

Inhaltsverzeichnis

Vorwort	11
Einleitung	13
Über dieses Buch.....	13
Der Aufbau dieses Buchs.....	13
Abschließende Bemerkung.....	15
Danksagung.....	17
Die Autoren	19
Der Fachgutachter.....	19
 Teil 1: Eine Reise ins Kernelland	 21
 1. Von Userland- zu Kernelland-Angriffen	 23
1.1 Einführung	23
1.2 Der Kernel und die Welt des Kernel-Hackings.....	24
1.2.1 Die Kunst der Ausnutzung.....	25
1.3 Warum funktioniert mein Userland-Exploit nicht mehr?	30
1.3.1 Kernelland- und Userland-Exploits im Vergleich	33
1.4 Der Kernel aus der Sicht eines Exploit-Autors	35
1.4.1 Userland-Prozesse und der Scheduler	35
1.4.2 Virtueller Arbeitsspeicher	36
1.4.3 Benutzerraum oberhalb des Kernelraums im Vergleich mit getrennten Adressräumen	38
1.5 Open-Source- und Closed-Source-Betriebssysteme.....	40
1.6 Zusammenfassung.....	41
1.6.1 Literatur	42

2. Klassifizierung von Kernelschwachstellen	43
2.1 Einführung	43
2.2 Dereferenzierung nicht initialisierter, nicht validierter und beschädigter Zeiger	44
2.3 Schwachstellen durch beschädigten Arbeitsspeicher	49
2.3.1 Schwachstellen des Kernelstacks.....	49
2.3.2 Schwachstellen des Kernelheaps	51
2.4 Integerprobleme	53
2.4.1 (Arithmetische) Integerüberläufe	53
2.4.2 Vorzeichenfehler	55
2.5 Race Conditions	57
2.6 Logikbugs	63
2.6.1 Referenzzählerüberlauf.....	63
2.6.2 Validierung der Eingaben von physischen Geräten	65
2.6.3 Vom Kernel hervorgerufene Userland-Schwachstellen.....	66
2.7 Zusammenfassung.....	69
 3. Der Weg zum erfolgreichen Kernel-Hacking	 71
3.1 Einführung	71
3.2 Die Architekturebene	73
3.2.1 Allgemeine Prinzipien	73
3.2.2 x86 und x86-64	80
3.3 Der Ausführungsschritt	84
3.3.1 Den Shellcode platzieren	84
3.3.2 Den Shellcode gestalten	92
3.3.3 Den Kernelzustand wiederherstellen.....	94
3.4 Der Auslöseschritt.....	98
3.4.1 Speicherbeschädigung	98
3.4.2 Race Conditions	113

3.5	Der Schritt zur Informationsgewinnung.....	118
3.5.1	Was uns die Umgebung mitteilt	119
3.5.2	Was uns die Umgebung nicht mitteilen möchte: Infoleaks	124
3.6	Zusammenfassung.....	126
3.6.1	Literatur	127

Teil 2: Die UNIX-Familie, Mac OS X und Windows **129**

4.	Die UNIX-Familie	131
4.1	Einführung	131
4.2	Die Mitglieder der UNIX-Familie	133
4.2.1	Linux	133
4.2.2	Solaris/OpenSolaris.....	144
4.2.3	BSD-Derivate	157
4.3	Der Ausführungsschritt	157
4.3.1	Das Rechtemodell von Linux missbrauchen	158
4.4	UNIX-Hacking in der Praxis	172
4.4.1	Hacking des Kernelheaps.....	172
4.4.2	Angriff auf den Slab-Allokator von OpenSolaris	173
4.4.3	Angriff auf den SLUB-Allokator von Linux 2.6.....	197
4.4.4	Stacküberläufe im (Linux-) Kernel	216
4.4.5	CVE-2009-3234, zum Zweiten	223
4.5	Zusammenfassung.....	235
5.	Mac OS X.....	237
5.1	Einführung	237
5.2	Überblick über XNU	239
5.2.1	Mach.....	239

5.2.2	BSD.....	240
5.2.3	IOKit.....	240
5.2.4	Systemaufruftabellen.....	241
5.3	Kerneldebugging.....	243
5.4	Kernelerweiterungen (Kext).....	253
5.4.1	IOKit.....	259
5.4.2	Überprüfen von Kernelerweiterungen	260
5.5	Der Ausführungsschritt	273
5.6	Hinweise für Exploits	275
5.6.1	Willkürliches Überschreiben des Arbeitsspeichers.....	275
5.6.2	Stacküberläufe.....	287
5.6.3	Exploits für den Speicherallocator	304
5.6.4	Race Conditions	319
5.6.5	Snow Leopard	319
5.7	Zusammenfassung.....	319

6. Windows.....321

6.1	Einführung	321
6.2	Überblick über den Windows-Kernel	323
6.2.1	Informationen über den Kernel gewinnen	324
6.2.2	DVWD (Dawn Vulnerable Windows Driver)	328
6.2.3	Interne Mechanismen des Kernels	330
6.2.4	Kerneldebugging.....	335
6.3	Der Ausführungsschritt	338
6.3.1	Das Autorisierungsmodell von Windows	338
6.3.2	Den Shellcode erstellen	348
6.4	Windows-Hacking in der Praxis	362
6.4.1	Stackpufferüberlauf.....	374
6.5	Zusammenfassung.....	395

Teil 3: Remote-Exploits 397

7. Die Herausforderung durch Remote-Kernelexploits 399

7.1	Einführung	399
7.2	Schwachstellen über das Netz angreifen	400
7.2.1	Mangel an offengelegten Informationen	401
7.2.2	Mangelnder Einfluss auf das Ziel	403
7.3	Die erste Anweisung ausführen	405
7.3.1	Direkte Umleitung des Ausführungsflusses	406
7.3.2	Willkürliches Überschreiben des Kernelarbeitspeichers	419
7.4	Remote-Payloads	421
7.4.1	Payload-Migration	422
7.5	Zusammenfassung	444

8. Anwendung in der Praxis am Beispiel von Linux 445

8.1	Einführung	445
8.2	Heapbeschädigung im SCTP-FWD-Abschnitt	446
8.2.1	Überblick über SCTP	446
8.2.2	Der anfällige Pfad	449
8.3	Der Remote-Exploit: Allgemeiner Überblick	453
8.4	Die Voraussetzungen zum willkürlichen Überschreiben des Arbeitsspeichers schaffen	454
8.4.1	Das Heaplayout über das Netzwerk anpassen	455
8.4.2	SCTP-Nachrichten erstellen: Vom relativen zum absoluten Überschreiben des Arbeitsspeichers	458
8.5	Den Shellcode installieren	464
8.5.1	Direkter Sprung vom Interruptkontext ins Userland	464
8.6	Den Shellcode ausführen	472
8.6.1	Den laufenden Prozess prüfen und die Funktion gettimeofday() emulieren	473

8.6.2	Die rückwärtige Verbindung ausführen	474
8.6.3	Vsyscall wiederherstellen	476
8.7	Zusammenfassung.....	477
8.8	Literatur	478

Teil 4:	Schlusswort	479
----------------	--------------------	------------

9.	Die Entwicklung des Kernels:	
	Angriff und Verteidigung in der Zukunft	481
9.1	Einführung	481
9.2	Kernelangriffe	482
9.2.1	Vertraulichkeit	482
9.2.2	Integrität.....	484
9.2.3	Verfügbarkeit	488
9.3	Kernelschutz	488
9.3.1	Bedrohungsanalyse und Modellierung	489
9.3.2	Kernelschutzmechanismen	490
9.3.3	Vertrauen in den Kernel.....	491
9.4	Virtualisierung	496
9.4.1	Die Sicherheit des Hypervisors	496
9.4.2	Sicherheit des Gastkernels	498
9.5	Zusammenfassung.....	498

Stichwortverzeichnis	501
-----------------------------------	------------