La sécurité

Tester la vulnérabilité de notre api.

Au minimum, le test de sécurité des applications Web nécessite l'utilisation d'un analyseur de vulnérabilité, tel que Netsparker ou Acunetix Web Vulnerability Scanner. Pour les tests authentifiés, un proxy HTTP tel que Burp Suite permet d'essayer de manipuler les connexions des utilisateurs, la gestion des sessions, les workflows des applications, etc.

D'autres outils sont disponibles si l'analyse du code source est une exigence, mais attention, on obtient ce pour quoi l'on paie avec les outils d'analyse du code source et, malheureusement, la plupart sont coûteux. Cette analyse n'est pas forcément des plus utiles, mais si elle constitue une exigence, il est difficile de s'y soustraire.

Les failles web les plus connus et comment s'en protéger.

1. Oubli de valider les entrées des utilisateurs.
Un classique, qui permet aux pirates de faire accepter des commandes au serveur à travers un formulaire web ou une simple URL, ou d'exécuter des contenus dynamiques (Javascript, par exemple) chez les autres utilisateurs d'un site.

2. Contrôle d'accès Inefficace.
Mauvaise mise en œuvre des outils de contrôle d'accès (fichier .htpasswd lisible par tous, mots de passes nuls par défaut, etc...).

3. Mauvaise gestion des sessions.
Cela permet aux pirates de "voler des sessions" d'autres utilisateurs (en devinant un numéro de session simple, en dérobant un cookie, ou en allant regarder les fichiers de sessions de PHP).

4. Cross Site Scripting.
Un autre grand classique, lui aussi lié à un manque de contrôle des entrées de l'utilisateur. Cette faille touche les sites web qui laissent les

internautes publier du code HTML susceptible d'être vu par les autres utilisateurs du site (dans un forum, par exemple). Cela permet d'exécuter des contenus dynamiques sur les navigateurs des internautes, avec les droits associés au site web.

5. Dépassement de mémoire tampon.
Une faille vieille comme le monde, qui frappe certains langages de programmation plus que d'autres (le C, par exemple). Si des composants CGI sont (mal) écrits dans ces langages, il peut-être simple de compromettre totalement le serveur par une telle attaque.

6. Injection de commandes.
Là encore, la source de la faille est un manque de contrôle des entrées de l'utilisateur. Elle permet au pirate de faire exécuter des commandes au serveur (au système d'exploitation ou à un serveur SQL, par exemple) en les attachant à une entrée web légitime avant que celle-ci ne soit transmise au serveur.

7. Mauvaise gestion des erreurs.
Les messages d'erreur utiles aux développeurs le sont souvent aussi pour les pirates ! Il faut donc penser à les supprimer une fois le développement terminé.

8. Mauvaise utilisation du chiffrement.
La mise en oeuvre du chiffrement au sein des applications web se révèle ardue. Des développeurs non spécialisés peuvent commettre des erreurs difficiles à déceler et créer ainsi une protection illusoire.

9. Failles dans l'administration distante.
C'est la voie royale : si les pages réservées aux administrateurs du site ne sont pas réellement protégées (authentification forte du client, chiffrement, contrôles réguliers...), un pirate peut prendre le contrôle du site sans avoir à pirater le serveur. Une aubaine, en quelque sorte.

10. Mauvaise configuration du serveur web et des applications.
Un classique : le serveur web qui permet de lister n'importe quel répertoire, ou les outils de développement qui laissent des versions temporaires des fichiers, lisibles par tout le monde. Avant de mettre un serveur en ligne, il est bon de faire le ménage et de bien comprendre toutes les options de ses fichiers de configuration...

Première bonne pratique : installer un pare-feu réseau et utiliser des DMZ

Mettre en place un pare-feu réseau est indispensable pour cloisonner les couches au sein de zones démilitarisées différentes (DMZ). Il est important de contrôler l'ouverture des flux sur ces pare-feu. Par exemple, il est absolument déconseillé d'ouvrir des flux depuis Internet vers la couche « données », cela irait à l'encontre de la sécurité du modèle 3 tiers car les couches « présentation » et « application » sont contournées.

Deuxième bonne pratique : chiffrer les flux

Il est aussi important de vérifier que les flux intercouches soient chiffrés à l'aide de protocoles tels que le HTTPS. Cela permet d'éviter des attaques de l'homme du milieu au sein de ce modèle. Il est à noter qu'il est très important de sensibiliser les utilisateurs à vérifier le certificat du site web. Cela permet d'éviter qu'un utilisateur se retrouve sur un site pirate.

Troisième bonne pratique : limiter les protocoles à risque

Certains protocoles fortement utilisés au sein des environnements Microsoft sont vecteurs de propagations virales. Par exemple, le protocole Netbios est à éviter au sein d'une infrastructure. Si toutefois, il faut ouvrir ce type de flux, il convient de parfaitement cloisonner les différents éléments et de mettre en place un plan d'actions en cas d'attaques virales (par exemple, la possibilité de pouvoir arrêter ce type de flux rapidement en cas de danger à l'aide d'un poste d'administration lui aussi cloisonné). La mise en œuvre d'IPS est aussi un moyen intéressant pour contrer les attaques qui pourraient véhiculer par ce type de flux.

Comment sécuriser l'application web au niveau de l'architecture?

Le pare-feu traditionnel n'étudie pas le contenu des flux qui pénètre dans vos architectures, il ne fait que les autoriser ou les bloquer en fonction de leur provenance. Si la provenance est légitime mais que le contenu du flux est corrompu, alors le risque est réel et non décelable par ce dernier.

Quatrième bonne pratique : installer un pare-feu applicatif (WAF ou *Web Application Firewall* en anglais)

Cet équipement est déployé en amont de la couche présentation afin de filtrer les attaques applicatives potentielles, telles que des Injection SQL ou les failles XSS.

Le pare-feu applicatif permet de se protéger contre les failles de l'OWASP (*Open Web Application Security Project* en anglais), qui est un organisme qui recense les dix risques de sécurité applicatifs les plus critiques sur Internet.

Comment fonctionne un pare-feu applicatif (WAF)?

Le pare-feu applicatif se place devant la couche présentation. Tous les flux applicatifs de type HTTP et HTTPS passent donc par lui avant d'accéder à la couche présentation de l'application.

Le filtrage se réalise de deux manières : soit par liste blanche, soit par liste noire.

– La liste blanche : la plus sûre mais la plus difficile à configurer

Son but consiste à autoriser uniquement le trafic sûr et à bloquer tout le reste (tout comme un pare-feu réseau). C'est une tâche difficile pour des applications qui génèrent beaucoup de données aléatoires. De plus, il est quasi impossible de connaître tout le bon trafic, ce qui peut créer des faux positifs. Une liste blanche mal générée risque d'altérer le fonctionnement de l'application.

– La liste noire : la plus simple mais la plus sensible

Lorsque la liste blanche est trop complexe à déterminer, il faut alors opter pour la liste noire.

Son fonctionnement est inverse et consiste à bloquer tout le mauvais trafic identifié et à autoriser le reste. Les pare-feu applicatifs disposent aujourd'hui de fonctionnalités complémentaires, comme la réputation web afin de compléter ce mécanisme et d'être ainsi plus efficace.

Comment renforcer la sécurité d'accès à une application web?

Pour accéder à la couche « présentation», il est aussi fortement recommandé de mettre en place un mécanisme d'authentification forte qui consiste à combiner deux moyens d'authentification (par exemple un mot de passe et une empreinte, ou un mot de passe et un jeton de sécurité).

Cinquième bonne pratique : privilégier une authentification forte

Une authentification faible de type nom d'utilisateur et mot de passe pourrait permettre à un attaquant de réaliser une attaque par force brute et ainsi de pouvoir accéder à l'application directement depuis Internet. Néanmoins, certains logiciels libres comme *fail2ban* peuvent empêcher les attaques par force brute. Ce système surveille les journaux des applications et bannit les adresses IP qui ont un trop grand nombre d'échecs d'authentification.

A défaut, utiliser des mots de passe différents et complexes pour chaque application, de plus de huit caractères et mélangeant chiffres, lettres et caractères spéciaux. Votre date de naissance est à proscrire, tout comme le nom de votre animal de compagnie ou de vos enfants, trop facilement identifiables sur Internet et les réseaux sociaux !

Ou rester informer sur les nouvelles failles.

CVE (Common Vulnerabilities and Exposures) est la référence mondiale en matière de failles et vulnérabilités. Malheureusement, elle n'est pas nécessairement la plus agréable en ce qui concerne la navigation et la recherche.

fr est l'une des sources francophones les plus utiles, avec des bulletins technologiques en français, une base assez complète et plusieurs flux RSS pour suivre les publications plus facilement.

CERT-FR, le centre gouvernemental de veille, d'alerte et de réponse aux attaques informatiques, qui dépend de l'ANSSI, est la base de données officielle française autour des vulnérabilités. C'est aussi une source indispensable de bonnes pratiques.

Sinon Google BRO .