**Apache : sécurité et durcissement de la configuration**

Désactiver les modules non utilisés

**Configuration */etc/apache2/conf-available/security.conf* (Debian) ou */etc/httpd/httpd.conf* (RH/CentOs) :**

ServerSignature Off

ServerTokens Prod

TraceEnable Off

FileETag None

<IfModule headers\_module>

Header always unset X-Powered-By

header always unset Composed-By

Header always set X-XSS-Protection “1; mode=block”

Header always set X-Frame-Options SAMEORIGIN

Header always set Strict-Transport-Security “max-age=31536000; includeSubDomains” # Only if full HTTPS Web Site

</IfModule>

Pour chaque virtual host :

<Directory "chemin\_appli">

Options ‐Indexes -FollowSymLinks

AllowOverride None

</Directory>''

**Configuration TLS dans */etc/apache2/mods-available/ssl.conf* (Debian) ou */etc/httpd/httpd.conf* (RH/CentOs) :**

SSLCipherSuite HIGH:!aNULL

SSLProtocol -all +TLSv1.2 +TLSv1.3

**Explications de chaque directive ci-dessous**

**Désactiver les modules inutiles**

De manière à réduire la surface d'attaque, il est recommandé de désactiver tous les modules Apache qui ne sont pas nécessaires à son fonctionnement. Par exemple, s'il s'agit d'un hébergement CMS en PHP, l'activation des modules python ou proxy ne sont pas nécessaires. Les configurations minimales sont décrites sur le [Wiki de la fondation Apache](https://wiki.apache.org/httpd/Minimal_Config), l'identification des modules strictement nécessaires dépend des applications hébergées. La gestion des modules se réalise avec les commandes *a2enmod* et *a2dismod* (Debian et dérivées) ou avec les directives *LoadModule* dans httpd.conf (RedHat et dérivées).

**Supprimer la signature du serveur**

Ce contenu est retourné dans l'entête de réponse HTTP Server: et peut se tester avec la commande curl.

Exemple de configuration incorrecte :

# curl -I https://intranet.zzzz.fr

HTTP/1.1 200 OK

Server: Apache/2.4.6 (Red Hat Enterprise Linux)

[...]

La modification des paramètres *ServerTokens* et *ServerSignature* masquera ces données précieuses pour un attaquant qui collecte ainsi des informations sur les technologies et versions utilisées, et déduit si les systèmes sont maintenus à jour ou vulnérables.

ServerSignature Off

ServerTokens Prod

Exemple de configuration correcte :

# curl -I https://intranet.inra.fr

HTTP/1.1 200 OK

Server: Apache

[...]

**Supprimer les entêtes ajoutées par certains logiciels**

Les serveurs d'applications (Tomcat, Glassfish…) ou les CMS (Wordpress, SPIP…) aiment ajouter des entêtes de réponses indiquant technologies et versions. Cela alimente les bases statistiques sur Internet sur l'utilisation et la popularité de ces outils. Bien évidemment, un attaquant utilisera immédiatement cette information pour connaître la version exacte du logiciel, s'il est à jour et ses vulnérabilités.

Exemple de configuration incorrecte :

# curl -I https://www.spip.net

HTTP/1.1 200 OK

X-Powered-By: PHP/5.6.26

Composed-By: SPIP 3.2.0 @ www.spip.net + spip(3.2.0),aide(1.0.0),archiviste(0.2.2),compagnon(1.6.1),dump(1.8.2),images(2.0.2),forum(1.11.3),jqueryui(1.12.1),mediabox(1.1.4),mots(2.8.4),organiseur(1.2.3),petitions(1.6.1),plan(2.2.5),porte\_plume(1.18.1),revisions(1.9.2),safehtml(1.5.1),sites(1.10.3),squelettes\_par\_rubrique(1.2.1),stats(1.1.8),svp(1.3.8),urls(2.1.6),vertebres(1.3.2),coloration\_code(0.9.11),crayons(1.26.14),facteur(3.4.11),notifications(3.5.13),nospam(1.5.18),opensearch(0.2.3),memoization(1.8.3),autorite(0.10.19),spip\_bonux(3.4.6),zcore(2.6.4),scssphp(1.4.0),languepreferee(0.4.7),precode(1.3.1),galactic(1.0.1),galactic\_spip\_net(1.0.2),iterateurs(1.0.6),queue(0.6.8),jquery(3.2.1),minidoc(1.0.3),ordoc(1.1.2),breves(1.4.0),compresseur(1.12.2),medias(2.20.16),tw(1.5.4),checkautobr(0.1.2),fulltext(1.1.20)

[...]

Par exemple, SPIP (par défaut) est particulièrement verbeux, car il indique sa version ainsi que celles de toutes les extensions installées! Dans cet exemple, dès qu'une vulnérabilité sera publiée sur SPIP 3.2.0, un attaquant se fera une joie de rechercher sur Internet tous les sites répondant avec cette entête pour identifier ses cibles et mener ses attaques. De la même manière, la version de PHP est retournée dans l'entête *X-Powered-By*.

Une fois activé le module *mod\_headers* d'Apache, ajouter une directive pour supprimer ces entêtes :

<IfModule headers\_module>

Header always unset X-Powered-By

header always unset Composed-By

</IfModule>

Une configuration correcte ne remonte pas ces entêtes.

Note : Bien évidemment, les configurations des serveurs d'application, du module PHP, des CMS permettent aussi de désactiver ces entêtes (majoritairement activées par défaut malheureusement).

**Désactivation du Directory Browsing**

Lors de la navigation, si un répertoire ne contient pas de page d'index, le serveur va gentiment en générer un et l'afficher à l'utilisateur qui peut alors simplement consulter les fichiers présents dans le répertoire. Il pourrait y trouver un fichier de configuration, un backup de base de données ou quelque information sensible malencontreusement copié dans un répertoire Web. C'est pourquoi, la désactivation de l'indexation automatique est recommandée. Aussi, quelques scénarios d'attaques peuvent abuser le parcours par liens symboliques ; s'ils ne sont pas nécessaires, il est recommandé de les désactiver également.

<Directory "chemin\_appli">

Options ‐Indexes -FollowSymLinks

</Directory>

Note : cette configuration peut être ajoutée de multiples façons. Celle présentée ici est la plus simple à appréhender mais peut être adaptée au contexte d'hébergement.

**Désactivation de l'en-tête Entity Tag**

L'en-tête *Etag* fournit des éléments d'informations sur les inodes. Non critique dans la plupart des cas, il s'agit néanmoins d'une fuite d'informations inutile qui peut être désactivée. Cette désactivation est exigée dans certains référentiels de sécurité.

FileETag None

Note : la présence de l'en-tête *Etag* est cependant nécessaire avec le module *Webdav mod\_dav\_fs*.

**Désactivation de la surcharge de configuration**

Apache autorise de surcharger sa configuration dans des fichiers normalisés dans son arborescence Web nommé .htaccess. Cela pose principalement 2 problèmes. L'exploitant ou l'administrateur technique d'un site hébergé peu regardant sur la sécurité peut désactiver des mécanismes de protection ou inclure des défauts de configuration. Aussi, un attaquant ayant obtenu une capacité d'altérer, remplacer ou créer un fichier .htaccess saura lui aussi en tirer profit pour abaisser le niveau de sécurité.

<Directory "chemin\_appli">

AllowOverride None

</Directory>

Note : cette configuration peut être ajoutée de multiples façons. Celle présentée ici est la plus simple à appréhender mais peut être adapté au contexte d'hébergement.

**Protection basique contre XSS réfléchi**

L'entête de réponse X-XSS-Protection indique au navigateur d'activer ses filtres anti XSS incorporés dans les navigateurs (depuis Internet Explorer 8). Cette protection est activée par défaut dans tous les navigateurs mais les utilisateurs peuvent choisir de la désactiver. L'ajout de cette entête force simplement l'activation. Même si le gain en termes de sécurité est relativement limité, cela reste une bonne pratique peu coûteuse à mettre en œuvre.

<IfModule headers\_module>

Header always set X-XSS-Protection “1; mode=block”

</IfModule>

Note : dans CSP, la directive *reflected-xss* est équivalente.

**Protection basique contre le click-jacking**

L'entête de réponse *X-Frame-Options* indique au navigateur ce qu'il peut ou ne peut pas faire avec l'inclusion de contenu avec les balises *frame*, *iframe* ou *object*. Dans la majorité des cas, l'inclusion de contenu tiers de sources inconnues devrait être interdit. En effet, en exploitant par exemple une faille XSS, un attaquant pourrait altérer une page en superposant des cadres transparents sur les boutons (par exemple) et rediriger l'utilisateur ou lui faire réaliser des actions à son insu. C'est pourquoi il est recommandé de positionner cet entête pour indiquer au navigateur qu'il ne peut charger du contenu tiers que s'il provient de la même origine.

<IfModule headers\_module>

Header always set X-Frame-Options SAMEORIGIN

</IfModule>

Note : il convient de s'assurer que le site hébergé n'utilise pas de manière légitime l'inclusion de contenu tiers depuis des sources de confiance. Dans ce cas, il est possible d'indiquer ces sources :

<IfModule headers\_module>

Header always set X-Frame-Options ALLOW-FROM www.inra.fr, www.cnrs.fr

</IfModule>

**Désactivation de la méthode TRACE**

La méthode TRACE est utilisée uniquement à des fins de débogage. Il a été montré qu'il existe des scénarios d'attaques exploitant la méthode TRACE pour contourner certains mécanismes de protection afin de détourner des sessions. Il est donc recommandé de désactiver cette méthode dans la configuration Apache.

TraceEnable Off

**Activation d'HTTP Strict Transport Security**

HSTS est une spécification qui impose l'utilisation intégrale d'HTTPS pour charger une page. La majorité des navigateurs modernes protègent déjà les utilisateurs lors de l'accès à un site HTTPS qui contient certaines requêtes en clair et sont alors bloquées. Cependant, l'utilisateur peut désactiver cette protection. HSTS force le navigateur à la respecter. De même, HSTS redirige en HTTPS toutes les requêtes HTTP vers le domaine cible.

<IfModule headers\_module>

Header always set Strict-Transport-Security “max-age=31536000; includeSubDomains”

</IfModule>

**Configuration TLS**

Les protocoles SSLv2 et SSLv3 doivent bien sûr être désactivés car ont été cassés. TLSv1.0 et TLSv1.1 doivent l'être également. Désormais, TLS v1.3 est suffisamment bien implémenté, est doit être proposé. Les suites de chiffrement doivent aussi être adaptées. Plutôt que de les choisir tous manuellement, Apache propose de définir des suites de niveaux élevées avec la valeur HIGH dans SSLCipherSuite. Un bon moyen de vérifier sa configuration TLS, pour un site disponible sur Internet, est [SSLLabs](https://www.ssllabs.com/ssltest" \o "https://www.ssllabs.com/ssltest).

SSLCipherSuite HIGH:!aNULL

SSLProtocol -all +TLSv1.2 +TLSv1.3