Sécurisation d’un site web

Sommaire

1. Audit de sécurité..... 2
2. Étude des failles de sécurité web courantes..... 2
3. Mise en place d'extensions de sécurité..... 12

1. Audit de sécurité

Wordpress est à jour avec les extensions (Kubio) et le mdp est sécurisé.

Il n’y a que mon compte

Les infos sensibles sont : CV, nom des lieux de stage pouvant conduire a une localisation de la ville

2. Étude des failles de sécurité web courantes

1) Qu'est-ce qu'une injection SQL et comment s'en protéger ?

Une injection SQL est une technique d'attaque utilisée par des pirates informatiques pour compromettre des systèmes informatiques qui utilisent des bases de données.

Elle implique l'insertion de code SQL malveillant dans les entrées utilisateur d'une application, souvent via des formulaires web ou d'autres interfaces utilisateur. L'objectif est de manipuler la logique de la requête SQL de manière à obtenir un accès non autorisé à la base de données ou à exécuter des opérations indésirables telles que la suppression ou la modification de données.

Pour se protéger contre les injections SQL, voici quelques pratiques :

1. Utiliser des requêtes préparées ou des procédures stockées : Les requêtes préparées permettent de séparer les instructions SQL de leurs données, ce qui rend l'injection SQL beaucoup plus difficile.

2. Valider et filtrer les entrées utilisateur : Toutes les entrées utilisateur doivent être validées et filtrées pour s'assurer qu'elles ne contiennent pas de caractères spéciaux ou de codes SQL malveillants.

3. Limiter les privilèges de la base de données : Accorder uniquement les privilèges nécessaires aux utilisateurs de la base de données peut réduire la surface d'attaque en cas de succès d'une injection SQL.

4. Utiliser un pare-feu d'application Web (WAF) : Un WAF peut aider à détecter et à bloquer les tentatives d'injection SQL en surveillant le trafic web et en appliquant des règles de sécurité.

5. Mettre à jour régulièrement le logiciel : Assurez-vous que tous les logiciels utilisés dans votre application, y compris le système de gestion de base de données (SGBD), sont régulièrement mis à jour pour corriger les vulnérabilités connues.

2) Expliquez le principe des attaques XSS (Cross-Site Scripting). Quels en sont les différents types ?

Les attaques XSS (Cross-Site Scripting) sont des techniques d'exploitation qui permettent à un attaquant d'injecter du code malveillant (généralement du code JavaScript) dans des pages web consultées par d'autres utilisateurs. Ces attaques exploitent les failles de sécurité des applications web qui permettent l'exécution de scripts côté client, souvent dans un navigateur web.

Le principe général des attaques XSS est d'insérer du code JavaScript non sécurisé dans les entrées utilisateur, telles que les champs de formulaire ou les paramètres d'URL, et de les envoyer à un serveur web. Lorsque d'autres utilisateurs accèdent à la page web contenant ces données, le code JavaScript malveillant s'exécute dans leur navigateur, ce qui peut entraîner diverses conséquences néfastes, telles que le vol de sessions utilisateur, la redirection vers des sites malveillants ou l'exécution d'actions non autorisées au nom de l'utilisateur.

Il existe plusieurs types d'attaques XSS :

1. **XSS stocké :** Aussi connu sous le nom de XSS persistant, cette attaque consiste à stocker du code JavaScript malveillant dans une base de données ou un autre espace de stockage côté serveur. Lorsque les utilisateurs consultent la page web, le code JavaScript est récupéré à partir de la base de données et exécuté dans leur navigateur.

2. **XSS réfléchi :** Dans ce type d'attaque, le code JavaScript malveillant est inclus dans un lien, un formulaire ou un autre élément d'une requête HTTP envoyée au serveur. Le serveur renvoie ensuite cette valeur à l'utilisateur dans une réponse

HTTP sans la modifier, ce qui permet à l'attaquant de l'exécuter dans le navigateur de la victime.

3. **XSS basé sur le DOM :** Cette variante d'attaque XSS exploite les vulnérabilités au niveau du Document Object Model (DOM) du navigateur. Le code JavaScript malveillant est injecté dans le DOM de la page web côté client et exécuté par le navigateur de la victime lors de la manipulation de l'arbre DOM par le code JavaScript de la page.

3) Pourquoi est-il important de bien configurer les permissions d'accès aux fichiers sur un serveur web ?

Il est important de bien configurer les permissions d'accès aux fichiers sur un serveur web pour plusieurs raisons :

1. **Protection des données sensibles :** En restreignant l'accès aux fichiers sensibles, tels que les fichiers de configuration, les bases de données et les fichiers contenant des informations confidentielles, vous réduisez le risque d'accès non autorisé et de vol de données.

2. **Prévention des attaques :** Des permissions mal configurées peuvent permettre à des attaquants d'exploiter des vulnérabilités pour accéder à des fichiers sensibles ou exécuter du code malveillant sur votre serveur. Une configuration appropriée des permissions peut aider à bloquer ces attaques.

3. **Garantie de l'intégrité des fichiers :** En limitant les permissions d'écriture aux fichiers uniquement lorsque cela est nécessaire et en restreignant les modifications aux utilisateurs autorisés, vous réduisez le risque de modification accidentelle ou malveillante des fichiers.

4. **Conformité aux réglementations de sécurité :** De nombreuses réglementations et normes de sécurité, telles que le Règlement général sur la protection des données (RGPD) en Europe, exigent que les entreprises prennent des mesures pour protéger les données personnelles. Une bonne configuration des permissions peut contribuer à la conformité à ces réglementations.

5. **Protection contre les erreurs de configuration :** Des erreurs de configuration peuvent entraîner des problèmes de sécurité et des failles de confidentialité. En

définissant des permissions appropriées dès le départ, vous réduisez le risque que de telles erreurs se produisent.

4) Qu'est-ce que le clickjacking et comment le prévenir ?

Le clickjacking, également connu sous le nom de "UI redressing", est une technique d'attaque utilisée pour tromper les utilisateurs et les inciter à cliquer sur des éléments web malveillants sans le savoir.

L'attaque consiste à superposer des éléments visuels transparents ou invisibles sur des sites légitimes, afin de détourner les clics de l'utilisateur vers des actions non souhaitées, telles que le téléchargement de logiciels malveillants, le partage de contenus sur les réseaux sociaux ou l'envoi involontaire d'informations sensibles.

Voici quelques méthodes pour prévenir le clickjacking :

1. **Utilisation de l'en-tête de politique de sécurité du contenu :** La CSP permet de contrôler les sources autorisées pour le chargement de ressources sur une page web. Vous pouvez configurer la CSP pour restreindre les cadres (iframes) à certaines sources autorisées, empêchant ainsi le clickjacking.

2. **Utilisation de la directive X-Frame-Options :** Cette directive HTTP permet de contrôler si une page peut être affichée dans un cadre (iframe). En configurant correctement cette directive, vous pouvez empêcher votre site web d'être intégré dans des cadres non autorisés.

3. **Utilisation de la directive de sécurité "frame-ancestors" dans la CSP :** Cette directive permet de spécifier les sites web autorisés à intégrer votre site web dans un cadre. En limitant les domaines autorisés, vous pouvez réduire le risque de clickjacking.

4. **Éducation des utilisateurs :** Sensibiliser les utilisateurs aux risques de clickjacking et les encourager à vérifier attentivement les actions qu'ils effectuent sur les sites web, en particulier lorsqu'ils sont invités à cliquer sur des éléments non familiers.

5) Décrivez le fonctionnement d'une attaque par déni de service distribué (DDoS)

Une attaque par déni de service distribué (DDoS) est une technique visant à rendre un service, une ressource ou une infrastructure informatique inaccessible aux utilisateurs légitimes en submergeant la cible avec un trafic réseau excessif. Voici comment fonctionne généralement une attaque DDoS :

1. **Recrutement de botnets :** Les attaquants utilisent généralement des botnets, des réseaux d'ordinateurs compromis et infectés par des logiciels malveillants, appelés "bots". Ces bots peuvent être des ordinateurs, des serveurs, des objets connectés ou d'autres appareils connectés à Internet. Les botnets sont contrôlés à distance par les attaquants, souvent à l'aide de logiciels malveillants.

2. **Coordination de l'attaque :** Les attaquants utilisent les botnets pour coordonner et lancer l'attaque DDoS. Ils envoient des instructions aux bots pour qu'ils génèrent et envoient un trafic réseau massif vers la cible désignée. Ce trafic peut être de nature variée, tels que des requêtes HTTP, des paquets UDP, des requêtes DNS, etc.

3. **Saturation des ressources :** Le trafic généré par les bots inonde la cible, saturant ses ressources réseau, ses capacités de traitement ou ses ressources système. Cela peut entraîner une surcharge des serveurs, des routeurs, des pare-feux ou d'autres composants de l'infrastructure, rendant le service inaccessible aux utilisateurs légitimes.

4. **Difficulté de mitigation :** En raison de la distribution géographique des bots et de leur nombre, il peut être difficile pour les défenseurs de l'infrastructure de filtrer ou de bloquer le trafic malveillant. De plus, l'identification et la neutralisation des bots individuels peuvent être complexes, car ils peuvent être répartis dans le monde entier et utiliser des techniques d'évasion pour éviter la détection.

5. **Durée variable :** Les attaques DDoS peuvent durer de quelques minutes à plusieurs jours, en fonction des ressources disponibles pour l'attaquant, de la résilience de la cible et des mesures de mitigation mises en place.

6) En quoi consiste la désérialisation non sécurisée d'objets et quel risque cela pose-t-il ?

La désérialisation non sécurisée d'objets est une vulnérabilité de sécurité qui survient lorsqu'une application accepte et traite des données sérialisées sans valider leur contenu ou leur origine. Cette vulnérabilité est souvent exploitée par des attaquants pour exécuter du code malveillant sur le serveur ou sur le client cible.

Voici comment cela fonctionne et quels sont les risques associés :

1. **Processus de désérialisation :** La désérialisation est le processus de conversion d'objets en données binaires (souvent dans un format comme JSON, XML, ou d'autres formats de sérialisation) pour les transmettre via le réseau ou les stocker dans des bases de données. Lorsque ces données sérialisées sont reçues par une application, elles sont désérialisées pour être retransformées en objets.

2. **Manque de validation :** Lorsque l'application effectue la désérialisation, elle ne vérifie pas toujours que les données sont sûres et légitimes. Les attaquants peuvent alors manipuler les données sérialisées en y insérant du code malveillant.

3. **Exécution de code malveillant :** Une fois que le code malveillant est désérialisé et exécuté par l'application, il peut avoir des conséquences graves. Cela peut inclure l'exécution de commandes système, l'accès à des ressources sensibles, la compromission du serveur ou l'exécution d'autres actions non autorisées.

4. **Risques pour la sécurité :** Les risques associés à la désérialisation non sécurisée d'objets sont nombreux. Ils incluent la compromission de la confidentialité et de l'intégrité des données, la prise de contrôle du système, le vol d'informations sensibles, les attaques par déni de service, et bien d'autres.

Pour prévenir les attaques liées à la désérialisation non sécurisée d'objets, il est important pour les développeurs de :

- Valider et filtrer les données sérialisées avant de les désérialiser.

- Limiter les privilèges de désérialisation aux seules classes et packages nécessaires.

- Utiliser des mécanismes de sécurité tels que la signature numérique ou le chiffrement pour vérifier l'origine et l'intégrité des données sérialisées

- Utiliser des bibliothèques de désérialisation sécurisées et mettre à jour régulièrement les frameworks et les dépendances pour bénéficier des correctifs de sécurité.

- Éduquer les développeurs sur les bonnes pratiques de sécurité en matière de désérialisation d'objets.

7) Qu'appelle-t-on une vulnérabilité de type "XXE" (XML eXternal Entity) ?

Une vulnérabilité de type "XXE" (XML eXternal Entity) est une faille de sécurité qui survient lorsqu'une application XML accepte et traite des entrées XML contenant des références à des entités externes. Les entités externes permettent de référencer des données stockées à l'extérieur du document XML en cours de traitement, telles que des fichiers locaux ou des ressources réseau.

Voici comment cela fonctionne et quels sont les risques associés :

1. **Traitement des entités externes :** Lorsqu'une application XML traite un document XML contenant des entités externes, elle peut accéder à ces entités pour les inclure dans le document en cours de traitement.

2. **Risque de lecture de fichiers sensibles :** Si une application XML n'est pas correctement configurée pour désactiver le chargement des entités externes ou pour restreindre leur accès aux ressources autorisées, un attaquant peut utiliser cette

vulnérabilité pour lire des fichiers sensibles présents sur le système de fichiers du serveur, tels que des fichiers de configuration, des mots de passe ou des données confidentielles.

3. **Risque d'accès réseau non autorisé :** En plus d'accéder aux fichiers locaux, une attaque XXE peut également être utilisée pour effectuer des requêtes HTTP vers des ressources réseau internes ou externes, permettant ainsi à l'attaquant de récupérer des informations sensibles ou de provoquer une déni de service en saturant le réseau.

4. **Risque d'injection de données malveillantes :** Les attaquants peuvent également utiliser une vulnérabilité XXE pour injecter du code malveillant dans les documents XML et exécuter des attaques telles que l'injection de commandes, l'injection de SQL ou d'autres formes d'attaques par injection.

8) Citez au moins trois bonnes pratiques essentielles pour sécuriser l'authentification sur un site web.

Pour sécuriser l'authentification sur un site web, voici trois bonnes pratiques essentielles :

1. **Utilisation de la connexion sécurisée HTTPS :** Assurez-vous que le processus d'authentification se déroule sur une connexion sécurisée HTTPS plutôt que HTTP. Le chiffrement HTTPS protège les informations sensibles, telles que les identifiants de connexion, contre l'interception par des tiers. Il empêche également les attaques de type "man-in-the-middle" qui pourraient compromettre les données d'authentification pendant leur transmission.

2. **Stockage sécurisé des mots de passe :** Ne stockez jamais les mots de passe en clair dans la base de données. Utilisez plutôt des fonctions de hachage sécurisées, telles que bcrypt ou PBKDF2, pour stocker les mots de passe de manière sécurisée. Assurez-vous également de saler les mots de passe avant de les hacher, ce qui ajoute une couche de sécurité supplémentaire en rendant les attaques par force brute plus difficiles.

3. I**mplémentation de la vérification en deux étapes (2FA) :** Encouragez l'utilisation de la vérification en deux étapes pour renforcer la sécurité de l'authentification. La 2FA exige non seulement un mot de passe, mais également une deuxième méthode d'authentification, telle qu'un code généré par une application

d'authentification ou envoyé par SMS. Cela rend beaucoup plus difficile pour un attaquant d'accéder au compte même s'il a réussi à compromettre le mot de passe.

9) Quelles sont les étapes clés d'une gestion saine des mises à jour et correctifs de sécurité ?

Une gestion saine des mises à jour et correctifs de sécurité est essentielle pour maintenir un niveau de sécurité optimal sur les systèmes informatiques. Voici les étapes clés à suivre :

1. **Identification des vulnérabilités :** Surveillez les sources d'informations sur les vulnérabilités, telles que les bulletins de sécurité, les annonces des fournisseurs de logiciels et les bases de données CVE (Common Vulnerabilities and Exposures), pour identifier les failles de sécurité qui pourraient affecter vos systèmes.

2. **Évaluation de l'impact :** Évaluez l'impact des vulnérabilités identifiées sur vos systèmes et vos données. Classez-les en fonction de leur gravité et de leur pertinence pour votre environnement.

3. **Planification des mises à jour :** Élaborez un plan de gestion des correctifs qui détaille les processus et les procédures à suivre pour appliquer les mises à jour de sécurité. Cela peut inclure la fréquence des mises à jour, les responsabilités des équipes de sécurité et des administrateurs système, ainsi que les canaux de communication pour informer les parties prenantes des changements à venir.

4. **Test des correctifs :** Avant de déployer les correctifs sur les systèmes de production, effectuez des tests approfondis pour vérifier qu'ils n'entraînent pas de dysfonctionnements ou de régressions. Cela peut impliquer la création d'environnements de test pour simuler les conditions de production.

5. **Déploiement des correctifs :** Appliquez les correctifs sur les systèmes de production de manière méthodique et planifiée. Assurez-vous de suivre les meilleures pratiques pour minimiser les interruptions de service et maximiser la disponibilité des systèmes pendant le processus de déploiement.

6. **Surveillance et validation :** Surveillez l'efficacité des correctifs déployés et vérifiez qu'ils ont résolu les vulnérabilités pour lesquelles ils ont été conçus. Utilisez

des outils de surveillance et de gestion des vulnérabilités pour détecter tout problème éventuel après le déploiement des correctifs.

7. **Documentation et rapport :** Documentez toutes les étapes du processus de gestion des correctifs, y compris les décisions prises, les correctifs appliqués et les résultats des tests. Produisez des rapports réguliers pour informer les parties prenantes de l'état de sécurité des systèmes et des mesures prises pour atténuer les risques.

10) Parmi ces failles, lesquelles concernent plus spécifiquement les sites WordPress ?

Les sites WordPress sont souvent vulnérables à plusieurs types de failles de sécurité en raison de leur popularité et de la complexité de l'écosystème WordPress. Voici quelques-unes des failles les plus courantes qui affectent spécifiquement les sites WordPress :

1. **Injection de code SQL (SQL Injection) :** Les attaques par injection SQL peuvent exploiter les vulnérabilités dans les plugins ou thèmes WordPress pour injecter du code SQL malveillant, compromettant ainsi la sécurité de la base de données du site.

2. **Cross-Site Scripting (XSS) :** Les vulnérabilités XSS peuvent permettre à un attaquant d'injecter du code JavaScript malveillant dans les pages web d'un site WordPress, souvent via des commentaires, des formulaires de contact ou d'autres entrées utilisateur non filtrées.

3. **Faille d'authentification :** Les attaques de force brute contre les comptes d'administration WordPress sont courantes, en particulier si les mots de passe sont faibles ou si le nom d'utilisateur "admin" est utilisé. Les vulnérabilités dans les mécanismes d'authentification peuvent également être exploitées pour compromettre les comptes d'utilisateurs légitimes.

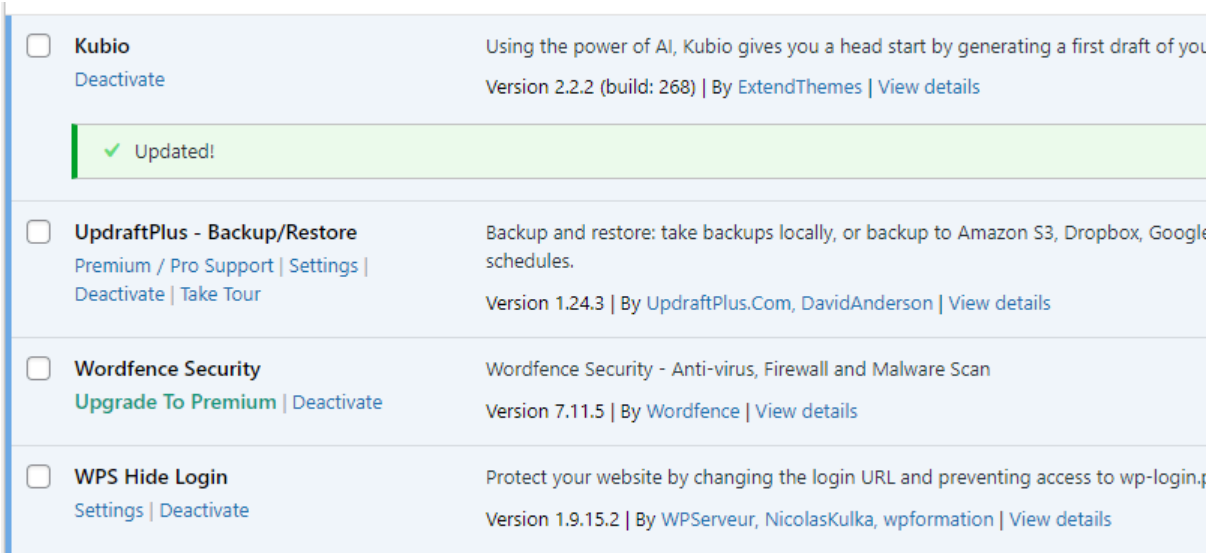
4. **Failles de sécurité des plugins et thèmes :** Les plugins et thèmes tiers peuvent contenir des failles de sécurité qui peuvent être exploitées pour compromettre un site WordPress. Il est important de maintenir tous les plugins et thèmes à jour et de n'utiliser que des sources fiables pour les télécharger.

5. **Mauvaise configuration des permissions de fichier :** Une mauvaise configuration des permissions de fichier sur le serveur WordPress peut permettre à un attaquant d'accéder à des fichiers sensibles ou de modifier le code source du site.

6. **Exposition de données sensibles :** Les informations sensibles telles que les noms d'utilisateur, les mots de passe ou les données personnelles peuvent être exposées en raison de failles de sécurité dans les plugins, les thèmes ou la configuration du site WordPress.

3. Mise en place d'extensions de sécurité

J'ai installé les extensions de sécurité mentionnée dans le site, les screens sont en dessous.



Les points suivants ont été effectués.

Quand j'aurai obtenu le certificat SSL mon site sera en HTTPS

6. Politique de sécurité

Portfolio, étudiant ou non, la sécurité concerne tout le monde, être étudiant n'épargne en aucun cas les attaques et pirates informatiques.

Les sauvegardes, mots de passe robustes sont très important, mais aussi attention a ce que l'on diffuse ou a ce que l'on publie dessus.

Si un intrus est repéré, il faut informer les autorités, fermer le site si on le peut ou informer le professeur dans notre cas.