

# Metasploit Framework ile Exploit Geliştirme

Fatih Özavcı

Bilgi Güveniği Danışmanı

fatih.ozavci at gamasec.net

gamasec.net/fozavci

Canberk Bolat

Güvenlik Araştırmacısı

canberk.bolat at gmail dotcom

cbolat.blogspot.com

# Kapsam

- Exploit Geliştirme Süreci
- Metasploit Framework Geliştirme Araçları
- Exploit, Auxiliary ve Post için Hazır Fonksiyonlar
  - Protokol ve Servis Yardımcıları
  - Fuzzing Araçları
  - Encoder, NOP, Kripto Araçları
- Örnek Exploit ve Modüller
- Bilinmesi Gerekenler
  - Ruby Dili ve Yapısı
  - Metasploit Framework Kullanımı





# Exploit ve Temel Kavramlar

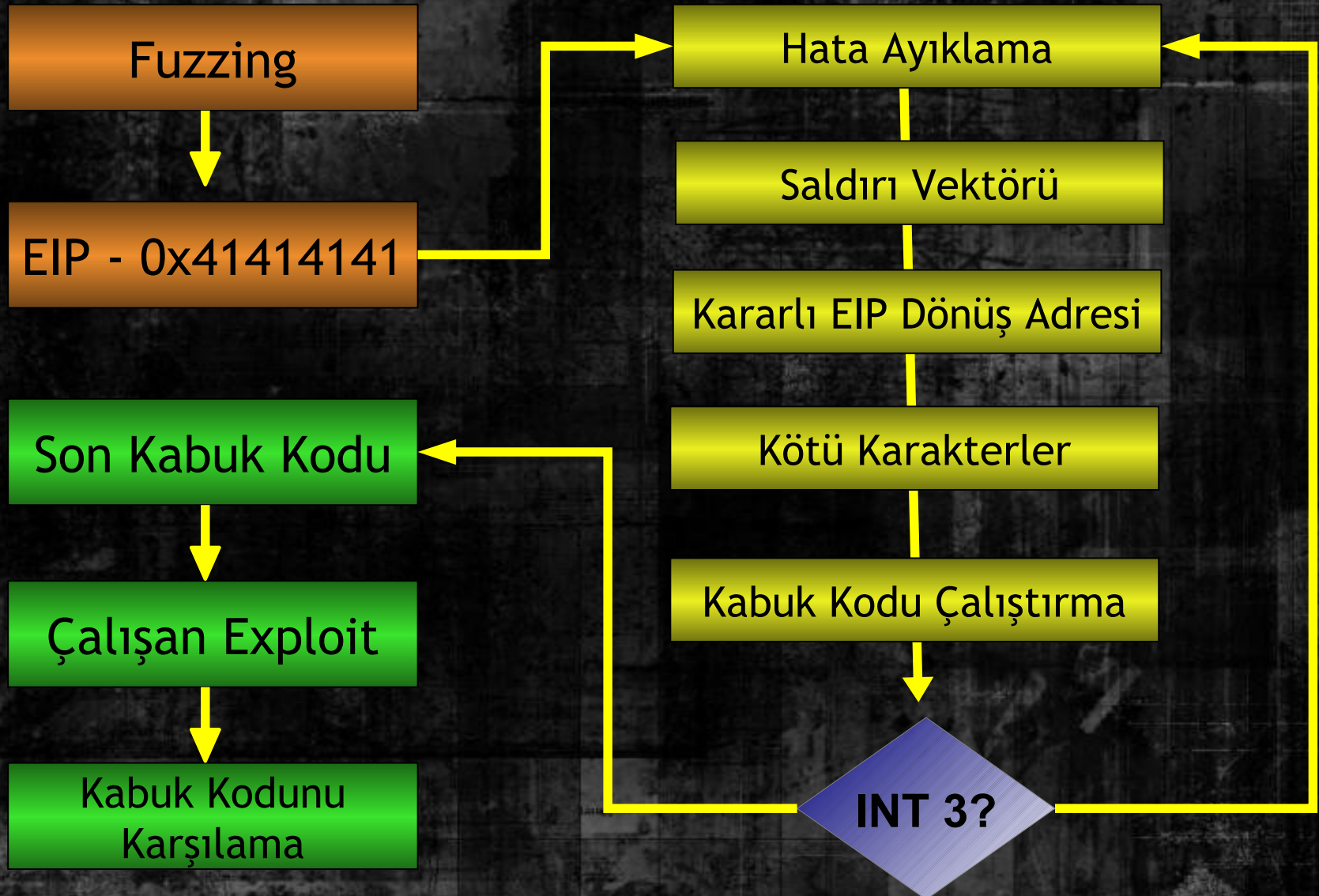
- Bir güvenlik açığına kullanarak normal-dışı bir işlem yapılmasını sağlayan yöntem veya yazılım

<http://sunucu/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir>

<http://sunucu/login.asp?uid=' OR 1='1>

- Payload / Shellcode : Exploit sonrası çalıştırılacak ve normal-dışı işlemi yapacak içerik, kabuk kodu
- NOP / NOPSlide : “Not Operation” bitleri, talimat içermeyen ve bellekte istenen yere ulaşıncaya kadar belleği dolduran bitler
- Encoder : Çalıştırılacak Shellcode’u değiştiren, Saldırı Önleme Sistemi veya Anti-Virüs tarafından yakalanmasını önleyen kodlar

# Güvenlik Açığından Exploit'e Gidiş



# Örnek Exploit Yapısı

## EIP Exploit

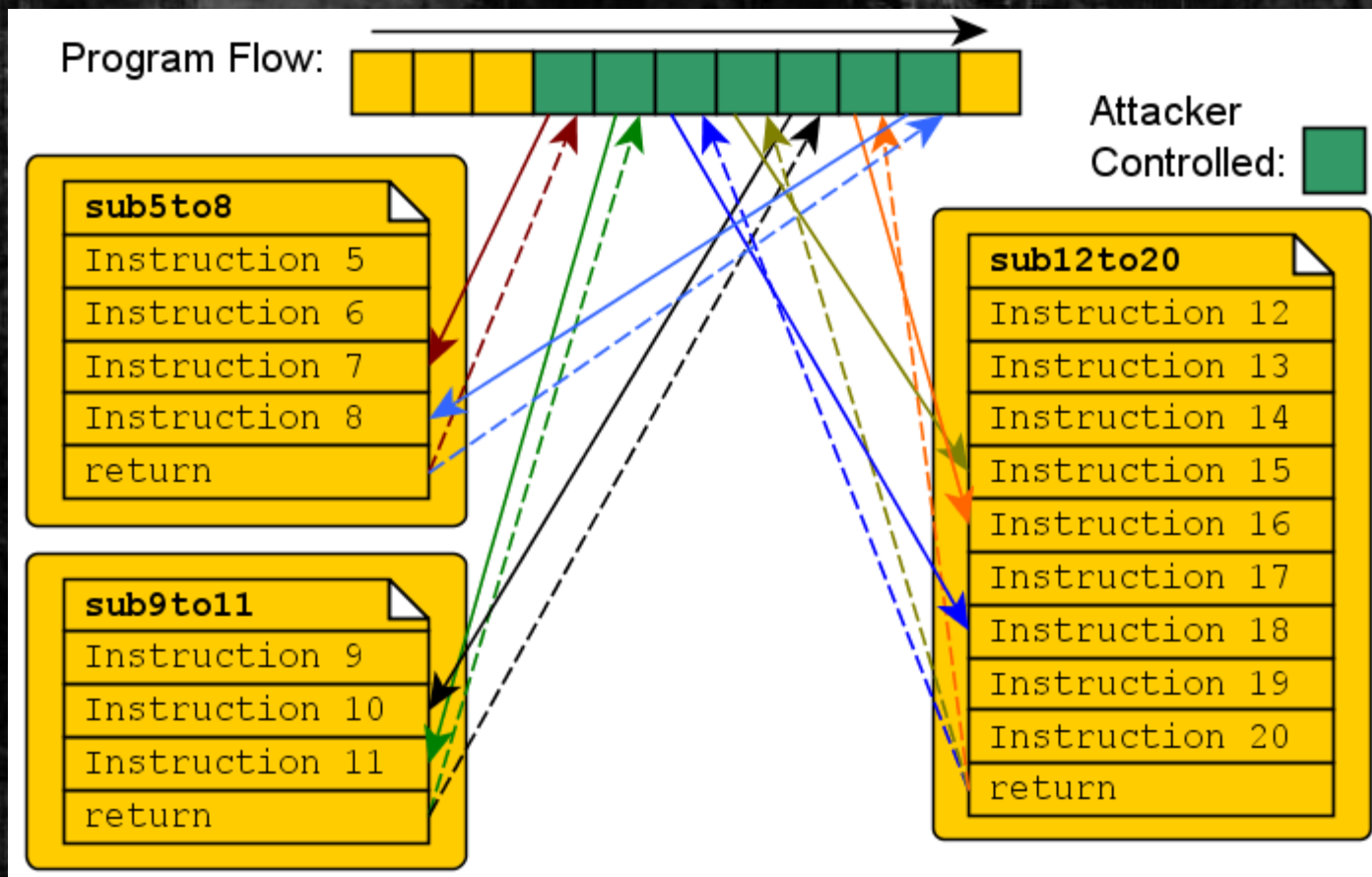


## SEH Exploit





# Return Oriented Programming



Resource: <http://www.auscert.org.au/render.html?it=13408>

# Return Oriented Programming

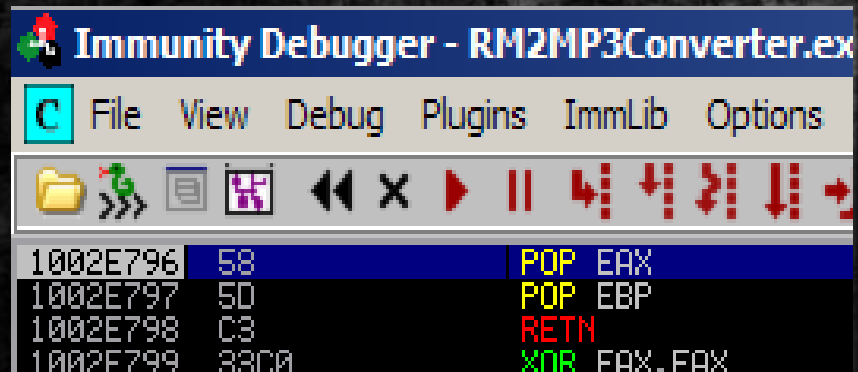
- Data Execution Prevention Bypass
  - VirtualAlloc
  - HeapCreate
  - SetProcessDEPPolicy
  - NtSetInformationProcess
  - VirtualProtect
  - WriteProcessMemory
- Code Reuse Technique
- Ret2lib
- ROP Gadgets

ROP / DEP Bypass

<http://cbolat.blogspot.com/2011/07/return-oriented-programming-dep-bypass.html>

# Return Oriented Programming

- ROP Gadget
  - RETN ve türevleri ile biten instruction'lar
  - CPU Register'ları ile oynamamızı sağlayan bir takım instruction'lar barındıran "instruction parçaları"



```
Immunity Debugger - RM2MP3Converter.exe
File View Debug Plugins ImmLib Options
[Folder] [Gears] [List] [CPU] [Back] [X] [Forward] [Pause] [Step Back] [Step Forward] [Step Into] [Step Out] [Next]
1002E796 58 POP EAX
1002E797 50 POP EBP
1002E798 C3 RETN
1002E799 33C0 XOR FAX, FAX
```

ROP / DEP Bypass

<http://cbolat.blogspot.com/2011/07/return-oriented-programming-dep-bypass.html>



# Return Oriented Programming

## VirtualProtect Function

Changes the protection on a region of committed pages in the virtual address space of the calling process.

To change the access protection of any process, use the **VirtualProtectEx** function.

### Syntax

```
BOOL WINAPI VirtualProtect(  
    __in LPVOID lpAddress,  
    __in SIZE_T dwSize,  
    __in DWORD flNewProtect,  
    __out PDWORD lpflOldProtect  
);
```

ROP / DEP Bypass

<http://cbolat.blogspot.com/2011/07/return-oriented-programming-dep-bypass.html>

# ROP Payload Yapısı



ROP / DEP Bypass

<http://cbolat.blogspot.com/2011/07/return-oriented-programming-dep-bypass.html>

# Genel Exploit'lerin Özellikleri

- Çok farklı programlama dillerinde sunulabilirler (binary,c,c++,perl,lisp,python)
- Açığın tek veya özel bir kullanımı üzerine geliştirilmiş olabilirler (..%c0%af.. veya ..%c0%qf..)
- Payload/Shellcode değeri özelleştirilemeyebilir (binary, açık hakkında kısıtlı bilgi)
- Kod kirli veya kötü niyetli yazılmış olabilir
- Herkesçe kullanıldığı için önlem alınmış olabilir
- Sadece açığı kanıtlamak için yazılmış kod örnekleri olabilir



# Exploit Geliştirme Süreci



# Hangi Araçlar Kullanılır

- Açık Bulunan Yazılımın Örneği !?
- Fuzzer (Karıştırıcı ve Değiştiriciler)
- Encoder (Kodlayıcılar)
- HEX Editörler
- Binary Analiz Araçları
- Debugger (Hata Ayıklayıcılar)
- Sniffer (Paket Yakalayıcılar)
- Protokol Çözümleyiciler
- Yorumlayıcılar / Derleyiciler (Interpreter/Compiler)
- Shellcode'lar
- SQL Sorguları



# Metasploit Framework

- Bileşenler
  - ~1000 Exploit ~250 Payload ~500 Yardımcı Araç ~30 Encoder
  - ~150 Exploit Sonrası Modül/Script
- Çok farklı türde Payload'lar kullanılabilir ve bağımsız olarak üretilebilir (Binary, Perl, Python, Shell, PHP)
- Meterpreter ile Hedef Tamamen Ele Geçirilebilir
- VNC ile Hedef Sisteme Grafik Arayüzle Bağlanılabilir
- Çok sayıda farklı encoder kullanılabilir (Shikata Ga Nai vb.)
- Konsol, Seri ve Grafik (Armitage) arayüzlerine sahip
- En güçlü özelliği Post-Exploitation yetenekleri (Meterpreter, VNC DLL Injection, Anti-Forensic, Process Migration vb.)



# Metasploit Framework'ün Avantajları

- Exploit ve Payload ayrımı
- Varolan Exploit'lerin Geliştirilmesi, Hedef Eklenmesi
  - Kaynak kodu açık Exploit'ler
  - 0 gün Exploit'leri
- Hazır fonksiyonlar ile daha az Exploit kodu
  - Sadece Offset ve Ret Değişimi ile Hedef Ekleme
  - Hazır Servisler ve Protokol Kütüphaneleri
  - Farklı İşlemci Mimarileri için Kodlayıcılar (Encoder)
  - Hazır Kabuk Kodları ve Yardımcı araçlar
  - Egg Hunting, Heap Spraying, ROP Destekleri

# Metasploit Framework Payload Seçimi

- Kabuk Kodları
  - Bind Shell
  - Exec
  - Reverse Shell
  - Staged Shell
- İleri Düzey Yöntemler
  - EggHunting
  - Meterpreter (Railgun, Powershell, Ruby Scripting)
  - DLL Injection (VNC, Özelleştirilmiş DLL)
- Her Kabuk Kodu için Karşılama
  - Multi/Handler
- Kodlama Araçları
  - Alpha2, Shika\_ga\_nai

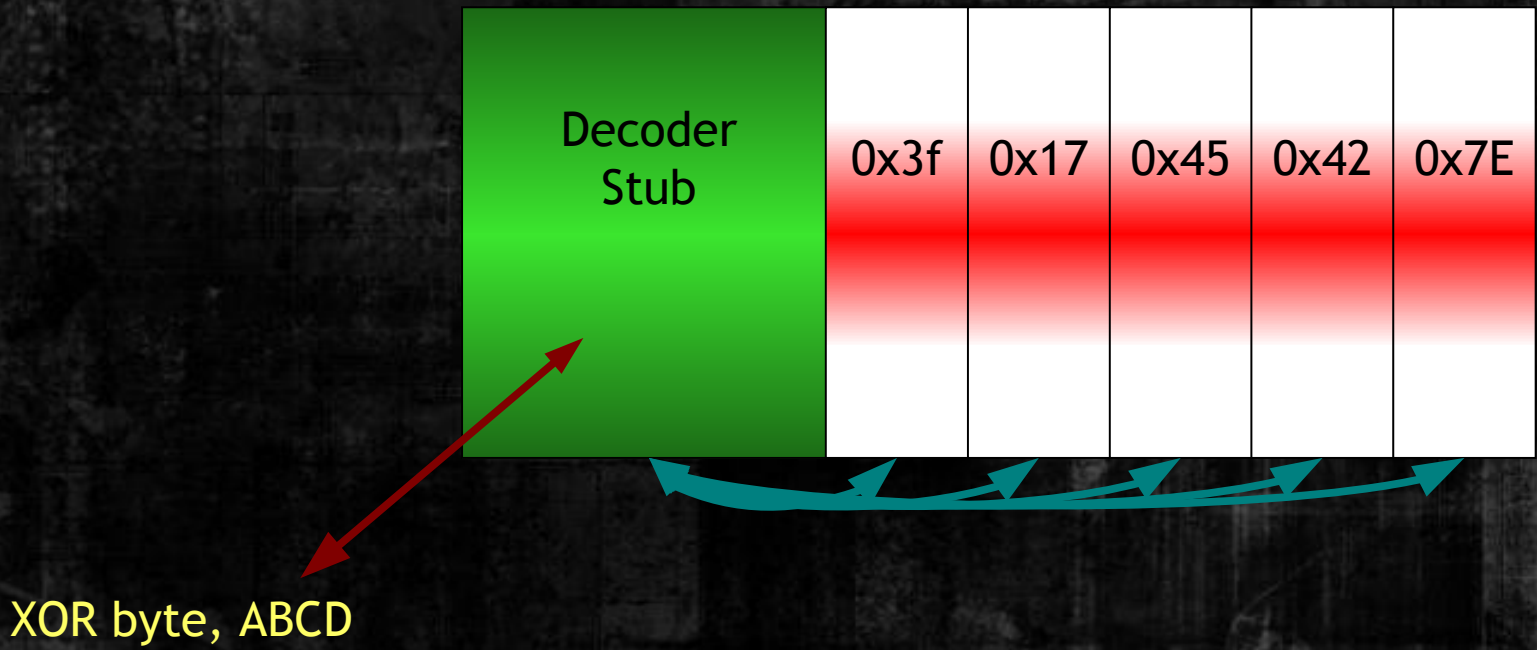
Kabuk Kodu

Decoder

Oudk Kbuak

# Encoder/Decoder

## Decoded Payload



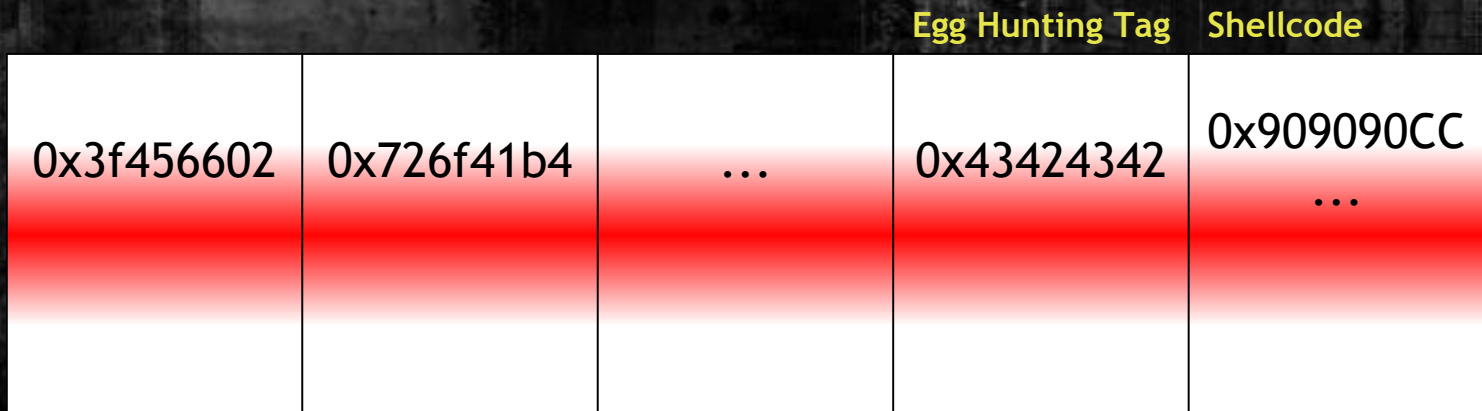


# Egg Hunting

- Yapısı Payload Decoder'lara çok benzemektedir
- Shellcode Memory'de bilmediğimiz bir adrestedir
- Egg Hunting payload Memory'de belirtilen özel bir string değeri arar
- O string değerinin bulunduğu yerden itibaren olan instruction'lar çalıştırılır
- Yani Shellcode!

# Egg Hunting

## Memory



IF (DWORD == 0x43424342)

JUMP TO DWORD+4

# Egg Hunting

```
00 33DB      xor ebx,ebx
02 6681CBFF0F or bx,0xffff
07 43        inc ebx
08 6A08      push byte +0x8
0A 53        push ebx
0B B80D5BE777 mov eax,0x77e75b0d ; IsBadReadPtr
10 FFD0      call eax
12 85C0      test eax,eax
14 75EC      jnz 0x2
16 B842434243 mov eax,0x43424342 ; TAG: CBCB
1B 8BFB      mov edi,ebx
1D AF        scasd          ; Compare EAX w/ ES:EDI
1E 75E7      jnz 0x7
20 AF        scasd          ; Compare EAX w/ ES:EDI
21 75E4      jnz 0x7
23 FFE7      jmp edi        ; JUMP to Shellcode
```



# Metasploit Framework Araçları

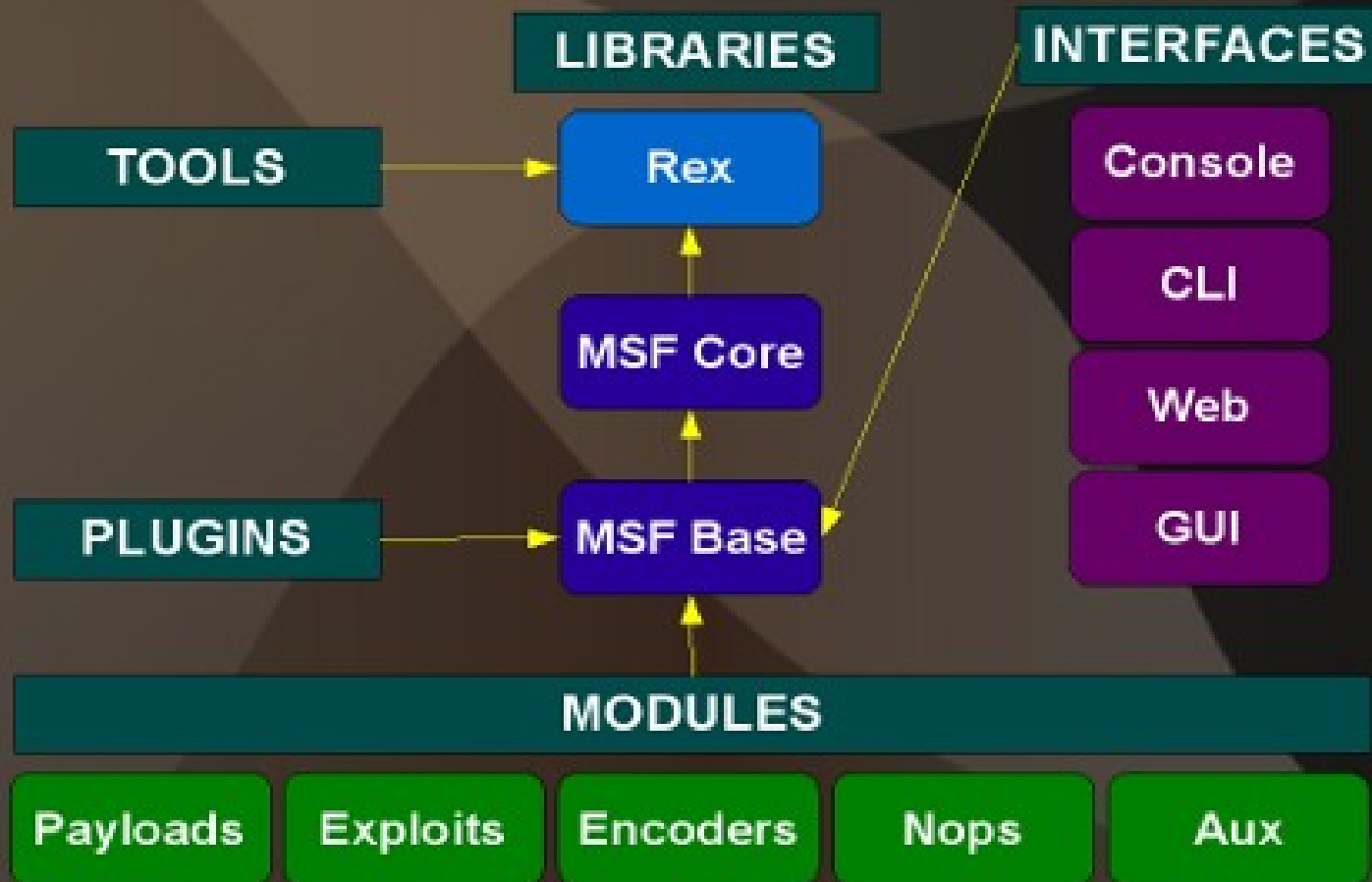
- MSFPayload - Kabuk Kodu Kodlama
- MSFEncode - Kabuk Kodu Kodlama
- MSFRop - ROP Gadget'lerinin Araştırılması
- Pattern\_Create/Pattern\_Offset - Bellek Alanı Hesaplama
- MSFPEScan/MSFElfScan - Dönüş Adresi Bulma
- Metasm\_Shell - Assembler/Disassembler

# Metasploit Framework Araçları

- MSFPayload - Kabuk Kodu Kodlama
- MSFEncode - Kabuk Kodu Kodlama
- MSFRop - ROP Gadget'lerinin Araştırılması
- Pattern\_Create/Pattern\_Offset - Bellek Alanı Hesaplama
- MSFPEScan/MSFElfScan - Dönüş Adresi Bulma
- Metasm\_Shell - Assembler/Disassembler

# REX Kütüphanesi ve Modül Yapısı

## *Metasploit Architecture*





# Exploit ve Modül Ayrımı

Metasploit Framework'e Göre, Kabuk Kodu Çalıştırılmıyorsa Exploit Değildir.

- Auxiliary Modülü
- Post Modülü

## Yardımcı Modül ile Exploit İşlemi

- Sahte Servis ile Bilgi Sızdırma, Fuzzing, Bilgi Toplama
- Dosya İşletme, Servis Engelleme, Sosyal Mühendislik
- Veritabanı, SAP, VoIP ve VM Analizi

## Post Modülü

- Yetki Yükseltme, Bilgi Toplama, Yönetim Ele Geçirme

# Protokol, Servis ve İstemci Sınıfları

Msf::Exploit::Remote::Ftp  
Msf::Exploit::Remote::FtpServer  
Msf::Exploit::Remote::HttpClient  
Msf::Exploit::Remote::HttpServer  
Msf::Exploit::Remote::HttpServer::HTML  
Msf::Exploit::Remote::MSSQL  
Msf::Exploit::Remote::MSSQL\_SQLI  
Msf::Exploit::Remote::MYSQL  
Msf::Exploit::Remote::TNS  
Msf::Exploit::Remote::Postgres  
Msf::Exploit::Remote::SMB  
Msf::Exploit::Remote::SMBServer  
Msf::Exploit::Remote::SMB::Authenticated

Msf::Exploit::Remote::Imap  
Msf::Exploit::Remote::Smtplib  
Msf::Exploit::Remote::SMTPDeliver  
Msf::Exploit::Remote::SunRPC  
Msf::Exploit::Remote::Tcp  
Msf::Exploit::Remote::TcpServer  
Msf::Exploit::Remote::TFTPServer  
Msf::Exploit::Remote::NDMP  
Msf::Exploit::Remote::Udp  
Msf::Exploit::Remote::WinRM



# Fuzzing ve Kabuk Kodu Hazırlama

## Kodlama ve Dönüştürme

- `Rex::Assembly::Nasm`
- `Rex::Encoder::Alpha2`
- `Rex::Encoder::Xor`
- `Rex::Encoder::NonAlpha`
- `Rex::Encoder::NonUpper`

## Karakter Üretme - `Rex::Text`

- `rand_text*`
- `md5`
- `encode/decode_base64`

## Fuzzing - `Msf::Auxiliary::Fuzzer`

- `#fuzz_numbers`
- `#fuzz_string_corrupt*`
- `#fuzz_strings`
- `#fuzzer_gen_string`
- `#fuzzer_number*`
- `#fuzzer_string_filepath_dos`
- `#fuzzer_string*`
- `#fuzzer_string_path*`
- `#fuzzer_string_uri*`



# Diğer Faydalı Sınıflar

## Kabuk Kodu Çalıştırma

- Msf::Exploit::Brute
- Msf::Exploit::BruteTargets
- Msf::Exploit::RopDb
- Rex::Exploitation::Seh
- Rex::Exploitation::Omelet::\*
- Rex::Exploitation::Egghunter::\*
- Rex::Exploitation::OpcodeDb::\*
- Rex::Exploitation::EncryptJS
- Rex::Exploitation::HeapLib
- Rex::Exploitation::JSObfu
- Rex::Exploitation::ObfuscateJS
- Rex::Exploitation::CmdStager\*

## Dosya Tipi Açıkları

- Msf::Exploit::FILEFORMAT
- Msf::Exploit::PDF
- Msf::Exploit::PDF\_Parse

## Exe Çalıştırma, Format String

- Msf::Exploit::FormatString
- Msf::Exploit::WbemExec

## Yerel Açıkların Kullanımı

- Msf::Exploit::Local::Linux
- Msf::Exploit::Local::LinuxKernel
- Msf::Exploit::Local::Unix
- Msf::Exploit::KernelMode

# Exploit ve Modül Örnekleri

## *Metasploit Framework Exploit ve Modüllerinin Örnekleme ile Analizi*



# Bağlantılar ve Referanslar

- Günlükler  
[fozavci.blogspot.com](http://fozavci.blogspot.com)  
[cbolat.blogspot.com](http://cbolat.blogspot.com)
- Türkçe Metasploit Framework Rehberi  
[www.gamasec.net/fozavci](http://www.gamasec.net/fozavci)
- Metasploit Project  
[www.metasploit.com](http://www.metasploit.com)
- Metasploit Unleashed  
[www.offensive-security.com/metasploit-unleashed/Main\\_Page](http://www.offensive-security.com/metasploit-unleashed/Main_Page)



Sorular ?





*Teşekkürler....*