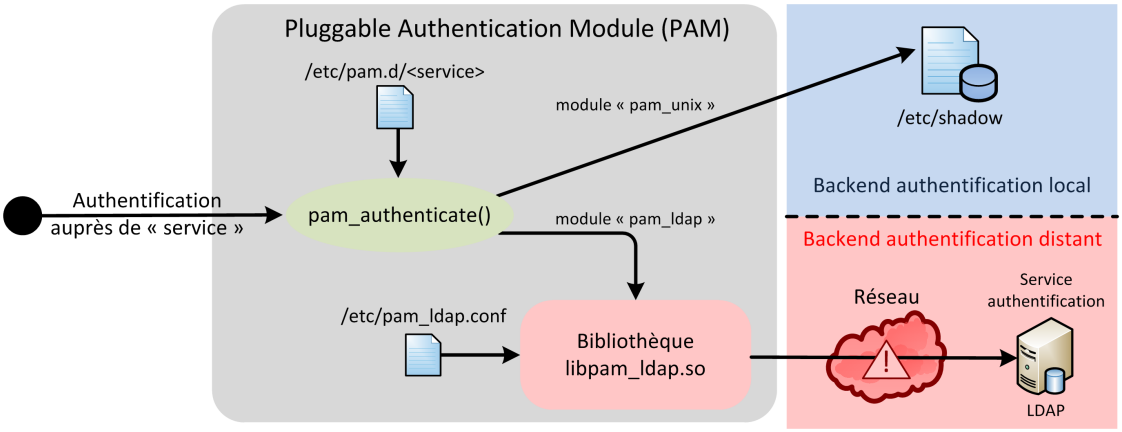


FIGURE 1 – Schéma d'une authentification PAM Unix et LDAP

Information

Certains protocoles comme `Kerberos` proposent des fonctions de communication sécurisée, par exemple le module `pam_krb5` si le `keytab host` est renseigné pour le système.
D'autres modules, tels `pam_ldap` ou `pam_mysql`, ne chercheront pas à établir de communication sécurisée entre le client PAM et l'application s'ils ne sont pas configurés explicitement en ce sens.

En plus de l'authentification, PAM permet de charger d'autres modules dont les fonctionnalités peuvent améliorer la sécurité de l'architecture, par exemple :

pam_faillock permet de bloquer temporairement un compte après un certain nombre d'échecs ;
pam_time permet de restreindre les accès à une plage horaire ;
pam_passwdqc permet d'appliquer des contraintes suivant une politique de complexité de mots de passe ;
pam_pwquality permet de tester la robustesse des mots de passe ;
pam_wheel permet de restreindre un accès aux utilisateurs membres d'un groupe particulier (`wheel` par défaut) ;
pam_mktemp permet de fournir des répertoires temporaires privés par utilisateur sous /tmp ;
pam_namespace permet de créer un espace de noms privés par utilisateur.

Exemple

Les fichiers de configuration PAM suivants sont situés sous `/etc/pam.d/` et portent le nom du service auquel ils sont associés. Seules les directives de configuration les concernant sont présentées.

- `/etc/pam.d/su` et `/etc/pam.d/su-l` pour limiter l'usage de `su` pour devenir root aux utilisateurs membres du groupe `wheel` seulement :

```
# Limite l'accès à root via su aux membres du groupe 'wheel'
auth    required  pam_wheel.so use_uid root_only
```

- `/etc/pam.d/passwd` pour fixer des règles de complexité des mots de passe :

```
# Au moins 12 caractères de 3 classes différentes parmi les majuscules,
```

```
# les minuscules, les chiffres et les autres en interdisant la répétition
# d'un caractère
password  required  pam_pwquality.so minlen=12 minclass=3  \
              dcredit=0 ucredit=0 lcredit=0 \
              ocredit=0 maxrepeat=1
```

- `/etc/pam.d/login` et `/etc/pam.d/sshd` pour bloquer automatiquement des comptes :

```
# Blocage du compte pendant 5 min après 3 échecs
auth    required  pam_faillock.so deny=3 unlock_time=300
```

Le stockage des mots de passe en clair est proscrit car la compromission du système permet à un attaquant de les réutiliser sans effort pour d'autres services. Leur protection ne doit pas reposer uniquement sur les droits d'accès à une base.

R68

Protéger les mots de passe stockés

Tout mot de passe doit être protégé par des mécanismes cryptographiques évitant de les exposer en clair à un attaquant. Pour cela, se référer à la *section 4.6 Stockage de mots de passe* du guide *Recommandations relatives à l'authentification multifacteur et aux mots de passe* [9].

Information

PAM peut être configuré afin d'utiliser `yescrypt` en ajoutant dans le fichier `/etc/pam.d/common-password` la directive suivante :

```
password    required    pam_unix.so obscure yescrypt rounds=11
```

Si le système est ancien et ne supporte pas yescrypt, il est recommandé d'utiliser `SHA-512crypt` en augmentant le nombre de tours :

```
password    required    pam_unix.so obscure sha512 rounds=65536
```

Pour plus d'information, se référer à `man 5 crypt`.

7.2.2 `Name Service Switch` ou service de gestion de noms

NSS (`Name Service Switch` ou service de gestion de noms) est la couche système qui se charge d'interroger et de manipuler les bases de données administratives. Il en existe plusieurs, `passwd`, `group`, `hosts`, `services`…Seules les bases de gestion des comptes utilisateur vont être étudiées ci-dessous.

Quand les comptes utilisateur sont stockés dans un annuaire externe (fréquemment LDAP), NSS se charge d'effectuer les requêtes auprès de l'annuaire dans le but de rendre les comptes visibles par le système. Généralement, ces requêtes sont anonymes et réalisées avec un canal de communication non protégé. Il est donc possible pour un attaquant de récupérer une liste de comptes valides auprès de l'annuaire voire même d'usurper le serveur annuaire interrogé par le NSS.

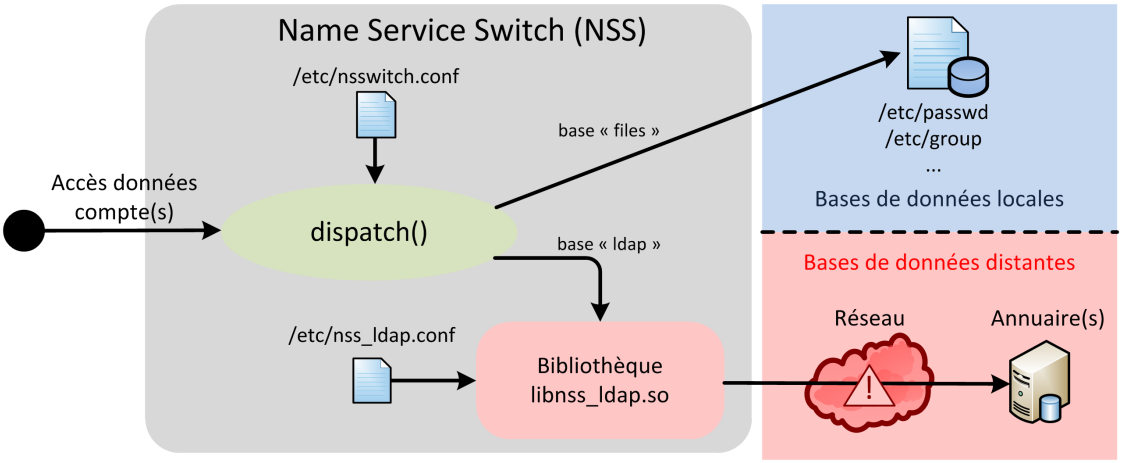


FIGURE 2 – Schéma des bases administratives de NSS

R69

M I

Sécuriser les accès aux bases utilisateur distantes

Lorsque les bases utilisateur sont stockées sur un serveur réseau distant, NSS doit être configuré pour établir une liaison sécurisée permettant au minimum d'authentifier le serveur et de protéger le canal de communication.

Pour accéder à certaines bases de données administratives et plus particulièrement à celles stockées sur un serveur réseau distant, NSS doit s'authentifier à l'aide d'un compte. Dans ce cas, le compte utilisé pour accéder à cette base de données doit être distinct des comptes utilisateur du système et il ne doit disposer que des droits strictement nécessaires pour interroger cette base de données.

R70

M I

Séparer les comptes système et d'administrateur de l'annuaire

Il est recommandé de ne pas avoir de recouvrement de compte entre ceux utilisés par le système d'exploitation et ceux utilisés pour administrer l'annuaire.
L'usage de comptes administrateur d'annuaire pour effectuer des requêtes d'énumération de comptes par NSS doit être prohibé.

7.2.3 Journalisation

Dans le cadre du principe de défense en profondeur, tout système doit être surveillé afin de détecter une activité suspecte et de pouvoir y réagir.

R71

M I R

Mettre en place un système de journalisation

Il est conseillé d'appliquer les recommandations de la note technique « *Recommandations de sécurité pour l'architecture d'un système de journalisation* » [7].

Le service `syslog` qui est le système de journalisation principal utilisé sous GNU/Linux peut être décomposé en deux parties :

- un serveur, généralement `syslog-ng`[37] ou `rsyslog`[38], qui collecte l'ensemble des évènements au format `syslog`,
- plusieurs clients qui envoient les évènements au serveur, principalement à partir de la fonction `syslog()` de la `glibc`.

Le serveur `syslog` est donc un élément fédérateur de journaux qui accède à un grand nombre de données en provenance de sources système diverses. N'importe quel service est susceptible de le solliciter en vue d'enregistrer un évènement, sans authentification.

Information

Il est d'usage d'avoir un serveur `syslog` résidant qui ne gère que les évènements soumis `localement` par les services du système et qui envoie éventuellement ces évènements à un serveur centralisé.

La sécurisation des journaux doit répondre aux deux niveaux de protection suivants :

- entre les services, afin qu'un service ne puisse ni manipuler ni accéder aux journaux enregistrés par un autre service ;
- au niveau du service, afin qu'en cas de compromission, celui-ci ne puisse lire, effacer ou altérer les évènements enregistrés.

R72

Mettre en place des journaux d'activité de service dédiés

Chaque service doit posséder un journal d'événements dédié sur le système. Ce journal ne doit être accessible que par le serveur `syslog`, et ne doit pas être lisible, modifiable ou supprimable par le service directement.

Information

L'usage de la fonction `syslog()` de la `glibc` est une solution envisageable. L'envoi des évènements peut être aussi réalisé par la commande shell `logger`[39].

Le service `auditd` est un service de journalisation avancé souvent présent sur les distributions GNU/Linux. Il permet d'enregistrer des opérations système spécifiques, voire d'alerter un administrateur lorsque des opérations privilégiées non prévues ont lieu.

Les évènements enregistrés dépendent des règles `auditd` écrites. Lorsqu'un enregistrement est déclenché, un second service, `audispd`, se charge de son traitement : évènement `syslog`, envoi de mail, écriture dans un fichier…

Le service `auditd` est capable de surveiller un grand nombre d'actions :

- appels système réalisés ;

37. https://www.syslog-ng.com
38. https://www.rsyslog.com
39. https://man7.org/linux/man-pages/man1/logger.1.html

- accès à une arborescence ou à des fichiers particuliers ;
- insertions de modules.

Consultez sa documentation pour plus de détails.

R73

M I R Journaliser l'activité système avec auditd

La journalisation de l'activité du système doit être faite au travers du service `auditd`.

Exemple de configuration `auditd`

Le fichier de configuration `/etc/audit/audit.rules` de `auditd` suivant enregistre les actions présentant un intérêt :

```
# Exécution de insmod, rmmod et modprobe
-w /sbin/insmod   -p x
-w /sbin/modprobe -p x
-w /sbin/rmmod    -p x
# Sur les distributions GNU/Linux récentes, insmod, rmmod et modprobe sont
# des liens symboliques de kmod
-w /bin/kmod      -p x

# Journaliser les modifications dans /etc/
-w /etc/ -p wa

# Surveillance de montage/démontage
-a exit,always -S mount -S umount2

# Appels de syscalls x86 suspects
-a exit,always -S ioperm -S modify_ldt
# Appels de syscalls qui doivent être rares et surveillés de près
-a exit,always -S get_kernel_syms -S ptrace
-a exit,always -S prctl

# Rajout du monitoring pour la création ou suppression de fichiers
# Ces règles peuvent avoir des conséquences importantes sur les
# performances du système
-a exit,always -F arch=b64 -S unlink -S rmdir -S rename
-a exit,always -F arch=b64 -S creat -S open -S openat -F exit=-EACCES
-a exit,always -F arch=b64 -S truncate -S ftruncate -F exit=-EACCES

# Rajout du monitoring pour le chargement, le changement et
# le déchargement de module noyau
-a exit,always -F arch=b64 -S init_module -S delete_module
-a exit,always -F arch=b64 -S finit_module

# Verrouillage de la configuration de auditd
-e 2
```

Information

Les journaux ainsi créés par `auditd` peuvent être verbeux en particulier quand de nombreuses activités sont rapportées. L'outil `aureport` permet de sélectionner les informations intéressantes en fonction de critères bien spécifiques :

- contexte sur des échecs d'authentification,
- rapport d'évènement sur certains fichiers ou répertoires,
- évènements anormaux (crash de logiciels, reconfiguration de cartes réseau…),
- …

7.2.4 Messagerie

La messagerie est un service couramment utilisé par le système pour informer sur certaines évolutions de son état. Cela se déroule généralement par l'envoi d'un message électronique à un destinataire spécifique, souvent un compte utilisateur.

Information

Un cas fréquemment rencontré est le service `cron`, qui soumet systématiquement un message électronique lorsque la tâche exécutée affiche des données sur sa sortie d'erreur (`stderr`).

Suivant les distributions, il arrive que le service de messagerie installé soit configuré en relai ouvert, c'est-à-dire qu'il accepte tout message électronique qui lui est soumis, mais avec la *socket* d'écoute uniquement sur l'interface réseau de la boucle locale (ou `loopback` abrégée `lo`). Dans la mesure du possible, il est préférable d'utiliser un service de redirection de messagerie électronique léger, par exemple `ssmtp`.

R74

Durcir le service de messagerie locale

Quand un service de messagerie est installé sur la machine, il doit être configuré pour n'accepter que :

- les messages à destination d'un compte utilisateur local à la machine ;
- les connexions sur l'interface réseau de la boucle locale, les connexions distantes au service de messagerie doivent être rejetées.

Les services utilisant la messagerie sont souvent configurés par défaut pour envoyer des messages à destination du compte de l'administrateur système (ou `root`). Conformément à la recommandation R33, ce compte doit être désactivé. Par conséquent, et pour ne perdre aucun message adressé à l'administrateur système, l'utilisation d' « alias » est recommandée.

R75

Configurer un alias de messagerie des comptes de service

Pour chaque service exploité sur le système, ainsi que pour le compte d'administrateur (ou `root`), un alias vers un compte utilisateur administrateur doit être configuré afin que celui-ci reçoive les notifications et les rapports adressés par messagerie électronique.

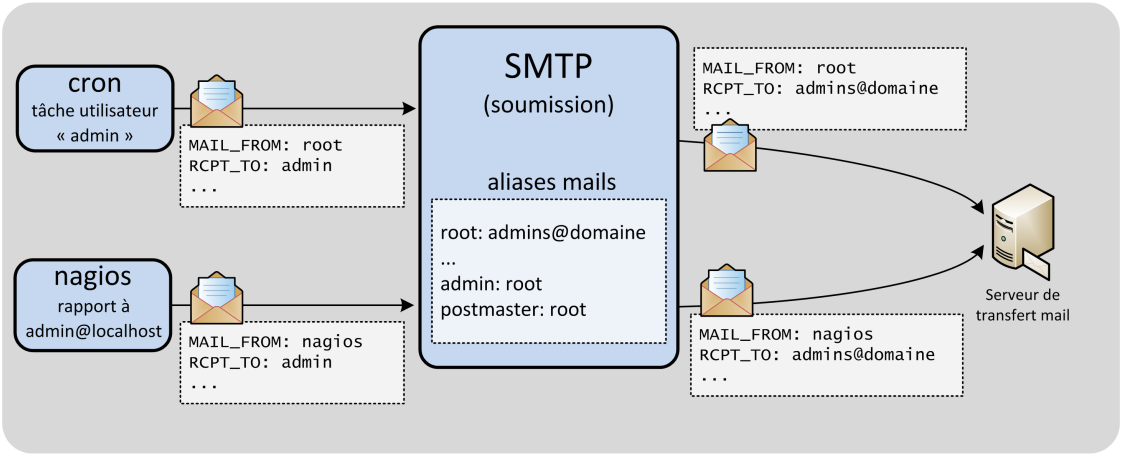


FIGURE 3 – Schéma d'envoi d'alertes par messagerie électronique à l'équipe d'administration

7.2.5 Surveillance du système de fichiers

L'opération de scellement d'un système de fichiers consiste en l'installation puis la configuration d'un service qui aura pour objectif de vérifier, au moins périodiquement, les modifications faites au niveau d'une arborescence. C'est une fonctionnalité présente au sein de la plupart des HIDS (`Host Intrusion Detection System` ou système de détection d'intrusion sur l'hôte). Le scellement permet de remonter des informations sur les modifications (écarts entre la version scellée et celle présente actuellement sur le système).

C'est une fonction utile aux administrateurs. Outre le fait qu'elle permet de conduire un audit périodique sur le système et de générer des rapports, elle permet aux équipes d'être informées sur les changements et donc d'obtenir un suivi des évolutions.

R76 M I R E

Sceller et vérifier l'intégrité des fichiers

Tout fichier qui ne revêt pas un caractère transitoire (comme des fichiers temporaires, des bases de données…) doit être surveillé par un logiciel de scellement. Cela inclut notamment : les répertoires contenant des exécutables, des bibliothèques, des fichiers de configuration, ainsi que tout fichier pouvant contenir des éléments sensibles (clés cryptographiques, mots de passe, données confidentielles…).

La compromission d'une machine pouvant s'accompagner d'une compromission de la base de scellement lorsque celle-ci est stockée localement, des mesures techniques doivent être mises en œuvre afin d'assurer que le contenu de la base reste autant que possible intègre.

R77 M I R E

Protéger la base de données des scellés

La base de données de scellement doit être protégée de tout accès frauduleux par des mécanismes de signature cryptographique (avec la clé utilisée pour la signature non enregistrée localement en clair), ou être éventuellement stockée sur une machine distincte de celle sur laquelle le scellement est réalisé.

Il existe de nombreuses solutions, les plus déployées sur les systèmes GNU/Linux sont Tripwire, Samhain [40] et AIDE (`Advanced Intrusion Detection Environment` ou environnement de détection d'intrusion avancé) [41].

7.3 Services réseau

Une attention toute particulière doit être portée sur les services exposés à des flux non maîtrisés, c'est-à-dire tout service communiquant avec des sources qui ne sont pas de confiance (Internet, bornes `Wi-Fi` publiques…) ou non authentifiées. Ce sont ceux qui sont préférentiellement attaqués et compromis car leur accès est libre.

Ils sont tous autant de portes ouvertes sur le système permettant un accès illégitime à celui-ci lorsque le service est vulnérable ou mal configuré : site `Web` permettant d'exécuter des commandes arbitraires, processus d'administration qui n'utilise pas un mécanisme d'authentification fiable, service réseau obsolète ayant une vulnérabilité exploitable…

R78

Cloisonner les services réseau

Les services [42] réseau doivent autant que possible être hébergés sur des environnements [43] distincts.
Cela évite d'avoir d'autres services potentiellement affectés si l'un d'eux se retrouve compromis sous le même environnement.

Le cloisonnement des environnements peut être réalisé de différentes façons (compte utilisateur dédié, conteneur spécifique, machine virtuelle ou physique…), chaque approche ayant des avantages et des inconvénients qu'il est nécessaire d'étudier : un compte utilisateur dédié permet toujours des accès aux ressources sur lesquels des droits incorrects sont positionnés (un audit assez fin des droits d'accès doit donc être réalisé), alors que les conteneurs et les machines virtuelles offrent un cloisonnement plus efficace, mais peuvent induire des efforts d'intégration supplémentaires.

Ces services doivent donc être durcis et surveillés, et ce malgré l'apparente opération d'authentification effectuée par le serveur. Le durcissement relève de l'ensemble des mesures techniques qui visent à retarder voire empêcher la compromission du service. Cette démarche s'applique dès la phase de conception (étude de séparation de privilèges, spécifications non ambiguës…) jusqu'à la réalisation (validation des entrées/sorties, configuration sécurisée…) et la maintenance.

R79

Durcir et surveiller les services exposés

Les services exposés à des flux non maîtrisés doivent être durcis et particulièrement surveillés.

40. https://www.la-samhna.de/samhain
41. https://aide.github.io
42. Le terme « service » est à prendre au sens large ici. Toute faille ou vulnérabilité exploitable avant une étape d'authentification est particulièrement concernée par cette recommandation.
43. Environnement est entendu au sens large : il représente l'ensemble des ressources et données disponibles et accessibles par le service en fonction des privilèges dont il dispose.

Cette surveillance consiste à caractériser le comportement du service, et à reporter tout écart par rapport à son fonctionnement nominal déduit des spécifications initiales attendues.

Exemple

Un service `Web` reposant sur une authentification `HTTP` basique peut être exploitable à plusieurs niveaux :

1. matériel (micrologiciel[44], pilote logiciel de cartes réseau…) ;
2. réseau (`Ethernet`, `IP`, `TCP`) ;
3. applicatif (couche `SSL/TLS`, en-têtes `HTTP` émis et reçus avant l'authentification).

Les services réseau sont souvent configurés par défaut en écoute sur l'ensemble des interfaces réseau, augmentant inutilement leur surface d'attaque.

R80

Réduire la surface d'attaque des services réseau

Tous les services réseau doivent être en écoute sur les interfaces réseau adéquates.

Information

La commande suivante permet de lister l'ensemble des services en écoute sur le réseau :

```
# sockstat
```

44. Communément appelé `firmware`.

Liste des recommandations

R1	Choisir et configurer son matériel	10
R2	Configurer le BIOS/UEFI	11
R3	Activer le démarrage sécurisé UEFI	12
R4	Remplacer les clés préchargées	12
R5	Configurer un mot de passe pour le chargeur de démarrage	14
R6	Protéger les paramètres de ligne de commande du noyau et l'`initramfs`	15
R7	Activer l'IOMMU	17
R8	Paramétrer les options de configuration de la mémoire	17
R9	Paramétrer les options de configuration du noyau	19
R10	Désactiver le chargement des modules noyau	20
R11	Activer et configurer le LSM `Yama`	21
R12	Paramétrer les options de configuration du réseau *IPv4*	21
R13	Désactiver le plan *IPv6*	22
R14	Paramétrer les options de configuration des systèmes de fichiers	23
R15	Paramétrer les options de compilation pour la gestion de la mémoire	24
R16	Paramétrer les options de compilation pour les structures de données du noyau	26
R17	Paramétrer les options de compilation pour l'allocateur mémoire	26
R18	Paramétrer les options de compilation pour la gestion des modules noyau	27
R19	Paramétrer les options de compilation pour les évènements anormaux	28
R20	Paramétrer les options de compilation pour les primitives de sécurité du noyau	28
R21	Paramétrer les options de compilation pour les `plugins` du compilateur	29
R22	Paramétrer les options de compilation pour la pile réseau	30
R23	Paramétrer les options de compilation pour divers comportements du noyau	30
R24	Paramétrer les options de compilation spécifiques aux architectures 32 bits	31
R25	Paramétrer les options de compilation spécifiques aux architectures x86_64 bits	31
R26	Paramétrer les options de compilation spécifiques aux architectures ARM	32
R27	Paramétrer les options de compilation spécifiques aux architectures `ARM` 64 bits	32

R28	Partitionnement type	34
R29	Restreindre les accès au dossier `/boot`	35
R30	Désactiver les comptes utilisateur inutilisés	36
R31	Utiliser des mots de passe robustes	36
R32	Expirer les sessions utilisateur locales	36
R33	Assurer l'imputabilité des actions d'administration	37
R34	Désactiver les comptes de service	38
R35	Utiliser des comptes de service uniques et exclusifs	38
R36	Modifier la valeur par défaut de UMASK	39
R37	Utiliser des fonctionnalités de contrôle d'accès obligatoire MAC	40
R38	Créer un groupe dédié à l'usage de `sudo`	41
R39	Modifier les directives de configuration `sudo`	42
R40	Utiliser des utilisateurs cibles non-privilégiés pour les commandes `sudo`	42
R41	Limiter l'utilisation de commandes nécessitant la directive `EXEC`	43
R42	Bannir les négations dans les spécifications `sudo`	43
R43	Préciser les arguments dans les spécifications `sudo`	43
R44	Éditer les fichiers de manière sécurisée avec `sudo`	44
R45	Activer les profils de sécurité `AppArmor`	45
R46	Activer `SELinux` avec la politique `targeted`	47
R47	Confiner les utilisateurs interactifs non privilégiés	48
R48	Paramétrer les variables `SELinux`	48
R49	Désinstaller les outils de débogage de politique `SELinux`	49
R50	Restreindre les droits d'accès aux fichiers et aux répertoires sensibles	50
R51	Changer les secrets et droits d'accès dès l'installation	50
R52	Restreindre les accès aux *sockets* et aux *pipes* nommées	51
R53	Éviter les fichiers ou répertoires sans utilisateur ou sans groupe connu	52
R54	Activer le `sticky bit` sur les répertoires inscriptibles	52
R55	Séparer les répertoires temporaires des utilisateurs	53
R56	Éviter l'usage d'exécutables avec les droits spéciaux `setuid` et `setgid`	53
R57	Éviter l'usage d'exécutables avec les droits spéciaux `setuid root` et `setgid root`	54
R58	N'installer que les paquets strictement nécessaires	55

R59	M	Utiliser des dépôts de paquets de confiance	55
R60	M I R	Utiliser des dépôts de paquets durcis	55
R61	M	Effectuer des mises à jour régulières	56
R62	M	Désactiver les services non nécessaires	57
R63	M I	Désactiver les fonctionnalités des services non essentielles	58
R64	M I R	Configurer les privilèges des services	59
R65	M I R	Cloisonner les services	60
R66	M I R E	Durcir les composants de cloisonnement	60
R67	M I	Sécuriser les authentifications distante par PAM	61
R68	M	Protéger les mots de passe stockés	63
R69	M I	Sécuriser les accès aux bases utilisateur distantes	64
R70	M I	Séparer les comptes système et d'administrateur de l'annuaire	64
R71	M I R	Mettre en place un système de journalisation	64
R72	M I R	Mettre en place des journaux d'activité de service dédiés	65
R73	M I R	Journaliser l'activité système avec auditd	66
R74	M I	Durcir le service de messagerie locale	67
R75	M I	Configurer un alias de messagerie des comptes de service	67
R76	M I R E	Sceller et vérifier l'intégrité des fichiers	68
R77	M I R E	Protéger la base de données des scellés	68
R78	M I R	Cloisonner les services réseau	69
R79	M I	Durcir et surveiller les services exposés	70
R80	M	Réduire la surface d'attaque des services réseau	70

Annexe A
Patchs au noyau Linux

Un certain nombre de patchs intrusifs existent pour ajouter des fonctionnalités de sécurité au noyau Linux. Ces patchs sont maintenus en dehors de l'arbre principal (ou `upstream`) en raison principalement de leur intrusivité et de l'impact fort que certaines fonctionnalités peuvent avoir sur les performances. Cette annexe ne prétend pas être exhaustive, mais vise plutôt à présenter quelques « exemples » de fonctionnalités que l'on peut retrouver dans certains d'entre eux.

Le projet `Kernel Self Protection Project` [45] (`KSPP`) s'attache à fournir directement dans le noyau les fonctionnalités acceptables pour toutes les machines et toutes les utilisations.

Le projet `grsecurity` [46] ainsi que `PaX` qu'il intègre a été pendant de nombreuses années la référence en termes de durcissement noyau. Ce patch extrêmement intrusif et complexe n'a jamais été considéré pour inclusion dans le noyau `upstream` et n'est maintenant plus disponible publiquement. Il n'est donc pas traité ici. Plusieurs nouveaux projets sont apparus pour combler ce vide.

Le projet `linux-hardened` en est un exemple. Les fonctionnalités de sécurité apportées sont des compléments à ce qu'apporte le projet `KSPP` et l'essentiel de celles-ci est extrait de `PaX` :

- des améliorations de l'ASLR (`Address Space Layout Randomisation` ou distribution aléatoire de l'espace d'adressage) ;
- des options de nettoyage de la mémoire après désallocation ;
- la déclaration de nombreuses variables `__ro_after_init`, qui ne sont donc plus accessibles en écriture après la phase d'initialisation du noyau ;
- des configurations par défaut plus restrictives que le noyau `uptream` ;
- des restrictions d'accès à l'infrastructure `perf` ;
- quelques limitations sur l'utilisation de *user namespace* et la connection de périphériques USB.

Information

L'option de configuration recommandée `extra_latent_entropy` des patchs fournis par le projet `linux-hardened` est à ajouter à la liste des paramètres du noyau lors du démarrage en plus de celles déjà présentes dans le fichier de configuration du chargeur de démarrage. Elle permet une forme très simple d'entropie latente extraite lors du démarrage du système et ajouté à l'entropie obtenue avec `GCC_PLUGIN_LATENT_ENTROPY`.

45. https://kernsec.org/wiki/index.php/Kernel_Self_Protection_Project
46. https://grsecurity.net

Information

Les options de configuration recommandées des patchs fournis par le projet `linux-hardened` détaillée dans cette liste permettent de compléter la configuration du noyau du paragraphe 5.2.2. Elles sont présentées telles que rencontrées dans le fichier de configuration `sysctl.conf` :

```
# Interdit l'ajout de tout nouveau périphérique USB. Cette protection est à
# considérer avec soin car à la fois efficace et très impactante, par exemple,
# lors du débranchement d'un clavier ou d'une souris USB.
# Cette option ne prémunit pas contre toutes les attaques possibles sur l'USB
# dans la mesure où le port peut continuer à être alimenté.
kernel.deny_new_usb = 0,
# Il est important de noter que la sémantique de cette option avec la valeur 3
# est modifiée par l'application du patch Linux-hardened (cf. article LWN -
# https://lwn.net/Articles/696216/ pour une discussion à ce sujet)
kernel.perf_event_paranoid=3,
# Restreint la manipulation des TTY potentiellement partagés entre une
# hiérarchie de processus dont les privilèges sont parfois hétérogènes.
kernel.tiocsti_restrict=1
# Atténue le risque d'attaque par mesure de temps (timing side channel).
kernel.device_sidechannel_restrict=1
```

Information

La liste ci-dessous présente les options recommandées de compilation des patchs fournis par le projet `linux-hardened` détaillée dans cette liste permettent de compléter la configuration statique du paragraphe 5.3 :

```
# Place des canary à la fin des allocations de SLAB
CONFIG_SLAB_CANNARY=y
# Utilise le maximum de bits aléatoires dans les adresses de base de mmap sur
# les processeurs x86_64. Cette option affecte aussi le nombre de bits utilisés
# pour les adresses de base de la stack;
CONFIG_ARCH_MMAP_RND_BITS=32
# Interdit aux utilisateurs non privilégiés d'utiliser les ioctl TIOCSTI pour
# exécuter des commandes arbitraires dans des processus qui partagent une
# session tty.
CONFIG_SECURITY_TIOCSTI_RESTRICT=y
# Vérfie que les nouvelles pages et les slabs allouées sont initialisées à 0 afin
# de détecter les bogues types "write-after-free"
CONFIG_PAGE_SANITIZE_VERIFY=y
CONFIG_SLAB_SANITIZE_VERIFY=y
```

Au même titre que `linux-hardened`, on peut citer `Lockdown Linux` qui est une suite de patchs dont le but est de verrouiller l'accès aux données du noyau depuis le monde utilisateur. Cette suite de patchs a été ajoutée dans la version 5.4 du noyau Linux [47].

Voici quelques fonctionnalités fournies par cette suite de patchs :

- tous les modules doivent être signés (il est nécessaire d'activer l'option de compilation du noyau `CONFIG_MODULE_SIG`, dans le cas contraire, les modules non signés pourront être chargés) ;
- pas d'utilisation d'`ioperm`, `iopl`, ni d'écriture dans `/dev/port` ;
- pas d'écriture dans `/dev/mem`, ni dans `/dev/kmem` ;
- pas d'hibernation ;
- accès restreint aux registres de configuration des périphériques PCI ;

47. https ://man7.org/linux/man-pages/man7/kernel_lockdown.7.html

- accès restreint aux registres MSR de configuration des processeurs x86 ;
- pas de `kexec` ;
- certaines restrictions sur l'ACPI ;

Information

La liste ci-dessous présente les options recommandées de compilation associées aux patchs fournis par `Lockdown Linux` :

```
# Active les patchs lockdown Linux et ce, dès le début du démarrage,
# restreignant l'accès aux fonctionnalités qui permettraient à l'utilisateur de
# modifier le noyau en cours d'exécution
CONFIG_SECURITY_LOCKDOWN_LSM=y
CONFIG_SECURITY_LOCKDOWN_LSM_EARLY=y
# Active le mode de confidentialité qui étend les restrictions ci-dessus aux
# fonctionnalités qui permettraient à l'utilisateur d'extraire des informations
# confidentielles contenues à l'intérieur du noyau.
CONFIG_LOCK_DOWN_KERNEL_FORCE_CONFIDENTIALITY=y
```

Enfin, le projet `CLIP OS` [13] élaboré par l'ANSSI propose un ensemble de configurations Linux ainsi que divers patchs de durcissement.

Comme on peut le voir sur ces exemples, les fonctionnalités de sécurité qu'apportent les patchs externes au noyau `upstream` sont diverses et ne se recoupent pas. Appliquer plusieurs patchs en cascade est une tâche difficile dans la mesure où ils sont développés indépendamment les uns des autres et que les modifications qu'ils font sur le noyau peuvent rentrer en conflit.

Attention

Le choix d'un durcissement éventuel à ajouter au noyau doit se faire au regard de l'usage envisagé de la machine pour laquelle le noyau est compilé.

Annexe B
Conformité de la configuration

Le SCAP (Security Content Automation Protocol) est une norme maintenue par le NIST (National Institute of Standards and Technology) pour permettre l'automatisation de la conformité de la configuration et l'analyse des vulnérabilités qui sont interopérables et efficaces. L'utilisation de scanners SCAP avec le contenu SCAP peut réduire la barrière et aider à maintenir la conformité.

OpenSCAP [48] est la seule implémentation de scanner open source certifiée NIST de la norme SCAP. ComplianceAsCode/content [49] est un projet qui fournit du contenu de sécurité dans divers formats, notamment SCAP, Ansible [50] et Bash. Il fournit des profils SCAP [51] alignés sur les niveaux de durcissement présentés dans ce guide et sont compatibles avec les distributions dérivées de Debian et Red Hat. Une liste des politiques de sécurité disponibles et de la documentation sur leur utilisation est disponible sur https://static.open-scap.org.

D'autres initiatives ont eu lieu pour émettre des recommandations sur la configuration du noyau Linux. La page Recommended Settings [52] du projet KSPP fournit des recommandations visant à obtenir une configuration durcie. Ces recommandations ne concernent que les options upstream du noyau, elles constituent une base saine sans toutefois se substituer aux recommandations de ce guide.

Le script Python kconfig-hardened-check [53] vérifie la conformité d'une configuration noyau vis-à-vis des Recommended Settings du projet KSPP.

Attention

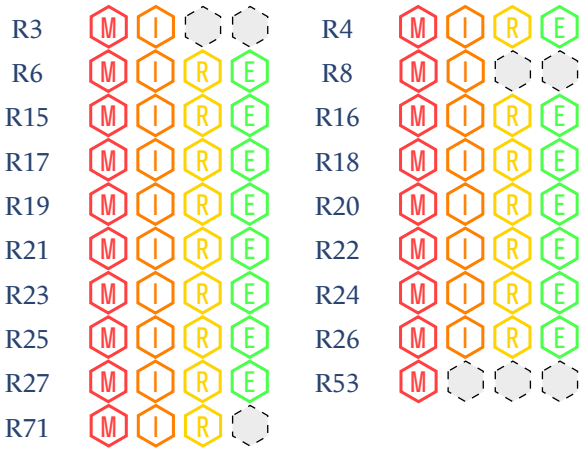
Ces implémentations, quoique pratiques, sont des commodités de vérification et ne sauraient se substituer à une lecture attentive et complète avec l'aide d'une expertise système et sécurité.

48. https://www.open-scap.org
49. https://github.com/ComplianceAsCode/content
50. https://www.ansible.com
51. https://github.com/ComplianceAsCode/content/releases
52. https://kernsec.org/wiki/index.php/Kernel_Self_Protection_Project/Recommended_Settings
53. https://github.com/a13xp0p0v/kconfig-hardened-check

Annexe C
Évolutions du guide

C.1 Nouvelles recommandations

Les recommandations suivantes font leur apparition dans la version 2.0 du guide :



C.2 Mises à jour entre les versions 1.2 et 2.0

Outre les mises à jour de forme, les mises à jour principales du fond sont les suivantes :

- Redéfinition des différents niveaux du système MIRE en section 1.2 Niveau de durcissement.
- Ajout du chapitre 2 Menaces et objectifs des attaquants.
- Refonte du chapitre 4 Configuration matérielle pour intégrer le démarrage sécurisé UEFI et le démarrage de confiance.
- Ajout du chapitre 5 Configuration du noyau Linux. Celui-ci comprend les options de la ligne de commande du noyau et les options de configuration pour la compilation du noyau Linux. Les options de configuration `sysctl` ont également été déplacées dans ce chapitre.
- Ajout de l'annexe A Patchs au noyau Linux.
- Ajout de l'annexe B Conformité de la configuration.
- Certaines recommandations ont également changé de niveau MIRE, notamment suite à la re-définition des niveaux MIRE. La matrice ci-dessous en section C.3 liste ces changements.

C.3 Matrice de rétrocompatibilité

Afin de permettre aux lecteurs ayant déjà travaillé sur la base de la première version de la note technique [4], dénommée v1.2 dans la suite du texte, il est proposé une matrice de rétrocompatibilité permettant de trouver les ajouts, suppressions ou équivalences de recommandations.

Attention

Cette matrice est un outil pour faciliter la lecture mais n'a pas vocation à établir une équivalence stricte entre les différentes versions du guide. La lecture détaillée des recommandations actualisées est fortement conseillée.

v1.2		Version actuelle	
Référence	Niveau	Référence	Niveau
R1	M	R62	M
R2	M I	R63	M I
R3	M I R	suppression	
R4	M I R E	R37	M I R
R5	M	suppression	
R6	M I R	R78	M I R
R7	M I R	Incluse dans R71	M I R
R8	M	R61	M
R9	M I	Support matériel dans R1 M I R Configuration du BIOS/UEFI dans R2	M I
R10	M I	Incluse dans R1	M I R
R11	M I R E	R7	M I R
R12	M I	R28	M I
R13	M I R	R29	M I R
R14	M I	R58	M
R15	M	R59	M
R16	M I R	R60	M I R
R17	M I R	R5	M I
R18	M	Étendue à l'ensemble des utilisateurs R31	M
R19	M I	R33	M I
R20	M I R	R51	M I R
R21	M I	R79	M I
R22	M I	Options réseau `IPv4` dans	

v1.2		Version actuelle	
Référence	Niveau	Référence	Niveau
		R12	M I
		Options réseau `IPv6` dans	
		R13	M I
R23	M I	Incluse dans	
		R9	M I
		R14	M I
R24	M I R	R10	M I R
R25	M I R	R11	M I
R26	M I R	R30	M
R27	M I	R34	M I
R28	M I R	R35	M I
R29	M I R	R32	M I
R30	M	suppression	
R31	M I	R67	M I
R32	M	R68	M
R33	M I	R69	M I
R34	M I	R70	M I
R35	M I R	R36	M I R
R36	M I	R50	M I
R37	M	R56	M
R38	M I R	R57	M I R
R39	M I	R55	M I
R40	M I	R54	M
R41	M I	R52	M I
R42	M	R80	M
R43	M I	suppression	
R44	M I	suppression	
R45	M I R E	suppression	
R46	M I	R72	M I R
R47	M I	Incluse dans	
		R71	M I R
R48	M I	R74	M I
R49	M I	R75	M I
R50	M I R	R73	M I R
R51	M I R E	R76	M I R E
R52	M I R E	R77	M I R E
R53	M I R	R64	M I R
R54	M I R	R66	M I R E

v1.2		Version actuelle	
Référence	Niveau	Référence	Niveau
R55	M I	R65	M I R
R56	M I	suppression	
R57	M I	R38	M I R
R58	M I	R39	M I
R59	M	suppression	
R60	M I	R40	M I
R61	M I R	R41	M I R
R62	M I	R42	M I
R63	M I	R43	M I
R64	M I	R44	M I
R65	M I R E	R45	M I R
R66	M I R E	R46	M I R E
R67	M I R E	R48	M I R E
R68	M I R E	R49	M I R E
R69	M I R E	R47	M I R E

Bibliographie

[1] *Documentation officielle en ligne de la dernière version du noyau Linux*.
Page web.
https://www.kernel.org/doc/html/latest/index.html.

[2] *Recommandations de configuration matérielle de postes clients et serveurs x86*.
Note technique DAT-NT-024/ANSSI/SDE/NP v1.0, ANSSI, mars 2015.
https://www.ssi.gouv.fr/nt-x86.

[3] *Recommandations pour un usage sécurisé d'(Open)SSH*.
Note technique DAT-NT-007/ANSSI/SDE/NP v1.2, ANSSI, août 2015.
https://www.ssi.gouv.fr/nt-ssh.

[4] *Recommandations de configuration d'un système GNU/Linux*.
Guide ANSSI-BP-028 v1.2, ANSSI, février 2019.
https://www.ssi.gouv.fr/reco-securite-systeme-linux.

[5] *Recommandations pour la mise en œuvre d'un site Web : maîtriser les standards de sécurité côté navigateur*.
Guide ANSSI-PA-009 v2.1, ANSSI, avril 2021.
https://www.ssi.gouv.fr/securisation-sites-web.

[6] *Recommandations relatives à l'administration sécurisée des systèmes d'information*.
Guide ANSSI-PA-022 v3.0, ANSSI, mai 2021.
https://www.ssi.gouv.fr/securisation-admin-si.

[7] *Recommandations de sécurité pour l'architecture d'un système de journalisation*.
Guide DAT-PA-012 v2.0, ANSSI, janvier 2022.
https://www.ssi.gouv.fr/journalisation.

[8] *Recommandations pour la mise en place de cloisonnement système*.
Guide ANSSI-PG-040 v1.0, ANSSI, décembre 2017.
https://www.ssi.gouv.fr/guide-cloisonnement-systeme.

[9] *Authentification multifacteurs et mots de passe*.
Guide ANSSI-PG-078 v1.0, ANSSI, octobre 2021.
https://www.ssi.gouv.fr/mots-de-passe/.

[10] *Instruction générale interministérielle n°1300*.
Référentiel, SGDSN, août 2021.
https://www.ssi.gouv.fr/igi1300.

[11] *Documentation de AppArmor*.
Page web, GitLab B.V.
https://gitlab.com/apparmor/apparmor/wikis/Documentation.

[12] *Site Web du CERT FR*.
Page web, ANSSI.
http://www.cert.ssi.gouv.fr/.

[13] *Système d'exploitation multiniveau sécurisé CLIP OS*.
Page web.
https://www.ssi.gouv.fr/administration/services-securises/clip.

[14] *Analyse de l'efficacité du service fourni par une IOMMU*.
Vincent Nicomette et Yves Deswarte Eric Lacombe, Fernand Lone Sang.
Publication scientifique, juin 2010.
https://www.sstic.org/2010/presentation/Analyse_de_l_efficacite_du_service_fourni_par_une_IOMMU/.

[15] *Licence ouverte / Open Licence v2.0*.
Page web, Mission Etalab, avril 2017.
https://www.etalab.gouv.fr/licence-ouverte-open-licence.

[16] *Site Web de GRUB*.
Page web.
http://www.gnu.org/s/grub/.

[17] *Documentation des variables SELinux (booleans) de Linux CentOS 5*.
Page web.
https://wiki.centos.org/TipsAndTricks/SelinuxBooleans.

[18] *Site Web des politiques SELinux du projet Fedora*.
Page web.
https://fedoraproject.org/wiki/SELinux/Policies.

[19] *Site Web du projet SELinux*.
Page web.
https://selinuxproject.org/page/Main_Page.

Version 2.0 - 03/10/2022 - ANSSI-BP-028
Licence ouverte / Open Licence (Étalab - v2.0)
ISBN : 978-2-11-167126-3 (papier)
ISBN : 978-2-11-167127-0 (numérique)
Dépôt légal : Octobre 2022

AGENCE NATIONALE DE LA SÉCURITÉ DES SYSTÈMES D'INFORMATION

ANSSI - 51, boulevard de La Tour-Maubourg, 75700 PARIS 07 SP
www.ssi.gouv.fr / conseil.technique@ssi.gouv.fr

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
Premier ministre
SGDSN