AIDE-MÉMOIRE

PROTÉGER SON SITE INTERNET

Objectif du document

Protéger adéquatement le site Internet de l'entreprise.

Puisque les sites Web sont l'un des éléments les plus exposés du système d'information de l'entreprise, leur sécurisation revêt une très grande importance.

Les menaces les plus connues pesant sur les sites Web sont les défigurations et les dénis de service :

- Une défiguration est une attaque qui permet à une personne malveillante de modifier le site pour remplacer son contenu légitime par un contenu qu'elle choisit dans le but de relayer un message politique, dénigrer le propriétaire du site ou tout simplement revendiquer son attaque;
- Un déni de service a quant à lui pour objet de rendre le site attaqué indisponible à ses utilisateurs légitimes.

Pour ces deux cas, l'impact sur le propriétaire du site est évidemment un déficit d'image et si le site en est un lucratif, un telle attaque peut représenter une perte monétaire.

L'externalisation de l'hébergement d'un site ne permet pas de transférer l'ensemble des risques d'intrusion au système d'information de l'hébergeur.

La protection contre ces menaces passe à la fois par des mesures préventives et par des mécanismes permettant de détecter les tentatives d'attaques.

Ci-dessous une liste non-exhaustive des bonnes pratiques à adopter afin de renforcer les protections d'un site Web contre ces attaques. Ces recommandations doivent bien entendu être adaptées en fonction du contexte d'emploi.

Architecture

L'architecture matérielle et logicielle du site Web et de son infrastructure d'hébergement doit respecter le principe de défense en profondeur

- Filtrage réseau;
- Filtrage par un pare-feu Web (*Web Application Firewall*).

Les composants applicatifs employés doivent être limités au strict nécessaire

- Supprimer les composants applicatifs inutiles des systèmes de gestion de contenu;
- Faire un inventaire des composants applicatifs employés.

L'administration du site doit se faire via des protocoles sécurisés

- Utiliser des protocoles sécurisés comme SSH. Il est déconseillé d'administrer le site via le protocole FTP, car celui-ci ne protège pas les mots de passe ni le contenu transmis;
- Vérifier que l'administration de votre site via une interface Web utilise le protocole HTTPS;
- Utiliser des mots de passe complexes pour s'authentifier à la plateforme d'administration;

- Optionnel : mettre un filtrage par adresse IP.

Le principe de moindre privilège doit être appliqué

- Associer le serveur HTTP à un compte utilisateur non privilégié;
- Seuls les fichiers nécessaires peuvent être servis aux clients HTTP;
- L'utilisateur associé au serveur Web ne doit pas avoir des droits en lecture (ni en écriture) sur les fichiers auxquels il n'a pas besoin d'accéder;
- Les droits sur la base de données doivent être gérés finement (utiliser des comptes utilisateurs SGBD différents selon les composants du site, accorder des droits uniquement sur les tables où cela s'impose).

Limiter les renseignements fournis sur le fonctionnement technique du site Web

- Supprimer les éléments visibles sur la page susceptibles d'indiquer les outils utilisés;
- Limiter les informations de débogage en cas d'erreur (ex. : requête SQL);
- Utiliser des pages d'erreur personnalisées;
- Rejeter les requêtes HTTP TRACE.

Filtrage réseau

- Établir une matrice des flux entrants et sortants (ports et destinations);
- Utiliser le protocole HTTPS pour les communications entre le client et le serveur;
- Utiliser le mécanisme HSTS pour indiquer aux navigateurs que les requêtes vers le site concerné doivent systématiquement être faites en HTTPS.

Applicatif

Principes généraux

- Tous les accès aux pages nécessitant un droit d'accès spécifique ne devraient pas être gérés du côté client;
- Aucune vérification ne doit être déléguée aux clients;
- Les données reçues en entrée de la part des clients doivent être préalablement traitées avant d'être interprétées par une instance quelconque (DBMS, navigateur…).

Protéger les sessions

- Les identifiants de session doivent être aléatoires et d'une entropie d'au moins 128 bits;
- Recourir au protocole HTTPS dès que l'on associe une session à des privilèges particuliers;
- Les attributs HTTPOnly et Secure (dans le cas d'un site en HTTPS) doivent être associés à l'identifiant de session;
- Établir une durée de validité des sessions selon les besoins d'affaires;
- Invalider la session après la déconnexion;
- Pour les opérations sensibles, forcer une réauthentification si la session est ancienne.

Protéger les fonctions de gestion d'authentification

- Empêcher l'énumération des noms d'usager: erreur générique en cas d'échec d'authentification;

- Utiliser des méthodes telles des liens de réactivation de mots de passe avec des jetons aléatoires à durée limitée;
- Utiliser des questions secrètes dont les réponses ne se trouveront pas facilement (ex. : réseaux sociaux).

Gestion des droits

- Détenir des comptes nominatifs;
- Formaliser la gestion des comptes (octroi, retrait, changement);
- Réviser les droits sur une base régulière;
- Ne pas inclure de permissions par défaut à tous les utilisateurs.

Stockage des mots de passe

- Les mots de passe doivent être stockés dans une forme transformée par une fonction cryptographique non réversible;
- La transformation des mots de passe doit faire intervenir une valeur (sel) aléatoire;
- Stocker les mots de passe dans un *Keystore* ou dans un fichier de propriétés accessible uniquement aux personnes autorisées.

Inclusion de contenus externes

- Limiter au strict nécessaire les inclusions de contenus tiers;
- Effectuer quelques contrôles élémentaires pour s'assurer de la fiabilité du contenu.

Pour plus de détails sur les mesures de protection, consultez l'**annexe 1.**

Mesures réactives

Un point de contact associé au site doit être facilement identifiable

- Courriel valide pour communiquer avec le responsable du site dans l'enregistrement SOA.

Surveillance

- Le site Web doit être régulièrement testé (balayage, test d'intrusion) pour déceler toute anomalie;
- Vérifier régulièrement l'intégrité du ou des répertoires stockant les fichiers associés au site Web sur le serveur concerné;
- L'apparition soudaine de nouveaux fichiers ou la modification de la configuration du serveur doivent déclencher une investigation.

Journalisation

- Définir une politique de journalisation;
- Journaliser le serveur Web, c'est-à-dire les informations suivantes : IP, heure et date des requêtes, URL demandée, code d'erreur, champs *Referer* des requêtes HTTP et champ *User-agent*;
- Journaliser les autres systèmes (pare-feu local, redémarrage de système, connexion aux interfaces d'administration).

En cas d'incident

- Rassembler et préserver toutes les informations disponibles pour permettre les investigations;
- Rechercher d'autres intrusions (pages d'hameçonnage, insertion de *malware*, modification de la configuration du site, installation de portes dérobées);
- S'assurer que le site remis en service ne comporte aucun élément malveillant. Si une sauvegarde est utilisée, il est impératif que celle-ci soit saine.

Complément d'information

Site de l'OWASP : www.owasp.org (« OWASP secure coding practices quick reference guide »)

Document de l'ANSSI : https://www.ssi.gouv.fr/uploads/IMG/pdf/NP_Securite_Web_NoteTech.pdf

Annexe 1 : Mesures de protection pour les principales attaques Web

Attaques	Mesures de protection
Injections SQL	Utiliser des « Requêtes préparées » • Détails d'implémentation supplémentaires en Java: http://docs.oracle.com/javase/7/docs/api/java/sql/PreparedStatement.html • En .NET, utiliser LINQ to SQL (*Language-Integrated Query*): http://msdn.microsoft.com/en-us/library/bb425822.aspx

XSS	S'assurer que toutes les données issues de sources externes qui sont incluses dans la page Web soient protégées, c'est-à-dire qu'elles aient subi un traitement empêchant leur interprétation dans le contexte où elles sont employées.
	
	En Java :
	• Encodage :
	  ○ **OWASP Java Encoder**
	  ○ **TagTags JavaServer Pages Standard Tag Library (JSTL)**
	  ○ **StringEscapeUtils - Apache Commons**
	  ○ **HtmlUtils.htmlEscape() – Spring**
	  ○ **HtmlEscapers.htmlEscapers().escape() – Google Guava**
	• Validation :
	  ○ **BeanValidation**
	En .net :
	• Validation :
	  ○ **RegularExpressionValidator**
	  ○ **RangeValidator**
	  ○ **CustomValidator**

Référence directe non sécurisée à un objet	Générer aléatoirement un identifiant non devinable de type GUID Génération d'un UUID (*Universally unique identifier*) en Java Possibilité d'implémenter un contrôle d'accès pour valider l'accès à la ressource
Composants vulnérables	Maintenir à jour et corriger les vulnérabilités des éléments entrant en jeu dans la mise en ligne d'un site Web (gestionnaire de contenu, extensions, langages utilisés, bibliothèques utilisées, logiciel fournisseur de service Web, logiciel d'administration du site, système d'exploitation)
CSRF	Ajouter un secret de type **jeton** • Mettre le jeton dans un POST ou un en-tête HTTP dans un champ caché (*hidden field*)