Les tests

Test fonctionnel

Les tests fonctionnels sont destinés à s'assurer que, dans le contexte d'utilisation réelle, le comportement fonctionnel obtenu est bien conforme avec celui attendu.

Un test fonctionnel a donc pour objectif de dérouler un scénario composé d'une liste d'actions, et pour chaque action d'effectuer une liste de vérifications validant la conformité de l'exigence avec l'attendu.

Il existe deux types de test fonctionnel.

Le test fonctionnel de progression, réalisé sur une exigence qui vient d'être développée. C'est donc la première fois que ce test est réalisé, avant que le produit ne soit livré en production.

Le test fonctionnel de non-régression, réalisé sur une exigence existant déjà avant les développements en cours. Il permet de s'assurer que les évolutions et les corrections effectuées dans une nouvelle version n'ont pas entraîné de régressions sur les fonctionnalités existantes.

Au cours d'un test fonctionnel, si une action échoue ou qu'une vérification n'est pas satisfaite, on est alors en présence d'un signe de non-qualité dans le système, plus communément appelé bug.

2 types de bugs

On distingue deux types de bugs.

Le bug d'implémentation, qui correspond à un problème introduit lors du développement initial d'une exigence.

Le bug de régression, qui correspond à un effet de bord introduit lors du développement d'une autre exigence ou lors de la maintenance corrective sur l'exigence en question ou sur une autre.

Quoi qu'il en soit, quel que soit le type de bug, une fois détecté, il faut ensuite le qualifier avec méthode et le remonter à l'équipe de développement. Celle-ci va alors lancer la phase de correction et produire une version corrigée qui va repasser à la moulinette des tests fonctionnels.

Tests ergonomiques

Vérifier le positionnement des différents blocs de contenus composant les pages : images, vidéos et objets graphiques, lisibilité des contenus, etc.

Failles web

L'attaque Brute Force

Cette attaque est utilisée en cryptanalyse et est une des plus simple à exploiter techniquement parlant, elle consiste à trouver un mot de passe, qu'il soit en clair ou hashé afin de pouvoir prendre le contrôle d'un compte administrateur sur un site web, sur un serveur FTP, une base de données, une box internet etc…

Le brute force consiste avant tout à tester toutes les combinaisons de caractères possibles jusqu'à trouver la bonne, cela peut être assez rapide si le mot de passe est "123456", quelques secondes pourraient suffire contrairement à un mot de passe du type "Cu44Dd5e5d8d88!!?dd$***e$Rred" qui lui, pourrait ne jamais être trouvé par cette technique, il faudrait plusieurs dizaines d'années à la machine pour trouver une telle combinaison.

L'attaque par brute force est donc souvent combinée à des dictionnaires qui sont des fichiers texte de plusieurs gigaoctet de noms, prénoms, mot du dictionnaire etc… ils sont généralement composés d'un mot par ligne, chaque mot sera testé.

Heureusement, il existe des méthodes de protection de l'attaque brute force.

- Utiliser des combinaisons qui se composent de nombreux types de caractères différents
- Etablir une authentification multifactorielle
- Sécuriser le mécanisme de mot de passe (bloquée la fonction en cas de nombreux échec)

La faille XSS (Cross-site scripting)

XSS qui correspond à **cross-site scripting**, c'est une faille de sécurité qui permet **d'injecter du contenu dans une page**. Pour la petite histoire cette vulnérabilité aurait dû porter le nom CSS mais ce dernier était déjà pris par les feuilles de style qui contiennent du code en cascade (Cascading Style Sheet) qui permet d'agencer les pages web et d'y mettre de la forme.

Pour tester l'existence d'une vulnérabilité XSS il suffit par exemple **d'injecter** une alerte Javascript dans un formulaire ou une url. Si l'alerte apparaît, alors, la faille est présente et est exploitable.

On distingue deux types de failles XSS :

1) XSS permanent

C’est lorsque le script est stocké sur le serveur externe (base de données). Il est donc récupéré et exécuté à tous moments sur le site par n’importe quel utilisateur.

2) XSS non permanent

Le script est souvent intégré à une URL et est exécuté sans être stocké sur un serveur.

Comment s’en prémunir

Il faut absolument utiliser les fonctions PHP htmlspecialchars() qui filtre les ‘<‘ et ‘>’ ou htmlentities() qui filtre toutes les entités html.

Ces fonctions doivent être utilisées sur des entrées utilisateurs qui s’afficheront plus tard sur votre site. Si elles ne sont pas filtrées, les scripts comme ceux que nous avons vus plus haut s’exécuteront avec tout le mal qui s’en suit.

La faille sql

Comme son nom l’indique, l’injection SQL (ou SQLi) est une méthode d’exploitation de faille de sécurité d’une application possédant des interactions avec une base de données. Le principe est d’injecter, dans une requête, du code SQL malveillant qui viendra modifier l’effet escompté et ainsi compromettre l’intégrité des données présentes dans la base. Cette technique est très souvent utilisée pour contourner les mécanismes d'authentification et d'autorisation d'une application web.

Une faille SQLi peut avoir de lourdes conséquences car un hackeur peut avoir un accès non autorisé aux données sensibles. Il sera en mesure de lire la base de données, y enregistrer de nouvelles données ou exécuter du code malveillant. Quand on connaît la valeur et l’importance des données il serait dommageable pour un site de subir une telle cyberattaque.

Comment se protéger contre les failles SQL injection ?

Afin de se protéger contre les failles SQLi, il est préférable d’utiliser un système de requêtes préparées (PDO pour PHP par exemple). La requête est compilée avant d’être exécutée pour s’assurer qu’elle ne contient pas de caractères d’échappement.

La faille upload

Cette faille peut apparaître lors de l'upload de fichiers sur un site : photo de profil, document pdf, image dans un message, etc. Elle profite de l'action effectuée pour mettre en ligne des fichiers malveillants PHP qui vont permettre au « hacker » de prendre le contrôle total de notre site.

Pour éviter cette vulnérabilité, il est important de :

Empêcher les utilisateurs d'envoyer des fichiers lorsque cela n'est pas une fonction primordiale pour votre site ou application

Interdire l'exécution de code depuis le dossier dans lequel sont stockés les fichiers uploadés sur votre site

Vérifier et autoriser l'extension des fichiers que vous tolérez via une liste blanche.

La faille include

Il s'agit d'une faille très dangereuse. Comme son nom l'indique, elle exploite une mauvaise utilisation de la fonction include. La plupart du temps, cette fonction est utilisée pour exécuter du code PHP qui se situe dans une autre page, permettant de se connecter à une base de données. Il existe deux type de failles include :

A distance : il s'agit de la faille include par excellence. C'est à la fois la plus courant et la plus facilement exploitable.

En local : cela revient à inclure des fichiers qui se trouvent sur le serveur du site. Une personne mal intentionnée pourra donc s'emparer, assez facilement, de votre fichier contenant vos mots de passes.

Pour se protéger de cette faille, rien de mieux que de la tester ! Il vous suffit d'inclure une page qui n'existe pas. Si l'URL de celle-ci est vulnérable, un message d'erreur vous sera transmis venant de PHP.

La faille upload

Cette faille peut apparaître lors de l'upload de fichiers sur un site : photo de profil, document pdf, image dans un message, etc. Elle profite de l'action effectuée pour mettre en ligne des fichiers malveillants PHP qui vont permettre au « hacker » de prendre le contrôle total de notre site.

Pour éviter cette vulnérabilité, il est important de :

Empêcher les utilisateurs d’envoyer des fichiers lorsque cela n’est pas une fonction primordiale pour votre site ou application

Interdire l’exécution de code depuis le dossier dans lequel sont stockés les fichiers uploadés sur votre site

Vérifier et autoriser l’extension des fichiers que vous tolérez via une liste blanche