Révision : Regex, CSRF, XSS

Table des matières

Regex

Regex est une suite de caractères permettant de rechercher, filtrer ou remplacer du texte selon un motif précis. C'est un outil puissant pour vérifier, extraire ou modifier du texte comme par exemple dans du codage afin de se faciliter la vie.

Regex symbole et signification :

Symbole	Signification

.	Permet de tout sélectionner
^	Indique le début d'une ligne
$	Indique la fin d'une ligne
*	Répète l'élément sélectionner 0 ou plusieurs fois
+	Répète l'élément sélectionner 1 ou plusieurs fois
?	Rend l'élément sélectionner facultatif
-	Permet de sélectionner une plage de lettre
{n}	Répète l'élément sélectionner précédent exactement n fois.
{n,}	Permet de préciser le nombre de récurrence d'un caractère
{x,y}	Permet de préciser le nombre de fois que le caractère peut apparaître
()	Permet de regrouper des caractère
\n	Permet de sélectionner un groupe
(?:)	Permet de séparer et ne pas être sélectionner dans une référence
|	Signifie « ou » (alternative)
\	Afin de sélectionner des caractères spéciaux comme des caractère normaux
\w	Sélectionne tout les caractère alphanumérique
\W	Sélectionne tous les caractères autres que lettres, chiffre
\d	Uniquement des caractère numérique
\D	Uniquement des caractère non numérique
\s	Uniquement les espaces
\S	Uniquement les non espaces

Faille CSRF

Qu'est-ce que c'est ?

Une faille CSRF (Cross-Site Request Forgery), ou falsification de requête intersite, est une vulnérabilité de sécurité web qui permet à un attaquant d'exécuter des actions à la place d'un utilisateur authentifié, sans son consentement.

Comment ça marche :

Cela passe par une url faussée envoyez à la cible afin de voler son compte sachant que l'utilisateur est déjà connecté au vrai site ce qui va permettre de récupérer ses informations et d'y accéder car dans l'url des informations seront rajoutée. Exploite la session active d'un utilisateur authentifié pour exécuter des actions malveillantes sur un site web. L'attaquant crée un lien ou un formulaire malveillant, souvent via un email ou un site piégé. Lorsque l'utilisateur clique dessus, son navigateur envoie automatiquement une requête au site légitime, avec ses cookies de session, ce qui permet à l'attaquant d'effectuer des actions sans le consentement de l'utilisateur.

Quel est son objective ?

L'objectif principal d'une attaque CSRF est de réaliser des actions malveillantes ou frauduleuses en exploitant la confiance d'un utilisateur authentifié sur un site. L'attaquant tire parti de l'absence de vérification de l'origine de la requête sur le site cible pour effectuer des actions telles que : Vol d'argent ; Modification de données sensibles ; Exploitation d'un service ; Propagation de l'attaque. Cette attaque permet à l'attaquant de prendre le contrôle

d'actions sensibles effectuées par l'utilisateur sur un site web, tout en exploitant la confiance que le site accorde à l'utilisateur authentifié.

Qui sont les cibles ?

- Les utilisateurs authentifiés : Toute personne connectée à un site web
- Les sites offrant des actions sensibles : Les plateformes où les utilisateurs peuvent effectuer des actions critiques
- Les sites sans protections appropriées : Les sites qui ne mettent pas en place de mécanismes de sécurité

Les utilisateurs actifs sur des sites où des actions sensibles sont réalisées, et plus particulièrement ceux sur des sites vulnérables ou mal sécurisés, sont les cibles principales de cette attaque.

Comment se protéger ?

- Token CSRF : générer un token unique pour chaque session utilisateur et l'inclure dans chaque requête sensible. Lorsqu'une requête est soumise, le serveur vérifie que le token envoyé correspond à celui stocké dans la session de l'utilisateur. Si les tokens ne correspondent pas, la requête est rejetée.
- Limiter l'usage des méthodes HTTP sensibles : Ne permettez pas que des actions sensibles soient effectuées via une simple requête GET. Utilisez plutôt des POST ou PUT, qui nécessitent généralement une action plus explicite de l'utilisateur.
- Authentification renforcée pour les actions critiques : Pour les actions sensibles demandez une revalidation de l'utilisateur (ex. : saisir à nouveau son mot de passe ou utiliser une méthode d'authentification à deux facteurs).

Conclusion :

En conclusion, l'attaque CSRF est une vulnérabilité sérieuse qui exploite la confiance d'un utilisateur authentifié pour effectuer des actions malveillantes sans son consentement. Cette faille peut entraîner des conséquences graves, telles que le vol d'argent ou la modification de données sensibles. Les cibles principales sont les utilisateurs connectés à des sites vulnérables, notamment ceux qui permettent des actions critiques sans protections adéquates. Pour se protéger contre ces attaques, il est essentiel d'utiliser des tokens CSRF, de limiter l'utilisation des méthodes HTTP sensibles, et d'implémenter des mesures d'authentification renforcée pour les actions critiques. La mise en œuvre de ces pratiques de sécurité permet de réduire considérablement les risques d'exploitation de cette vulnérabilité.

Faille XSS

Qu’est-ce que c’est ?

Une faille XSS (Cross-Site Scripting) est une vulnérabilité de sécurité qui permet à un attaquant d'injecter du code malveillant (souvent du JavaScript) dans une page web vue par d'autres utilisateurs.

Comment ça marche ?

L'attaquant insère du code malveillant dans un formulaire, un champ de recherche ou une URL d'un site vulnérable, généralement en exploitant l'absence de validation ou d'échappement des entrées utilisateur. Ce code malveillant est ensuite exécuté par le navigateur de la victime lorsqu'elle charge la page web concernée. Selon le type d'attaque XSS, le code peut être injecté dans une page HTML, un script ou un attribut, affectant ainsi l'expérience de l'utilisateur.

Attaque XSS persistant/non persistant différence ?

La faille XSS persistante (ou stockée) se produit lorsque le code malveillant injecté par l'attaquant est stocké sur le serveur (par exemple, dans une base de données) et est exécuté chaque fois que la page contenant le contenu malveillant est chargée par une victime. En revanche, la faille XSS non persistante (ou réfléchie) intervient lorsque le code malveillant est directement exécuté dans le navigateur de la victime, généralement après que cette dernière ait cliqué sur un lien ou soumis un formulaire contenant la charge malveillante. La principale différence réside dans le fait que l'attaque persistante reste active et affecte tous les utilisateurs qui accèdent à la page, tandis que l'attaque non persistante n'affecte que la victime ayant cliqué sur le lien ou soumis le formulaire.

Qui sont les cibles ?

Utilisateurs : Les attaquants visent principalement les utilisateurs qui naviguent sur des sites vulnérables aux attaques XSS. Ces utilisateurs peuvent être victimes de vol de cookies, de données sensibles, ou être redirigés vers des sites malveillants qui exploitent leur session pour commettre des actions non autorisées.

Les sites web vulnérables : Tout site qui ne valide pas correctement ou ne filtre pas les données des utilisateurs est une cible potentielle.

Administrateurs et modérateurs de sites : Un attaquant peut cibler des administrateurs ou modérateurs de sites pour prendre le contrôle de fonctionnalités critiques.

Comment se protéger ?

Validation des entrées : Vérifiez et nettoyez toutes les entrées utilisateur pour éviter l'injection de code malveillant.

Échappement des sorties : Assurez-vous que tout contenu dynamique insérer dans les pages web est correctement échappé pour empêcher l'exécution de JavaScript.

Éviter l'insertion directe de données utilisateurs dans le HTML/JS sans validation préalable.

Conclusion :

La faille XSS (Cross-Site Scripting) permet à un attaquant d'injecter du code malveillant dans une page web, affectant les utilisateurs en volant leurs informations sensibles ou en manipulant le contenu de la page. Les attaques XSS peuvent être persistantes ou non persistantes, selon que le code malveillant soit stocké sur le serveur ou exécuté immédiatement dans le navigateur de la victime. Les cibles principales sont les utilisateurs, les sites vulnérables et les administrateurs. Pour se protéger, il est essentiel de valider les entrées, d'échapper les sorties et de ne jamais insérer directement des données non sécurisées dans du code HTML ou JavaScript.

Injection SQL

L'injection SQL est une vulnérabilité qui permet à un attaquant d'insérer du code SQL malveillant dans une requête, exploitant ainsi des failles de validation des entrées. L'objectif est de manipuler la base de données pour voler, modifier ou supprimer des données. Pour se protéger, il est crucial d'utiliser des requêtes préparées et des paramètres liés dans les requêtes SQL, ainsi que de valider et échapper toutes les entrées utilisateurs.

Faille Include

La faille include permet à un attaquant d'inclure des fichiers distants ou locaux dans un script PHP, souvent pour exécuter du code malveillant. L'objectif est de charger des fichiers non autorisés pour obtenir des informations sensibles ou exécuter des actions malveillantes. Pour se protéger, il faut désactiver l'inclusion de fichiers distants dans la configuration PHP et valider soigneusement les chemins de fichiers fournis par l'utilisateur.

Envoi de fichiers malicieux

L'envoi de fichiers malicieux consiste à uploader des fichiers (ex. : scripts, virus) sur un serveur vulnérable. L'objectif est d'exécuter du code malveillant ou de compromettre le serveur. Pour se protéger, limitez les types de fichiers autorisés à être téléchargés, utilisez des contrôles côté serveur (comme vérifier les extensions et effectuer des analyses antivirus) et restreignez les permissions d'écriture sur les répertoires sensibles.

Attaque Temporelle

Une attaque temporelle exploite les différences de temps de réponse d'un serveur pour extraire des informations sensibles, comme un mot de passe ou une clé de cryptage. L'objectif est de deviner des informations en analysant les retards dans les réponses. Pour se protéger, utilisez des fonctions de comparaison constantes (qui prennent le même temps) et introduisez des délais aléatoires dans les réponses du serveur pour rendre l'attaque plus difficile.

Hacher les mots de passe

Le hachage des mots de passe consiste à transformer un mot de passe en une chaîne de caractères fixe et irréversible à l'aide d'un algorithme de hachage. Ces algorithmes génèrent un "haché" unique basé sur l'entrée et un sel pour éviter les attaques par dictionnaire ou table de hachage pré-calculée. Le sel rend chaque haché unique, même pour les mots de passe identiques. Une fois un mot de passe haché, il ne peut pas être "déhaché", ce qui signifie qu'il est impossible de retrouver le mot de passe original à partir du haché, augmentant ainsi la sécurité.

Session :

Cela se passe coté serveur, il va crée un espace temporaire de stockage afin de suivre l'activité de l'utilisateur qui s'est connecté au préalable. Les sessions exister avant les token. Ainsi le serveur permet de crée une session ID en stockant nos informations et à chaque action elle vérifie la session avec l'id.

Cookie de Session :

Cela permet de stocker temporairement les informations d'un utilisateur et de maintenir sa connexion pendant la durée de sa navigation. Les informations de l'utilisateur seront associées à une session utilisateur et seront supprimés lorsque l'utilisateur mettras fin à sa session sur un site Web de manière automatique.

Token :

C'est une chaîne de caractères générer après l'authentification d'un utilisateur par le serveur afin de s'assurer qu'il s'agit bien de lui car c'est une preuve d'identité qui va lui permettre d'accéder et de s'identifier dans certaines ressources. Cela permet de remplacer le mot de passe de manière générale. Chaque token peu importe son type doit expirer au bout d'un certain temps.

Token CSRF :

C'est un jeton qui permet de se défendre contre les attaques CSRF. Il doit être unique à chaque session utilisateur et il doit être difficile à deviner car générer aléatoirement.

Baerer token :

Cela provient des API, c'est un jeton de type authentification qui permet de prouver qu'on est autorisée à accéder à certaines ressources. Cependant le token ne varie pas selon les utilisateurs, il est le même pour tous c'est-à-dire que si on possède ce token qui n'a pas expiré alors on peut consulter les informations qui y sont relié. Il permet donc seulement de gagner du temps au lieu d'envoyer et de rentrer ses informations à chaque requête

JWT (jison) Token :

Un JSON Web Token (JWT) est un jeton généré par un serveur lors de l'authentification d'un utilisateur. Il contient des informations signées avec une clé privée du serveur. Le jeton est envoyé au client et utilisé pour s'authentifier dans chaque requête

HTTP suivante. Le serveur vérifie la signature pour s'assurer de l'intégrité du jeton et de son authenticité. Ainsi le navigateur envoi une requête pour se connecter puis coté server cela va crée un jeton jwt avec une clé secrète et va renvoyez le jeton jwt au navigateur. L'utilisateur va encore faire une requête au sein du site et coté server il va vérifier si la signature du jeton JWT est correcte et récupérer les informations nécessaires puis il renvoi la réponse à l'utilisateur qui est donc le client.

Ce portfolio est développé dans le cadre de mon épreuve de BTS SIO option SLAM. Il vise à présenter mes compétences techniques ainsi que les projets sur lesquels j'ai travaillé au cours de ma formation.

Langage utiliser pour la création du portfolio :

- HTML5
- JavaScript (pour le mini jeux)
- CSS3

Fonctionnalité :

- Galerie de projets développés au cours du BTS
- Présentation de mes compétences et certifications

Charte graphique :

- Blanc/Noir/Rouge

- le site contient des liens PDF et des images

Que contient mon portfolio :

En-tête : le menu de navigation ainsi que mon numéro

Accueil : une brève présentation suivie du CV et de mes certifications.

Compétences techniques : ce que j'ai appris à utiliser durant mes deux années en BTS

Tableau de synthèse : Tableau qui récapitule mes compétences acquises

Veille : Sujet qui porte sur l'iot durant une année pour voir les améliorations dans ce secteur

Projets : les projets effectuer au cours de ces deux années

Stage : Mes rapport de stage durant les 5 semaines de la première année et mes 7 semaines de stage de la seconde année

Pied de page : Mes contact