Université des Sciences et de la Technologie
Houari Boumediène

FACULTÉ D'ÉLECTRONIQUE ET D'INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

RAPPORT PROJET : SÉCURISER UN SERVEUR WEB

MODULE : SÉCURITÉ DES SYSTÈMES D'EXPLOITATION

PRÉSENTÉ PAR :

— Bennabi Abdelhakim.
— Bensaid Nabil.
— Bentorcha Camelia.
— Medouni Lotfi.
— Namane Madjid.
— Toumi Nassima.

Master I SSI - 2015/2016

Table des matières

Chapitre 1

Introduction

Le site web d'une entreprise représente sa vitrine vers le monde extérieur, il permet également de fournir un certain nombre de services aux clients. Son bon fonctionnement affecte donc directement celle-ci, de ce fait, l'entreprise doit assurer sa bonne configuration et sa sécurité à travers un certain nombre de mécanismes afin d'assurer les services attendus.

1.1 Services fournis :

Le serveur web doit contenir les fichiers nécessaires au déploiement du site web de l'entreprise, ces fichiers devront être accessibles au public en lecture, il faut également permettre aux utilisateurs autorisés d'administrer le serveur et contrôler le contenu du site.

1.2 Classification des ressources :

Confidentialité : Le site web d'une entreprise doit être public, de ce fait les fichiers de ce site web ne doivent pas être protégés pour la lecture. Les autres fichiers du serveur tels que les fichiers de *log* doivent quant à eux être protégés à travers des mécanismes de cryptage et de contrôle d'accès discrétionnaires et obligatoires pour définir les privilèges et éviter les escalations de ceux-ci.

Intégrité : L'intégrité de tous les fichiers du serveur doit être préservée, y compris les fichiers accessibles au public, un défacement du site web pourrait nuire à l'image de l'entreprise ou son fonctionnement tout entier. Il faut donc mettre en place des mécanismes de contrôle d'accès DAC et MAC.

Disponibilité : Le serveur doit permettre en tous temps à tout utilisateur autorisé à accéder aux services demandés. Les contrôles d'accès ne devront donc pas interdire à l'utilisateur un accès qu'il est sensé obtenir, il faut également prévenir les dénis de service (DoS).

1.3 Profils d'utilisateurs :

Nous pouvons avoir plusieurs types d'utilisateurs selon leur position dans l'entreprise et leurs tâches à accomplir, nous les regrouperons par profils et veillerons à leur accorder les "moindres privilèges" :

1.3.1 Utilisateurs physiques :

Administrateurs système :

Un compte d'administrateur système est un compte d'utilisateur qui permet d'effectuer des modifications affectant le système et d'autres utilisateurs. Les administrateurs peuvent modifier des paramètres de sécurité, installer des logiciels et des matériels, et accéder à tous les fichiers de l'ordinateur. Ils sont également habilités à modifier d'autres comptes d'utilisateurs.
L'utilisateur admin dispose d'un accès en lecture/écriture sur les objets et les comptes utilisateurs.

Administrateurs réseau :

Les administrateurs réseau doivent s'assurer que les services de communication réseau fonctionnent correctement.

Développeurs web :

Sont chargés de la création et de la mise à jour du contenu du site web, devront donc avoir les droits de modification sur le répertoire correspondant (*/var/www* par exemple).

Visiteurs du site web :

Les hôtes extérieurs au réseau local doivent passer par l'application serveur web afin de consulter le site web, par contre les hôtes du réseau local doivent pouvoir accéder à celui-ci à partir de leurs comptes.
Les comptes doivent avoir le droit de lecture sur les fichiers du répertoire où se trouvent les pages du site web (*/var/www* par exemple).

1.3.2 Utilisateurs processus :

Tout processus doit se voir assigner un utilisateur spécifique avec des privilèges qui lui permettent de s'exécuter correctement sans privilèges additionnels, en général un processus ne doit pas être lancé avec les privilèges *root*, il faut également éviter qu'il soit lancé en tant que *nobody*. Ceci permet de personnaliser les privilèges et de les contrôler dans un premier temps, et de mieux analyser les fichiers log pour retracer les activités des applications.
Chaque utilisateur processus doit être confiné à son contexte d'exécution, de ce fait même si une de ses vulnérabilités venait à être exploitée elle ne pourrait pas être utilisée pour accéder aux fichiers appartenant à d'autres applications/utilisateurs.

Chapitre 2

Système d'exploitation du serveur

2.1 Choix de la distribution :

La distribution installée sur la machine du serveur doit être choisie selon un certain nombre de paramètres :

- Stabilité : Pas/peu de bugs/vulnérabilités.
- Cycle de vie (lifecycle) : durée pendant laquelle la distribution est maintenue, patchée, et au bout de laquelle l'OS doit obligatoirement être mis à jour. Un upgrade de système d'exploitation risque de fortement perturber le fonctionnement du serveur, de ce fait il est conseillé de choisir des distributions avec un long cycle de vie.
- Gestion de packages : Les deux gestionnaires *apt-get* et *yum* sont très similaires et présentent à peu près les même avantages. Une légère différence est que *yum* est beaucoup plus souple et user-friendly lors de la création/gestion des repositories.
- Nombre de packages proposés dans le repository standard et leur récence.
- Popularité : Ceci signifie qu'il existe une large communauté sur les forums pour aider en cas de problèmes et qu'une riche documentation officielle et non-officielle. Dans le monde Open Source ceci veut dire plus de contributeurs et donc plus de stabilité.

La distribution **CentOS** est basée sur RHEL (Red Hat Enterprise Edition) et est connue pour être stable, a un cycle de vie de 10 ans, il utilise *yum* pour la gestion des packages. Il est vrai qu'il ne présente pas autant de choix de packages dans le repository standard mais les packages voulus peuvent toujours être téléchargés à partir d'autres repositories.

Il garde également la même version de software durant tout son cycle de vie, ceci peut être interprété comme un gage de stabilité et une garantie de ne pas rencontrer des problèmes de compatibilité mais peut poser problème aux développeurs qui souhaitent toujours utiliser les dernières versions de logiciels, langages...etc La version 7 de CentOS est sortie très récemment et comporte de versions très récentes des logiciels dont notre serveur web aura besoin.

CentOS est également la distribution serveur la plus populaire depuis plusieurs années.Pour toutes ces raisons, nous avons porté notre choix sur cette distribution pour notre serveur web. [1] [2]

2.2 Schéma de partitionnement :

Pour notre système d'exploitation nous allons plutôt choisir de faire une séparation par partitions des différents répertoires importants tels que le : /home, /, /var...
Nous avons basé notre choix sur le fait de pouvoir :

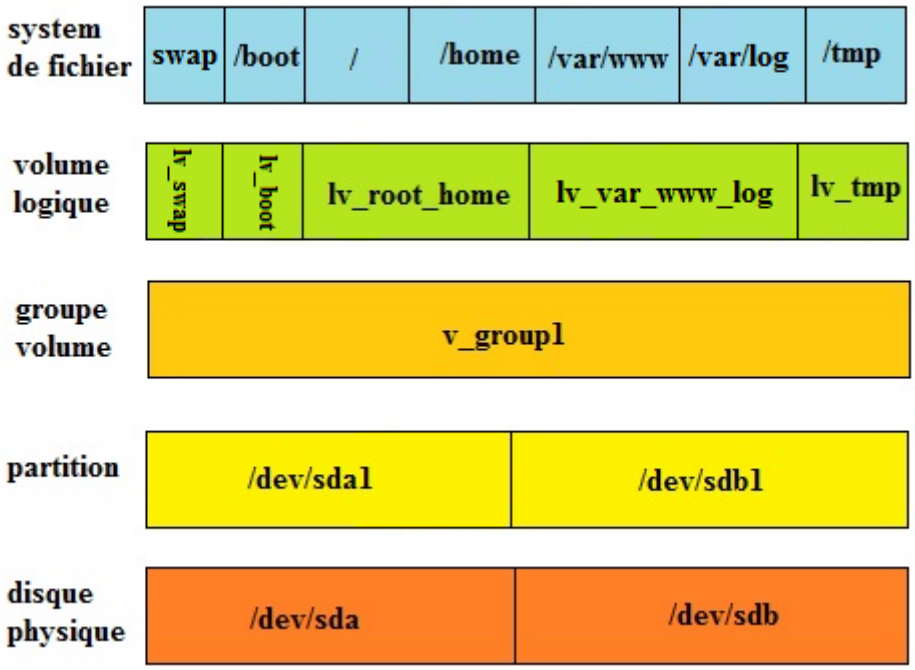
- Installer une nouvelle version (ou une nouvelle distribution), sans perdre la majorité de la personnalisation apportée, configuration ..., cette dernière opération implique la suppression des données qui existent, on peut réaliser cette séparation lors de l'installation du système.
- Elle simplifie le redimensionnement/migration vers une partition de plus grande de taille.
- Minimise les pertes en cas de dommage de la partition /, les données dans le /home ne sont pas perdues pour autant.
- L'utilisation de différents systèmes de gestion de fichiers, pour un temps d'accès rapide aux les données utilisées fréquemment, ou la fiabilité pour les données sensibles.

Pour effectuer cette opération de partitionnement on fait recours au LVM (Logical volume manager), celui-ci offre des avantages intéressants pour notre implémentation on peut citer :

- On peut regrouper n'importe quel nombre de disques comme étant un seul groupe logique.
- Nous pourrons avoir des volumes logiques étendus sur plusieurs disques.
- Réaliser une migration à chaud des volumes logiques utilisé par des services vers d'autres disques sans redémarrer les services.
- Permet de redimensionner de manière dynamique les volumes logiques.

Nous proposons ce schéma de partitionnement du serveur web :

1. http ://inthebox.webmin.com/choosing-a-linux-distribution-for-web-server
2. http ://www.infoworld.com/article/2687088/linux/how-to-choose-a-linux-server-distribution.html



— La partition *boot* est dédiée au démarrage du système.
— La partition *SWAP*, nécessaire au fonctionnement du système.
— Le dossier */* contient les fichiers système.
— Le dossier */home* contient les fichiers personnels des utilisateurs, en les mettant sur une partition séparée nous pourrons les récupérer dans le cas où le système d'exploitation est compromis.
— Le dossier */var/www* ou tout dossier contenant les fichiers du serveur web accessibles de l'extérieur, ceci en fait un emplacement vulnérable. **Remarque :** Pour des raisons de sécurité nous changerons ce chemin par défaut car les attaquent les ciblent en premier, il est donc préférable de cacher l'emplacement des fichiers dans notre système. Il sera accessible aux développeurs et à l'utilisateur du processus du serveur web.
— Le dossier */var/log* Contient les fichiers logs du système, le mettre sur une partition à part permettra de contrôler sa taille pour protéger le système contre les attaquent qui visent à augmenter la taille du fichier log de manière à provoquer un Dos. Il sera accessible uniquement aux administrateurs.
— Le dossier */tmp* est un dossier partagé et donc considéré comme vulnérable, il est donc préférable de le mettre sur une partition isolée, il faudra également mettre le *bit sticky* à 1 pour qu'un utilisateur ne puisse pas supprimer les fichiers qui ne lui appartiennent pas, nous devrons aussi empêcher l'exécution de scripts avec le flag *noexec*.

Remarque : Les flags *noexec*, *nodev* et *nosuid* devront être utilisés autant que possible afin de restreindre les accès.

2.3 Installation :

L'installation du serveur doit se faire hors ligne car chaque système d'exploitation comporte des vulnérabilités de conception et de configuration par défaut qui peuvent facilement être exploitées afin de compromettre le serveur dès sa mise en ligne.
Nous devons également installer le minimum de services/packages : tout service ajoute son lot de vulnérabilités, il faut aussi considérer le fait qu'un grand nombre de services augmente la taille des fichiers log inutilement et complique leur analyse.

2.4 Patch vulnérabilités :

Tout système d'exploitation propose des patchs qui permettent de corriger les vulnérabilités découvertes.
Là encore les patchs ne peuvent pas être téléchargés directement par le serveur qui est encore vulnérable et non configuré, il faut donc télécharger les patchs sur une machine différente protégée et les transférer à travers le réseau local sécurisé ou alors en les copiant sur un CD puis les appliquer.

2.5 Anti-spyware et anti-malware :

Leur installation permet de protéger le serveur contre les programmes et scripts malveillants qui pourraient porter atteindre à la confidentialité des informations à leur intégrité ou alors aux performances du système.

2.6 Configuration :

2.6.1 Sécurisation du BIOS et du GRUB :

La protection par mots de passe associés au BIOS (ou un équivalent du BIOS) et au chargeur de démarrage (bootloader) peut empêcher que des utilisateurs non-autorisés ayant un accès physique à la machine ne les démarrent à partir d'un média amovible ou ne s'octroient un accès super-utilisateur par le biais du mode mono-utilisateur.

BIOS :

Le mot de passe BIOS permettra de :

— Empêcher toute modification des paramètres du BIOS — Si un intrus a accès au BIOS, il ne pourra pas le configurer de manière à démarrer à partir d'une disquette ou d'un CD-ROM. De la sorte, nous l'empêcherons d'entrer dans un mode de secours ou un mode mono-utilisateur, lui permettant alors de lancer des processus arbitraires sur le système ou de copier des données confidentielles.
— Empêcher le démarrage du système — Certains BIOS permettent de protéger le processus de démarrage à l'aide d'un mot de passe. Lorsque cette fonctionnalité est activée,

tout agresseur est obligé de saisir un mot de passe avant que le BIOS ne puisse lancer le chargeur de démarrage.

Le bootloader GRUB :

Nous affecterons un mot de passe au GRUB, ce mot de passe sera demandé lors de toute tentative de modification des options de configuration ou lorsqu'on essaie de se connecter en mode *single user*. Un haché MD5 du mot de passe sera généré et devra être ajouté au fichier de configuration du GRUB. [3]

2.6.2 Gestion des disques :

Cryptage partitions :

Il existe des mécanismes qui permettent de crypter les systèmes de fichiers des partition que nous considérons sensibles, ceci permet de protéger les données dans le cas où les disques physiques sont dérobés.
L'attaquant devra dans ce cas introduire le mot de passe avant de pouvoir accéder aux données.

Désactivation montage automatique :

Le montage automatique des volumes externes peut être dangereux surtout si l'attaquant arrive à faire exécuter des scripts malveillants, il faut donc le désactiver, ceci peut être effectué en ajoutant le flag "noauto" dans le fichier */etc/fstab*.

2.6.3 Gestion des utilisateurs/mots de passe :

Mots de passe :

Les mots de passe devront être de taille dépassant le 8 caractères avec utilisation des majuscules, chiffres et caractères spéciaux afin de les renforcer.
Ils devront également être changés régulièrement, nous devrons donc fixer un délai au bout duquel ils devront être changés lors de la création de l'utilisateur, ainsi que le délai après lequel le compte sera désactivé si le mot de passe n'a pas été modifié car le compte serait considéré comme vulnérable.

Comptes par défaut et accès root :

Comptes invités : Les comptes par défaut et invités doivent être désactivés car inutiles et sources de vulnérabilités.

Single user : La connexion en mode peut être empêchée avec le mot de passe du GRUB comme cité précédemment.

3. http ://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-fr-4/s1-wstation-boot-sec.html

Commande su : Le login en tant que root peut être dangereux, celui-ci est désactivé par défaut sur Ubuntu mais pas sur CentOS, il faut donc effectuer la configuration adéquate afin d'empêcher la connexion sur ce compte en changeant le shell de connexion dans le fichier */etc/passwd* à */sbin/nologin*, les utilisateurs pourront toujours exécuter des commandes avec les privilèges root grâce à la commande *sudo*, après avoir introduit le mot de passe.

2.6.4 Services/protocoles :

Notre stratégie sera la suivante : on n'active que les services qui seront utilisés et indispensables au bon fonctionnement de notre implémentation car chaque lot de services apporte bien évidemment de nouvelles fonctionnalités qui peuvent augmenter le confort des utilisateurs/administrateurs mais apporte aussi un lot de vulnérabilité et limiter ces services limitera en partie cette présence de vulnérabilité donc nous-nous sommes simplement focalisés sur l'essentiel à savoir les services suivants :

- **HTTPS :** qui sera la version sécurise du protocole HTTP, on le sécurise en utilisant le service SSL/TLS qu'offre notre système, il exploitera le port 443.
- **PHP :** Notre serveur web ne se limitera pas à héberger des sites statiques, la plupart des sites web actuellement sont dynamiques et celui d'une entreprise le sera très certainement, c'est donc pour cela nous avons intégré ce dernier.
- **DNS :** Nous savons bien entendu que les personnes sont plus apte à mémoriser un nom logique qu'une suite de chiffre formant l'adresse IP, c'est pour cela que nous avons intégré ce service qui fera la traduction de l'adresse IP vers son nom de domaine associé. Il est préférable d'installer ce service sur une machine séparée dans le réseau.
- **FTP** : Nous avons décidé d'offrir à nos administrateurs web la possibilité de développer leur site web à distance, ce service sera donc nécessaire pour uploader et télécharger les fichiers qu'ils utilisent.
- Nous avons aussi décidé de mettre en place un système de **PKI** qui pourra générer des certificats afin d'authentifier notre serveur web comme étant un serveur valide ce qui est pour nous indispensable dans le cas où nous intégrons des systèmes de communication ou d'échange de données entre les utilisateurs et notre serveur, il faut leur permettre de vérifier qu'ils sont en train de communiquer avec notre serveur.
- Nous avons parlé plus haut de la possibilité d'offrir un accès distant à notre serveur, nous devons nous assurer que ces communications sont sécurisées et chiffrées afin de garantir la confidentialité et l'intégrité des échanges, pour cela nous avons opté pour la solution **SSH** qu'offre également CentOs car ce dernier offre un certain niveau de sécurité comme par exemple :

  - C'est un protocole orienté connexion utilisant exclusivement le protocole de transport TCP, ce qui permet d'assurer une fiabilité quant à la transmission de données.
  - Il utilise une cryptographie a clé publique qui permet de prouver l'authenticité de l'utilisateur à distance ainsi que la signature pour confirmer l'identité d'un individu ou hôte.
  - Les données sont chiffrées via des algorithmes de chiffrement symétrique tel qu'AES, 3DES. . . .
- Nous avons préféré le protocole SSL pour sécuriser les échanges entre les utilisateurs et

le serveur web implémenté.SSL peut être préférable dans nombreux cas «Messagerie, web...», des applications qui mettent en contact un très grand nombre de personnes, et pour lesquelles le mécanisme de certificats mis en place dans le navigateur côté client, et sur les serveurs fonctionne parfaitement bien et est utilisé préalablement par un large panel, ce qui va permettre une parfaite compatibilité avec ce qui existe au paravent tout en garantissant un très grand niveau de sécurité. SSL s'appuiera donc sur les certificats générés par notre PKI pour son fonctionnement.

— Nous savons que dans un système distribué il est très difficile d'assurer une parfaite synchronisation, CentOS nous permet de remédier a se problème en offrant un service nommé **NTP** (Network Time Protocol) qui permet de synchroniser, via un réseau informatique, l'horloge locale d'ordinateurs sur une référence d'heure. Cela va aussi jouer un rôle très important dans l'implémentation du logging et l'interprétation de plusieurs fichiers log de machines différentes après incident.

— **PAM (Pluggable Authentication Module)** : Permet de définir une stratégie d'authentification sans avoir à recompiler les logiciels existants.

— **LDAP (Lightweight Directory Access Protocol) :** Est un protocole pour l'accès à un répertoire de services, plus précisément les services basés sur le X.500. Basé sur le modèle client-serveur, il fonctionne sur TCP/IP.
Le modèle d'information du LDAP est basé sur des entrées, une entrée est une collection d'attributs qui a un nom globale distingué (DN) [Distinguished Name], chaque attribut a un type et une ou plusieurs valeurs, ces derniers sont des chaînes de caractères mnémoniques (CN) [Common Name],ou (mail) pour des adresse e-mail.
Notre choix s'est porté sur OpenLDAP car il offre entre autres :
  - L'authentification des machines.
  - L'authentification des utilisateurs.
  - Carnet d'adresse.
  - Un mécanisme de réplication.

Remarque : Bien entendu notre architecture peut être amenée à se développer et nous serons dans l'obligation d'ajouter ou de supprimer des services, cela reste flexible mais le principe reste le même : seuls les service exploitables seront activés.

2.6.5 Contrôle d'accès :

L'accès aux fichiers doit être réglementé selon les besoins des utilisateurs et la classification des informations/ressources. Nous utiliserons le contrôle d'accès discrétionnaire combiné avec le contrôle d'accès obligatoire pour une sécurité accrue.

DAC :

umask : La valeur du umask permet de spécifier les valeurs de droits par défaut lors de la création de fichiers, la valeur conseillée est **027**, ceci voudra dire que le propriétaire aura tous les droits, que les membres du groupes auront le droit de lire et d'exécuter et que les autres utilisateurs n'auront aucun droit.

Bit sticky : Doit être mis à 1 pour les répertoires partagés en particulier, de ce fait seuls les propriétaires des fichiers pourront les supprimer.

SetUID : Doit généralement être mis à 0 surtout pour les processus créés par le root ou les comptes ayant de nombreux privilèges.

SetGID : Pratique pour la gestion des accès aux fichiers dans des répertoires partagés.

MAC (SELinux) :

SELinux permet d'implémenter une politique d'accès obligatoire, une fois que le contrôle DAC a autorisé l'accès, SELinux vérifie sa propre base de permissions afin de garantir ou non cet accès.
Nous utiliserons SELinux avec la politique *targeted* en mode Type Enforcement qui se base sur les domaines/types des sujets et objets avant de garantir les accès.
Les utilisateurs Linux sont mappés à des utilisateurs SELinux, les privilèges/restrictions de ceux-ci leur seront appliqués. Par défaut un utilisateur créé se verra mappé à un utilisateur*unconfined-u* qui n'est pas sujet aux autorisations de SELinux, ceci constitue donc une faille et cette configuration par défaut doit être changée afin que les nouveaux utilisateurs se voient appliquer le plus de restrictions possibles puis selon leur rôle recevront plus de privilèges.
Les paramètres SELinux peuvent être consultés grâce aux commandes *sesearch* et *semanage*. [4]

2.6.6 Accès distant :

Comme cité précédemment, nous offrirons aux administrateurs et développeurs web un accès distant sécurisé au serveur à travers le protocole SSH.

Ce protocole est destiné aux administrateurs réseaux dans un environnement Unix, en effet de par les possibilités de terminal à distance qu'il offre, il a les caractéristiques idéales, de la connexion qui permet de configurer à distance les paramètres d'un réseau tout en gardant les fondement de la sécurité bien en évidence.
Nous devrons spécifier les utilisateurs pour lesquels la connexion à distance est autorisée, utiliser des ports non standards pour que la communication ne soit pas détectée avec un port scanning et de ce fait autoriser le flux à travers ce port dans le firewall.

2.7 Maintenance :

2.7.1 Scan vulnérabilités et pen-testing :

[5] [6] Une fois la configuration effectuée, nous devons lancer un scan de vulnérabilités afin de détecter tout oubli/erreur de configuration ou toute vulnérabilité non patchée. Il existe plusieurs scanners de vulnérabilités dont nous citons : Internet Security Systems (ISS) Internet Scanner, Metasploit, Nessus, Retina et SAINT.

4. http ://doc.fedora-fr.org/wiki/SELinux
5. NIST SP800-123
6. NIST SP800-44v2

Le pen-testing permettra d'effectuer une série d'attaques sur le serveur afin d'analyser son comportement face à celles-ci et s'assurer qu'il est bien protégé contre ces attaques. Il faut néanmoins effectuer ces tests sur un serveur dupliqué au cas où l'une des attaques cause des dommages irréversibles au système.

2.7.2 Logging :

Pour le logging nous utiliseront le logiciel *rsyslog*, en plus des fonctionnalités de *syslogd*, il permet la transmission des fichiers log via TCP, la possibilité d'utiliser des formats de bases de données et permet de crypter les données de logging lors de leur envoi à l'emplacement de stockage distant. [7]

2.7.3 Audit :

L'audit du système, à travers des logiciels tels qu'*auditd* permet aux administrateurs système d'effectuer le suivi des activités en détectant les accès non autorisés ou les modifications des données. Par défaut, auditd contrôle les "AVC denials" de SELinux, les logins système, les modifications sur les comptes ainsi que les événements d'authentification. Tous ces événements sont enregistrées dans le fichier */var/log/audit/audit.log*. [8]

2.7.4 Scan fichiers log :

L'analyse et interprétation manuelle des fichiers log peut être fastidieuse, il est préférable d'utiliser des outils qui effectuent des scans réguliers des fichiers log et détectent les événements anormaux, parmi les scanners de fichiers log que nous pouvons citer : analog, cronolog, swatch, wwwstat. [9]

2.7.5 Planification mises à jour :

Les mises à jour des applications ainsi que les patchs doivent être effectuées régulièrement, il faut donc planifier leur téléchargement. Toutefois, ces mises à jour ne doivent pas être installées avant d'avoir été vérifiées et testées sur un deuxième serveur afin de s'assurer qu'elles ne perturbent pas le fonctionnement du serveur de production.

2.7.6 Planification de la procédure de sauvegarde et de reprise :

Il est conseillé d'effectuer des snapshots et backups rotationnels du système périodiquement et de transférer ceux-ci sur des machines distantes (de préférence en dehors du site de production pour les protéger des dommages physiques) pour les stocker de manière à pouvoir récupérer une version récente des données du système (peut être effectué avec *rsyslog*).
Il est également préférable dans la mesure du possible d'avoir un serveur redondant avec la dernière configuration valide connue qui serait mis en ligne au cas où le serveur principal est hors service, ceci permettrait d'assurer la disponibilité des services.

7. CIS_CentOS_Linux_7_Benchmark_v1.1.0
8. CIS_CentOS_Linux_7_Benchmark_v1.1.0
9. NIST SP800-44v2

Remarque : Il est conseillé d'utiliser *anacron* lors des planifications car il permet d'effectuer les tâches programmées même si le serveur était hors service au moment où elles devaient l'être.

Chapitre 3

Serveur web :

3.1 Serveur choisi :

Nous avons choisi d'installer le serveur web **Apache** pour les raisons suivantes :

— Le fait qu'il soit Open Source.
— Sa popularité, ce qui garantit une grande communauté sur les forums d'aide.
— Sa modularité, Apache est très flexible et permet de choisir les modules à charger.
— Ses modules et options de sécurité tels que *mod_security*.
— Sa portabilité, peut être installé sur la plupart des systèmes d'exploitation.

3.2 Installation :

Les configurations de serveur par défaut installent souvent une grande variété de services augmentant inutilement le risque pour le système. Les services et daemons d'exécution sur le serveur Web Apache devraient se limiter à ceux qui sont nécessaires. Un serveur Web doit fonctionner en tant que tel, et si possible ne doit pas être mélangé avec d'autres fonctions principales telles que la messagerie, DNS, base de données ou middleware.

3.3 Configuration :

Le serveur web doit être confiné dans un *chroot jail*, de ce fait même s'il est compromis le reste du système sera préservé. De plus nous devons activer une multitude de modules et mettre en place plusieurs mécanismes de contrôle :

3.3.1 Documentation :

La documentation dans le serveur ainsi que les scripts d'essai doivent être supprimés, nous devons également configurer le serveur pour ne pas révéler sa version ou les modules installés, ce qui pourrait être utile à l'attaquant pour retrouver les vulnérabilités connues et les exploiter. Ceci peut être effectué avec la directive *ServerTokens* qui doit être mise à la valeur "Prod".

Nous devons également désactiver le *directory listing* qui permet de lister le contenu des répertoires du serveur.

3.3.2 Protection contre les attaques XSS :

La protection contre les attaques XSS (Cross-Site Scripting) peut être activée simplement en modifiant le fichier de configuration *httpd.conf*.

3.3.3 Protection contre les DDos :

Les DDos peuvent être minimisés grâce à une série de mesures :

- Minimiser la valeur des timeouts.
- Activer la directive *KeepAlive* qui permet de réutiliser les mêmes sessions pour les mêmes utilisateurs.
- Le module *mod_evasive* est également très efficace.
- Afin de protéger le serveur contre les DDos dus aux attaques de TCP syn flooding, il est conseillé d'activer les TCP Cookies qui permettent d'enregistrer les connexions en attente directement dans le kernel ce qui permettra au serveur de continuer à accepter les demandes de connexion et donc un utilisateur légitime pourra toujours accéder aux services.

3.3.4 Modules :

En réduisant au minimum les modules activés à ceux qui sont réellement utilisés, nous réduisons le nombre de "portes" et cela permet donc de réduire la surface d'attaque du site.
De même ayant moins de modules signifie moins de logiciels qui pourraient contenir des vulnérabilités. Parmi les recommandations émises par le CIS pour la configuration d'un serveur Apache nous pouvons citer : [1]

Activer le module mod_security :

Ce module agit en tant que firewall et permet de suivre le trafic en temps réel, il permet également de protéger les sites web des attaques force brute.
Mod_security détecte et protège aussi des malwares basés web, des trjoans, bots, crawlers, scanners et d'une grande partie des attaques web.

Activer les modules d'authentification et d'autorisation :

Les modules *authn* * fournissent une authentification, tandis que les modules *authz* * fournissent autorisation.

1. CIS_Apache_HTTP_Server_2.2_Benchmark_v3.3.1

Activer le module Log Config

Le module *log_config* prévoit l'enregistrement flexible des demandes des clients, et prévoit la configuration de l'information dans chaque journal. Le Logging est critique pour l'utilisation de la surveillance.

Désactiver les Modules WebDAV

WebDAV est une extension du protocole HTTP qui permet aux clients de créer, déplacer et supprimer des fichiers et des ressources sur le serveur Web. WebDAV a des problèmes de sécurité car il peut permettre aux clients de modifier les fichiers non autorisés sur le serveur Web. Par conséquent, les modules WebDav (*mod_dav* et *mod_dav_fs*) doivent être désactivés.

Désactiver les module Status

Le module *mod_status* fournit des statistiques sur les performances du serveur en cours. il est recommandé que ce module soit désactivé :

Désactiver le module Autoindex

Le module autoindex est généré automatiquement par la page Web et permet de lister le contenu des répertoires sur le serveur. Il peut révéler des fichiers qui ne sont pas destinés à l'être. Les listes de répertoires automatisés ne doivent pas être activées car elles peuvent révéler des informations utiles à un attaquant comme les conventions de nommage et les chemins de répertoire.

Désactiver le module User Directory

La directive UserDir doit être désactivée afin que les répertoires personnels de l'utilisateur ne soient pas accessibles via le site web avec un tilde ( ) précédant le nom d'utilisateur. La directive définit également le nom de chemin du répertoire qui sera accessible. Par exemple : • http ://example.com/ Camelia/ pourrait accéder à public_html, sous-répertoire du répertoire personnel de l'utilisateur Camelia. • La directive UserDir pourrait mapper / root dans le répertoire racine (/).

Désactiver le module Info

Le module mod_info fournit des informations sur la configuration du serveur via l'accès à un emplacement de l'URL /server-info. Il est recommandé que ce module ne soit pas activé : • Une fois que mod_info est chargé dans le serveur, sa capacité de gestionnaire est disponible dans les fichiers .htaccess et peut laisser fuir des informations sensibles à partir des directives de configuration des autres modules Apache tels que les chemins du système, les noms d'utilisateur , mots de passe, noms de bases de données , etc.

3.4 Autorisations/propriétés DAC :

La Sécurité au niveau du système d'exploitation est le fondement vital requis pour un serveur Web sécurisé. Cette section se concentrera sur les autorisations et les privilèges de la plate-forme OS.

3.4.1 Exécuter le serveur Web Apache en tant que non-utilisateur root

Bien qu'Apache fonctionne habituellement avec les privilèges root pour écouter sur le port 80 et 443, il peut et doit fonctionner comme un non - utilisateur root pour effectuer les services Web.
Une des meilleures façons de réduire l'exposition à l'attaque lors de l'exécution d'un serveur Web est de créer un utilisateur non privilégié et unique ainsi qu'un groupe pour l'application serveur.
Le " nobody "ou" daemon " qui vient par défaut sur les variantes d'Unix ne doit pas être utilisé pour exécuter le serveur Web, puisque le compte est couramment utilisé pour d'autres services démon séparés.

3.4.2 Donner au compte utilisateur un Shell non valide

Le compte apache ne doit pas être utilisé comme un compte régulier de connexion, et devrait se voir attribuer un shell invalide ou nologin pour assurer que le compte ne puisse pas être utilisé pour se connecter.
Les comptes de service tels que le compte apache représentent un risque si elles peuvent être utilisées pour obtenir un shell de connexion au système.

3.4.3 Verrouiller le compte d'utilisateur Apache

Le compte d'utilisateur sous lequel Apache fonctionne ne devrait pas avoir un mot de passe valide, et doit être verrouillé pour empêcher les connexions. En général, il ne devrait pas être une nécessité pour quiconque d'utiliser "su" et lorsqu'il y a un besoin, "sudo" devrait être utilisé à la place, car il ne nécessiterait pas le mot de passe de compte apache.

3.4.4 Définir la propriété sur les répertoires et fichiers Apache

Les répertoires et les fichiers d'Apache devraient appartenir à root. Cela vaut pour tous les répertoires de logiciels Apache et fichiers installés.
Restreindre la propriété des fichiers et des répertoires permettra de réduire la probabilité de modifications non autorisées de ces ressources.

3.4.5 Group Id sur des répertoires et fichiers Apache

Les répertoires et les fichiers doivent être réglés pour avoir un groupe Id de racine. Cela vaut pour tous les répertoires de logiciels et fichiers installés.
La seule exception prévue est que le web documentroot Apache est susceptible d'avoir besoin

d'un groupe désigné pour autoriser le contenu Web d'être mis à jour par le biais d'un processus de gestion du changement.

3.4.6 Restreindre l'accès en écriture sur les répertoires et fichiers Apache

L'autorisation sur les répertoires Apache devrait être rwx r-x r-x (755) et les autorisations de fichier doivent être similaires. Cela vaut pour tous les répertoires de logiciels et fichiers installés à l'exception dans certains cas, les documents de la racine web sont susceptibles d'avoir besoin d'un groupe désigné pour permettre au contenu Web d'être modifié.
L'accès en écriture est susceptible d'être très utile pour une modification non autorisée du contenu Web, des fichiers de configuration ou d'un logiciel pour les attaques malveillantes.

3.4.7 Configurer les droits d'accès aux répertoires

La directive *allow* d'apache permet de spécifier les dossiers accessibles à distance ainsi que les adresses IP autorisées/interdites afin d'assurer la disponibilité du contenu web.

3.4.8 Configurer les droits d'accès au fichiers .ht*

Les fichiers .ht* tels que *.htaccess* contiennent beaucoup d'informations de configuration qui ne doivent pas être révélées.

3.5 Utilisation de SSL/TLS :

Comme cité précédemment, le protocole SSL est compatible avec Apache et permet de sécuriser les échanges et d'authentifier le serveur grâce à des certificats X509.

3.6 Apache et SELinux :

Il existe plusieurs utilisateurs/types SELinux prédéfinis avec des droits spécifiques aux cas d'utilisation, par exemple nous avons :

— **httpd_sys_content_t** : Lecture seule des fichiers utilisés par Apache.
— **httpd_sys_rw_content_t** : Fichiers et répertoires pouvant être lus et modifiés.
— **httpd_log_t** : Utilisé par Apache pour le logging.

3.7 Logging :

3.7.1 Configuration Syslog pour Error Logging

Il est facile pour les error logs du serveur Web d'être négligés dans le processus de surveillance des journaux, et pourtant les attaques au niveau des applications sont devenues communes et sont extrêmement importants pour la détection d'attaques précoce, ou des erreurs internes.

3.7.2 Configuration Access Log

La directive LogFormat définit le format et les informations à inclure dans les entrées du journal d'accès. La directive CustomLog spécifie le fichier journal, syslog.
Les journaux d'accès du serveur sont également précieux pour une variété de raisons.Ils peuvent être utilisés pour déterminer quelles ressources sont utilisées le plus.
Plus important encore, ils peuvent être utilisés pour étudier le comportement anormal qui peut être une indication qu'une attaque. Si le serveur enregistre uniquement les erreurs, il serait alors très difficile d'enquêter sur les incidents.

3.7.3 Log Storage et Rotation

Il est important qu'il y ait suffisamment d'espace disque sur la partition qui contiendra tous les fichiers journaux, et que la rotation des journaux soit configurée pour conserver au moins 3 mois ou 13 semaines si l'enregistrement central n'est pas utilisé pour le stockage. Il est conseillé de ne pas tenir tous les fichiers journaux Apache sur la partition racine du système d'exploitation. Cela pourrait entraîner un déni de service contre le serveur hébergeur en remplissant la partition racine et de provoquer un crash du système. Pour cette raison, il est recommandé que les fichiers journaux soient stockés sur une partition dédiée (notre cas pour la partition /var/-log). l'Enquête sur les incidents nécessite souvent l'accès à plusieurs journaux, ce qui explique pourquoi il est important de les garder disponibles au moins 3 mois.

Chapitre 4

Trafic réseau :

Comme le serveur web doit être accessible par l'extérieur, il devra être placé dans la zone DMZ du réseau, ceci l'expose à plusieurs attaques à travers le réseau ce qui impose une protection accrue et un contrôle des flux entrants et sortants.

4.1 xinetd et TCPWrappers :

4.1.1 xinetd :

Le service *inetd* étant considéré comme vulnérable, nous devons le désactiver et utiliser *xinetd* à sa place.
xinetd permet d'effectuer l'écoute des ports à la place des applications, permet également de mutualiser les contrôles d'accès.
Le service peut être configuré à partir des fichiers suivants :

— /etc/xinetd.conf : Contient des paramètres de configuration généraux qui influencent tous les services placés sous le contrôle de xinetd. Ce fichier n'est lu que lors du lancement du service xinetd, par conséquent, afin que des changements apportés à la configuration puissent prendre effet, l'administrateur doit redémarrer le service xinetd.
Parmi les options qui peuvent être spécifiées dans ce fichier :

- Le nombre maximal de requêtes que le service peut gérer à un moment donné.
- Création d'un fichier de journalisation dans le répertoire */var/log* et spécification des cas dans lesquels la journalisation est effectuée ainsi que les informations enregistrées.
- Nombre maximal de connexions par secondes à un service donné, si ce nombre est atteint le service est désactivé pendant un laps de temps.

— /etc/xinetd.d/ — Le répertoire contient les fichiers de configuration relatifs à chaque service géré par xinetd.
[1]

1. http ://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-fr-4/s1-tcpwrappers-xinetd-config.html

4.1.2 TCPWrappers :

TCP Wrappers est un outil de contrôle d'accès entrant à des services au niveau applicatif, les applications concernées sont celles compilées avec la librairie *libwrappers* et qui utilisent le protocole de transport TCP.

Offre plusieurs avantages par rapport aux autres techniques de contrôle de service réseau comme la transparence à la fois au client et au service réseau enveloppé, ainsi que la gestion centralisée des multiples protocoles - TCP Wrappers fonctionnent séparément des services de réseau qu'ils protègent, ce qui permet de nombreuses applications de serveur de partager un ensemble commun de fichiers de configuration de contrôle d'accès, ce qui rend la gestion plus simple. Il permet également de scanner les paquets cryptés.

Pour déterminer si un client est autorisé à se connecter à un service, TCP Wrappers vérifie séquentiellement le contenu des fichiers : */etc/hosts.allow* et */etc/hosts.deny*. Toute modification à hosts.allow ou hosts.deny prennent effet immédiatement, sans redémarrage des services réseau.
Parce que les règles d'accès à hosts.allow sont appliquées en premier, elles ont préséance sur les règles spécifiées dans hosts.deny . Par conséquent, si l' accès à un service est autorisé dans hosts.allow , une règle refusant l'accès à ce même service dans hosts.deny est ignorée.
Les règles de chaque fichier sont lues à partir du haut vers le bas et la première règle de correspondance pour un service donné est le seul appliqué. L'ordre des règles est extrêmement important.
Si aucune règle pour le service ne se trouve dans les deux fichiers, ou si aucun fichier existe, l'accès au service est accordé, il est donc essentiel d'ajouter une ligne "deny all" à la fin du fichier hosts.deny afin d'interdire tout trafic qui n'a pas été autorisé précédemment.

Nous ferons en sorte que seuls les flux autorisés pour les protocoles actifs du serveur soient acceptés, le reste du trafic devra être rejeté.
[2]

4.2 Pare-feu :

Du fait que TCPWrappers ne prend en charge que le trafic entrant TCP pour les applications compilées avec la librairie *libwrappers*, il ne peut assurer une protection complète, nous devons donc configurer un pare-feu pour contrôler tous les flux entrants et sortants.

Nous avons opté pour l'implémentation de pare-feu **IPTables** au niveau de notre serveur web qui s'avère être une infrastructure en ligne de commande permettant de filtrer le trafic entrant et sortant et par conséquent augmenter fortement le niveau de sécurité pour notre implémentation, nous devons juste mettre en place une stratégie qui limite le transit entre les utilisateurs et notre serveur web au minimum nécessaire, ce qui limitera les risques d'intrusions.

2. http ://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-fr-4/s1-tcpwrappers-access.html

Tout paquet entrant ou sortant est analysé afin de déterminer notamment sa source et sa destination ou encore les numéros de ports, ceux-ci seront comparés aux entrées de la table afin de savoir si le paquet est accepté ou rejeté.

La stratégie que nous avons adopté est simple et repose sur deux points :

— On bloque tout le trafic entrant par défaut.

— On autorise au cas par cas : le trafic appartenant ou lié à des connexions déjà établies et le trafic à destination des serveurs (Web, SSH, etc.) que nous souhaitons mettre à disposition.

Le serveur web se trouvera dans la DMZ afin d'être accessible à partir du réseau local et de l'extérieur (internet).

Remarque : Cela ne protégera pas notre serveur d'une attaque de type déni de service (DoS).

4.3 IDS et IPS :

Pour le choix de notre IDS/IPS nous avons longuement hésité entre plusieurs solutions, mais au final SNORT nous a paru l'IDS/IPS qui répondait le plus à nos besoins.

Pourquoi SNORT ?

— Coût : Le logiciel est Open Source et gratuit.

— Stabilité, vitesse et robustesse : Depuis sa création, l'objectif principal que se sont fixé les développeurs de SNORT était de le garder le plus léger possible afin de faire face à toujours plus de bande passante. Côté bugs ils sont presque inexistants vu que SORT est l'un des plus anciens IDS/IPS (1998).

— Les préprocesseurs utilisés par SNORT analysent le flux de données en temps réel ce qui permet une détection plus efficace des paquets malveillants. Mais ce n'est pas tout, ces préprocesseurs sont surtout caractérisés par leur capacité à vaincre plusieurs techniques d'évasion d'IDS.

— La flexibilité : SNORT est très flexible dans son déploiement et cela est dû au fait qu'on peut personnaliser les signatures existantes, voir même crée nos propres règles.

— Soutien de l'industrie : La communauté Open Source de SNORT est très active et tient des mises à jour régulières (voir même plusieurs par semaine) des signatures afin de contrer les ravages causés par les vers sur internet.