Secrétariat général

Service des Politiques Support
et des Systèmes d'Information

Centre de Prestations et d'Ingénierie Informatiques
Département Opérationnel de l'Ouest

Mai 2018

# DESCRIPTION DU PLUGIN CISEC
# ANTI SCANNER DE VULNÉRABILITÉ POUR SPIP

C.Imberti – mis à jour le 24/05/2018

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE DE L'ENVIRONNEMENT,
DE L'ÉNERGIE ET DE LA MER
www.developpement-durable.gouv.fr

MINISTÈRE DU LOGEMENT,
ET DE L'HABITAT DURABLE
www.logement.gouv.fr

## Historique des versions du document

| Version | Date | Commentaire |
|---|---|---|
| 1 | 11/04/2016 | |
| 1.1 | 12/10/2016 | Ajout d'une constante dans le chapitre 4.3 |
| 1.2 | 24/05/2018 | Ajout de constantes dans le chapitre 4.3 et enrichissement du contenu du courriel |

## Auteur du document

**Christophe IMBERTI** - SG/SPSSI/CP2I/DO Ouest
Chargé de mission auprès du chef du DO Ouest

# SOMMAIRE

# 1. Les objectifs de ce plugin

Les scanners de vulnérabilité de site web sont des logiciels qui analysent les pages d'un site web (“crawling”), puis effectuent des requêtes HTTP en ajoutant des codes malicieux dans l'URL (ainsi que dans les variables POST, etc.) et analysent le contenu de la page obtenue pour détecter si le code malicieux a été filtré ou non.

L'utilisation de scanners est intéressante dans le cadre de l'audit de sécurité d'un site. En revanche, elle est dangereuse lorsque ce sont des pirates qui utilisent ces scanners. Par ailleurs, certains scanners sollicitent fortement le serveur, par exemple avec une cadence de 80 requêtes par seconde. Enfin, les variations d'URL, effectuées par les scanners de vulnérabilité, remplissent inutilement le cache de SPIP.  A noter que l'usage non autorisé d'un de ces logiciels sur un site est susceptible de relever de l'article 323 du code pénal.

Les objectifs de ce plugin sont les suivants :

- Il s'agit de ne pas donner les véritables pages au scanner de vulnérabilité. En effet, il faut l'empêcher de conduire ses tests de vulnérabilité.
- Il convient de réduire le plus possible le temps de traitement des pages demandées par le scanner de vulnérabilité (une fois détecté).
- Il s'agit d'éviter l'impact, sur le cache de SPIP, des variations d'URL envoyées par le scanner de vulnérabilité.

Le plugin tient compte des contraintes suivantes :

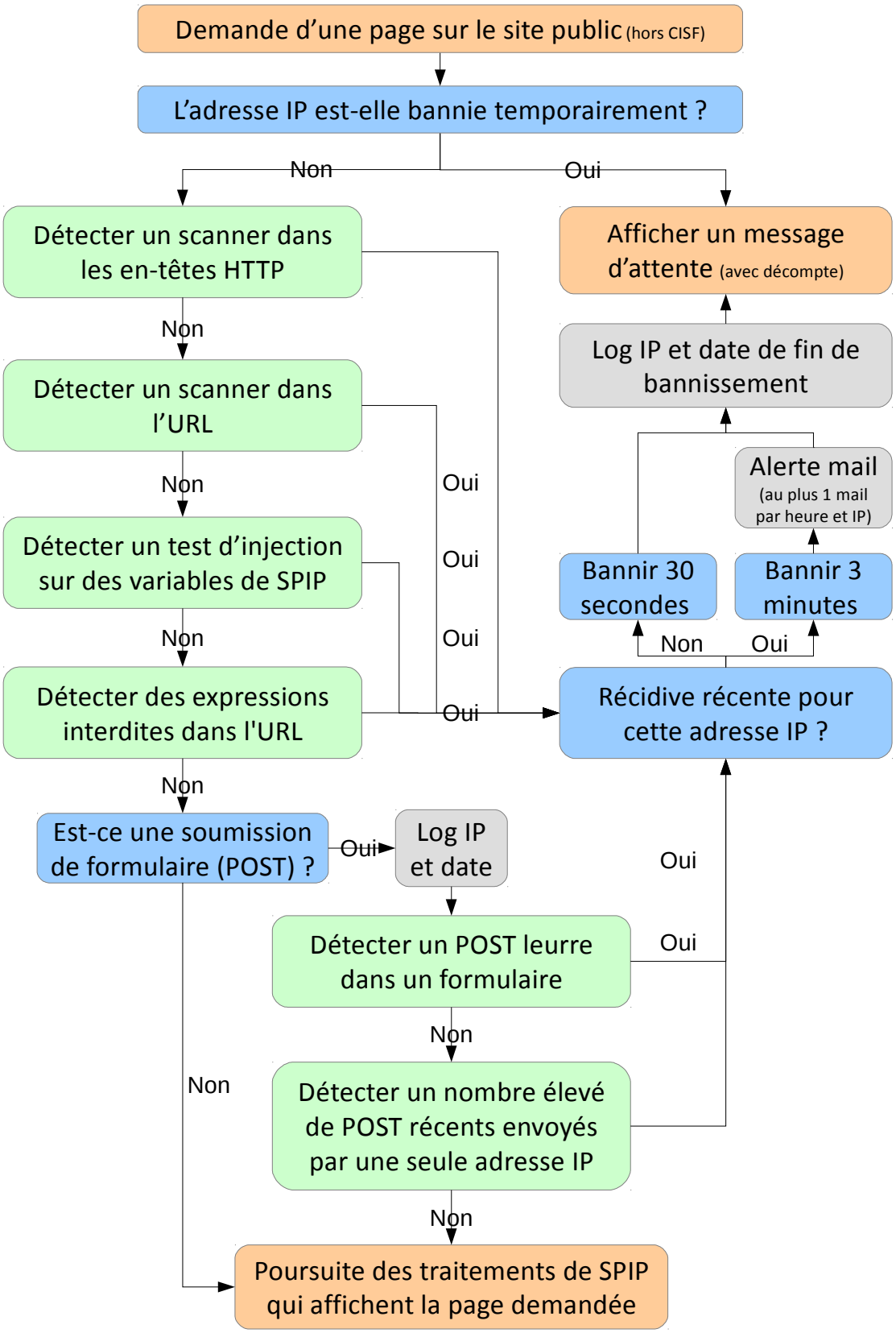
- En cas de bannissement, il est nécessaire d'offrir une réponse compréhensible pour tenir compte d'éventuels faux positifs.
- Le bannissement doit être temporaire, afin de limiter l'impact d'éventuels faux positifs.
- L'internaute doit pouvoir utiliser la recherche et sur n'importe quel mot (quelle que soit la langue).
- Les robots d'indexation des moteurs de recherche doivent pouvoir continuer à faire leur travail et ne doivent pas être affectés par des faux positifs.
- Le temps de traitement ajouté par le plugin doit être le plus faible possible.
- Le plugin doit être compatible avec les adresses IP v4 et IP v6.
- Il est impératif de tenir compte des traitements effectués en amont par l'écran de sécurité de SPIP.

Le périmètre porte sur le site public uniquement. Le périmètre ne porte pas sur les attaques de type spam (un plugin pour SPIP existe déjà sur ce sujet), ni sur les attaques de type déni de service ou déni de service distribué.

## 2. Les fonctionnalités de ce plugin

### Détecter et bannir un scanner de vulnérabilités

SG/SPSSI/CPI/DO Ouest/C. Imberti – 24/03/2016



## 2.1 Détecter un scanner dans les en-têtes HTTP

Le plugin détecte dans les en-têtes HTTP :

- la présence de variables propriétaires d'un scanner de vulnérabilité
- la présence d'un User-Agent d'un scanner.

Cette détection est intéressante, mais elle n'est pas suffisante. En effet, je ne dispose pas des signatures de tous les scanners existants et surtout certains scanners permettent de modifier leur signature.

## 2.2 Détecter un scanner dans l'URL

Le plugin détecte dans l'URL demandée la présence d'expressions spécifiques à certains scanners et qui ne risquent pas d'être utilisées par un utilisateur dans une recherche ou comme valeur d'une variable de SPIP ou d'un plugin.

Cette détection est intéressante, mais elle n'est pas suffisante. En effet, je ne dispose pas des mots spécifiques pour chaque scanner existant et surtout certains scanners sont "open source", donc ils sont modifiables.

Remarque : la liste de ces mots peut être enrichie via une constante à placer dans un fichier d'options (cf. annexe).

## 2.3 Détecter une tentative d'injection sur des variables de SPIP

Lorsque l'écran de sécurité de SPIP détecte certaines URL malicieuses, il effectue un nettoyage de l'URL pour certaines et affiche un message d'erreur "403" pour d'autres. Dans les deux cas, il ne bannit pas l'adresse IP. Aussi un scanner peut essayer d'autres URL malicieuses qui ne seront pas forcément détectées par SPIP.

Le plugin teste la valeur de certaines variables, propres à SPIP, afin de détecter une tentative d'injection. Par exemple, il vérifie que :

- forcer_lang doit être égal à "true" ou "false".
- lang doit être de type alpha avec possibilité d'underscore (en tenant compte que l'écran de sécurité de SPIP nettoie le contenu de la variable "lang" en amont en mettant un espace à la place d'un caractère interdit.).
- "debut_..." doit correspondre à un nombre.

Ces injections peuvent provenir d'un scanner, d'une tentative d'injection manuelle, etc.

## 2.4 Détecter des expressions interdites dans l'URL

Le plugin détecte certaines expressions interdites dans une URL, et qui ne risquent pas d'être utilisées comme expression recherchée par un utilisateur ou comme valeur d'une variable de SPIP ou d'un plugin, par exemple : ../../../../

## 2.5 Détecter un POST leurre dans un formulaire

Le plugin teste l'existence du contenu d'un POST correspondant à un champ leurre, ajouté dans un formulaire par le plugin chargé de la lutte anti spam.

Le plugin, chargé de la lutte anti spam, effectue ce test lors de la phase de vérification des données envoyées par le formulaire. C'est-à-dire qu'un scanner de vulnérabilité qui envoie 10 000 requêtes par heure sur un formulaire (c'est un cas réel) sollicitera autant de fois cette phase de vérification des données envoyées par le formulaire.
Aussi, il est intéressant de bénéficier de la période de bannissement pendant laquelle il ne sera plus nécessaire d'effectuer cette phase de vérification, puisque l'adresse IP bannie sera rejetée en amont.
Par ailleurs, le bannissement temporaire bloquera toutes les actions de l'adresse IP bannie, même si elles portent sur une autre page du site web.

## 2.6 Détecter un nombre élevé de POST récents envoyés par une seule adresse IP

Le plugin détecte si au moins l'un de ces deux seuils est dépassé :

- Au moins N POST d'une même adresse IP pendant une seconde.
- Au moins M POST d'une même adresse IP pendant 10 secondes.

L'adresse IP et la date du POST (timestamp) sont stockés dans un fichier de logs dédié au flood sur formulaire, à chaque POST envoyé (sur le site public).

Si une adresse IP est déjà bannie et que la date, à laquelle son bannissement expire, n'est pas échue, tout nouveau POST ne sera pas enregistré, afin d'éviter de saturer le fichier de log dédié au flood sur formulaire.

Pour favoriser les performances, seule la fin du fichier de logs sera examinée pour détecter un dépassement de seuil d'une adresse IP (environ les N derniers POST). Si c'est le cas, cette adresse IP sera temporairement bannie.

Remarque : La présente détection est indépendante de celle du plugin anti spam. En effet, un robot de spam utilise une cadence très faible et c'est le rôle du plugin anti spam de détecter les robots de spam.

## 2.7 Bannir un scanner temporairement

La durée d'un bannissement temporaire sera de 30 secondes, ou plus en cas de récidive.

Si une adresse IP doit être bannie et qu'elle a déjà été bannie, avec une date d'expiration qui est échue depuis moins de 3 minutes, on considère que c'est une récidive et la durée de bannissement passe à 3 minutes pour cette adresse IP.

Si une adresse IP est bannie, à chaque nouvelle requête HTTP provenant de cette adresse IP, un message d'attente est affiché et mentionne la durée d'attente (un script javascript effectue un décompte pour faire patienter). Par exemple :

```
Votre demande est suspendue,
votre adresse IP a été enregistrée
Veuillez attendre : N secondes
```

Le message doit avoir un en-tête HTTP spécifique pour ne pas perturber les robots d'indexation en cas de faux positif. Par exemple :

```
HTTP/1.1 503 Service Unavailable
Status: 503 Service Unavailable
Retry-After: 30
Expires: ...
Cache-Control: no-cache, must-revalidate
Pragma: no-cache
```

Un bannissement est enregistré dans un fichier de log dédié aux adresses bannies, avec la forme suivante : Une virgule, une adresse IP, un underscore, une date d'expiration (timestamp de fin de bannissement), une virgule. Par exemple :
,10.xx.xx.xx_1458230602,

Une version détaillée est enregistrée dans un autre fichier de log, pour pouvoir rechercher les causes plus tard. Elle comprend la date, l'adresse IP et l'URL demandée. Exemple :
2016-04-11 14:57:54 XX.XX.XX.XX (pid 4623) REQUEST_URI :
/spip.php?page=recherche&lang=fr&forcer_lang=.....&recherche=zzz

Si une adresse IP est déjà bannie et que la date à laquelle son bannissement expire n'est pas échue, toute nouvelle tentative ne sera pas enregistrée, afin d'éviter de saturer le fichier de log dédié aux adresses bannies.

Pour savoir si une adresse IP est bannie, il convient de rechercher le dernier bannissement de cette adresse dans le fichier de log et de lire sa date d'expiration.

## 2.8 Prévenir par courriel (sans saturer)

En cas de récidive, un courriel est automatiquement envoyé à l'adresse du webmestre, qui figure dans le menu [Configuration] de SPIP.
Si la constante '_CISEC_EMAIL' est définie dans un fichier d'options, avec comme valeur une adresse email avec une syntaxe valide, le courriel sera envoyé à cette adresse (et non pas à celle du webmestre).

Exemple de contenu du courriel :

```
Sujet : Nom du site : Bannissement temporaire de l'adresse IP XXX.XX.XX.XX
Date : Fri, 8 Apr 2016 12:13:53 +0200
De : robot giseh - ne pas repondre <robot-giseh....@....fr>
Pour : webmestre....@....fr

Ceci est un message automatique du site : Nom du site : .... (adresse du site)
Bannissement temporaire de l'adresse IP XXX.XX.XX.XX
pour la raison suivante : ...
```

L'adresse IP et la date de cet envoi sont enregistrées dans un fichier de log dédié à ces alertes par courriels, avec la forme suivante : Une virgule, une adresse IP, un underscore, la date d'envoi du courriel, une virgule. Par exemple :
,10.xx.xx.xx_1458230602,

Pour éviter de saturer la boîte aux lettres du webmestre, au plus un courriel est envoyé par tranche d'une heure pour une même adresse IP.

## 2.9 Contenu des fichiers de log

| Fichiers de log | Contenu |
|---|---|
| cisec_bannir.log | Pour chaque bannissement : l'adresse IP et la date d'expiration du bannissement (timestamp). |
| cisec_detail.log | Pour chaque bannissement : la date, l'adresse IP, la cause et l'URL demandée. |
| cisec_mail.log | Pour chaque mail envoyé par le plugin : l'adresse IP et la date d'envoi du mail à l'adresse du webmestre. |
| cisec_post.log | Pour chaque POST envoyé sur le site public : l'adresse IP et la date du POST (timestamp). |

Si la taille d'un fichier de log est supérieure à la taille maximale des logs de SPIP (par défaut 100 Ko), une rotation est effectuée. La rotation s'effectue sur le même nombre de fichiers que pour les logs de SPIP (par défaut 4 fichiers).

# 3. Mesure des gains obtenus

### 3.1.1 Gains obtenus avec le plugin CISEC

| **Mesures** (1) | **Scanner A** | **Scanner B** | **Scanner C** |
|---|---|---|---|
| **Sans le plugin CISEC** | | | |
| Durée du scan | 1 h 14 min | **1 h 47 min** (3) | 0 h 42 min |
| Nombre de requêtes HTTP envoyées (4) | 91 285 | **498 097** | 60 008 |
| Nombre de requêtes par seconde | 21 requêtes/sec | **78** requêtes/sec | 24 requêtes/sec |
| Load average max sur 1 minute | 9,3 | **21,1** | 3,0 |
| Load average max sur 5 minutes | 8,9 | **18,7** | 1,7 |
| Taille du cache de SPIP (5) | 205 Mo | **377** Mo (6) | 170 Mo |
| **Avec le plugin CISEC** (2) | | | |
| Durée du scan | 3 secondes | 49 secondes | 5 secondes |
| Nombre de requêtes HTTP envoyées (4) | 55 | 7 418 | 133 |
| Load average max sur 1 minute | 0,6 | 1,3 | 0,6 |
| Taille du cache de SPIP (5) | 1 Mo | 1 Mo | 1 Mo |

(1) Scan avec comme URL de base la page de résultats de recherche.

(2) Le plugin CISEC tente de détecter et, le cas échéant, bannir temporairement un scanner.

(3) Arrêt manuel du scan à l'approche des 500 000 requêtes (sinon le scan dure plus de 15h avec 2 millions de requêtes).

(4) Certaines requêtes ne concernent pas SPIP, mais le serveur.

(5) Le cache de SPIP était vide au départ.

(6) 377 Mo après 1 heure de scan (59 894 fichiers). Si le scan dure plus d'une heure, SPIP 3.0 va progressivement purger le cache. Aussi, le cache a été mesuré au bout d'une heure.

L'examen des logs montre que le scanner demande la page de départ et reçoit une réponse "503" (celle du plugin CISEC). Comme il n'obtient pas le contenu de la page de départ, il ne peut pas effectuer de "crawling" sur les liens contenus dans la page de départ. Il effectue des variations d'URL de la page de départ puis il s'arrête.

***Le plugin CISEC s'avère très efficace***
***pour contrer ces 3 scanners de vulnérabilités.***

### 3.1.2 Gains obtenus avec le plugin CISEC sans la détection de signatures

Cette mesure vise à simuler le cas d'un scanner dont on ne dispose pas des signatures.

Pour mesurer les gains apportés sans la détection de signatures, le plugin CISEC a été temporairement modifié, afin de désactiver la détection d'un scanner dans les en-têtes HTTP et la détection d'un scanner dans l'URL.

| **Mesures** | **Scanner B** | **Scanner B face au plugin CISEC** (sans la détection de signatures dans les en-têtes HTTP et dans l'URL) |
|---|---|---|
| Pourcentage de code "200" (OK) | **56 %** | **10 %** |
| Pourcentage de code "403" (accès interdit)(1) | 7 % | 7 % |
| Pourcentage de code "404" (page non trouvée)(5) | 35 % | 12 % (5) |
| Pourcentage de code "503" (message renvoyé par le plugin cisec) | 0 % | 70 % |
| Durée du scan | 1 h 47 (2) | 33 min (2) (3) |
| Nombre de requêtes HTTP envoyées | 498 097 (2) | 498 097 (2) |
| Nombre de requêtes par seconde | 78 req/s | 252 req/s (3) |
| Load average max sur 1 minute | 21,1 | 20,2 |
| Load average max sur 5 minutes | 18,7 | 17,4 |
| Taille du cache de SPIP | 377 Mo (4) | 22 Mo (4) |
| Nombre de fichiers dans le cache de SPIP | **59 894** (4) | **2 846** (4) |

(1) SPIP dispose d'un écran de sécurité, qui effectue un contrôle, non exhaustif, d'URL malicieuses. Lorsque SPIP détecte une URL malicieuse, il effectue un nettoyage de l'URL pour certaines et affiche un message d'erreur "403" pour d'autres (dans les deux cas, il ne bannit pas l'adresse IP, aussi un scanner peut essayer d'autres URL malicieuses qui ne seront pas forcément détectées par SPIP).

(2) Arrêt manuel du scan à l'approche des 500 000 requêtes (sinon le scan dure plus de 15h avec 2 millions de requêtes). Cela permet une comparaison sur le même nombre de requêtes.

(3) Le scanner B adapte sa cadence de requêtes au temps de réponse du serveur. Comme l'envoi du message de bannissement est très rapide, la cadence des requêtes augmente, mais pas le load average.

(4) Cette mesure a été effectuée après 1 heure de scan, pour éviter toute purge automatique de SPIP.

(5) Le bannissement intervient en amont de l'affichage éventuel de la page "404", aussi il est normal que le pourcentage de code "404" soit plus faible avec le plugin CISEC.

L'examen des logs montre que le message de bannissement (code "503") est envoyé par le plugin CISEC pour 70 % des requêtes. En revanche, le scanner ne s'arrête pas. En effet, le scanner demande la page de départ et la reçoit. Il peut ainsi effectuer un "crawling" sur les liens contenus dans la page de départ, etc.

Seules 10 % des requêtes du scanner obtiennent la page demandée, contre 56 % sans le plugin CISEC.

***L'objectif de ne pas donner les véritables pages au scanner de vulnérabilité est atteint à 90 %*** *lorsque la détection de signatures est inactive.*

Un test de charge a été effectué afin de mesurer les performances du site, en termes de nombre de pages par seconde, pendant que le scanner est banni.

en **pages par seconde**

| Site avec les données « catalogue » | Le scanner n'est pas banni (1) | Le scanner **est banni** (1) | Ecart |
|---|---|---|---|
| **Page dans le cache de SPIP** | | | |
| Page d'accueil (2) | 91,6 | 732 | De 1 à 8,0 |
| Article 8 (2) | 96,3 | 732 | De 1 à 7,6 |
| Rubrique 57 (2) | 97,6 | 732 | De 1 à 7,5 |
| *Moyenne* | *95,2* | *732* | *De 1 à 7,7* |
| **Calcul d'une page** | | | |
| Page d'accueil (2) | 30,7 | 732 | De 1 à 23,8 |
| Article 8 (2) | 36,9 | 732 | De 1 à 19,8 |
| Rubrique 57 (2) | 40,6 | 732 | De 1 à 18,0 |
| *Moyenne* | *36,1* | *732* | De 1 à 20,3 |

(1) SPIP 3.0.22 avec le plugin CISEC.

(2) Page uniquement (**SANS** la feuille de style, sans les fichiers javascripts, sans les images, etc.). En effet, contrairement à un navigateur, le scanner charge la page uniquement.

Lorsque le scanner est banni, ce qui est le cas pour 70 % des requêtes, l'envoi du message de bannissement à la place de la page demandée, permet de multiplier les performances du site par un facteur compris entre 7 et 24.

***L'objectif de réduire le temps de traitement des pages demandées par le scanner de vulnérabilité*** *(une fois détecté)* ***est largement atteint*** *lorsque la détection de signatures est inactive.*

La **durée du scan est plus courte**, toutefois le load average reste élevé. En effet, le scanner B adapte sa cadence de requêtes au temps de réponse du serveur.

En revanche le nombre de requêtes reste très élevé.

Le **nombre de fichiers dans le cache** de SPIP **est divisé par 21** (2 846 contre 59 894). En effet, lorsque le plugin CISEC envoie le message de bannissement, il arrête le traitement de SPIP et aucune page n'est mise en cache.

### 3.1.3 Gains apportés uniquement par la détection d'un nombre élevé de POST

Cette mesure vise à simuler le cas d'un scanner qui examine un formulaire et qui n'aurait pas été détecté en amont par l'une des actions suivantes :

- Détecter un scanner dans les en-têtes HTTP
- Détecter un scanner dans l'URL
- Détecter une tentative d'injection sur des variables de SPIP
- Détecter des expressions interdites dans l'URL
- Détecter un POST leurre dans un formulaire

Pour mesurer les gains apportés par la détection d'un nombre élevé de POST envoyés par une seule adresse IP :

- Un formulaire a été modifié pour enregistrer une trace dans les logs, chaque fois qu'il effectue la phase de vérification.
- Le plugin CISEC a été temporairement modifié, afin de désactiver toutes les détections effectuées en amont de la détection d'un nombre élevé de POST.

Après un scan, effectué avec le scanner C sur ce formulaire, l'examen des logs montre que **le formulaire a été utilisé seulement 6 fois au lieu de 2 885 fois**.

***La détection d'un nombre élevé de POST envoyés par une seule adresse IP est très efficace*** *(dans le cas d'un scanner non détecté par les autres mécanismes situés en amont dans le plugin CISEC).*

Remarques :

- Avec le plugin CISEC, le formulaire a été utilisé 3 fois à 15h19:07 puis 3 fois à 15h19:38, c'est-à-dire après les 30 secondes du premier bannissement. Soit un total de 6 fois.
- Sans le plugin CISEC, le formulaire a été utilisé 2 885 fois, ce qui correspond à une cadence de 15,8 requêtes par seconde.

# 4. Annexe

## 4.1 Compatibilité

Le plugin a été testé avec SPIP 2.1, SPIP 3.0, SPIP 3.1 et SPIP 3.2.
Il est compatible avec PHP 5 (y compris PHP 5.6), PHP 7.0 et PHP 7.1.

## 4.2 Installation

Le plugin s'installe comme tous les plugins, cf. http://www.spip.net/fr_article3396.html .

## 4.3 Constante facultative

Ces constantes sont **facultatives**. Si on souhaite les utiliser, il convient de les placer dans un fichier d'options (le fichier config/mes_options.php ou bien le fichier d'options d'un autre plugin).

Exemple (en commentaires) :

```
// Pour ajouter des mots interdits dans les URLS
//
// Les mots devront être séparés par un "pipe"
// (comme la liste des BOTS dans l'écran de sécurité de SPIP).
// Exemple :
// define('_CISEC_MOTS','codedangereux|evilcode');


// Si la constante _CISEC_EMAIL est définie,
// avec comme valeur une adresse email avec une syntaxe valide
// (avec lettres, chiffres, arobase, points, tirets, underscores),
// le courriel sera envoyé à cette adresse (et non pas à l'adresse du webmestre
// qui figure dans le menu [Configuration] de SPIP).
// Exemple :
// define('_CISEC_EMAIL','administrateur@test.fr');


// Possibilité d'augmenter les seuils (mais pas de les diminuer)
// Au moins N POST d'une même adresse IP pendant une seconde.
// Au moins M POST d'une même adresse IP pendant 10 secondes.
// Exemple :
// define('_CISEC_MAX_POST_EN_1_S',5);
// define('_CISEC_MAX_POST_EN_10_S',15);


// Pour ceux qui utilisent un reverse proxy
// et qui ne definissent pas la valeur de $ip dans mes_options.php
// (et qui n'utilisent pas mod_rpaf ou mod_remoteip)
// il est possible de definir l'ordre de recherche personnalisee de l'adresse IP
// valeur par defaut : 'HTTP_X_FORWARDED_FOR,REMOTE_ADDR'
// Exemple :
// define('_CI_ORDRE_RECHERCHE_IP','HTTP_X_FORWARDED_FOR,REMOTE_ADDR');
```