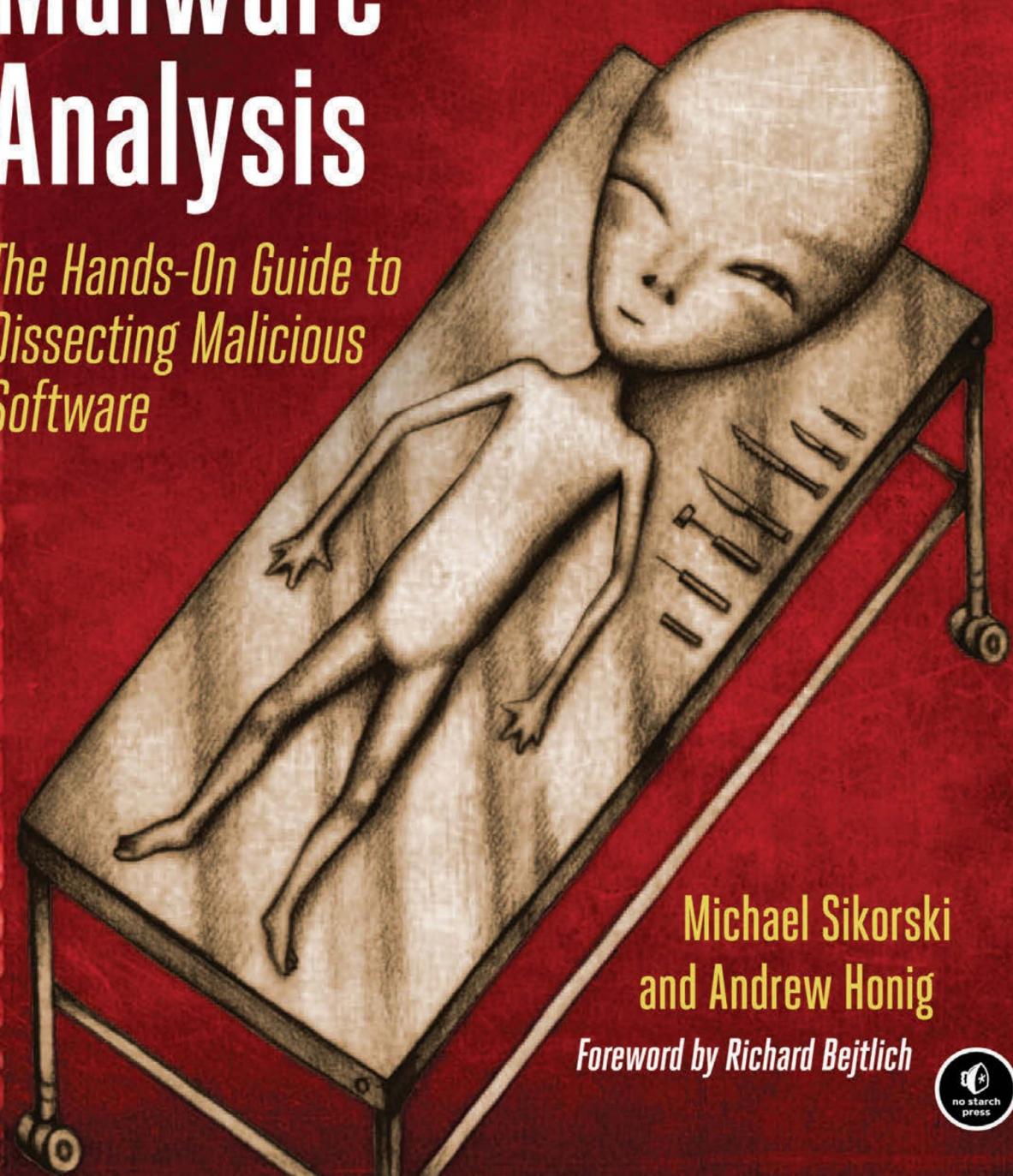


Practical Malware Analysis

*The Hands-On Guide to
Dissecting Malicious
Software*



Michael Sikorski
and Andrew Honig

Foreword by Richard Bejtlich



ELOGIOS PARA EL ANÁLISIS PRÁCTICO DE MALWARE

Libro de análisis forense digital del año, premios FORENSIC 4CAST 2013

“Una introducción práctica al análisis de malware. Se lo recomendaría a cualquiera que quiera analizar malware de Windows”.

—Ilfak Guilfanov, CREADOR DE IDA PRO

“El libro que todo analista de malware debería tener a mano”.

—Richard Bejtlich, director de estrategia empresarial de MANDIANT y fundador de TAOSECURITY

“Este libro hace exactamente lo que promete en la portada: está repleto de detalles y tiene un enfoque intensamente práctico, pero está lo suficientemente bien organizado como para que puedas tenerlo a mano como referencia”.

—Mary Branscombe, ZDNET

“Si estás empezando en el análisis de malware, o si llegas al análisis desde otra disciplina, te recomiendo que tengas un poco de olfato”.

—Paul Baccas, SEGURIDAD DESNUDA DE SOPHOS

“Un excelente curso intensivo sobre análisis de malware”.

—Dino Dai Zovi, CONSULTOR DE SEGURIDAD INDEPENDIENTE

“La guía más completa para el análisis de malware, que ofrece una cobertura detallada de todas las habilidades esenciales necesarias para comprender los desafíos específicos que presenta el malware moderno”.

—Chris Eagle, PROFESOR PRINCIPAL DE CIENCIAS DE LA COMPUTACIÓN EN LA NAVAL ESCUELA DE POSGRADO

“Una excelente introducción al análisis de malware. Todos los capítulos contienen explicaciones técnicas detalladas y ejercicios prácticos de laboratorio para que pueda familiarizarse de inmediato con el malware real”.

—Sebastian Porst, INGENIERO DE SOFTWARE DE GOOGLE

“Acerca la ingeniería inversa a lectores de todos los niveles. Los laboratorios, técnicamente ricos y accesibles, le permitirán comprender mejor el arte y la ciencia de la ingeniería inversa. Creo firmemente que este se convertirá en el texto de referencia para aprender a analizar malware en el futuro”.

—Danny Quist, PHD, FUNDADOR DE LA COMPUTACIÓN OFENSIVA

“Un libro maravilloso... escrito por autores expertos que poseen el don excepcional de poder comunicar sus conocimientos a través de la palabra escrita”.

—Richard Austin, IEEE CIPHER

“Si solo lee un libro sobre malware o busca adentrarse en el mundo del análisis de malware, este es el libro que debe adquirir”.

—Patrick Engebretson, PROFESOR DE IA , UNIVERSIDAD ESTATAL DE DAKOTA Y

AUTOR DE Los conceptos básicos de piratería informática y pruebas de penetración

“Un excelente complemento a los materiales del curso para un curso avanzado de nivel de posgrado sobre seguridad de software o sistemas de detección de intrusiones. Los

laboratorios son especialmente útiles para que los estudiantes aprendan los métodos para realizar ingeniería inversa, analizar y comprender el software malicioso”.

—Sal Stolfo, PROFESOR, UNIVERSIDAD DE COLUMBIA

“La explicación de las herramientas es clara, la presentación del proceso es lúcida y el trabajo de investigación en sí es fascinante. Todo presentado con claridad y con el nivel justo para que los desarrolladores sin experiencia previa en este área en particular puedan participar plenamente. Muy recomendable.”

—Dr. Dobb

“Este libro es como tener tu propio profesor personal de análisis de malware sin los costosos costes de formación”.

—Dustin Schultz, EL EXPLOIT

“Recomiendo encarecidamente este libro a cualquiera que quiera iniciarse en el análisis de malware o simplemente busque una buena referencia de escritorio sobre el tema”.

—Pete Arzamendi, 403 LABS

“No entiendo cómo alguien que tiene una responsabilidad directa con la seguridad de los sistemas Windows puede justificar su falta de familiaridad con estas herramientas”.

—Stephen Northcutt, INSTITUTO SANS

PRÁCTICO ANÁLISIS DE MALWARE

La guía práctica para
Diseccionando lo malicioso
Software

por Michael Sikorski y Andrew Honig



San Francisco

ANÁLISIS PRÁCTICO DE MALWARE. Copyright © 2012 por Michael Sikorski y Andrew Honig.

Todos los derechos reservados. No se puede reproducir ni transmitir ninguna parte de esta obra en ninguna forma ni por ningún medio, electrónico o mecánico, incluidas la fotocopia, la grabación o cualquier sistema de almacenamiento o recuperación de información, sin el permiso previo por escrito del propietario de los derechos de autor y del editor.

ISBN-10: 1-59327-290-1

ISBN-13: 978-1-59327-290-6

Editorial: William Pollock

Editor de producción: Alison Law

Ilustración de portada: Hugh D'Andrade

Diseño de interiores: Octopod Studios

Editores de desarrollo: William Pollock y Tyler Ortman

Revisor técnico: Stephen Lawler

Correctora de estilo: Marilyn Smith

Compositor: Riley Hoffman

Correctora de pruebas: Irene Barnard

Indizador: Nancy Guenther

Para obtener información sobre distribución, traducciones o ventas al por mayor, comuníquese directamente con No Starch Press, Inc.:

Prensa sin almidón, Inc.

245 8th Street, San Francisco, CA 94103

Teléfono: 1.415.863.9900; info@nostarch.com; www.nostarch.com

Datos de catalogación en publicación de la Biblioteca del Congreso

Sikorski, Michael.

Análisis práctico de malware: la guía práctica para disecionar software malicioso / por Michael Sikorski y Andrew Honig.

pág. cm.

ISBN 978-1-59327-290-6 -- ISBN 1-59327-290-1

1. Malware (software informático). 2. Virus informáticos. 3. Depuración en informática. 4. Seguridad informática.

I. Honig, Andrew. II. Título.

QA76.76.C68S534 2012

005.8'4--dc23

2012000214

No Starch Press y el logotipo de No Starch Press son marcas registradas de No Starch Press, Inc. Otros nombres de productos y empresas mencionados en este documento pueden ser marcas comerciales de sus respectivos propietarios. En lugar de utilizar un símbolo de marca registrada cada vez que aparece un nombre de marca registrada, utilizamos los nombres solo de manera editorial y en beneficio del propietario de la marca registrada, sin intención de infringir la marca registrada.

La información contenida en este libro se distribuye "tal como está", sin garantía. Si bien se han tomado todas las precauciones durante la preparación de este trabajo, ni los autores ni No Starch Press, Inc. tendrán responsabilidad alguna ante ninguna persona o entidad con respecto a cualquier pérdida o daño causado o presuntamente causado directa o indirectamente por la información contenida en él.

BREVE CONTENIDO

Acerca de los autores	xix
Prólogo de Richard Bejtlich	xxi
Agradecimientos	xxv
Introducción	xxvii
Capítulo 0: Introducción al análisis de malware	1
PARTE 1: ANÁLISIS BÁSICO	
Capítulo 1: Técnicas estáticas básicas.....	9
Capítulo 2: Análisis de malware en máquinas virtuales.....	29
Capítulo 3: Análisis dinámico básico.....	39
PARTE 2: ANÁLISIS ESTÁTICO AVANZADO	
Capítulo 4: Curso intensivo sobre desmontaje de x86	65
Capítulo 5: IDA Pro	87
Capítulo 6: Reconocimiento de construcciones de código C en ensamblador.....	109
Capítulo 7: Análisis de programas maliciosos de Windows.....	135
PARTE 3: ANÁLISIS DINÁMICO AVANZADO	
Capítulo 8: Depuración.....	167

Capítulo 9: OllyDbg	179
Capítulo 10: Depuración del kernel con WinDbg.....	205
PARTE 4: FUNCIONALIDAD DEL MALWARE	
Capítulo 11: Comportamiento del malware	231
Capítulo 12: Ejecución de malware encubierto	253
Capítulo 13: Codificación de datos	269
Capítulo 14: Firmas de red enfocadas en malware.....	297
PARTE 5: INGENIERÍA ANTI-REVERSA	
Capítulo 15: Antidesmontaje.....	327
Capítulo 16: Anti-depuración	351
Capítulo 17: Técnicas anti-máquinas virtuales	369
Capítulo 18: Empacadores y desempacadores.....	383
PARTE 6: TEMAS ESPECIALES	
Capítulo 19: Análisis de Shellcode	407
Capítulo 20: Análisis en C++	427
Capítulo 21: Malware de 64 bits.....	441
Apéndice A: Funciones importantes de Windows.....	453
Apéndice B: Herramientas para el análisis de malware.....	465
Apéndice C: Soluciones de los laboratorios	477
Índice.....	733

CONTENIDO EN DETALLE

ACERCA DE LOS AUTORES	XIX
Acerca del revisor técnico	xx
Acerca de los autores colaboradores	xx
PRÓLOGO de Richard Bejtlich	xxi
EXPRESIONES DE GRATITUD	xxv
Agradecimientos individuales	xxv
INTRODUCCIÓN	xxvii
¿Qué es el análisis de malware?	xxviii
Requisitos previos	xxviii
Aprendizaje práctico y directo	xxix
¿Qué hay en el libro?	xxx
0	
MANUAL DE ANÁLISIS DE MALWARE	1
Los objetivos del análisis de malware	1
Técnicas de análisis de malware	2
Análisis estático básico	2
Análisis dinámico básico	2
Análisis estático avanzado	3
Análisis dinámico avanzado	3
Tipos de malware	3
Reglas generales para el análisis de malware	5
PARTE 1	
ANÁLISIS BÁSICO	
1	
TÉCNICAS ESTÁTICAS BÁSICAS Análisis	9
antivirus: un primer paso útil	10
Hashing: una huella dactilar del malware	10
Encontrar cadenas	11
Malware empaquetado y ofuscado	13
Archivos de embalaje	13
Detección de empacadores con PEID 14	
Formato de archivo ejecutable portátil	14
Bibliotecas y funciones vinculadas	15
Vinculación estática, en tiempo de ejecución y dinámica	15

Exploración de funciones vinculadas dinámicamente con Dependency Walker	16
Funciones importadas	18
Funciones exportadas	18
Análisis estático en la práctica	18
PotentialKeylogger.exe: un ejecutable descomprimido	18
PackedProgram.exe: un callejón sin salida	21
Los encabezados y secciones de archivos PE	21
Examen de archivos PE con PEvew	22
Visualización de la sección de recursos con Resource Hacker	25
Uso de otras herramientas de archivos PE	26
Resumen de encabezados PE	26
PE	26
Conclusión	26
Laboratorios	27

2	
EL ANÁLISIS DE MALWARE SE ENCUENTRA EN MÁQUINAS VIRTUALES	29
La estructura de una máquina virtual	30
Creación de su máquina de análisis de malware	31
Configuración de VMware	31
Uso de su máquina de análisis de malware	34
Conexión de malware a Internet	34
Conexión y desconexión de dispositivos periféricos	34
Toma de instantáneas	35
Transferencia de archivos desde una máquina virtual	36
Riesgos de usar VMware para el análisis de malware	36
Grabación/reproducción: ejecutar su computadora en reversa	37
Conclusión	37

3	
ANÁLISIS DINÁMICO BÁSICO	39
Sandboxes: el enfoque rápido y sucio	40
Uso de un sandbox de malware	40
Desventajas de los sandboxes	41
Ejecución de malware	42
Monitoreo con Process Monitor	43
La pantalla de Procmon	44
Filtrado en Procmon	44
Visualización de procesos con Process Explorer	47
La pantalla de Process Explorer	47
Uso de la opción Verificar	48
Comparación de cadenas	49
Uso de Dependency Walker	49
Ánalysis de documentos maliciosos	50
Comparación de instantáneas del registro con Regshot	50

Simulación de una red	51	Uso de
ApateDNS	51	Monitoreo con
Netcat	52	Detección de paquetes con
Wireshark	53	Uso de
INetSim	55	Herramientas dinámicas básicas
en la práctica	56	
Conclusión	60	
Laboratorios	61	

PARTE 2 ANÁLISIS ESTÁTICO AVANZADO

4

CURSO INTENSIVO SOBRE EL DESMONTAJE DE UN X86 65

Niveles de abstracción	66	Ingeniería
inversa	67	La arquitectura
x86	68	Memoria
principal	69	
Instrucciones	69	Códigos de operación
y endianidad	70	
Operandos	70	
Registros	71	Instrucciones
simples	73	La
pila	77	
Condicionales	80	
Ramificación	80	Instrucciones
Rep	81	Método principal y desplazamientos de
C	83	Más información: Manuales de la arquitectura x86 de
Intel	85	Conclusión

85

5 IDA PRO 87

Carga de un ejecutable	88	La interfaz de IDA
Pro	89	Modos de la ventana de
desensamblaje	89	Ventanas útiles para el
análisis	91	Regreso a la vista
predeterminada	92	Navegación por IDA
Pro	92	
Búsqueda	94	Uso de referencias
cruzadas	95	Referencias cruzadas de
código	95	Referencias cruzadas de
datos	96	Análisis de
funciones	97	Uso de opciones de
gráficos	98	

Mejora del desensamblado	100	Cambio de nombre de
ubicaciones	100	
Comentarios	100	Formato de
operando	100	Uso de constantes con
nombre	102	Redefinición de código y
datos	103	Extensión de IDA con
complementos	103	Uso de scripts de
IDC	104	
IDAPython	105	Uso de complementos
comerciales	106	
Conclusión	106	
Laboratorios	107	
6		
RECONOCIMIENTO DE CONSTRUCCIONES DE CÓDIGO C EN ENSAMBLAJE		109
Variables globales y variables locales	110	Desensamblado
de operaciones aritméticas	112	Reconocimiento de instrucciones
if	113	Análisis gráfico de funciones con IDA
Pro	114	Reconocimiento de instrucciones if
anidadas	114	Reconocimiento de
bucles	116	Búsqueda de bucles
for	116	Búsqueda de bucles
while	118	Comprensión de las convenciones
de llamadas a funciones	119	cdecl
119 stdcall	120	
fastcall	120	Push vs.
Move	120	121 122
Análisis de declaraciones switch		
Si Estilo		
Tabla de saltos		123
Desensamblado de matrices		127
Identificación de estructuras		
128 Análisis del recorrido de listas enlazadas		
130 Conclusión	132	
Laboratorios	133	
7		
ANALIZANDO PROGRAMAS MALICIOSOS DE WINDOWS		135
La API de Windows	136	Tipos y notación
húngara	136	
Identificadores	137	Funciones del
sistema de archivos	137	Archivos
especiales	138	El Registro de
Windows	139	Claves raíz del
Registro	140	
Regedit	140	Programas que se
ejecutan automáticamente	140	141 Funciones comunes del
Registro		

Análisis del código de registro en la práctica	141
scripts de registro con archivos .reg	142
redes	143
Berkeley	143
redes	144
WinINet	145
ejecución	145
DLL	145
Procesos	147
Subprocesos	149
Coordinación entre procesos con mutexes	151
Servicios	152
El modelo de objetos de componentes	154
Excepciones: cuando las cosas van mal	154
Modo kernel frente a modo de usuario	157
Modo kernel frente a modo de usuario nativa	158
La API	159
Conclusión	161
Laboratorios	162

PARTE 3 ANÁLISIS DINÁMICO AVANZADO

8	
DEPURACIÓN	167
Depuradores de nivel de fuente y de nivel de ensamblaje	168
kernel y de modo de usuario	168
depurador	169
paso	169
paso a paso por dentro	170
interrupción	171
Excepciones	175
y segunda oportunidad	176
comunes	176
depurador	177
práctica	177
178	
9	
OLYDBG (OllyDBG)	179
Carga de malware	180
ejecutable	180
ejecución	181
OllyDbg	181
memoria	183
Rebase	184
subprocesos y pilas	185
código	186

Puntos de interrupción	188
software	188
condicionales	189
hardware	190
memoria	190
DLL	191
Rastreo	192
estándar	192
llamadas	193
ejecución	193
Ivy	193
excepciones	194
parches	195
Shellcode	196
asistencia	197
Complementos	197
OllyDump	198
depurador	198
comandos	198
Marcadores	199
scripts	200
Conclusión	201
Laboratorios	202

10

DEPURACIÓN DEL KERNEL CON WINDDBG	205
Controladores y código del núcleo	206
depuración del núcleo	207
Uso de	
WinDbg	210
Lectura desde la memoria	210
Operadores aritméticos	211
Configuración de puntos de interrupción	211
Listado de módulos	212
Microsoft	212
Búsqueda de símbolos	212
Visualización de información de estructura	213
Configuración de símbolos de Windows	215
Depuración del núcleo en la práctica	215
Observación del código del espacio de usuario	215
Observación del código del modo núcleo	217
Búsqueda de objetos del controlador	220
Rootkits	221
Análisis de rootkits en la práctica	222
Carga de Interrupciones	225
controladores	226
Windows 7 y versiones x64	226
Conclusión	227
Laboratorios	228

PARTE 4

FUNCIONALIDAD DE MALWARE

11		
COMPORTAMIENTO DEL MALWARE		231
Descargadores y lanzadores	231	Puertas
traseras	232	Reverse
Shell	232	
RATs	233	
Botnets	234	Comparación de RATs y
Botnets	234	Ladrones de
credenciales	234	Intercepción de
GINA	235	Volcado de
hash	236	Registro de pulsaciones de
teclas	238	Mecanismos de
persistencia	241	El Registro de
Windows	241	Binarios del sistema
troyanizados	243	Secuestro de orden de carga de
DLL	244	Escalada de privilegios
245 Uso de SeDebugPrivilege	246	Cubriendo sus huellas:
rootkits en modo usuario	247	Enganche
IAT	248	Enganche en
línea	248	
Conclusión	250	
Laboratorios	251	
12		
LANZAMIENTO DE MALWARE ENCUBIERTO		253
Lanzadores	253	Inyección de
procesos	254	Inyección de
DLL	254	Inyección
directa	257	Reemplazo de
procesos	257	Inyección de
ganchos	259	Ganchos locales y
remotos	260	Keyloggers que utilizan
ganchos	260	Uso de
SetWindowsHookEx	260	Asignación de objetivos a
subprocesos	261	
Desvíos	262	Inyección de
APC	262	Inyección de APC desde el espacio de
usuario	263	Inyección de APC desde el espacio del
núcleo	264	Conclusión
265 Laboratorios	266	

13**CODIFICACIÓN DE DATOS****269**

El objetivo de analizar algoritmos de codificación	270	Cifrados
simples	270	Cifrado
César	270	
XOR	271	Otros esquemas de
codificación simples	276	
Base64	277	Algoritmos criptográficos
comunes	280	Reconocimiento de cadenas e
importaciones	281	Búsqueda de constantes
criptográficas	282	Búsqueda de contenido de alta
entropía	283	Codificación
personalizada	285	Identificación de la codificación
personalizada	285	Ventajas de la codificación personalizada para el
atacante	288	Decodificación
288 Autodescodificación	288	Programación manual de
funciones de decodificación	289	Uso de instrumentación para descifrado
genérico	291	
Conclusión	294	
Laboratorios	295	

14**FIRMAS DE RED CENTRADAS EN MALWARE****297**

Contramedidas de red	297	Observación del
malware en su hábitat natural	298	Indicaciones de actividad
maliciosa	298	OPSEC = Seguridad de
operaciones	299	Investigar de forma segura a un
atacante en línea	300	Tácticas de
indirección	300	Obtención de información de
dominio y dirección IP	300	Contramedidas de red basadas en
contenido	302	Detección de intrusiones con
Snort	303	Una mirada más
profunda	304	Combinación de técnicas de
análisis dinámico y estático	307	El peligro del
sobreanálisis	308	Ocultarse a simple
vista	308	Comprensión del código
circundante	312	Encontrar el código de
red	313	Conocer las fuentes de contenido de
red	314	Datos codificados de forma rígida frente a datos
efímeros	314	Identificar y aprovechar los pasos de
codificación	315	Crear una firma
317	318	Analizar las rutinas de análisis
múltiples elementos	320	Apuntar a
atacante	321	Comprender la perspectiva del
Conclusión	322	
Laboratorios	323	

PARTE 5

INGENIERÍA ANTI-REVERSA

15	
ANTIDESMONTAJE	327
Comprensión del antidesensamblado	328
desensamblado 329 Desensamblado	
lineal 329 Desensamblado orientado al	
flujo 331 Técnicas	
antidesensamblado 334 Instrucciones de salto con el mismo	
destino 334 Una instrucción de salto con una condición	
constante 336 Desensamblado imposible	
337 Instrucciones NOP con IDA Pro 340 Ocultación del control de	
flujo 340 El problema del puntero de	
función 340 Adición de referencias cruzadas de código faltante en IDA	
Pro 342 Abuso del puntero de retorno 342	
Mal uso de controladores de excepciones estructuradas 344 Cómo frustrar el análisis	
de marcos de pila 347	
Conclusión 349 Prácticas de	
laboratorio 350	
16	
ANTI-DEPURACIÓN	351
Detección del depurador de Windows	352
Windows 352 Uso de la API de	
estructuras 352 Comprobación manual de	
sistema 353 Comprobación de residuos del	
depurador 356 Identificación del comportamiento del	
356 Escaneo de INT	
357 Realización de sumas de comprobación de código 357 Comprobaciones de	
tiempo 357 Interferencia con la funcionalidad del	
depurador 359 Uso de devoluciones de llamadas	
TLS 359 Uso de excepciones	
361 Inserción de interrupciones 362 Vulnerabilidades	
del depurador 363 Vulnerabilidades del encabezado	
PE 363 La vulnerabilidad	
OutputDebugString 365	
Conclusión 365	
Laboratorios 367	
17	
TÉCNICAS ANTIMÁQUINAS VIRTUALES	369
Artefactos de VMware	370
artefactos de VMware 372 Cómo comprobar si hay artefactos de	
memoria 373	

Instrucciones vulnerables	373 Uso de la técnica Red
Pill Anti-VM	374 Uso de la técnica No
Pill	375 Consulta del puerto de comunicación de
E/S	375 Uso de la instrucción
str	377 Instrucciones Anti-VM
x86	377 Resaltado de Anti-VM en IDA
Pro	377 Uso de
ScoopyNG	379 Ajuste de la
configuración	379 Escapar de la máquina
virtual	380
Conclusión	380
Laboratorios	381

18

EMPAQUETADORES Y DESEMBALAJE 383

Anatomía del empaquetador	384 El stub de
desempaquetado	384 Carga del
ejecutable	384 Resolución de
importaciones	385 El salto de
cola	386 Desempaquetado
ilustrado	386 Identificación de programas
empaquetados	387 Indicadores de un programa
empaquetado	387 Cálculo de
entropía	387 Opciones de
desempaquetado	388 Desempaquetado
automático	388 Desempaquetado
manual	389 Reconstrucción de la tabla de importación
con Import Reconstructor	390 Búsqueda del
OEP	391 Reparación manual de la tabla
de importación	395 Consejos y trucos para empaquetadores
comunes	397 UPX
397 PECompact	397
ASPack	398
Petite	398
WinUpack	398
Themida	400 Análisis sin desempaquetar
completamente	400 DLL
empaquetadas	401
Conclusión	402
Laboratorios	403

PARTE 6 TEMAS ESPECIALES

19

ANÁLISIS DE SHELLCODE 407

Carga de Shellcode para análisis	408
--	-----

Código independiente de la posición	408	Identificación de la ubicación de ejecución
409 Uso de call/ pop	409	Uso de
fnstenv	411	Resolución manual de
símbolos	413	Búsqueda de kernel32.dll en la memoria
413 Análisis de datos de exportación de PE	413	Análisis de datos de exportación de
415 Uso de nombres exportados con hash	417	Un ejemplo completo de Hello World
418 Codificaciones de Shellcode	421	Sleds
NOP	422	Búsqueda de
Shellcode	423	
Conclusión	424	
Laboratorios	425	

20**ANÁLISIS EN C++****427**

Programación orientada a objetos	427	El puntero
this	428	Sobrecarga y alteración
430 Herencia y anulación de funciones	432	Funciones virtuales y no virtuales
432 Uso de Vtables	434	Reconocimiento de una Vtable
435 Creación y destrucción de objetos	437	
Conclusión	438	Prácticas de laboratorio
	439	

21**MALWARE DE 64 BITS****441**

¿Por qué malware de 64 bits?	442	Diferencias en la arquitectura x64
	443	Diferencias en la convención de llamada x64 y el uso de la pila
444 Manejo de excepciones de 64 bits	447	Windows de 32 bits en Windows de 64 bits
de 64 bits sobre la funcionalidad del malware	448	447 Sugerencias
Conclusión	449	
Laboratorios	450	

A**FUNCIONES IMPORTANTES DE WINDOWS****453****B****HERRAMIENTAS PARA ANÁLISIS DE MALWARE****465**

do

SOLUCIONES PARA LABORATORIOS

477

Laboratorio 1-1	477
Laboratorio 1-2	479
Laboratorio 1-3	480
Laboratorio 1-4	481
Laboratorio 3-1	482
Laboratorio 3-2	485
Laboratorio 3-3	490
Laboratorio 3-4	492
Laboratorio 5-1	494
Laboratorio 6-1	501
Laboratorio 6-2	503
Laboratorio 6-3	507
Laboratorio 6-4	511
Laboratorio 7-1	513
Laboratorio 7-2	517
Laboratorio 7-3	519
Laboratorio 9-1	530
Laboratorio 9-2	539
Laboratorio 9-3	545
Laboratorio 10-1	548
Laboratorio 10-2	554
Laboratorio 10-3	560
Laboratorio 11-1	566
Laboratorio 11-2	571
Laboratorio 11-3	581
Laboratorio 12-1	586
Laboratorio 12-2	590
Laboratorio 12-3	597
Laboratorio 12-4	599
Laboratorio 13-1	607
Laboratorio 13-2	612
Laboratorio 13-3	617
Laboratorio 14-1	626
Laboratorio 14-2	632
Laboratorio 14-3	637
Laboratorio 15-1	645
Laboratorio 15-2	646
Laboratorio 15-3	652
Laboratorio 16-1	655
Laboratorio 16-2	660
Laboratorio 16-3	665
Laboratorio 17-1	670
Laboratorio 17-2	673
Laboratorio 17-3	678
Laboratorio 18-1	684
Laboratorio 18-2	685
Laboratorio 18-3	686
Laboratorio 18-4	689
Laboratorio 18-5	691
Laboratorio 19-1	696
Laboratorio 19-2	699
Laboratorio 19-3	703
Laboratorio 20-1	712
Laboratorio 20-2	713
Laboratorio 20-3	717
Laboratorio 21-1	723
Laboratorio 21-2	728

ÍNDICE

733

ACERCA DE LOS AUTORES

Michael Sikorski es consultor de seguridad informática en Mandiant. Realiza ingeniería inversa de software malicioso para respaldar las investigaciones de respuesta a incidentes y brinda soluciones de seguridad especializadas de investigación y desarrollo a la base de clientes federales de la empresa. Mike creó una serie de cursos sobre análisis de malware y los imparte a una variedad de audiencias, incluido el FBI y Black Hat. Llegó a Mandiant desde el Laboratorio Lincoln del MIT, donde realizó investigaciones sobre mapeo pasivo de redes y pruebas de penetración. Mike también es egresado del Programa Interdisciplinario de Sistemas y Redes (SNIP) de tres años de la NSA. Mientras estuvo en la NSA, contribuyó a la investigación en técnicas de ingeniería inversa y recibió múltiples premios por inventiva en el campo del análisis de redes.

Andrew Honig es un experto en seguridad de la información del Departamento de Defensa. Imparte cursos sobre análisis de software, ingeniería inversa y programación de sistemas Windows en la Escuela Nacional de Criptología y es un profesional certificado en seguridad de sistemas de información. Andy ha sido reconocido públicamente por varios ataques de día cero en productos de virtualización de VMware y ha desarrollado herramientas para detectar software malicioso innovador, incluido software malicioso en el núcleo. Es un experto en el análisis y la comprensión de software tanto malicioso como no malicioso y tiene más de 10 años de experiencia como analista en la industria de la seguridad informática.

Acerca del revisor técnico

Stephen Lawler es el fundador y presidente de una pequeña empresa de consultoría de seguridad y software informático. Stephen ha trabajado activamente en seguridad de la información durante más de siete años, principalmente en ingeniería inversa, análisis de malware e investigación de vulnerabilidades. Fue miembro del equipo de análisis de malware de Mandiant y colaboró en intrusiones informáticas de alto perfil que afectaron a varias empresas de Fortune 100. Anteriormente trabajó en la división Security and Mission Assurance (SMA) de ManTech International, donde descubrió numerosas vulnerabilidades de día cero y técnicas de explotación de software como parte de los esfuerzos continuos de garantía de software. En una vida anterior que no tenía nada que ver con la seguridad informática, fue el desarrollador principal del componente de simulador de sonar del programa SMMTT de la Marina de los EE. UU.

Acerca de los autores colaboradores

Nick Harbour es analista de malware en Mandiant y un veterano experimentado en el negocio de la ingeniería inversa. Su carrera de 13 años en seguridad de la información comenzó como examinador forense informático e investigador en el Laboratorio de Informática Forense del Departamento de Defensa. Durante los últimos seis años, Nick ha estado en Mandiant y se ha centrado principalmente en el análisis de malware. Es investigador en el campo de las técnicas anti-ingeniería inversa y ha escrito varios empaquetadores y herramientas de ofuscación de código, como PE-Scrambler. Ha dado presentaciones en Black Hat y Defcon varias veces sobre el tema de técnicas anti-ingeniería inversa y anti-análisis forense. Es el desarrollador principal y profesor de un curso de Análisis avanzado de malware de Black Hat.

Lindsey Lack es director técnico de Mandiant y cuenta con más de doce años de experiencia en seguridad de la información, especializándose en ingeniería inversa de malware, defensa de redes y operaciones de seguridad. Ha ayudado a crear y operar un Centro de Operaciones de Seguridad, ha liderado esfuerzos de investigación en defensa de redes y ha desarrollado soluciones de alojamiento seguro. Anteriormente ha ocupado cargos en el Laboratorio Nacional de Investigación de Garantía de la Información, la Oficina Ejecutiva del Presidente (EOP), Cable and Wireless y el Ejército de los EE. UU. Además de una licenciatura en Ciencias de la Computación de la Universidad de Stanford, Lindsey también recibió una maestría en Ciencias de la Computación con énfasis en garantía de la información de la Escuela de Postgrado Naval.

Jerrold "Jay" Smith es consultor principal en Mandiant, donde se especializa en ingeniería inversa de malware y análisis forense. En este puesto, ha contribuido a muchas respuestas a incidentes ayudando a una variedad de clientes de empresas de Fortune 500. Antes de unirse a Mandiant, Jay trabajó en la NSA, pero no se le permite hablar de eso. Jay tiene una licenciatura en ingeniería eléctrica y ciencias de la computación de la UC Berkeley y una maestría en ciencias de la computación de la Universidad Johns Hopkins.

PREFACIO

Pocas áreas de la seguridad digital parecen tan asimétricas como aquellas que involucran malware, herramientas defensivas y sistemas operativos.

En el verano de 2011, asistí a la conferencia de apertura de Peiter (Mudge) Zatko en Black Hat en Las Vegas, Nevada. Durante su charla, Mudge presentó la naturaleza asimétrica del software moderno. Explicó cómo analizó 9000 archivos binarios de malware y contó un promedio de 125 líneas de código (LOC) para su conjunto de muestra.

Se podría argumentar que las muestras de Mudge incluían sólo malware “simple” o “peatonal”. Se podría preguntar, ¿qué pasa con algo verdaderamente “armado”? Algo como (aguanta la respiración) Stuxnet. Según Larry L. Constantine,¹ Stuxnet incluía alrededor de 15.000 LOC y, por lo tanto, tenía un tamaño 120 veces mayor que una muestra de malware promedio de 125 LOC. Stuxnet era altamente especializado y específico, lo que probablemente explica su tamaño superior al promedio.

Dejando el mundo del malware por un momento, el editor de texto que estoy usando (gedit, el editor de texto de GNOME) incluye gedit.c con 295 LOC, y gedit.c es solo uno de los 128 archivos fuente totales (junto con 3 directorios más) publicados.

1. <http://www.informit.com/articles/article.aspx?p=1686289>

en el repositorio de código fuente de GNOME GIT para gedit.² Contando los 128 archivos y 3 directorios se obtienen 70.484 LOC. La proporción de LOC de aplicaciones legítimas con respecto a malware es de más de 500 a 1. En comparación con una herramienta bastante sencilla como un editor de texto, ¡una muestra de malware promedio parece muy eficiente!

El número de 125 LOC de Mudge me pareció un poco bajo, porque existen diferentes definiciones de "malware". Muchas aplicaciones maliciosas existen como "suites", con muchas funciones y elementos de infraestructura. Para capturar este tipo de malware, conté lo que se podría considerar razonablemente como los elementos "fuente" del troyano Zeus (.cpp, .obj, .h, etc.) y conté 253.774 LOC.

Al comparar un programa como Zeus con una de las muestras promedio de Mudge, ahora vemos una relación de más de 2000 a 1.

Mudge comparó entonces el LOC del malware con los recuentos de los productos de seguridad diseñados para interceptar y derrotar al software malicioso. Citó 10 millones como su estimación para el LOC encontrado en los productos defensivos modernos. Para hacer los cálculos más fáciles, imagino que hay productos con al menos 12,5 millones de líneas de código, lo que eleva la relación de LOC ofensivo a LOC defensivo al nivel de 100.000 a 1. En otras palabras, por cada 1 LOC de potencia de fuego ofensiva, los defensores escriben 100.000 LOC de bastión defensivo.

Mudge también comparó las LOC de malware con los sistemas operativos que esas muestras de malware están diseñadas para subvertir. Los analistas estiman que Windows XP se creó a partir de 45 millones de LOC, y nadie sabe cuántas LOC crearon Windows 7.

Mudge citó 150 millones como cifra para los sistemas operativos modernos, presumiblemente pensando en las últimas versiones de Windows. Reduzcamos esa cifra a 125 millones para simplificar las matemáticas y obtenemos una relación de 1 millón a 1 entre el tamaño del sistema operativo objetivo y el tamaño del arma maliciosa capaz de abusar de él.

Detengámonos un momento para resumir la perspectiva que ha producido nuestro ejercicio de conteo LOC:

120:1 Stuxnet a malware promedio

500:1 Editor de texto simple a malware promedio

2000:1 Paquete de malware a malware promedio

Herramienta defensiva de 100 000:1 para combatir el malware promedio

1.000.000:1 Sistema operativo de destino para malware promedio

Desde el punto de vista de un defensor, las proporciones de herramientas defensivas y sistemas operativos objetivo en relación con las muestras de malware promedio parecen bastante sombrías. ¡Incluso cambiar el tamaño de la suite de malware por el tamaño promedio no parece mejorar mucho la situación del defensor! Parece que los defensores (y sus proveedores) dedican mucho esfuerzo a producir miles de LOC, solo para ver cómo los intrusos hábiles y ágiles que tienen mucho menos LOC las destruyen.

¿Qué puede hacer un defensor? La respuesta es tomar ejemplo de lo que hace cualquier líder que no tiene suficientes recursos: redefinir un "obstáculo" como una "oportunidad". Olvídense del tamaño de las herramientas defensivas y de los sistemas operativos de destino: no hay mucho que pueda hacer al respecto. Alégrese del hecho de que las muestras de malware son tan pequeñas (relativamente hablando) como lo son.

2. <http://git.gnome.org/browse/gedit/tree/gedit?id=3.3.1>

Imaginemos que intentamos entender cómo funciona una herramienta de defensa a nivel de código fuente, donde están esperando esos 12,5 millones de LOC. Es una tarea abrumadora, aunque algunos investigadores se asignan este tipo de proyectos favoritos. Para un ejemplo increíble, lea "Sophail: A Critical Analysis of Sophos Antivirus" de Tavis Ormandy,³ también presentado en Black Hat Las Vegas en 2011. Este tipo de análisis gigantesco es la excepción y no la regla.

En lugar de preocuparse por millones de LOC (o cientos o decenas de miles), concéntrese en el área de mil o menos, el lugar donde se puede encontrar una parte significativa del malware del mundo. Como defensor, su objetivo principal con respecto al malware es determinar qué hace, cómo se manifiesta en su entorno y qué hacer al respecto. Al tratar con muestras de tamaño razonable y las habilidades adecuadas, tiene la oportunidad de responder a estas preguntas y, por lo tanto, reducir el riesgo para su empresa.

Si los autores de malware están dispuestos a proporcionar las muestras, los autores del libro que estás leyendo están aquí para proporcionarte las habilidades. Practical Malware Analysis es el tipo de libro que creo que todo analista de malware debería tener a mano.

Si eres principiante, leerás el material introductorio y práctico que necesitas para entrar en acción. Si eres un practicante intermedio, te llevará al siguiente nivel. Si eres un ingeniero avanzado, encontrarás esas gemas adicionales que te impulsarán aún más alto, y podrás decir "lee este excelente manual" cuando te hagan preguntas las personas a las que asesoras.

Practical Malware Analysis es en realidad dos libros en uno: primero, es un texto que muestra a los lectores cómo analizar el malware moderno. Podrías haber comprado el libro solo por ese motivo y haberte beneficiado enormemente de sus instrucciones. Sin embargo, los autores decidieron ir más allá y, básicamente, escribir un segundo libro. Este volumen adicional podría haberse llamado Análisis de malware aplicado y consta de ejercicios, respuestas breves e investigaciones detalladas que se presentan al final de cada capítulo y en el Apéndice C. Los autores también escribieron todo el malware que utilizan como ejemplos, lo que garantiza un entorno rico y seguro para el aprendizaje.

Por lo tanto, en lugar de desesperarse por las aparentes asimetrías que enfrenta la era digital, Los defensores de los derechos de autor deben estar contentos de que el malware en cuestión adopte la forma que tiene actualmente. Si cuentan con libros como Practical Malware Analysis, tendrán la ventaja que necesitan para detectar y responder mejor a las intrusiones en su empresa o en la de sus clientes. Los autores son expertos en estos ámbitos y encontrarán consejos extraídos de las primeras líneas, no teorizados en un laboratorio de investigación aislado. Disfruten de la lectura de este libro y sepan que cada pieza de malware que analicen y analicen aumenta los costos del oponente al exponer sus artes oscuras a la luz del conocimiento.

Richard Bejtlich (@taosecurity)

Director de seguridad de Mandiant y fundador de TaoSecurity

Parque Manassas, Virginia

2 de enero de 2012

3. <http://dl.packetstormsecurity.net/papers/virus/Sophail.pdf>

EXPRESIONES DE GRATITUD

Gracias a Lindsey Lack, Nick Harbour y Jerrold “Jay” Smith por contribuir con capítulos en sus áreas de especialización. Gracias a nuestro revisor técnico Stephen Lawler, quien revisó por sí solo más de 50 laboratorios y todos nuestros capítulos. Gracias a Seth Summersett, William Ballenthin y Stephen Davis por contribuir con el código para este libro.

Un agradecimiento especial a todos los de No Starch Press por su esfuerzo. Alison, Bill, Travis y Tyler: nos alegró trabajar con ustedes y todos los demás en No Starch Press.

Agradecimientos individuales

Mike: Dedico este libro a Rebecca. No podría haberlo hecho sin tener una persona tan comprensiva y amorosa en mi vida.

Andy: Me gustaría agradecer a Molly, Claire y Eloise por ser la mejor familia que un hombre podría tener.

INTRODUCCIÓN

Suena el teléfono y los chicos de la red te dicen que te han hackeado y que están robando información confidencial de tus clientes de tu red. Comienzas tu investigación comprobando tus registros para identificar los hosts implicados. Analizas los hosts con antivirus.

El software busca el programa malicioso y tiene suerte cuando detecta un troyano llamado TROJ.snapAK. Borra el archivo para intentar limpiar el sistema y utiliza la captura de red para crear una firma de sistema de detección de intrusos (IDS) para asegurarse de que no haya otras máquinas infectadas. Luego tapas el agujero que crees que usaron los atacantes para entrar y te aseguras de que no vuelva a suceder.

Luego, varios días después, los chicos de la red vuelven y te dicen que están robando datos confidenciales de tu red. Parece el mismo ataque, pero no tienes idea de qué hacer. Claramente, tu firma IDS falló, porque hay más máquinas infectadas y tu software antivirus no está brindando suficiente protección para aislar la amenaza. Ahora la alta gerencia exige una explicación de lo que sucedió, y todo lo que puedes decirles sobre el malware es que era TROJ.snapAK. No tienes las respuestas a las preguntas más importantes y pareces un poco tonto.

¿Cómo se puede determinar exactamente qué hace TROJ.snapAK para poder eliminar la amenaza? ¿Cómo se puede escribir una firma de red más eficaz? ¿Cómo se puede averiguar si hay otras máquinas infectadas con este malware? ¿Cómo puede asegurarse de haber eliminado todo el paquete de malware y no solo una parte? ¿Cómo puede responder a las preguntas de la gerencia sobre lo que hace el programa malicioso?

Lo único que puedes hacer es decirle a tu jefe que necesitas contratar consultores externos costosos porque no puedes proteger tu propia red. Esa no es realmente la mejor manera de mantener tu trabajo seguro.

Ah, pero afortunadamente fuiste lo suficientemente inteligente como para conseguir una copia de Practical Análisis de malware. Las habilidades que aprenderá en este libro le enseñarán a responder esas preguntas difíciles y le mostrarán cómo proteger su red contra malware.

¿Qué es el análisis de malware?

El software malicioso, o malware, desempeña un papel en la mayoría de los incidentes de seguridad e intrusión informática. Cualquier software que haga algo que cause daño a un usuario, un ordenador o una red puede considerarse malware, incluidos los virus, los troyanos, los gusanos, los rootkits, el scareware y el spyware. Si bien las distintas encarnaciones de malware hacen todo tipo de cosas diferentes (como verá a lo largo de este libro), como analistas de malware, tenemos un conjunto básico de herramientas y técnicas a nuestra disposición para analizar malware.

El análisis de malware es el arte de diseccionar el malware para entender cómo funciona, cómo identificarlo y cómo derrotarlo o eliminarlo. Y no es necesario ser un hacker experto para realizar un análisis de malware.

Con millones de programas maliciosos circulando y cada día se detectan más, el análisis de malware es fundamental para cualquier persona que responda a incidentes de seguridad informática. Y, con la escasez de profesionales en análisis de malware, existe una gran demanda de analistas de malware capacitados.

Dicho esto, este no es un libro sobre cómo encontrar malware. Nuestro enfoque se centra en cómo analizar el malware una vez que se ha encontrado. Nos centramos en el malware que se encuentra en el sistema operativo Windows (el sistema operativo más común en uso en la actualidad), pero las habilidades que aprenda le serán de gran utilidad para analizar malware en cualquier sistema operativo. También nos centramos en los archivos ejecutables, ya que son los archivos más comunes y más difíciles con los que se encontrará. Al mismo tiempo, hemos optado por evitar hablar de scripts maliciosos y programas Java. En su lugar, profundizamos en los métodos utilizados para diseccionar amenazas avanzadas, como puertas traseras, malware encubierto y rootkits.

Prerrequisitos

Independientemente de sus antecedentes o experiencia con el análisis de malware, encontrará algo útil en este libro.

Los capítulos 1 a 3 tratan técnicas básicas de análisis de malware que incluso aquellos sin experiencia en seguridad o programación podrán utilizar para realizar la clasificación de malware. Los capítulos 4 a 14 cubren técnicas más intermedias.

Material que le proporcionará las principales herramientas y habilidades necesarias para analizar la mayoría de los programas maliciosos. Estos capítulos requieren algunos conocimientos de programación. El material más avanzado de los capítulos 15 a 19 será útil incluso para analistas de malware experimentados, ya que cubre estrategias y técnicas para analizar incluso los programas maliciosos más sofisticados, como los programas que utilizan técnicas antidesensamblado, antidepuración o empaquetado.

Este libro le enseñará cómo y cuándo utilizar diversos análisis de malware. Técnicas. Comprender cuándo utilizar una técnica en particular puede ser tan importante como conocer la técnica, ya que utilizar la técnica incorrecta en la situación incorrecta puede ser una pérdida de tiempo frustrante. No cubrimos todas las herramientas, porque las herramientas cambian todo el tiempo y lo importante son las habilidades básicas. Además, utilizamos muestras de malware realistas a lo largo del libro (que puede descargar desde <http://www.practicalmalwareanalysis.com/> o <http://www.nostarch.com/malware.htm>) para exponerlo a los tipos de cosas que verá al analizar malware del mundo real.

Aprendizaje práctico y directo

Nuestra amplia experiencia en la enseñanza de ingeniería inversa y análisis de malware nos ha enseñado que los estudiantes aprenden mejor cuando practican las habilidades que están aprendiendo. Hemos descubierto que la calidad de los laboratorios es tan importante como la calidad de la clase y, sin un componente de laboratorio, es casi imposible aprender a analizar malware.

Para ello, los ejercicios de laboratorio que se incluyen al final de la mayoría de los capítulos le permiten practicar las habilidades enseñadas en ese capítulo. Estos laboratorios lo desafían con malware realista diseñado para demostrar los tipos de comportamiento más comunes que encontrará en el malware del mundo real. Los laboratorios están diseñados para reforzar los conceptos enseñados en el capítulo sin abrumarlo con información no relacionada. Cada laboratorio incluye uno o más archivos maliciosos (que se pueden descargar desde <http://www.practicalmalwareanalysis.com/> o <http://www.nostarch.com/malware.htm>), algunas preguntas para guiarlo a través del laboratorio, respuestas breves a las preguntas y un análisis detallado del malware.

Los laboratorios están pensados para simular escenarios realistas de análisis de malware. Por ello, tienen nombres de archivo genéricos que no ofrecen ninguna información sobre la funcionalidad del malware. Al igual que con el malware real, comenzarás sin información y tendrás que usar las habilidades que has aprendido para reunir pistas y descubrir qué hace el malware.

La cantidad de tiempo requerido para cada laboratorio dependerá de su experiencia. Puede intentar completar el laboratorio usted mismo o seguir el análisis detallado para ver cómo se utilizan las distintas técnicas en la práctica.

La mayoría de los capítulos contienen tres prácticas. La primera práctica es generalmente la más fácil y la mayoría de los lectores deberían poder completarla. La segunda práctica está pensada para ser moderadamente difícil y la mayoría de los lectores necesitarán ayuda de las soluciones. La tercera práctica está pensada para ser difícil y solo los lectores más expertos podrán completarla sin la ayuda de las soluciones.

¿Qué hay en el libro?

El análisis práctico de malware comienza con métodos sencillos que se pueden utilizar para obtener información de programas maliciosos relativamente poco sofisticados y continúa con técnicas cada vez más complicadas que se pueden utilizar para abordar incluso los programas maliciosos más sofisticados. Esto es lo que encontrará en cada capítulo:

El Capítulo 0, “Introducción al análisis de malware”, establece el proceso general y la metodología del análisis de malware.

El Capítulo 1, “Técnicas estáticas básicas”, enseña formas de obtener información desde un ejecutable sin ejecutarlo.

El Capítulo 2, “Análisis de malware en máquinas virtuales”, lo guía en la configuración de máquinas virtuales para usarlas como un entorno seguro para ejecutar malware.

El Capítulo 3, “Análisis dinámico básico”, enseña técnicas fáciles de usar pero efectivas para analizar un programa malicioso ejecutándolo.

El Capítulo 4, “Un curso intensivo sobre ensamblaje x86”, es una introducción al lenguaje ensamblador x86, que proporciona una base para usar IDA Pro y realizar análisis en profundidad de malware.

El capítulo 5, “IDA Pro”, muestra cómo utilizar IDA Pro, una de las herramientas de análisis de malware más importantes. Utilizaremos IDA Pro en el resto del libro.

El Capítulo 6, “Reconocimiento de construcciones de código C en ensamblador”, proporciona ejemplos de código C en ensamblador y le enseña cómo comprender la funcionalidad de alto nivel del código de ensamblador.

El Capítulo 7, “Análisis de programas maliciosos de Windows”, cubre una amplia gama de conceptos específicos de Windows que son necesarios para comprender los programas maliciosos de Windows.

El Capítulo 8, “Depuración”, explica los conceptos básicos de la depuración y cómo Utilice un depurador para analistas de malware.

El Capítulo 9, “OllyDbg”, le muestra cómo utilizar OllyDbg, el depurador más popular entre los analistas de malware.

El Capítulo 10, “Depuración del kernel con WinDbg”, cubre cómo usar el depurador WinDbg para analizar malware y rootkits en modo kernel.

El Capítulo 11, “Comportamiento del malware”, describe la funcionalidad común del malware y le muestra cómo reconocer esa funcionalidad al analizar el malware.

El capítulo 12, “Lanzamiento de malware encubierto”, analiza cómo analizar una clase particularmente sigilosa de programas maliciosos que ocultan su ejecución dentro de otro proceso.

El Capítulo 13, “Codificación de datos”, demuestra cómo el malware puede codificar datos para dificultar la identificación de sus actividades en el tráfico de la red o en el host víctima.

El Capítulo 14, "Firmas de red enfocadas en malware", le enseña cómo usar el análisis de malware para crear firmas de red que superen a las firmas creadas solo con el tráfico capturado.

El capítulo 15, "Anti-Desensamblaje", explica cómo algunos autores de malware diseñar su malware para que sea difícil de desmantelar y cómo reconocer y derrotar estas técnicas.

El capítulo 16, "Anti-depuración", describe los trucos que utilizan los autores de malware para dificultar la depuración de su código y cómo superar esos obstáculos.

El Capítulo 17, "Técnicas anti-máquinas virtuales", demuestra las técnicas que utiliza el malware para dificultar su análisis en una máquina virtual y cómo evitar esas técnicas.

El Capítulo 18, "Empaquetadores y desempaquetadores", enseña cómo el malware utiliza el empaquetado para ocultar su verdadero propósito y luego proporciona un enfoque paso a paso para desempaquetar programas empaquetados.

El capítulo 19, "Análisis de shellcode", explica qué es el shellcode y presenta consejos y trucos específicos para analizar shellcode malicioso.

El Capítulo 20, "Análisis de C++", le enseña cómo el código C++ se ve diferente una vez compilado y cómo realizar análisis de malware creado con C++.

El Capítulo 21, "Malware de 64 bits", analiza por qué los autores de malware pueden utilizar malware de 64 bits y lo que necesita saber sobre las diferencias entre x86 y x64.

El Apéndice A, "Funciones importantes de Windows", describe brevemente las funciones de Windows. funciones comúnmente utilizadas en malware.

El Apéndice B, "Herramientas para el análisis de malware", enumera herramientas útiles para el análisis de malware. analistas.

El Apéndice C, "Soluciones para los laboratorios", proporciona las soluciones para los laboratorios. incluido en los capítulos de todo el libro.

Nuestro objetivo a lo largo de este libro es brindarle las habilidades necesarias para analizar y derrotar a todo tipo de malware. Como verá, cubrimos una gran cantidad de material y utilizamos laboratorios para reforzar el material. Cuando haya terminado de leer este libro, habrá aprendido las habilidades que necesita para analizar cualquier malware, incluidas técnicas simples para analizar rápidamente malware común y técnicas complejas y sofisticadas para analizar incluso el malware más enigmático.

Empecemos.

0

MANUAL DE ANÁLISIS DE MALWARE

Antes de entrar en detalles sobre cómo analizar malware, debemos definir cierta terminología, cubrir los tipos comunes de malware y presentar los enfoques fundamentales para el análisis de malware. Cualquier software que Cualquier cosa que cause daño al usuario, al ordenador o a la red (como virus, troyanos, gusanos, rootkits, scareware y spyware) puede considerarse malware. Si bien el malware aparece en muchas formas diferentes, se utilizan técnicas comunes para analizarlo. La elección de la técnica que emplee dependerá de sus objetivos.

Los objetivos del análisis de malware

El objetivo del análisis de malware suele ser proporcionar la información necesaria para responder a una intrusión en la red. Normalmente, el objetivo será determinar exactamente qué ha ocurrido y asegurarse de haber localizado todos los equipos y archivos infectados. Al analizar un malware sospechoso, el objetivo será determinar exactamente qué puede hacer un binario sospechoso en particular, cómo detectarlo en la red y cómo medir y contener sus daños.

Una vez que identifique qué archivos requieren un análisis completo, es hora de desarrollar Firmas para detectar infecciones de malware en su red. Como aprenderá a lo largo de este libro, el análisis de malware se puede utilizar para desarrollar firmas basadas en host y en red.

Las firmas o indicadores basados en el host se utilizan para detectar código malicioso en los equipos de las víctimas. Estos indicadores suelen identificar archivos creados o modificados por el malware o cambios específicos que realiza en el registro. A diferencia de las firmas de antivirus, los indicadores de malware se centran en lo que el malware hace en un sistema, no en las características del malware en sí, lo que los hace más eficaces para detectar malware que cambia de forma o que se ha eliminado del disco duro.

Las firmas de red se utilizan para detectar códigos maliciosos mediante el control del tráfico de red. Las firmas de red se pueden crear sin análisis de malware, pero las firmas creadas con la ayuda de análisis de malware suelen ser mucho más efectivas, ya que ofrecen una mayor tasa de detección y menos falsos positivos.

Después de obtener las firmas, el objetivo final es averiguar exactamente cómo funciona el malware. Esta suele ser la pregunta más frecuente de los directivos, que quieren una explicación completa de una intrusión importante. Las técnicas detalladas que aprenderá en este libro le permitirán determinar el propósito y las capacidades de los programas maliciosos.

Técnicas de análisis de malware

En la mayoría de los casos, al realizar un análisis de malware, solo tendrá el ejecutable del malware, que no será legible para las personas. Para entenderlo, utilizará una variedad de herramientas y trucos, cada uno de los cuales revelará una pequeña cantidad de información. Necesitará utilizar una variedad de herramientas para ver el panorama completo.

Existen dos enfoques fundamentales para el análisis de malware: estático y dinámico. El análisis estático implica examinar el malware sin ejecutarlo.

El análisis dinámico implica la ejecución del malware. Ambas técnicas se clasifican en básicas o avanzadas.

Análisis estático básico

El análisis estático básico consiste en examinar el archivo ejecutable sin ver las instrucciones reales. El análisis estático básico puede confirmar si un archivo es malicioso, brindar información sobre su funcionalidad y, en ocasiones, brindar información que le permitirá generar firmas de red simples. El análisis estático básico es sencillo y puede ser rápido, pero es en gran medida ineficaz contra malware sofisticado y puede pasar por alto comportamientos importantes.

Análisis dinámico básico

Las técnicas básicas de análisis dinámico implican ejecutar el malware y observar su comportamiento en el sistema para eliminar la infección, generar firmas efectivas o ambas cosas. Sin embargo, antes de poder ejecutar el malware de forma segura, debe configurar un entorno que le permita estudiar el malware en ejecución.

malware sin riesgo de dañar su sistema o red. Al igual que las técnicas básicas de análisis estático, las técnicas básicas de análisis dinámico pueden ser utilizadas por la mayoría de las personas sin conocimientos profundos de programación, pero no serán efectivas con todo el malware y pueden pasar por alto funciones importantes.

Análisis estático avanzado

El análisis estático avanzado consiste en aplicar ingeniería inversa a los componentes internos del malware cargando el ejecutable en un desensamblador y observando las instrucciones del programa para descubrir qué hace el programa. Las instrucciones las ejecuta la CPU, por lo que el análisis estático avanzado le indica exactamente qué hace el programa. Sin embargo, el análisis estático avanzado tiene una curva de aprendizaje más pronunciada que el análisis estático básico y requiere conocimientos especializados de desensamblado, construcciones de código y conceptos del sistema operativo Windows, todo lo cual aprenderá en este libro.

Análisis dinámico avanzado

El análisis dinámico avanzado utiliza un depurador para examinar el estado interno de un ejecutable malicioso en ejecución. Las técnicas de análisis dinámico avanzado proporcionan otra forma de extraer información detallada de un ejecutable. Estas técnicas son más útiles cuando se intenta obtener información que es difícil de recopilar con otras técnicas. En este libro, le mostraremos cómo utilizar el análisis dinámico avanzado junto con el análisis estático avanzado para analizar por completo el malware sospechoso.

Tipos de malware

Al realizar un análisis de malware, descubrirá que, a menudo, puede acelerar el análisis si hace conjeturas fundamentadas sobre lo que intenta hacer el malware y luego confirma esas hipótesis. Por supuesto, podrá hacer conjeturas más acertadas si conoce el tipo de cosas que suele hacer el malware. A tal efecto, a continuación se indican las categorías en las que se divide la mayoría del malware:

Puerta trasera Código malicioso que se instala en una computadora para permitir el acceso del atacante. Las puertas traseras generalmente permiten que el atacante se conecte a la computadora con poca o ninguna autenticación y ejecute comandos en el sistema local.

Botnet Similar a una puerta trasera, en el sentido de que permite al atacante acceder al sistema, pero todas las computadoras infectadas con la misma botnet reciben las mismas instrucciones de un único servidor de comando y control.

Descargador Código malicioso que existe únicamente para descargar otro código malicioso. Los atacantes suelen instalar los descargadores cuando obtienen acceso por primera vez a un sistema. El programa descargador descargará e instalará código malicioso adicional.

Malware que roba información Malware que recopila información del equipo de la víctima y, por lo general, la envía al atacante. Algunos ejemplos son los rastreadores, los capturadores de hash de contraseñas y los registradores de pulsaciones de teclas. Este malware se utiliza normalmente para obtener acceso a cuentas en línea, como cuentas de correo electrónico o de banca en línea.

Lanzador Programa malicioso utilizado para ejecutar otros programas maliciosos.

Por lo general, los lanzadores utilizan técnicas no tradicionales para lanzar otros programas maliciosos con el fin de garantizar el sigilo o un mayor acceso a un sistema.

Rootkit Código malicioso diseñado para ocultar la existencia de otro código. Los rootkits suelen combinarse con otro malware, como una puerta trasera, para permitir el acceso remoto al atacante y dificultar que la víctima detecte el código.

Scareware Malware diseñado para asustar a un usuario infectado para que compre algo.

Generalmente tiene una interfaz de usuario que lo hace parecer un antivirus u otro programa de seguridad. Informa a los usuarios que hay un código malicioso en su sistema y que la única forma de deshacerse de él es comprar su "software", cuando en realidad, el software que vende no hace nada más que eliminar el scareware.

Malware que envía spam Malware que infecta el equipo de un usuario y luego lo utiliza para enviar spam. Este malware genera ingresos para los atacantes al permitirles vender servicios de envío de spam.

Gusano o virus Código malicioso que puede copiarse a sí mismo e infectar computadoras adicionales.

El malware suele abarcar varias categorías. Por ejemplo, un programa puede tener un keylogger que recopila contraseñas y un componente de gusano que envía spam. No se obsesione demasiado con clasificar el malware según su funcionalidad.

El malware también se puede clasificar en función de si el objetivo del atacante es masivo o específico. El malware masivo, como el scareware, adopta el enfoque de escopeta y está diseñado para afectar a la mayor cantidad de máquinas posible. De los dos objetivos, es el más común y suele ser el menos sofisticado y más fácil de detectar y defenderse porque el software de seguridad lo tiene como objetivo.

El malware dirigido, como una puerta trasera única en su tipo, está diseñado para una organización específica. El malware dirigido es una amenaza mayor para las redes que el malware masivo, porque no está muy extendido y sus productos de seguridad probablemente no lo protegerán de él. Sin un análisis detallado del malware dirigido, es casi imposible proteger su red contra ese malware y eliminar las infecciones. El malware dirigido suele ser muy sofisticado y su análisis a menudo requerirá las habilidades de análisis avanzadas que se tratan en este libro.

Reglas generales para el análisis de malware

Terminaremos este manual con varias reglas a tener en cuenta al realizar el análisis.

En primer lugar, no te obsesiones demasiado con los detalles. La mayoría de los programas de malware son... Es muy grande y complejo, y no es posible comprender todos los detalles. Concéntrate en las características clave. Cuando se encuentre con secciones difíciles y complejas, intente obtener una visión general antes de quedarse atascado en los detalles.

En segundo lugar, recuerda que existen diferentes herramientas y enfoques para distintos trabajos. No existe un único enfoque. Cada situación es diferente y las distintas herramientas y técnicas que aprenderás tendrán funciones similares y, a veces, superpuestas. Si no tienes suerte con una herramienta, prueba con otra.

Si se queda atascado, no se demore demasiado en un solo problema; pase a otra cosa. Intente analizar el malware desde un ángulo diferente o simplemente pruebe un enfoque diferente.

Por último, recuerda que el análisis de malware es como un juego del gato y el ratón. Se desarrollan nuevas técnicas de análisis de malware y los autores de malware responden con nuevas técnicas para frustrar el análisis. Para tener éxito como analista de malware, debe poder reconocer, comprender y derrotar estas técnicas, y responder a los cambios en el arte del análisis de malware.

PARTE 1

ANÁLISIS BÁSICO

1

TECNICAS ESTATICAS BASICAS

Comenzamos nuestra exploración del análisis de malware con el análisis estático, que suele ser el primer paso para estudiar el malware. El análisis estático describe el proceso de analizar el código o la estructura de un programa para determinar su función. El programa en sí no se ejecuta en este momento. Por el contrario, cuando se realiza un análisis dinámico, el analista ejecuta el programa, como aprenderá en el Capítulo

En este capítulo se analizan varias formas de extraer información útil de ejecutables. En este capítulo, analizaremos las siguientes técnicas:

Utilizar herramientas antivirus para confirmar la malicia

Uso de hashes para identificar malware

Obtener información de las cadenas, funciones y encabezados de un archivo

Cada técnica puede proporcionar información diferente y las que utilices Depende de tus objetivos. Normalmente, utilizarás varias técnicas para recopilar la mayor cantidad de información posible.

Análisis antivirus: un primer paso útil

Al analizar por primera vez un posible malware, un buen primer paso es ejecutarlo en varios programas antivirus, que posiblemente ya lo hayan identificado.

Pero las herramientas antivirus no son perfectas, por supuesto. Se basan principalmente en una base de datos de fragmentos identificables de código sospechoso (firmas de archivos), así como en análisis de comportamiento y de coincidencia de patrones (heurística) para identificar archivos sospechosos. Un problema es que los creadores de malware pueden modificar fácilmente su código, cambiando así la firma de su programa y evadiendo los escáneres de virus. Además, el software antivirus a menudo no detecta malware poco común porque simplemente no está en la base de datos. Por último, la heurística, aunque suele ser eficaz para identificar códigos maliciosos desconocidos, puede ser burlada por malware nuevo y único.

Dado que los distintos programas antivirus utilizan diferentes firmas y heurísticas, resulta útil ejecutar varios programas antivirus diferentes contra el mismo malware sospechoso. Sitios web como VirusTotal (<http://www.virustotal.com/>) le permite cargar un archivo para que lo analicen varios motores antivirus. VirusTotal genera un informe que proporciona la cantidad total de motores que marcaron el archivo como malicioso, el nombre del malware y, si está disponible, información adicional sobre el malware.

Hashing: una huella digital para el malware

El hash es un método común que se utiliza para identificar de forma única el malware. El software malicioso se ejecuta a través de un programa de hash que produce un hash único que identifica ese malware (una especie de huella digital). La función hash Message-Digest Algorithm 5 (MD5) es la que se utiliza con más frecuencia para el análisis de malware, aunque el Secure Hash Algorithm 1 (SHA-1) también es popular.

Por ejemplo, usar el programa md5deep disponible gratuitamente para calcular el hash del programa Solitario que viene con Windows generaría el siguiente resultado:

```
C:\>md5deep c:\WINDOWS\system32\sol.exe  
373e7a863a1a345c60edb9e20ec32311 c:\WINDOWS\system32\sol.exe
```

El hash es 373e7a863a1a345c60edb9e20ec32311.

La calculadora WinMD5 basada en GUI, que se muestra en la Figura 1-1, puede calcular y mostrar hashes para varios archivos a la vez.

Una vez que tenga un hash único para un fragmento de malware, puede usarlo de la siguiente manera:

Utilice el hash como etiqueta.

Comparta ese hash con otros analistas para ayudarlos a identificar malware.

Busque ese hash en línea para ver si el archivo ya ha sido identificado.

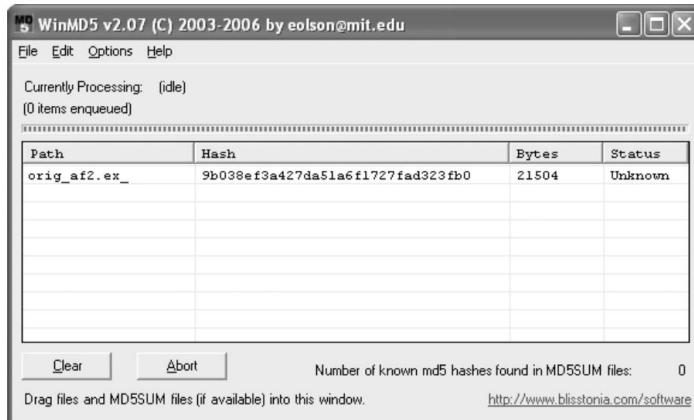


Figura 1-1: Salida de WinMD5

Encontrar cadenas

Una cadena en un programa es una secuencia de caracteres como “the”. Un programa contiene cadenas si imprime un mensaje, se conecta a una URL o copia un archivo a una ubicación específica.

La búsqueda en cadenas puede ser una forma sencilla de obtener pistas sobre la funcionalidad de un programa. Por ejemplo, si el programa accede a una URL, verá la URL a la que accedió almacenada como una cadena en el programa. Puede utilizar el programa Strings (<http://bit.ly/ic4pIL>) para buscar cadenas en un ejecutable, que normalmente se almacenan en formato ASCII o Unicode.

NOTA: Microsoft utiliza el término cadena de caracteres ancha para describir su implementación de cadenas Unicode, que varía ligeramente de los estándares Unicode. En este libro, cuando nos referimos a Unicode, nos referimos a la implementación de Microsoft.

Tanto los formatos ASCII como Unicode almacenan caracteres en secuencias que terminan con un terminador NULL para indicar que la cadena está completa. Las cadenas ASCII utilizan 1 byte por carácter, y Unicode utiliza 2 bytes por carácter.

La Figura 1-2 muestra la cadena BAD almacenada como ASCII. La cadena ASCII se almacena como los bytes 0x42, 0x41, 0x44 y 0x00, donde 0x42 es la representación ASCII de una letra mayúscula B, 0x41 representa la letra A, y así sucesivamente. El 0x00 al final es el terminador NULL.

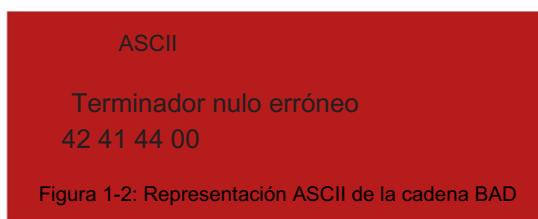


Figura 1-2: Representación ASCII de la cadena BAD

La Figura 1-3 muestra la cadena BAD almacenada como Unicode. La cadena Unicode se almacena como los bytes 0x42, 0x00, 0x41, etc. La B mayúscula se representa con los bytes 0x42 y 0x00, y el terminador NULL son dos bytes 0x00 seguidos.



Cuando Strings busca cadenas ASCII y Unicode en un ejecutable, ignora el contexto y el formato, de modo que puede analizar cualquier tipo de archivo y detectar cadenas en todo el archivo (aunque esto también significa que puede identificar bytes de caracteres como cadenas cuando no lo son). Strings busca una secuencia de tres letras o más de caracteres ASCII y Unicode, seguida de un carácter de terminación de cadena.

A veces, las cadenas detectadas por el programa Strings no son cadenas reales. Por ejemplo, si Strings encuentra la secuencia de bytes 0x56, 0x50, 0x33, 0x00, la interpretará como la cadena VP3. Pero es posible que esos bytes no representen realmente esa cadena; podrían ser una dirección de memoria, instrucciones de CPU o datos utilizados por el programa. Strings deja en manos del usuario la tarea de filtrar las cadenas no válidas.

Afortunadamente, la mayoría de las cadenas no válidas son obvias, porque no representan texto legítimo. Por ejemplo, el siguiente extracto muestra el resultado de ejecutar Strings en el archivo bp6.ex_:

```
C:>cadenas bp6.ex_
VP3
VW3
mierda
_____
99.124.22.1
mi-@
Obtener diseño
GDI32.DLL
Establecer diseño
MJC
¡La DLL del sistema de correo no es válido! Enviar correo no pudo enviar el mensaje.
```

En este ejemplo, las cadenas en negrita se pueden ignorar. Normalmente, si una cadena está corto y no corresponde a las palabras, probablemente no tenga sentido.

Por otra parte, las cadenas GetLayout en _____ y SetLayout en _____ son funciones de Windows utilizadas por la biblioteca de gráficos de Windows. Podemos identificarlas fácilmente como cadenas significativas porque los nombres de las funciones de Windows normalmente comienzan con una letra mayúscula y las palabras posteriores también comienzan con una letra mayúscula.

GDI32.DLL en _____ tiene sentido porque es el nombre de una biblioteca de vínculos dinámicos (DLL) común de Windows utilizada por programas de gráficos. (Los archivos DLL contienen código ejecutable que se comparte entre varias aplicaciones).

Como puedes imaginar, el número 99.124.22.1 en _____ es una dirección IP. Lo más probable es que sea uno que el malware utilizará de alguna manera.

Finalmente, en _____, ¡la DLL del sistema de correo no es válido! Enviar correo no pudo enviar el mensaje. es un mensaje de error. A menudo, la información más útil que se obtiene al ejecutar Strings se encuentra en los mensajes de error. Este mensaje en particular revela dos

Cosas: El malware en cuestión envía mensajes (probablemente a través del correo electrónico) depende de una DLL del sistema de correo. Esta información sugiere que podríamos querer comprobar los registros de correo electrónico en busca de tráfico sospechoso y que otra DLL (DLL del sistema de correo) podría estar asociada con este malware en particular. Tenga en cuenta que la DLL faltante en sí no es necesariamente maliciosa; el malware a menudo utiliza bibliotecas y DLL legítimas para lograr sus objetivos.

Malware empaquetado y ofuscado

Los creadores de malware suelen utilizar el empaquetamiento o la ofuscación para dificultar la detección o el análisis de sus archivos. Los programas ofuscados son aquellos cuya ejecución el creador del malware ha intentado ocultar. Los programas empaquetados son un subconjunto de los programas ofuscados en los que el programa malicioso está comprimido y no se puede analizar. Ambas técnicas limitarán gravemente sus intentos de analizar estáticamente el malware.

Los programas legítimos casi siempre incluyen muchas cadenas. El malware empaquetado u ofuscado contiene muy pocas cadenas. Si al buscar un programa con cadenas, descubre que solo tiene unas pocas cadenas, probablemente esté ofuscado o empaquetado, lo que sugiere que puede ser malicioso. Probablemente deba realizar más análisis estáticos para investigar más.

NOTA: El código empaquetado y ofuscado a menudo incluirá al menos las funciones LoadLibrary y GetProcAddress, que se utilizan para cargar y obtener acceso a funciones adicionales.

Archivos de embalaje

Cuando se ejecuta el programa empaquetado, también se ejecuta un pequeño programa contenedor para descomprimir el archivo empaquetado y luego ejecutar el archivo descomprimido, como se muestra en la Figura 1-4. Cuando se analiza estáticamente un programa empaquetado, solo se puede analizar el pequeño programa contenedor. (El Capítulo 18 analiza el empaquetado y desempaquetado con más detalle).



Figura 1-4: El archivo de la izquierda es el ejecutable original, con todas las cadenas, importaciones y demás información visible. A la derecha hay un ejecutable empaquetado. Todas las cadenas, importaciones y demás información del archivo empaquetado están comprimidas y son invisibles para la mayoría de las herramientas de análisis estático.

Detección de empacadores con PEiD

Una forma de detectar archivos empaquetados es con el programa PEiD. Puede utilizar PEiD para detectar el tipo de empacador o compilador empleado para crear una aplicación, lo que facilita enormemente el análisis del archivo empaquetado. La Figura 1-5 muestra información sobre el archivo orig_af2.exe según lo informado por PEiD.

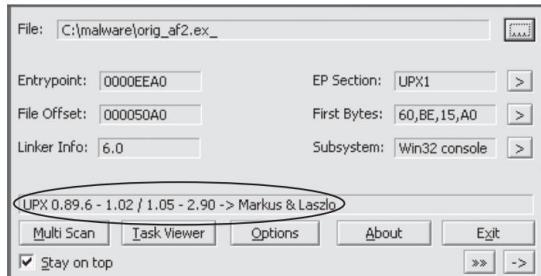


Figura 1-5: El programa PEiD

NOTA El desarrollo y soporte para PEiD se ha interrumpido desde abril de 2011, pero es

Sigue siendo la mejor herramienta disponible para la detección de empacadores y compiladores. En muchos casos, también identificará qué empacador se utilizó para empaquetar el archivo.

Como puede ver, PEiD ha identificado el archivo como empaquetado con la versión UPX 0.89.6-1.02 o 1.05-2.90. (Por ahora, ignore la otra información que se muestra aquí. Analizaremos este programa con más detalle en el Capítulo 18).

Cuando se empaqueta un programa, es necesario descomprimirlo para poder realizar cualquier análisis. El proceso de descompresión suele ser complejo y se explica en detalle en el Capítulo 18, pero el programa de compresión UPX es tan popular y fácil de usar para descomprimir que merece una mención especial aquí. Por ejemplo, para descomprimir malware empaquetado con UPX, simplemente se descargaría UPX (<http://upx.sourceforge.net/>) y ejecútelo de la siguiente manera, utilizando el programa empaquetado como entrada:

```
upx -d ProgramaEmpacado.exe
```

NOTA: Muchos complementos de PEiD ejecutarán el archivo ejecutable de malware sin previo aviso. (Consulte el Capítulo 2 para obtener información sobre cómo configurar un entorno seguro para ejecutar malware). Además, como todos los programas, especialmente los que se utilizan para el análisis de malware, PEiD puede estar sujeto a vulnerabilidades. Por ejemplo, la versión 0.92 de PEiD contenía un desbordamiento de búfer que permitía a un atacante ejecutar código arbitrario. Esto habría permitido que un escritor de malware astuto escribiera un programa para explotar la máquina del analista de malware. Asegúrese de utilizar la última versión de PEiD.

Formato de archivo ejecutable portátil

Hasta ahora hemos hablado de herramientas que escanean archivos ejecutables sin tener en cuenta su formato. Sin embargo, el formato de un archivo puede revelar mucho sobre la funcionalidad del programa.

El formato de archivo Portable Executable (PE) es utilizado por ejecutables, código objeto y DLL de Windows. El formato de archivo PE es una estructura de datos que contiene la información necesaria para que el cargador del sistema operativo Windows administre el código ejecutable empaquetado. Casi todos los archivos con código ejecutable que carga Windows están en formato de archivo PE, aunque algunos formatos de archivo heredados aparecen en raras ocasiones en malware.

Los archivos PE comienzan con un encabezado que incluye información sobre el código, el tipo de aplicación, las funciones de biblioteca requeridas y los requisitos de espacio.

La información del encabezado PE es de gran valor para el analista de malware.

Bibliotecas y funciones vinculadas

Una de las piezas de información más útiles que podemos obtener sobre un ejecutable es la lista de funciones que importa. Las importaciones son funciones utilizadas por un programa que en realidad están almacenadas en un programa diferente, como bibliotecas de código que contienen funcionalidades comunes a muchos programas. Las bibliotecas de código se pueden conectar al ejecutable principal mediante enlaces.

Los programadores vinculan las importaciones a sus programas para no tener que volver a implementar determinadas funciones en varios programas. Las bibliotecas de código se pueden vincular de forma estática, en tiempo de ejecución o de forma dinámica. Saber cómo se vincula el código de la biblioteca es fundamental para comprender el malware, ya que la información que podemos encontrar en el encabezado del archivo PE depende de cómo se haya vinculado el código de la biblioteca. En esta sección, analizaremos varias herramientas para ver las funciones importadas de un ejecutable.

Vinculación estática, en tiempo de ejecución y dinámica

El enlace estático es el método menos utilizado para enlazar bibliotecas, aunque es común en programas UNIX y Linux. Cuando una biblioteca está enlazada estáticamente a un ejecutable, todo el código de esa biblioteca se copia en el ejecutable, lo que hace que el ejecutable crezca en tamaño. Al analizar el código, es difícil diferenciar entre el código enlazado estáticamente y el código del propio ejecutable, porque nada en el encabezado del archivo PE indica que el archivo contiene código enlazado.

Si bien no es muy popular en programas amigables, la vinculación en tiempo de ejecución se usa comúnmente en malware, especialmente cuando está empaquetada u ofuscada. Los ejecutables que usan la vinculación en tiempo de ejecución se conectan a las bibliotecas solo cuando se necesita esa función, no al inicio del programa, como sucede con los programas vinculados dinámicamente.

Varias funciones de Microsoft Windows permiten a los programadores importar funciones vinculadas que no se encuentran en el encabezado de archivo de un programa. De estas, las dos más utilizadas son LoadLibrary y GetProcAddress. También se utilizan LdrGetProcAddress y LdrLoadDLL . LoadLibrary y GetProcAddress permiten que un programa acceda a cualquier función de cualquier biblioteca del sistema, lo que significa que cuando se utilizan estas funciones, no se puede saber de forma estática a qué funciones está vinculada el programa sospechoso.

De todos los métodos de vinculación, la vinculación dinámica es la más común y la más interesante para los analistas de malware. Cuando las bibliotecas se vinculan dinámicamente, el sistema operativo host busca las bibliotecas necesarias cuando se carga el programa. Cuando el programa llama a la función de la biblioteca vinculada, esa función se ejecuta dentro de la biblioteca.

El encabezado del archivo PE almacena información sobre cada biblioteca que se cargará y cada función que utilizará el programa. Las bibliotecas utilizadas y las funciones llamadas suelen ser las partes más importantes de un programa, y su identificación es especialmente importante, ya que nos permite adivinar lo que hace el programa. Por ejemplo, si un programa importa la función URLDownloadToFile, es posible que suponga que se conecta a Internet para descargar algún contenido que luego almacena en un archivo local.

Exploración de funciones vinculadas dinámicamente con Dependency Walker

El programa Dependency Walker (<http://www.dependencywalker.com/>), distribuido con algunas versiones de Microsoft Visual Studio y otros paquetes de desarrollo de Microsoft, enumera únicamente funciones vinculadas dinámicamente en un ejecutable.

La figura 1-6 muestra el análisis de Dependency Walker de SERVICES.EX_. El panel del extremo izquierdo en muestra el programa así como las DLL que se están importando, es decir, KERNEL32.DLL y WS2_32.DLL.

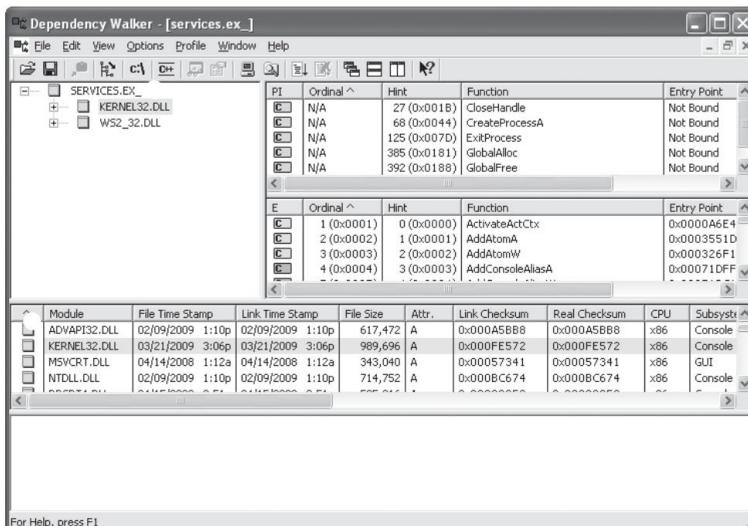


Figura 1-6: El programa Dependency Walker

Al hacer clic en KERNEL32.DLL se muestran las funciones importadas en el panel superior derecho en . Vemos varias funciones, pero la más interesante es CreateProcessA, que nos dice que el programa probablemente creará otro proceso y nos sugiere que, al ejecutar el programa, debemos estar atentos al lanzamiento de programas adicionales.

El panel central derecho en enumera todas las funciones de KERNEL32.DLL que se pueden importar (información que no nos resulta particularmente útil). Observe la columna en los paneles y etiquetada como Ordinal. Los ejecutables pueden importar funciones

por ordinal en lugar de nombre. Al importar una función por ordinal, el nombre de la función nunca aparece en el ejecutable original y puede resultar más difícil para un analista averiguar qué función se está utilizando. Cuando el malware importa una función por ordinal, puede averiguar qué función se está importando buscando el valor ordinal en el panel .

Los dos paneles inferiores (y) enumeran información adicional sobre las versiones de DLL que se cargarían si ejecutara el programa y cualquier error informado, respectivamente.

Las DLL de un programa pueden brindarle mucha información sobre su funcionalidad. Por ejemplo, En la Tabla 1-1 se enumeran las DLL comunes y lo que indican sobre una aplicación.

Tabla 1-1: DLL comunes

DLL	Descripción
Kernel32.dll	Esta es una DLL muy común que contiene funciones básicas, como acceso y manipulación de memoria, archivos y hardware.
Advapi32.dll	Esta DLL proporciona acceso a componentes avanzados del núcleo de Windows, como el Administrador de servicios y el Registro.
Usuario32.dll	Esta DLL contiene todos los componentes de la interfaz de usuario, como botones, barras de desplazamiento y componentes para controlar y responder a las acciones del usuario.
Gdi32.dll	Esta DLL contiene funciones para mostrar y manipular gráficos.
Ntdll.dll	Esta DLL es la interfaz con el núcleo de Windows. Los ejecutables generalmente no importan este archivo directamente, aunque siempre lo importa indirectamente Kernel32.dll. Si un ejecutable importa este archivo, significa que el autor pretendía utilizar una funcionalidad que normalmente no está disponible para los programas de Windows. Algunas tareas, como ocultar funcionalidad o manipular procesos, utilizarán esta interfaz.
WSock32.dll y Ws2_32.dll	Se trata de archivos DLL de red. Un programa que accede a cualquiera de ellos probablemente se conecte a una red o realice tareas relacionadas con la red.
Wininet.dll	Esta DLL contiene funciones de red de nivel superior que implementan protocolos como FTP, HTTP y NTP.

CONVENCIONES PARA NOMBRAR FUNCIONES

Al evaluar funciones de Windows que no conoce, vale la pena tener en cuenta algunas convenciones de nombres, ya que aparecen con frecuencia y pueden confundirlo si no las reconoce. Por ejemplo, a menudo encontrará nombres de funciones con un sufijo Ex , como CreateWindowEx. Cuando Microsoft actualiza una función y la nueva función es incompatible con la anterior, Microsoft continúa brindando soporte a la función anterior. La nueva función recibe el mismo nombre que la anterior, con un sufijo Ex agregado.

Las funciones que se han actualizado significativamente dos veces tienen dos sufijos Ex en sus nombres.

Muchas funciones que aceptan cadenas como parámetros incluyen una A o una W al final de sus nombres, como.CreateDirectoryW. Esta letra no aparece en la documentación de la función; simplemente indica que la función acepta un parámetro de cadena y que hay dos versiones diferentes de la función: una para cadenas ASCII y otra para cadenas de caracteres anchos. Recuerde omitir la A o la W final cuando busque la función en la documentación de Microsoft.

Funciones importadas

El encabezado del archivo PE también incluye información sobre funciones específicas utilizadas por un ejecutable. Los nombres de estas funciones de Windows pueden darle una buena idea sobre lo que hace el ejecutable. Microsoft hace un excelente trabajo documentando la API de Windows a través de la biblioteca Microsoft Developer Network (MSDN). (También encontrará una lista de funciones utilizadas comúnmente por malware en el Apéndice A).

Funciones exportadas

Al igual que las importaciones, las DLL y los EXE exportan funciones para interactuar con otros programas y códigos. Normalmente, una DLL implementa una o más funciones y las exporta para que las utilice un ejecutable que luego puede importarlas y usarlas.

El archivo PE contiene información sobre qué funciones exporta un archivo.

Dado que las DLL se implementan específicamente para proporcionar la funcionalidad que utilizan los archivos EXE, las funciones exportadas son las más comunes en las DLL. Los archivos EXE no están diseñados para proporcionar funcionalidad a otros archivos EXE y las funciones exportadas son poco frecuentes. Si descubre exportaciones en un ejecutable, a menudo le proporcionarán información útil.

En muchos casos, los autores de software nombran sus funciones exportadas de una manera que proporcione información útil. Una convención común es utilizar el nombre que se utiliza en la documentación de Microsoft. Por ejemplo, para ejecutar un programa como servicio, primero debe definir una función ServiceMain . La presencia de una función exportada llamada ServiceMain le indica que el malware se ejecuta como parte de un servicio.

Lamentablemente, aunque la documentación de Microsoft llama a esta función ServiceMain (y es habitual que los programadores hagan lo mismo), la función puede tener cualquier nombre. Por lo tanto, los nombres de las funciones exportadas tienen un uso limitado contra el malware sofisticado. Si el malware utiliza exportaciones, a menudo omitirá los nombres por completo o utilizará nombres poco claros o engañosos.

Puede ver la información de exportación utilizando el programa Dependency Walker. Se analiza en "Exploración de funciones vinculadas dinámicamente con Dependency Walker" en la página 16. Para obtener una lista de las funciones exportadas, haga clic en el nombre del archivo que desea examinar. Volviendo a la Figura 1-6, la ventana muestra todas las funciones exportadas de un archivo.

Análisis estático en la práctica

Ahora que comprende los conceptos básicos del análisis estático, examinemos algunos programas maliciosos reales. Analizaremos un posible keylogger y luego un programa empaquetado.

PotentialKeylogger.exe: un ejecutable descomprimido

La Tabla 1-2 muestra una lista abreviada de funciones importadas por PotentialKeylogger.exe, recopiladas mediante Dependency Walker. Como vemos tantas importaciones, podemos concluir de inmediato que este archivo no está empaquetado.

Tabla 1-2: Lista abreviada de DLL y funciones importadas desde PotentialKeylogger.exe

Kernel32.dll	Usuario32.dll	User32.dll (continuación)
CrearDirectorioW	InicioAplazarVentanaPos	Mostrar ventana
CrearArchivoW	Llamar a NextHookEx	Para UnicodeEx
Crear hilo	CrearDialogParamW	Menú emergente de seguimiento
EliminarArchivoW	CrearVentanaExW	Menú emergente de seguimiento
Proceso de salida	DefWindowProcW	Traducir mensaje
BuscarCerrar	Parámetro del cuadro de diálogo	DesengancharWindowsHookEx
BuscarPrimerArchivoW	Fin del diálogo	Anular registro de ClassW
BuscarSiguienteArchivoW	ObtenerMensajeW	Anular registro de fecha de acceso rápido
ObtenerLíneaDeComandoW	Obtener métricas del sistema	
Obtener proceso actual	ObtenerVentanaLongW	GDI32.dll
Obtener hilo actual	Obtener ventanarectángulo	Obtener objeto de stock
Obtener tamaño de archivo	ObtenerTextoDeVentanaW	Establecer modo Bk
Obtener identificador de módulo W	InvalidarRect	Establecer color de texto
Obtener montón de procesos	¿Está marcado el botón Dlg?	
Obtener nombre de ruta cortaW	¿Está habilitada la ventana?	Shell32.dll
Asignación de montón	CargarCursorW	Línea de comandos para ArgvW
Montón libre	Icono de cargaW	SHChangeNotificar
¿Está presente el depurador?	CargarMenuW	SHObtenerRutaDeCarpetasW
Vista de mapa del archivo	MapaVirtualKeyW	EjecutarExW de Shell
Proceso abierto	Puntos de la ventana del mapa	Ejecutar ShellW
Leer archivo	Cuadro de mensajesW	
EstablecerPunteroDeArchivo	RegistrarClaseExW	Advapi32.dll
Escribir archivo	RegistrarseHotKey	Clave de cierre de registro
	EnviarMensajeA	RegDeleteValueW
	Establecer datos del portapapeles	RegOpenUsuarioActual
	EstablecerDlgItemTextW	RegOpenKeyExW
	EstablecerTextoDeVentanaW	Valor de consulta de registro ExW
	EstablecerWindowsHookExW	Valor del conjunto de registros ExW

Como la mayoría de los programas de tamaño medio, este ejecutable contiene una gran cantidad de un gran número de funciones importadas. Lamentablemente, solo una pequeña minoría de esas funciones son particularmente interesantes para el análisis de malware. A lo largo de este libro, cubriremos las importaciones de software malicioso, centrándonos en las funciones más interesantes desde el punto de vista del análisis de malware.

Cuando no esté seguro de lo que hace una función, deberá buscarla.

Para ayudar a orientar su análisis, en el Apéndice A se enumeran muchas de las funciones de mayor interés para los analistas de malware. Si una función no aparece en el Apéndice A, búsqüela en MSDN en línea.

Como analista nuevo, pasará tiempo buscando muchas funciones que no son muy interesantes, pero rápidamente comenzará a aprender qué funciones podrían ser importantes y cuáles no. Para los fines de este ejemplo, le mostraremos una gran cantidad de importaciones que no son interesantes, para que pueda

familiarizarse con el análisis de una gran cantidad de datos y centrarse en algunos fragmentos clave de información.

Normalmente, no sabríamos que este malware es un keylogger potencial y tendríamos que buscar funciones que nos den pistas. Nos centraremos únicamente en las funciones que nos den pistas sobre la funcionalidad del programa.

Las importaciones de Kernel32.dll en la Tabla 1-2 nos indican que este software puede abrir y manipular procesos (como OpenProcess, GetCurrentProcess y GetProcessHeap) y archivos (como ReadFile, CreateFile y WriteFile). Las funciones FindFirstFile y FindNextFile son particularmente interesantes y podemos utilizarlas para buscar en directorios.

Las importaciones de User32.dll son aún más interesantes. La gran cantidad de funciones de manipulación de la interfaz gráfica de usuario (como RegisterClassEx, SetWindowText y ShowWindow) indica una alta probabilidad de que este programa tenga una interfaz gráfica de usuario (aunque la interfaz gráfica de usuario no necesariamente se muestra al usuario).

La función SetWindowsHookEx se utiliza habitualmente en programas espía y es la forma más popular en que los keyloggers reciben entradas del teclado. Esta función tiene algunos usos legítimos, pero si sospecha que se trata de un malware y ve esta función, probablemente esté buscando una función de keylogger.

También es interesante la función RegisterHotKey , que registra una combinación de teclas (como CTRL-SHIFT-P) de modo que, cada vez que el usuario presione esa combinación de teclas, se notifique a la aplicación. Independientemente de qué aplicación esté activa en ese momento, una tecla de acceso rápido llevará al usuario a esa aplicación.

Las importaciones de GDI32.dll están relacionadas con los gráficos y simplemente confirman que el programa probablemente tenga una interfaz gráfica de usuario. Las importaciones de Shell32.dll nos indican que este programa puede iniciar otros programas, una característica común tanto a los programas legítimos como a los programas maliciosos.

Las importaciones de Advapi32.dll nos indican que este programa utiliza el registro, lo que a su vez nos indica que debemos buscar cadenas que se parezcan a claves de registro. Las cadenas de registro se parecen mucho a directorios. En este caso, encontramos la cadena Software\Microsoft\Windows\CurrentVersion\Run, que es una clave de registro (que suele utilizar el malware) que controla qué programas se ejecutan automáticamente cuando se inicia Windows.

Este ejecutable también tiene varias exportaciones: LowLevelKeyboardProc y LowLevelMouseProc. La documentación de Microsoft dice: "El archivo LowLevelKeyboardProc El procedimiento de gancho es una función de devolución de llamada definida por la aplicación o la biblioteca que se utiliza con la función SetWindowsHookEx . En otras palabras, esta función se utiliza con SetWindowsHookEx para especificar qué función se llamará cuando se produzca un evento específico (en este caso, el evento de teclado de bajo nivel). La documentación de SetWindowsHookEx explica además que esta función se llamará cuando se produzcan determinados eventos de teclado de bajo nivel.

La documentación de Microsoft utiliza el nombre LowLevelKeyboardProc y el programador en este caso también lo hizo. Pudimos obtener información valiosa porque el programador no ocultó el nombre de una exportación.

Con la información obtenida a partir de un análisis estático de estas importaciones y exportaciones, podemos extraer algunas conclusiones importantes o formular algunas hipótesis sobre este malware. Por un lado, parece probable que se trate de un keylogger local que utiliza SetWindowsHookEx para registrar las pulsaciones del teclado.

Supongamos que tiene una interfaz gráfica de usuario que se muestra solo a un usuario específico y que la tecla de acceso rápido registrada con RegisterHotKey especifica la tecla de acceso rápido que el usuario malintencionado ingresa para ver la interfaz gráfica de usuario del keylogger y acceder a las pulsaciones de teclas registradas. Podemos especular más a partir de la función de registro y la existencia de Software\Microsoft\Windows\CurrentVersion\Run que este programa configura para cargarse al iniciar el sistema.

PackedProgram.exe: Un callejón sin salida

La Tabla 1-3 muestra una lista completa de las funciones importadas por un segundo programa malicioso desconocido. La brevedad de esta lista nos indica que este programa está empaquetado u ofuscado, lo que se confirma aún más por el hecho de que este programa no tiene cadenas legibles. Un compilador de Windows no crearía un programa que importe una cantidad tan pequeña de funciones; incluso un programa Hello, World tendría más.

Tabla 1-3: DLL y funciones importadas desde PackedProgram.exe

Kernel32.dll	Usuario32.dll
Obtener identificador de módulo A	Cuadro de mensaje A
CargarLibraryA	
Obtener dirección de proceso	
Proceso de salida	
Asignación virtual	
Virtual Gratis	

El hecho de que este programa esté empaquetado es una pieza valiosa de información, pero su naturaleza empaquetada también nos impide aprender más sobre el programa mediante el análisis estático básico. Necesitaremos probar técnicas de análisis más avanzadas, como el análisis dinámico (que se trata en el Capítulo 3) o el desempaquetado (que se trata en el Capítulo 18).

Los encabezados y secciones del archivo PE

Los encabezados de archivos PE pueden proporcionar considerablemente más información que solo las importaciones. El formato de archivo PE contiene un encabezado seguido de una serie de secciones. El encabezado contiene metadatos sobre el archivo en sí. Después del encabezado se encuentran las secciones reales del archivo, cada una de las cuales contiene información útil. A medida que avancemos en el libro, continuaremos analizando estrategias para ver la información en cada una de estas secciones. Las siguientes son las secciones más comunes e interesantes de un archivo PE:

.text La sección .text contiene las instrucciones que ejecuta la CPU. Todas las demás secciones almacenan datos e información complementaria. Generalmente, esta es la única sección que se puede ejecutar y debería ser la única sección que incluya código.

.rdata La sección .rdata normalmente contiene la información de importación y exportación, que es la misma información disponible desde ambas dependencias.

Walker y PEview. Esta sección también puede almacenar otros datos de solo lectura utilizados por el programa. A veces, un archivo contendrá una sección .idata y .edata , que almacenan la información de importación y exportación (consulte la Tabla 1-4).

.data La sección .data contiene los datos globales del programa, a los que se puede acceder desde cualquier parte del programa. Los datos locales no se almacenan en esta sección ni en ningún otro lugar del archivo PE. (Abordaremos este tema en el Capítulo 6).

.rsrc La sección .rsrc incluye los recursos utilizados por el ejecutable que no se consideran parte del ejecutable, como iconos, imágenes, menús y cadenas. Las cadenas se pueden almacenar en la sección .rsrc o en el programa principal, pero a menudo se almacenan en la sección .rsrc para compatibilidad con varios idiomas.

Los nombres de las secciones suelen ser consistentes en todo el compilador, pero pueden variar entre diferentes compiladores. Por ejemplo, Visual Studio utiliza .text para el código ejecutable, pero Borland Delphi utiliza CODE. A Windows no le importa el nombre real, ya que utiliza otra información en el encabezado PE para determinar cómo se utiliza una sección. Además, los nombres de las secciones a veces se ofuscan para dificultar el análisis. Afortunadamente, los nombres predeterminados se utilizan la mayor parte del tiempo. La Tabla 1-4 enumera los más comunes que encontrará.

Tabla 1-4: Secciones de un archivo PE para un ejecutable de Windows

Descripción del ejecutable	
.texto	Contiene el código ejecutable
.rdatos	Contiene datos de solo lectura a los que se puede acceder globalmente dentro del programa.
.datos	Almacena datos globales a los que se accede a través del programa.
.idata	A veces está presente y almacena la información de la función de importación; si esta sección no está presente, la información de la función de importación se almacena en la sección .rdata
.edata	A veces está presente y almacena la información de la función de exportación; si esta sección no está presente, la información de la función de exportación se almacena en la sección .rdata
.pdatos	Presente sólo en ejecutables de 64 bits y almacena información de manejo de excepciones.
.rsrc	Almacena los recursos que necesita el ejecutable.
.reubicación	Contiene información para la reubicación de archivos de la biblioteca.

Examinar archivos PE con PEview

El formato de archivo PE almacena información interesante en su encabezado. Podemos utilizar la herramienta PEview para explorar la información, como se muestra en la Figura 1-7.

En la figura, el panel izquierdo en muestra las partes principales de un encabezado PE. La entrada IMAGE_FILE_HEADER está resaltada porque está seleccionada actualmente.

Las dos primeras partes del encabezado PE: IMAGE_DOS_HEADER y MS-DOS Programa Stub—son históricos y no ofrecen información de particular interés a nosotros.

La siguiente sección del encabezado PE, IMAGE_NT_HEADERS, muestra los encabezados NT. La firma es siempre la misma y se puede ignorar.

La entrada IMAGE_FILE_HEADER , resaltada y mostrada en el panel derecho en , contiene información básica sobre el archivo. La marca de fecha y hora

La descripción en nos indica cuándo se compiló este ejecutable, lo que puede resultar muy útil en el análisis de malware y la respuesta a incidentes. Por ejemplo, un tiempo de compilación antiguo sugiere que se trata de un ataque anterior y los programas antivirus pueden contener firmas para el malware. Un tiempo de compilación nuevo sugiere lo contrario.

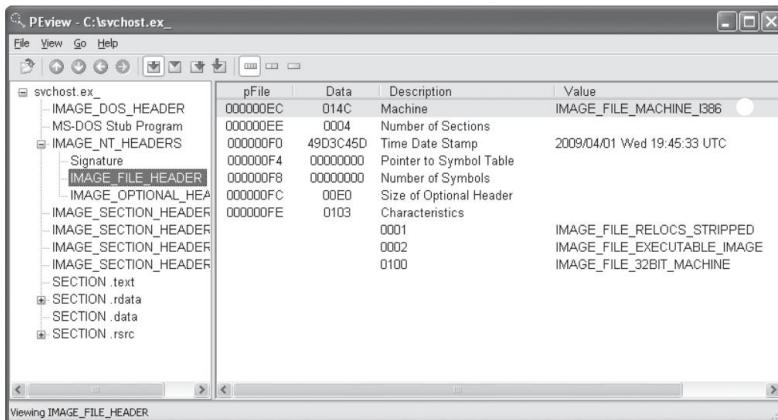


Figura 1-7: Visualización de IMAGE_FILE_HEADER en el programa PEview

Dicho esto, el tiempo de compilación es un poco problemático. Todos los programas Delphi utilizan un tiempo de compilación del 19 de junio de 1992. Si ves ese tiempo de compilación, probablemente estés viendo un programa Delphi y no sabrás realmente cuándo se compiló. Además, un escritor de malware competente puede falsificar fácilmente el tiempo de compilación. Si ves un tiempo de compilación que no tiene sentido, probablemente haya sido falsificado.

La sección IMAGE_OPTIONAL_HEADER incluye varios datos importantes. La descripción del subsistema indica si se trata de un programa de consola o de GUI. Los programas de consola tienen el valor IMAGE_SUBSYSTEM_WINDOWS_CUI y ejecutar dentro de una ventana de comandos. Los programas GUI tienen el valor IMAGE_SUBSYSTEM_WINDOWS_GUI y se ejecuta dentro del sistema Windows. También se utilizan subsistemas menos comunes, como Native o Xbox.

La información más interesante proviene de los encabezados de sección, que se encuentran en IMAGE_SECTION_HEADER, como se muestra en la Figura 1-8. Estos encabezados se utilizan para describir cada sección de un archivo PE. El compilador generalmente crea y nombra las secciones de un ejecutable, y el usuario tiene poco control sobre estos nombres. Como resultado, las secciones suelen ser consistentes de un ejecutable a otro (ver Tabla 1-4) y cualquier desviación puede ser sospechosa.

Por ejemplo, en la Figura 1-8, el tamaño virtual en nos indica cuánto espacio se asigna a una sección durante el proceso de carga. El tamaño de los datos sin procesar en Muestra el tamaño de la sección en el disco. Estos dos valores deberían ser iguales, ya que los datos deberían ocupar tanto espacio en el disco como en la memoria. Es normal que haya pequeñas diferencias, que se deben a diferencias entre la alineación en la memoria y en el disco.

Los tamaños de sección pueden ser útiles para detectar ejecutables empaquetados. Por ejemplo, si el tamaño virtual es mucho mayor que el tamaño de los datos sin procesar, sabrá que la sección ocupa más espacio en la memoria que en el disco. Esto suele ser indicativo de código compacto, en particular si la sección .text es más grande en la memoria que en el disco.

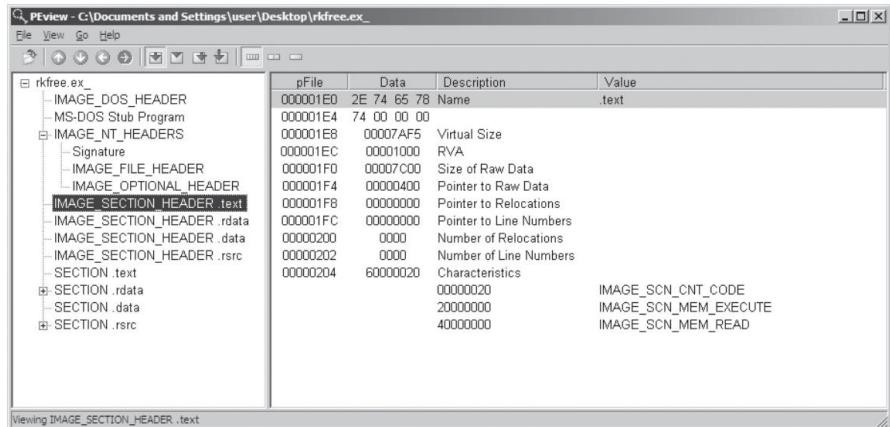


Figura 1-8: Visualización de la sección IMAGE_SECTION_HEADER .text en el programa PEview

La Tabla 1-5 muestra las secciones de PotentialKeylogger.exe. Como puede ver, las secciones .text, .rdata y .rsrc tienen un tamaño virtual y un tamaño de datos sin procesar de aproximadamente el mismo tamaño. La sección .data puede parecer sospechosa porque tiene un tamaño virtual mucho mayor que el tamaño de los datos sin procesar, pero esto es normal para la sección .data en los programas de Windows. Pero tenga en cuenta que esta información por sí sola no nos dice que el programa no sea malicioso; simplemente muestra que es probable que no esté empaquetado y que el encabezado del archivo PE fue generado por un compilador.

Tabla 1-5: Información de la sección para PotentialKeylogger.exe

Sección	Tamaño virtual	Tamaño de los datos sin procesar
.text	7AF5	7C00
.datos	17A0	0200
.rdatos	1AF5	1C00
.rsrc	72B8	7400

La Tabla 1-6 muestra las secciones de PackedProgram.exe. Las secciones de este archivo tienen varias anomalías: las secciones denominadas Dijfpds, .sdfuok y Kijjil son inusuales, y las secciones .text, .data y .rdata son sospechosas.

La sección .text tiene un valor de tamaño de datos sin procesar de 0, lo que significa que no ocupa espacio en el disco, y su valor de tamaño virtual es A000, lo que significa que se asignará espacio para el segmento .text. Esto nos indica que un empaquetador descomprimirá el código ejecutable en la sección .text asignada.

Tabla 1-6: Información de la sección PackedProgram.exe

Nombre	Tamaño virtual	Tamaño de los datos sin procesar
.text	A000	0000
.datos	3000	0000
.rdatos	4000	0000
.rsrc	19000	3400

Tabla 1-6: Información de la sección PackedProgram.exe (continuación)

Nombre	Tamaño virtual	Tamaño de los datos sin procesar
Dije 20000	0000	
.sdfuok34000	3313F	
Kijijl	1000	0200

Cómo ver la sección de recursos con Resource Hacker

Ahora que hemos terminado de ver el encabezado del archivo PE, podemos ver algunas de las secciones. La única sección que podemos examinar sin conocimientos adicionales de capítulos posteriores es la sección de recursos. Puede utilizar la herramienta gratuita Resource Hacker que se encuentra en <http://www.angusj.com/> para explorar el archivo .rsrc.

Sección. Cuando hace clic en los elementos de Resource Hacker, verá las cadenas, los íconos y los menús. Los menús que se muestran son idénticos a los que utiliza el programa. La Figura 1-9 muestra la pantalla de Resource Hacker para el programa Calculadora de Windows, calc.exe.

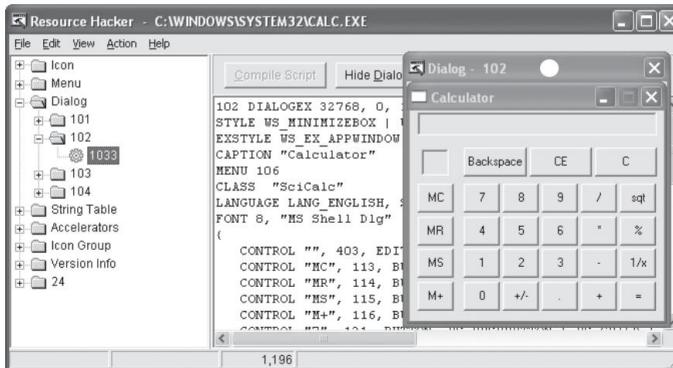


Figura 1-9: La pantalla de la herramienta Resource Hacker para calc.exe

El panel de la izquierda muestra todos los recursos incluidos en este ejecutable. Cada carpeta raíz que se muestra en el panel izquierdo en almacena un tipo diferente de recurso. Las secciones informativas para el análisis de malware incluyen:

La sección Icono enumera las imágenes que se muestran cuando el ejecutable está en una lista de archivos.

La sección Menú almacena todos los menús que aparecen en varias ventanas, como los menús Archivo, Editar y Ver. Esta sección contiene los nombres de todos los menús, así como el texto que se muestra para cada uno. Los nombres deberían darle una buena idea de su funcionalidad.

La sección Diálogo contiene los menús de diálogo del programa. El diálogo en muestra lo que el usuario verá al ejecutar calc.exe. Si no supiéramos nada más sobre calc.exe, podríamos identificarlo como un programa de calculadora simplemente mirando este menú de diálogo.

La sección Tabla de cadenas almacena cadenas.

La sección Información de la versión contiene un número de versión y, a menudo, el

nombre de la empresa y una declaración de derechos de autor.

La sección .rsrc que se muestra en la Figura 1-9 es típica de las aplicaciones de Windows. y puede incluir todo lo que requiera un programador.

NOTA El malware, y ocasionalmente el software legítimo, a menudo almacenan un programa incrustado o controlador aquí y, antes de que se ejecute el programa, extraen el ejecutable o controlador incorporado. Resource Hacker le permite extraer estos archivos para su análisis individual.

Uso de otras herramientas de archivos PE

Hay muchas otras herramientas disponibles para explorar un encabezado PE. Dos de las herramientas más útiles son PEBrowse Professional y PE Explorer.

PEBrowse Professional (<http://www.smidgeonsoft.prohosting.com/pebrowse-pro-file-viewer.html>) es similar a PEview. Permite ver los bytes de cada sección y muestra los datos analizados. PEBrowse Professional presenta mejor la información de la sección de recursos (.rsrc).

PE Explorer (<http://www.heaventools.com/>) tiene una interfaz gráfica de usuario muy completa que le permite navegar por las distintas partes del archivo PE. Puede editar ciertas partes del archivo PE y su editor de recursos incluido es ideal para explorar y editar los recursos del archivo. La principal desventaja de la herramienta es que no es gratuita.

Resumen del encabezado PE

El encabezado PE contiene información útil para el analista de malware y continuaremos examinándolo en los capítulos siguientes. La Tabla 1-7 analiza la información clave que se puede obtener de un encabezado PE.

Tabla 1-7: Información en el encabezado PE

Campo	Información revelada
Importaciones	Funciones de otras bibliotecas que utiliza el malware
Exportaciones	Funciones del malware que deben ser llamadas por otros programas o bibliotecas
Hora Fecha Sello Hora en que se compiló el programa	
Secciones	Nombres de las secciones del archivo y sus tamaños en el disco y en la memoria
Subsistema	Indica si el programa es una aplicación de línea de comandos o GUI
Recursos	Cadenas, iconos, menús y otra información incluida en el archivo

Conclusión

Con un conjunto de herramientas relativamente sencillas, podemos realizar un análisis estático del malware para obtener cierta información sobre su funcionamiento. Sin embargo, el análisis estático suele ser solo el primer paso y, por lo general, es necesario realizar más análisis. El siguiente paso es configurar un entorno seguro para poder ejecutar el malware y realizar un análisis dinámico básico, como verá en los dos próximos capítulos.

LABORATORIOS

El objetivo de los laboratorios es brindarle la oportunidad de practicar las habilidades enseñadas en el capítulo. Para simular un análisis realista de malware, se le brindará poca o ninguna información sobre el programa que está analizando. Al igual que todos los laboratorios de este libro, los archivos básicos de laboratorio de análisis estático tienen nombres genéricos para simular malware desconocido, que generalmente utilizan nombres sin sentido o engañosos.

Cada uno de los laboratorios consta de un archivo malicioso, algunas preguntas, respuestas breves a las preguntas y un análisis detallado del malware. Las soluciones de los laboratorios se incluyen en el Apéndice C.

Los laboratorios incluyen dos secciones de respuestas. La primera sección consta de respuestas breves, que se deben utilizar si usted mismo realizó el laboratorio y solo desea verificar su trabajo. La segunda sección incluye explicaciones detalladas para que pueda seguir nuestra solución y aprender cómo encontramos las respuestas a las preguntas planteadas en cada laboratorio.

Laboratorio 1-1

En este laboratorio se utilizan los archivos Lab01-01.exe y Lab01-01.dll. Utilice las herramientas y técnicas descritas en el capítulo para obtener información sobre los archivos y responder las preguntas que aparecen a continuación.

Preguntas

1. Sube los archivos a <http://www.VirusTotal.com/> y mira los informes.
¿Alguno de los archivos coincide con alguna firma antivirus existente?
2. ¿Cuándo se compilaron estos archivos?
3. ¿Hay algún indicio de que alguno de estos archivos esté comprimido u ofuscado?
Si es así, ¿cuáles son estos indicadores?
4. ¿Hay alguna importación que indique lo que hace este malware? Si es así, ¿qué importaciones?
¿Son ellos?
5. ¿Existen otros archivos o indicadores basados en el host que pueda buscar en los sistemas infectados?
6. ¿Qué indicadores basados en red podrían usarse para encontrar este malware en máquinas infectadas?
7. ¿Cuál crees que es el propósito de estos archivos?

Laboratorio 1-2

Analice el archivo Lab01-02.exe.

Preguntas

1. Sube el archivo Lab01-02.exe a <http://www.VirusTotal.com/>. ¿Coincide con alguna definición de antivirus existente?
2. ¿Existen indicios de que este archivo está comprimido u ofuscado? Si es así, ¿cuáles son esos indicadores? Si el archivo está comprimido, descomprímalo si es posible.
3. ¿Hay alguna importación que indique la funcionalidad de este programa? Si es así, ¿cuáles son y qué te indican?
4. ¿Qué indicadores basados en el host o en la red podrían usarse para identificar este malware en las máquinas infectadas?

Laboratorio 1-3

Analice el archivo Lab01-03.exe.

Preguntas

1. Sube el archivo Lab01-03.exe a <http://www.VirusTotal.com/>. ¿Coincide con alguna definición de antivirus existente?
2. ¿Existen indicios de que este archivo está comprimido u ofuscado? Si es así, ¿cuáles son esos indicadores? Si el archivo está comprimido, descomprímalo si es posible.
3. ¿Hay alguna importación que indique la funcionalidad de este programa? Si es así, ¿cuáles son y qué te indican?
4. ¿Qué indicadores basados en el host o en la red podrían usarse para identificar este malware en las máquinas infectadas?

Laboratorio 1-4

Analice el archivo Lab01-04.exe.

Preguntas

1. Sube el archivo Lab01-04.exe a <http://www.VirusTotal.com/>. ¿Coincide con alguna definición de antivirus existente?
2. ¿Existen indicios de que este archivo está comprimido u ofuscado? Si es así, ¿cuáles son esos indicadores? Si el archivo está comprimido, descomprímalo si es posible.
3. ¿Cuándo se compiló este programa?
4. ¿Hay alguna importación que indique la funcionalidad de este programa? Si es así, ¿cuáles son y qué te indican?
5. ¿Qué indicadores basados en el host o en la red podrían usarse para identificar este malware en las máquinas infectadas?
6. Este archivo tiene un recurso en la sección de recursos. Utilice Resource Hacker para examinar ese recurso y luego utilícelo para extraerlo. ¿Qué puede aprender del recurso?

2

ANÁLISIS DE MALWARE EN MÁQUINAS VIRTUALES

Antes de poder ejecutar malware para realizar análisis dinámicos, debe configurar un entorno seguro. El malware nuevo puede estar lleno de sorpresas y, si lo ejecuta en una máquina de producción, puede propagarse rápidamente a otras máquinas. máquinas de la red y será muy difícil eliminarlo. Un entorno seguro le permitirá investigar el malware sin exponer su máquina ni otras máquinas de la red a riesgos inesperados e innecesarios.

Puede utilizar máquinas físicas o virtuales dedicadas para estudiar el malware de forma segura. El malware se puede analizar utilizando máquinas físicas individuales en redes aisladas. Se trata de redes aisladas con máquinas que están desconectadas de Internet o de cualquier otra red para evitar que el malware se propague.

Las redes con espacios de aire permiten ejecutar malware en un entorno real sin poner en riesgo a otros equipos. Sin embargo, una desventaja de este escenario de prueba es la falta de conexión a Internet. Muchos programas maliciosos dependen de una conexión a Internet activa para actualizaciones, comandos y control, y otras funciones.

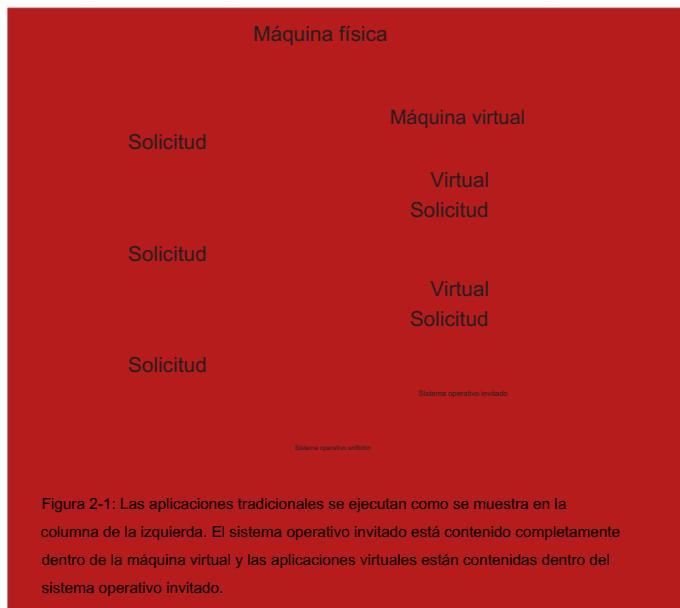
Otra desventaja de analizar el malware en equipos físicos en lugar de virtuales es que puede resultar difícil eliminarlo. Para evitar problemas, la mayoría de las personas que prueban el malware en equipos físicos utilizan una herramienta como Norton Ghost para administrar imágenes de respaldo de sus sistemas operativos (OS), que restauran en sus equipos después de haber completado el análisis.

La principal ventaja de utilizar máquinas físicas para el análisis de malware es A veces, el malware puede ejecutarse de forma diferente en las máquinas virtuales. Mientras analiza el malware en una máquina virtual, algunos malware pueden detectar que se están ejecutando en una máquina virtual y se comportarán de forma diferente para frustrar el análisis.

Debido a los riesgos y desventajas que conlleva el uso de máquinas físicas para analizar malware, las máquinas virtuales son las más utilizadas para el análisis dinámico. En este capítulo, nos centraremos en el uso de máquinas virtuales para el análisis de malware.

La estructura de una máquina virtual

Las máquinas virtuales son como una computadora dentro de otra computadora, como se ilustra en la Figura 2-1. Un sistema operativo invitado se instala dentro del sistema operativo host en una máquina virtual, y el sistema operativo que se ejecuta en la máquina virtual se mantiene aislado del sistema operativo host. El malware que se ejecuta en una máquina virtual no puede dañar el sistema operativo del host. Y si el malware daña la máquina virtual, simplemente puede reinstalar el sistema operativo en la máquina virtual o devolver la máquina virtual a un estado limpio.



VMware ofrece una serie popular de productos de virtualización de escritorio que se pueden utilizar para analizar malware en máquinas virtuales. VMware Player es gratuito y se puede utilizar para crear y ejecutar máquinas virtuales, pero carece de algunas funciones necesarias para un análisis eficaz de malware. VMware Workstation cuesta un poco menos de 200 dólares y, por lo general, es la mejor opción para el análisis de malware.

incluye características como la creación de instantáneas, que le permite guardar el estado actual de una máquina virtual, y la capacidad de clonar o copiar una máquina virtual existente.

Existen muchas alternativas a VMware, como Parallels, Microsoft Virtual PC, Microsoft Hyper-V y Xen. Estas varían en cuanto a compatibilidad y características con sistemas operativos host e invitados. Este libro se centrará en el uso de VMware para la virtualización, pero si prefiere otra herramienta de virtualización, este debate también le resultará relevante.

Creación de su máquina de análisis de malware

Por supuesto, antes de poder utilizar una máquina virtual para el análisis de malware, es necesario crear una. Este libro no trata específicamente sobre la virtualización, por lo que no le explicaremos todos los detalles. Cuando se le presenten opciones, su mejor opción, a menos que sepa que tiene requisitos diferentes, es elegir las configuraciones de hardware predeterminadas. Elija el tamaño del disco duro en función de sus necesidades.

VMware utiliza el espacio en disco de forma inteligente y redimensionará su disco virtual de forma dinámica en función de sus necesidades de almacenamiento. Por ejemplo, si crea un disco duro de 20 GB pero almacena solo 4 GB de datos en él, VMware reducirá el tamaño del disco duro virtual en consecuencia. Un tamaño de disco virtual de 20 GB suele ser un buen comienzo. Esa cantidad debería ser suficiente para almacenar el sistema operativo invitado y cualquier herramienta que pueda necesitar para el análisis de malware. VMware tomará muchas decisiones por usted y, en la mayoría de los casos, estas decisiones serán suficientes.

A continuación, instalará el sistema operativo y las aplicaciones. La mayoría de los programas maliciosos y las herramientas de análisis de programas maliciosos funcionan en Windows, por lo que probablemente instalará Windows como su sistema operativo virtual. Al momento de escribir este artículo, Windows XP sigue siendo el sistema operativo más popular (sorprendentemente) y el objetivo de la mayoría de los programas maliciosos. Centraremos nuestras exploraciones en Windows XP.

Después de haber instalado el sistema operativo, puede instalar cualquier aplicación necesaria. Siempre puedes instalar aplicaciones más tarde, pero normalmente es más fácil si configuras todo a la vez. El Apéndice B contiene una lista de aplicaciones útiles para el análisis de malware.

A continuación, instalará VMware Tools. En el menú VMware, seleccione VM Instale VMware Tools para comenzar la instalación. VMware Tools mejora la experiencia del usuario al hacer que el mouse y el teclado respondan mejor. También permite el acceso a carpetas compartidas, la transferencia de archivos mediante arrastrar y soltar, y otras funciones útiles que analizaremos en este capítulo.

Después de haber instalado VMware, es hora de realizar algunas configuraciones.

Configuración de VMware

La mayoría de los programas maliciosos incluyen funciones de red. Por ejemplo, un gusano realizará ataques de red contra otras máquinas en un intento de propagarse. Pero no conviene permitir que un gusano acceda a su propia red, ya que podría propagarse a otras computadoras.

Al analizar malware, probablemente desee observar la actividad de red del malware para comprender la intención del autor, crear firmas o probar el programa por completo. VMware ofrece varias opciones de red para redes virtuales, como se muestra en la Figura 2-2 y se analiza en las siguientes secciones.

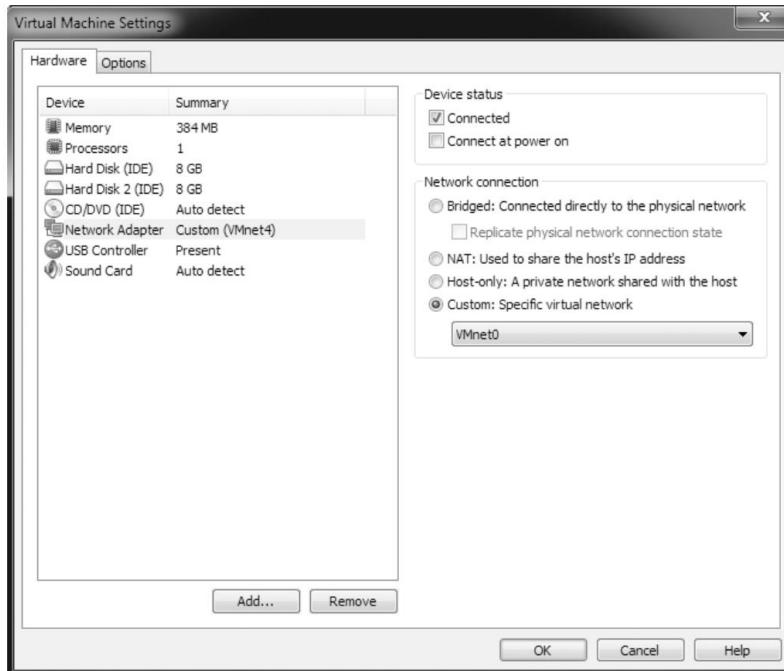


Figura 2-2: Opciones de configuración de red virtual para un adaptador de red

Desconectar la red

Aunque puede configurar una máquina virtual para que no tenga conectividad de red, normalmente no es una buena idea desconectar la red. Hacerlo será útil solo en ciertos casos. Sin conectividad de red, no podrá analizar la actividad maliciosa de la red.

Aun así, si tiene motivos para desconectar la red en VMware, puede hacerlo quitando el adaptador de red de la máquina virtual o desconectando el adaptador de red de la red eligiendo VM → Dispositivos extraíbles.

También puede controlar si un adaptador de red se conecta automáticamente cuando se enciende la máquina marcando la casilla Conectar al encender.

casilla de verificación (ver Figura 2-2).

Configuración de redes de solo host

La red de solo host, una función que crea una LAN privada separada entre el sistema operativo host y el sistema operativo invitado, se utiliza comúnmente para el análisis de malware. Una LAN de solo host no está conectada a Internet, lo que significa que el malware está contenido dentro de su máquina virtual, pero se le permite cierta conectividad de red.

NOTA: Al configurar el equipo host, asegúrese de que esté completamente actualizado, como protección en caso de que el malware que está probando intente propagarse. Es una buena idea configurar un firewall restrictivo para el host desde la máquina virtual para ayudar a evitar que el malware se propague al host. El firewall de Microsoft que viene con Windows XP Service Pack 2 y versiones posteriores está bien documentado y proporciona suficiente protección. Sin embargo, incluso si los parches están actualizados, el malware podría propagarse mediante un exploit de día cero contra el sistema operativo host.

La figura 2-3 ilustra la configuración de red para redes de solo host.

Cuando se habilita la red solo para host, VMware crea un adaptador de red virtual en el host y en las máquinas virtuales, y conecta ambos sin tocar el adaptador de red físico del host. El adaptador de red físico del host sigue conectado a Internet o a otra red externa.



Figura 2-3: Redes de solo host en VMware

Uso de múltiples máquinas virtuales

Una última configuración combina lo mejor de todas las opciones. Requiere varias máquinas virtuales conectadas por una LAN pero desconectadas de Internet y de la máquina host, de modo que el malware esté conectado a una red, pero la red no esté conectada a nada importante.

La Figura 2-4 muestra una configuración personalizada con dos máquinas virtuales conectadas entre sí. En esta configuración, una máquina virtual está configurada para analizar malware y la segunda máquina proporciona servicios. Las dos máquinas virtuales están conectadas al mismo conmutador virtual VMNet. En este caso, la máquina host sigue conectada a la red externa, pero no a la máquina que ejecuta el malware.



Figura 2-4: Redes personalizadas en VMware

Cuando utilice más de una máquina virtual para el análisis, le resultará útil combinar las máquinas como un equipo de máquinas virtuales. Cuando sus máquinas se unan como parte de un equipo de máquinas virtuales, podrá administrar sus configuraciones de energía y red en conjunto. Para crear un nuevo equipo de máquinas virtuales, seleccione Archivo Nuevo Equipo.

Cómo utilizar su máquina de análisis de malware

Para aprovechar al máximo la funcionalidad del malware en cuestión, debe simular todos los servicios de red de los que depende el malware. Por ejemplo, el malware normalmente se conecta a un servidor HTTP para descargar malware adicional. Para observar esta actividad, deberá otorgarle acceso al malware a un servidor del Sistema de nombres de dominio (DNS) para resolver la dirección IP del servidor, así como a un servidor HTTP para responder a las solicitudes. Con la configuración de red personalizada que acabamos de describir, la máquina que proporciona los servicios debería estar ejecutando los servicios necesarios para que el malware se comunique. (Analizaremos una variedad de herramientas útiles para simular servicios de red en el próximo capítulo).

Conexión de malware a Internet

En ocasiones, querrá conectar su equipo con malware a Internet para proporcionar un entorno de análisis más realista, a pesar de los riesgos obvios. El mayor riesgo, por supuesto, es que su equipo realice una actividad maliciosa, como propagar malware a otros hosts, convertirse en un nodo de un ataque de denegación de servicio distribuido o simplemente enviar spam. Otro riesgo es que el creador del malware pueda notar que se está conectando al servidor de malware e intentando analizarlo.

Nunca debe conectar malware a Internet sin antes realizar algún análisis para determinar qué podría hacer el malware al conectarse.

Entonces conéctese sólo si se siente cómodo con los riesgos.

La forma más común de conectar una máquina virtual a Internet mediante VMware es con un adaptador de red en puente, que permite que la máquina virtual se conecte a la misma interfaz de red que la máquina física. Otra forma de conectar malware que se ejecuta en una máquina virtual a Internet es utilizar el modo de traducción de direcciones de red (NAT) de VMware.

El modo NAT comparte la conexión IP del host a Internet. El host

Actúa como un enrutador y traduce todas las solicitudes de la máquina virtual para que provengan de la dirección IP del host. Este modo es útil cuando el host está conectado a la red, pero la configuración de la red dificulta, si no imposibilita, conectar el adaptador de la máquina virtual a la misma red.

Por ejemplo, si el host utiliza un adaptador inalámbrico, el modo NAT se puede utilizar fácilmente para conectar la máquina virtual a la red, incluso si la red inalámbrica tiene habilitado el acceso protegido Wi-Fi (WPA) o la privacidad equivalente por cable (WEP). O bien, si el adaptador del host está conectado a una red que solo permite la conexión de ciertos adaptadores de red, el modo NAT permite que la máquina virtual se conecte a través del host, evitando así la configuración de control de acceso de la red.

Conexión y desconexión de dispositivos periféricos

Los dispositivos periféricos, como los CD-ROM y las unidades de almacenamiento USB externas, plantean un problema particular para las máquinas virtuales. La mayoría de los dispositivos se pueden conectar a la máquina física o a la máquina virtual, pero no a ambas.

La interfaz de VMware le permite conectar y desconectar dispositivos externos a las máquinas virtuales. Si conecta un dispositivo USB a una máquina mientras la ventana de la máquina virtual está activa, VMware conectará el dispositivo USB al invitado y no al host, lo que puede resultar indeseable, considerando la creciente popularidad de los gusanos que se propagan a través de dispositivos de almacenamiento USB. Para modificar esta configuración, seleccione VM Configuration Controlador USB y desmarque la casilla Conectar automáticamente nuevos dispositivos USB para evitar que los dispositivos USB se conecten a la máquina virtual.

Tomando instantáneas

La toma de instantáneas es un concepto exclusivo de las máquinas virtuales. Las instantáneas de máquinas virtuales de VMware permiten guardar el estado actual de una computadora y volver a ese punto más tarde, de forma similar a un punto de restauración de Windows.

La línea de tiempo de la Figura 2-5 ilustra cómo funciona la toma de instantáneas. A las 8:00, toma una instantánea del equipo. Poco después, ejecuta la muestra de malware. A las 10:00, vuelve a la instantánea. El sistema operativo, el software y otros componentes del equipo vuelven al mismo estado en el que estaban a las 8:00, y todo lo que ocurrió entre las 8:00 y las 10:00 se borra como si nunca hubiera sucedido. Como puede ver, la toma de instantáneas es una herramienta extremadamente poderosa. Es como una función de deshacer integrada que le ahorra la molestia de tener que reinstalar el sistema operativo.



Después de haber instalado su sistema operativo y las herramientas de análisis de malware, y de haber Configuró la red y tomó una instantánea. Utilizó esa instantánea como base, una instantánea de borrón y cuenta nueva. A continuación, ejecutó el malware, completó el análisis y luego guardó los datos y volvió a la instantánea base para poder volver a hacerlo todo de nuevo.

Pero, ¿qué pasa si estás en medio de un análisis de malware y quieres hacerlo? ¿Quiere hacer algo diferente con su máquina virtual sin borrar todo su progreso? El Snapshot Manager de VMware le permite volver a cualquier instantánea en cualquier momento, sin importar qué instantáneas adicionales se hayan tomado desde entonces o qué haya sucedido con la máquina. Además, puede ramificarse sus instantáneas para que sigan rutas diferentes. Eche un vistazo al siguiente flujo de trabajo de ejemplo:

1. Mientras analiza la muestra de malware 1, se frustra y desea probar otra muestra.
2. Toma una instantánea del análisis de malware de la muestra 1.
3. Regresas a la imagen base.

4. Comienza a analizar la muestra de malware 2.
5. Toma una instantánea para tomar un descanso.

Cuando regrese a su máquina virtual, podrá acceder a cualquiera de las instantáneas en cualquier momento, como se muestra en la Figura 2-6. Los dos estados de la máquina son completamente independientes y puede guardar tantas instantáneas como espacio en disco tenga.

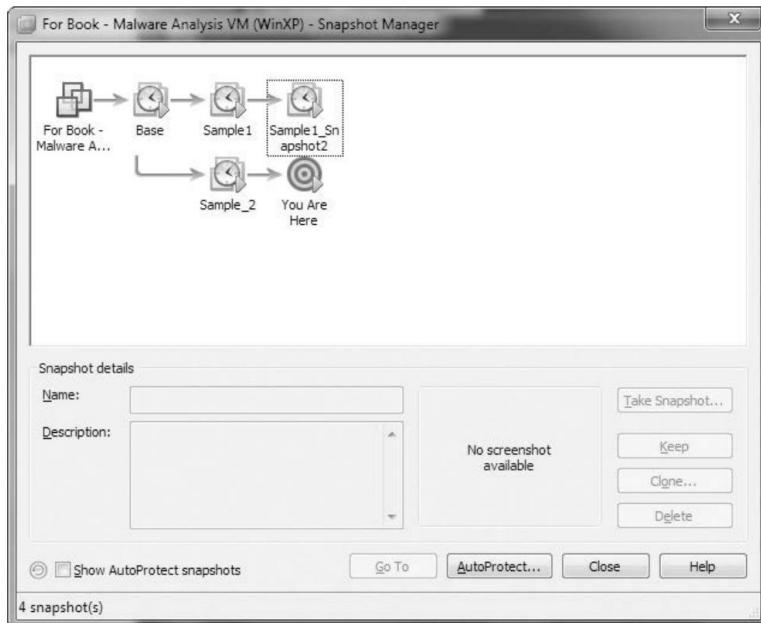


Figura 2-6: VMware Snapshot Manager

Transferencia de archivos desde una máquina virtual

Una desventaja de usar instantáneas es que cualquier trabajo realizado en la máquina virtual se pierde cuando se vuelve a una instantánea anterior. Sin embargo, puede guardar su trabajo antes de cargar la instantánea anterior transfiriendo los archivos que desea conservar al sistema operativo host mediante la función de arrastrar y soltar de VMware.

Siempre que VMware Tools esté instalado en el sistema operativo invitado y ambos sistemas ejecuten Windows, debería poder arrastrar y soltar un archivo directamente desde el sistema operativo invitado al sistema operativo host. Esta es la forma más sencilla y fácil de transferir archivos.

Otra forma de transferir datos es con las carpetas compartidas de VMware. Se puede acceder a una carpeta compartida tanto desde el sistema operativo host como desde el invitado, de forma similar a una carpeta compartida de Windows.

Los riesgos de utilizar VMware para el análisis de malware

Algunos programas maliciosos pueden detectar cuándo se están ejecutando dentro de una máquina virtual y se han publicado muchas técnicas para detectar precisamente esa situación. VMware no considera que esto sea una vulnerabilidad y no toma medidas explícitas para evitarlo.

detección, pero algunos programas maliciosos se ejecutan de manera diferente cuando se ejecutan en una máquina virtual, lo que dificulta la vida a los analistas de programas maliciosos. (El Capítulo 17 analiza estas técnicas anti-VMware con más detalle).

Y, como todo software, VMware ocasionalmente tiene vulnerabilidades. Estas pueden ser explotadas, provocando que el sistema operativo host se bloquee, o incluso se usen para ejecutar código en el sistema operativo host. Aunque solo existen unas pocas herramientas públicas o formas bien documentadas para explotar VMware, se han encontrado vulnerabilidades en la función de carpetas compartidas y se han publicado herramientas para explotar la funcionalidad de arrastrar y soltar. Asegúrese de mantener su versión de VMware completamente parcheada.

Y, por supuesto, incluso después de tomar todas las precauciones posibles, siempre existe algún riesgo cuando se analiza malware. Independientemente de lo que haga, e incluso si está ejecutando su análisis en una máquina virtual, debe evitar realizar análisis de malware en cualquier máquina crítica o sensible.

Grabar/Reproducir: Cómo ejecutar el ordenador en sentido inverso

Una de las funciones más interesantes de VMware es la de grabación y reproducción. Esta función de VMware Workstation graba todo lo que sucede para que puedas reproducir la grabación más tarde. La grabación ofrece una fidelidad del 100 %; cada instrucción que se ejecuta durante la grabación original se ejecuta durante una reproducción. Incluso si la grabación incluye una condición de carrera de una en un millón que no puedes replicar, se incluirá en la reproducción.

VMware también tiene una función de captura de películas que graba solo la salida de video, pero la función de grabación/reproducción ejecuta las instrucciones de la CPU del sistema operativo y los programas. Y, a diferencia de una película, puede interrumpir la ejecución en cualquier momento para interactuar con la computadora y realizar cambios en la máquina virtual. Por ejemplo, si comete un error en un programa que no tiene una función de deshacer, puede restaurar su máquina virtual al punto anterior a ese error para hacer algo diferente.

A medida que presentamos más herramientas a lo largo de este libro, examinaremos muchas maneras más potentes de utilizar la función de grabación y reproducción. Volveremos a esta función en el Capítulo 8.

Conclusión

La ejecución y el análisis de malware mediante VMware y máquinas virtuales implica los siguientes pasos:

1. Comience con una instantánea limpia sin malware ejecutándose en ella.
2. Transfiera el malware a la máquina virtual.
3. Realice su análisis en la máquina virtual.
4. Tome sus notas, capturas de pantalla y datos de la máquina virtual y transferirlo a la máquina física.
5. Revierta la máquina virtual a la instantánea limpia.

A medida que se lanzan nuevas herramientas de análisis de malware y se actualizan las herramientas existentes, deberá actualizar su imagen base limpia. Simplemente instale las herramientas y las actualizaciones y luego tome una nueva instantánea limpia.

Para analizar malware, normalmente es necesario ejecutar el malware para observar su Comportamiento. Al ejecutar malware, debe tener cuidado de no infectar su computadora o redes. VMware le permite ejecutar malware en un entorno seguro y controlable, y le proporciona las herramientas que necesita para limpiar el malware cuando haya terminado de analizarlo.

A lo largo de este libro, cuando hablamos de la ejecución de malware, asumimos que está ejecutando el malware dentro de una máquina virtual.

3

ANÁLISIS DINÁMICO BÁSICO

El análisis dinámico es cualquier examen que se realiza después de ejecutar un malware. Las técnicas de análisis dinámico son el segundo paso en el proceso de análisis de malware.

El análisis dinámico generalmente se realiza después de lo básico.

El análisis estático ha llegado a un callejón sin salida, ya sea por ofuscación, empaquetamiento o porque el analista ha agotado las técnicas de análisis estático disponibles.

Puede implicar monitorear el malware mientras se ejecuta o examinar el sistema después de que el malware se haya ejecutado.

A diferencia del análisis estático, el análisis dinámico permite observar la verdadera funcionalidad del malware, ya que, por ejemplo, la existencia de una cadena de acción en un binario no significa que la acción se ejecutará realmente. El análisis dinámico también es una forma eficaz de identificar la funcionalidad del malware. Por ejemplo, si el malware es un keylogger, el análisis dinámico puede permitirle localizar el archivo de registro del keylogger en el sistema, descubrir los tipos de registros que mantiene, descifrar a dónde envía su información, etc. Este tipo de información sería más difícil de obtener utilizando únicamente técnicas estáticas básicas.

Aunque las técnicas de análisis dinámico son extremadamente potentes, se deben realizar solo después de que se haya completado el análisis estático básico, ya que el análisis dinámico puede poner en riesgo su red y su sistema. Las técnicas dinámicas tienen sus limitaciones, ya que no todas las rutas de código pueden ejecutarse cuando se ejecuta un programa malicioso. Por ejemplo, en el caso de un programa malicioso de línea de comandos que requiere argumentos, cada argumento podría ejecutar una funcionalidad diferente del programa y, sin conocer las opciones, no podría examinar dinámicamente toda la funcionalidad del programa. Su mejor opción será utilizar técnicas dinámicas o estáticas avanzadas para averiguar cómo obligar al programa malicioso a ejecutar toda su funcionalidad. Este capítulo describe las técnicas básicas de análisis dinámico.

Sandboxes: el enfoque rápido y sencillo

Se pueden utilizar varios productos de software todo en uno para realizar análisis dinámicos básicos, y los más populares utilizan tecnología sandbox. Un sandbox es un mecanismo de seguridad para ejecutar programas no confiables en un entorno seguro sin temor a dañar sistemas "reales". Los sandboxes comprenden entornos virtualizados que a menudo simulan servicios de red de alguna manera para garantizar que el software o malware que se está probando funcionará con normalidad.

Uso de un Sandbox contra malware

Muchos entornos sandbox de malware, como Norman SandBox, GFI Sandbox, Anubis, Joe Sandbox, ThreatExpert, BitBlaze y Comodo Instant Malware Analysis, Analizará malware de forma gratuita. Actualmente, Norman SandBox y GFI Sandbox (anteriormente CWSandbox) son los más populares entre los profesionales de la seguridad informática.

Estos entornos aislados brindan resultados fáciles de entender y son excelentes para la clasificación inicial, siempre que esté dispuesto a enviar su malware a los sitios web de los entornos aislados. Si bien los entornos aislados están automatizados, puede optar por no enviar malware que contenga información de la empresa a un sitio web público.

NOTA: Puedes comprar herramientas sandbox para uso interno, pero son extremadamente caras.

En cambio, puede descubrir todo lo que estos entornos aislados pueden encontrar utilizando las técnicas básicas que se describen en este capítulo. Por supuesto, si tiene mucho malware para analizar, puede que valga la pena comprar un paquete de software de entornos aislados que se pueda configurar para procesar el malware rápidamente.

La mayoría de los entornos aislados funcionan de manera similar, por lo que nos centraremos en un ejemplo: GFI Sandbox. La Figura 3-1 muestra la tabla de contenidos de un informe en PDF generado al ejecutar un archivo a través del análisis automatizado de GFI Sandbox. El informe de malware incluye una variedad de detalles sobre el malware, como la actividad de red que realiza, los archivos que crea, los resultados del análisis con VirusTotal y pronto.



Figura 3-1: Resultados de muestra de GFI Sandbox para win32XYZ.exe

Los informes generados por GFI Sandbox varían en la cantidad de secciones que contienen, según los hallazgos del análisis. El informe de GFI Sandbox tiene seis secciones en la Figura 3-1, como se muestra a continuación:

- La sección Resumen del análisis enumera información del análisis estático y una descripción general de alto nivel de los resultados del análisis dinámico.
- La sección Actividad de archivo enumera los archivos que se abren, crean o eliminan. Cada proceso afectado por el malware.
- La sección Mutexes creados enumera los mutex creados por el malware.
- La sección Actividad del Registro enumera los cambios realizados al registro.
- La sección Actividad de red incluye la actividad de red generada por el malware, incluida la configuración de un puerto de escucha o la realización de una solicitud de DNS.
- La sección Resultados de VirusTotal enumera los resultados de un análisis de VirusTotal del software malicioso.

Desventajas del Sandbox

Los entornos aislados de malware tienen algunas desventajas importantes. Por ejemplo, el entorno aislado simplemente ejecuta el ejecutable, sin opciones de línea de comandos. Si el ejecutable de malware requiere opciones de línea de comandos, no ejecutará ningún código que se ejecute solo cuando se proporcione una opción. Además, si el malware en cuestión está esperando que se devuelva un paquete de comando y control antes de ejecutar una puerta trasera, esta no se ejecutará en el entorno aislado.

Es posible que el sandbox tampoco registre todos los eventos, porque ni usted ni el El sandbox puede esperar lo suficiente. Por ejemplo, si el malware está configurado para dormir durante un día antes de realizar una actividad maliciosa, es posible que no se dé cuenta de ese evento. (La mayoría de los sandboxes conectan la función Dormir y la configuran para que duerma solo brevemente, pero hay más de una forma de dormir y los sandboxes no pueden tener en cuenta todas ellas).

Otros posibles inconvenientes incluyen los siguientes:

El malware suele detectar cuándo se está ejecutando en una máquina virtual y, si se detecta una máquina virtual, el malware puede dejar de ejecutarse o comportarse de forma diferente. No todos los entornos aislados tienen en cuenta este problema.

Algunos programas maliciosos requieren la presencia de determinadas claves de registro o archivos en el sistema que podrían no encontrarse en el entorno protegido. Es posible que estos archivos sean necesarios para contener datos legítimos, como comandos o claves de cifrado.

Si el malware es una DLL, ciertas funciones exportadas no se invocarán correctamente, porque una DLL no se ejecutará tan fácilmente como un ejecutable.

Es posible que el sistema operativo del entorno sandbox no sea el adecuado para el malware. Por ejemplo, el malware podría bloquearse en Windows XP pero funcionar correctamente en Windows 7.

Un sandbox no puede decirle qué hace el malware. Puede informarle información básica Funcionalidad, pero no puede decirle que el malware es una utilidad de volcado de hash de Security Accounts Manager (SAM) personalizada o una puerta trasera de registro de teclas cifradas, por ejemplo. Esas son conclusiones que debe sacar por su cuenta.

Ejecución de malware

Las técnicas básicas de análisis dinámico no servirán de nada si no consigues que el malware se ejecute. Aquí nos centraremos en la ejecución de la mayoría de malware que encontrarás (EXE y DLL). Aunque normalmente te resultará bastante sencillo ejecutar malware ejecutable haciendo doble clic en el ejecutable o ejecutando el archivo desde la línea de comandos, puede resultar complicado ejecutar DLL maliciosas porque Windows no sabe cómo ejecutarlas automáticamente.

(Discutiremos los aspectos internos de las DLL en profundidad en el Capítulo 7).

Veamos cómo puede iniciar archivos DLL para tener éxito en el desempeño. formación de análisis dinámico.

El programa rundll32.exe está incluido en todas las versiones modernas de Windows. Proporciona un contenedor para ejecutar una DLL utilizando esta sintaxis:

```
C:\>rundll32.exe DLLname, Exportar argumentos
```

El valor de exportación debe ser un nombre de función o un ordinal seleccionado de la tabla de funciones exportadas en la DLL. Como aprendió en el Capítulo 1, puede utilizar una herramienta como PEview o PE Explorer para ver la tabla de exportación. Por ejemplo, el archivo rip.dll tiene las siguientes exportaciones:

Instalar
Desinstalar

La instalación parece ser una forma probable de ejecutar rip.dll, así que ejecutemos el malware de la siguiente manera:

```
C:\>rundll32.exe rip.dll, Instalar
```

El malware también puede tener funciones que se exportan por ordinal, es decir, como una función exportada con solo un número ordinal, que analizamos en profundidad en el Capítulo 1. En este caso, aún puede llamar a esas funciones con rundll32.exe utilizando el siguiente comando, donde 5 es el número ordinal que desea llamar, precedido por el carácter # :

```
C:\>rundll32.exe xyzzy.dll, #5
```

Debido a que las DLL maliciosas con frecuencia ejecutan la mayor parte de su código en DLLMain (llamado desde el punto de entrada de la DLL) y, dado que DLLMain se ejecuta siempre que se carga la DLL, a menudo puede obtener información de forma dinámica al forzar la carga de la DLL mediante rundll32.exe. De manera alternativa, puede incluso convertir una DLL en un ejecutable modificando el encabezado PE y cambiando su extensión para forzar a Windows a cargar la DLL como lo haría con un ejecutable.

Para modificar el encabezado PE, borre el indicador IMAGE_FILE_DLL (0x2000) del campo Características en IMAGE_FILE_HEADER. Si bien este cambio no ejecutará ninguna función importada, ejecutará el método DLLMain y puede provocar que el malware se bloquee o finalice inesperadamente. Sin embargo, siempre que sus cambios hagan que el malware ejecute su carga útil maliciosa y usted pueda recopilar información para su análisis, el resto no importa.

También puede ser necesario instalar el malware DLL como un servicio, a veces con una Exportación conveniente como InstallService, como se indica en ipr32x.dll:

```
C:\>rundll32 ipr32x.dll, InstalarServicioNombreServicio  
C:\>net start NombreServicio
```

Se debe proporcionar el argumento ServiceName al malware para que pueda instalarse y ejecutarse. El comando net start se utiliza para iniciar un servicio en un sistema Windows.

NOTA: Cuando vea una función ServiceMain sin una función exportada adecuada, como Install o InstallService, es posible que deba instalar el servicio manualmente. Puede hacerlo utilizando el comando sc de Windows o modificando el registro para un servicio no utilizado y luego utilizando net start en ese servicio. Las entradas de servicio se encuentran en el registro en HKLM\SYSTEM\CurrentControlSet\Services.

Monitoreo con Process Monitor

Process Monitor, o procmon, es una herramienta de monitoreo avanzada para Windows que brinda una manera de monitorear ciertas actividades del registro, sistema de archivos, red, procesos y subprocesos. Combina y mejora la funcionalidad de dos herramientas heredadas: FileMon y RegMon.

Aunque procmon captura una gran cantidad de datos, no captura todo. Por ejemplo, puede perderse la actividad del controlador de dispositivo de un componente en modo usuario que se comunica con un rootkit a través de los controles de E/S del dispositivo, así como ciertas llamadas de GUI, como SetWindowsHookEx. Aunque procmon puede ser una herramienta útil, por lo general no se debe utilizar para registrar la actividad de la red, ya que no funciona de manera uniforme en todas las versiones de Microsoft Windows.

ADVERTENCIA En este capítulo, utilizaremos herramientas para probar malware de forma dinámica. Cuando pruebe malware, asegúrese de proteger sus equipos y redes mediante una máquina virtual, como se explicó en el capítulo anterior.

Procmon monitorea todas las llamadas del sistema que puede recopilar tan pronto como se ejecuta. Debido a que existen muchas llamadas del sistema en una máquina Windows (a veces más de 50.000 eventos por minuto), normalmente es imposible revisarlas todas. Como resultado, debido a que procmon usa RAM para registrar eventos hasta que se le indica que deje de capturar, puede bloquear una máquina virtual que usa toda la memoria disponible. Para evitar esto, ejecute procmon durante períodos de tiempo limitados. Para evitar que procmon capture eventos, seleccione Archivo Capturar eventos. Antes de usar procmon para el análisis, primero borre todos los eventos capturados actualmente para eliminar los datos irrelevantes seleccionando Editar Borrar pantalla. A continuación, ejecute el malware en cuestión con la captura activada. Después de unos minutos, puede interrumpir la captura de eventos.

La pantalla Procmon

Procmon muestra columnas configurables que contienen información sobre eventos individuales, incluido el número de secuencia del evento, la marca de tiempo, el nombre del proceso que causa el evento, la operación del evento, la ruta utilizada por el evento y el resultado del evento. Esta información detallada puede ser demasiado larga para caber en la pantalla o puede ser difícil de leer por algún otro motivo. Si encuentra que alguno de estos es el caso, puede ver los detalles completos de un evento en particular haciendo doble clic en su fila.

La Figura 3-2 muestra una colección de eventos de procmon que ocurrieron en una máquina que ejecutaba un programa malicioso llamado mm32.exe. Al leer la columna Operación, podrá saber rápidamente qué operaciones realizó mm32.exe en este sistema, incluidos los accesos al registro y al sistema de archivos. Una entrada que vale la pena destacar es la creación de un archivo C:\Documents and Settings\All Users\Application Data\mw2mmgr.txt en el número de secuencia 212 utilizando CreateFile. La palabra SUCCESS en la columna Resultado se indica que esta operación fue exitosa.

Seq	Time	Process Name	Operation	Path	Result	Detail
200	1:55:31	mm32.exe	CloseFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	
201	1:55:31	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 11.776, Length: 1.024, I/O Flags
202	1:55:31	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 12.800, Length: 32.768, I/O Flags
203	1:55:31	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 1.024, Length: 9.216, I/O Flags
204	1:55:31	mm32.exe	ReadOpenKey	HKEY\Software\Microsoft\Windows NT\CurrentVersion\image File Exec	NAME NOT ...	Desired Access: Read
205	1:55:31	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 45.568, Length: 25.088, I/O Flags
206	1:55:31	mm32.exe	QueryOpen	Z:\Malware\imagehelp.dll	NOMBRE NO ...	
207	1:55:31	mm32.exe	QueryOpen	C:\WINDOWS\system32\imagehelp.dll	SUCCESS	CreationTime: 2/28/2006 8:00:00 AM,
208	1:55:31	mm32.exe	CreateFile	C:\WINDOWS\system32\imagehelp.dll	SUCCESS	Desired Access: Execute/Traverse, S
209	1:55:31	mm32.exe	CloseFile	C:\WINDOWS\system32\imagehelp.dll	SUCCESS	
210	1:55:31	mm32.exe	ReadOpenKey	HKEY\Software\Microsoft\Windows NT\CurrentVersion\image File Exec	NAME NOT ...	Desired Access: Read
211	1:55:31	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 10.240, Length: 1.536, I/O Flags
212	1:55:31	mm32.exe	CreateFile	C:\Documents and Settings\All Users\Application Data\mw2mmgr.txt	SUCCESS	Desired Access: Generic Write, Read
213	1:55:31	mm32.exe	ReadFile	C:\\$Directory	SUCCESS	Offset: 12.288, Length: 4.096, I/O Flags
214	1:55:31	mm32.exe	CreateFile	Z:\Malware\mm32.exe	SUCCESS	Desired Access: Generic Read, Disc
215	1:55:31	mm32.exe	ReadFile	Z:\Malware\mm32.exe	SUCCESS	Offset: 0, Length: 64

Figura 3-2: Ejemplo de Procmon mm32.exe

Filtrado en Procmon

No siempre es fácil encontrar información en procmon cuando se buscan miles de eventos, uno por uno. Ahí es donde la capacidad de filtrado de procmon es clave.

Puede configurar procmon para que filtre un ejecutable que se esté ejecutando en el sistema.

Esta función es especialmente útil para el análisis de malware, ya que permite configurar un filtro para el malware que se está ejecutando. También puede filtrar llamadas de sistema individuales, como RegSetValue, CreateFile, WriteFile u otras llamadas sospechosas o destructivas.

Cuando el filtrado de procmon está activado, filtra únicamente los eventos registrados. Todos los eventos registrados siguen estando disponibles aunque el filtro solo muestre una cantidad limitada de información. Configurar un filtro no es una forma de evitar que procmon consuma demasiada memoria.

Para configurar un filtro, seleccione Filtro → Filtro para abrir el menú Filtro, como se muestra en la imagen superior de la Figura 3-3. Al configurar un filtro, primero seleccione una columna para filtrar usando el cuadro desplegable en la parte superior izquierda, sobre el botón Restablecer. Los filtros más importantes para el análisis de malware son Nombre del proceso, Operación y Detalle. A continuación, seleccione un comparador, eligiendo entre opciones como Es, Contiene y Menor que. Por último, elija si se trata de un filtro que se incluirá o excluirá de la pantalla. Dado que, de forma predeterminada, la pantalla mostrará todas las llamadas del sistema, es importante reducir la cantidad que se muestra.

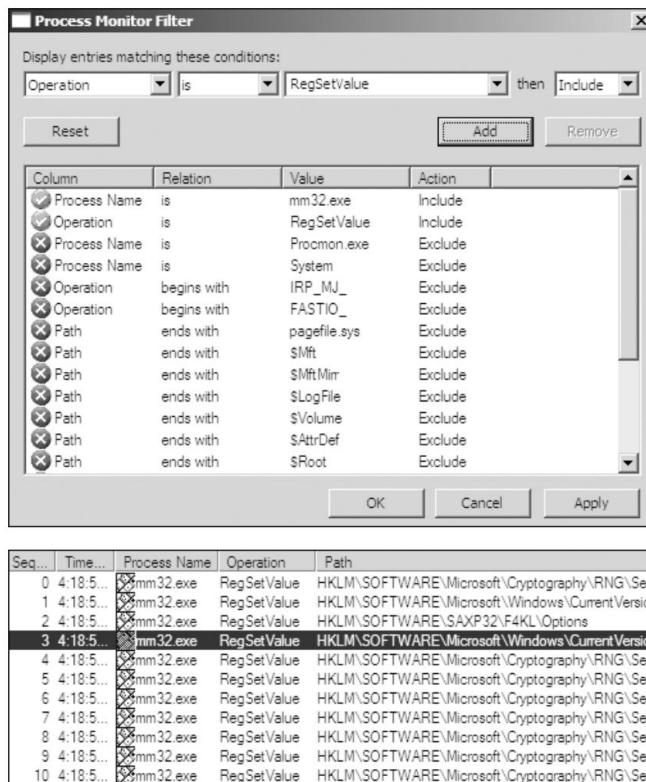


Figura 3-3: Configuración de un filtro procmon

NOTA: Procmon utiliza algunos filtros básicos de forma predeterminada. Por ejemplo, contiene un filtro que excluye procmon.exe y otro que excluye el archivo de paginación del registro, porque se accede a él con frecuencia y no proporciona información útil.

Como puede ver en las dos primeras filas de la Figura 3-3, estamos filtrando por Nombre de proceso y Operación. Hemos agregado un filtro por Nombre de proceso igual a mm32.exe que está activo cuando la Operación está configurada en RegSetValue.

Después de haber elegido un filtro, haga clic en Agregar para cada uno y, a continuación, haga clic en Aplicar. Como resultado de la aplicación de nuestros filtros, la ventana de visualización que se muestra en la imagen inferior muestra solo 11 de los 39.351 eventos, lo que nos permite ver con más facilidad que mm32.exe realizó un RegSetValue de la clave de registro HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Sys32V2Controller (número de secuencia 3 usando RegSetValue).

Al hacer doble clic en este evento RegSetValue , se revelarán los datos escritos en esta ubicación, que es la ruta actual al malware.

Si el malware extraído otro ejecutable y lo ejecutó, no se preocupe, porque esa información sigue estando allí. Recuerde que el filtro controla únicamente la visualización. Se capturan todas las llamadas del sistema que se produjeron cuando ejecutó el malware, incluidas las llamadas del sistema del malware extraído por el ejecutable original. Si ve algún malware extraído, cambie el filtro para que se muestre el nombre extraído y, a continuación, haga clic en Aplicar. Se mostrarán los eventos relacionados con el malware extraído.

Procmon ofrece filtros automáticos útiles en su barra de herramientas. Los cuatro filtros marcados con un círculo en la Figura 3-4 filtran por las siguientes categorías:

Registro Al examinar las operaciones del registro, puede saber cómo se instala un fragmento de malware en el registro.

Sistema de archivos Explorar la interacción del sistema de archivos puede mostrar todos los archivos que crea el malware o los archivos de configuración que utiliza.

Actividad del proceso Investigar la actividad del proceso puede indicarle si el malware generó procesos adicionales.

Red La identificación de las conexiones de red puede mostrarle cualquier puerto en el que el malware esté escuchando.

Los cuatro filtros están seleccionados de forma predeterminada. Para desactivar un filtro, simplemente haga clic en el botón Icono en la barra de herramientas correspondiente a la categoría.



Figura 3-4: Botones de filtro para procmon

NOTA: Si su malware se ejecuta en el momento del arranque, utilice las opciones de registro de arranque de procmon para instalar proc-mon como un controlador de inicio para capturar eventos de inicio.

El análisis de los eventos registrados de procmon requiere práctica y paciencia, ya que Muchos eventos son simplemente parte de la forma estándar en que se inician los ejecutables. Cuanto más utilice procmon, más fácil le resultará revisar rápidamente la lista de eventos.

Visualización de procesos con Process Explorer

El Process Explorer, gratuito de Microsoft, es un administrador de tareas extremadamente poderoso que debe ejecutarse cuando esté realizando un análisis dinámico.

Puede proporcionar información valiosa sobre los procesos que se ejecutan actualmente en un sistema.

Puede utilizar Process Explorer para enumerar los procesos activos, las DLL cargadas por un proceso, diversas propiedades del proceso e información general del sistema. También puede utilizarlo para eliminar un proceso, cerrar la sesión de los usuarios e iniciar y validar procesos.

La pantalla del explorador de procesos

Process Explorer supervisa los procesos que se ejecutan en un sistema y los muestra en una estructura de árbol que muestra las relaciones entre procesos secundarios y primarios. Por ejemplo, en la Figura 3-5 puede ver que services.exe es un proceso secundario de winlogon.exe, como lo indica la llave izquierda.

Process	PID	CPU	Description	Company Name
System Idle Process	0	96.97		
Interrupts	n/a		Hardware Interrupts	
DPCs	n/a		Deferred Procedure ...	
System	4			
smss.exe	580		Windows NT Session... Microsoft Corp...	
csrss.exe	652		Client Server Runtim... Microsoft Corp...	
{	684		Windows NT Logon ... Microsoft Corp...	
winlogon.exe	728	3.03	Services and Control... Microsoft Corp...	
services.exe	884		VMware Activation H... VMware, Inc.	
vmacthl.exe	896		Generic Host Proces... Microsoft Corp...	
svchost.exe	980		Generic Host Proces... Microsoft Corp...	
svchost.exe	1024		Generic Host Proces... Microsoft Corp...	
svchost.exe	204		Windows Security Ce... Microsoft Corp...	
wsrndf.exe	1076		Generic Host Proces... Microsoft Corp...	
svchost.exe	1188		Generic Host Proces... Microsoft Corp...	
svchost.exe	1292		Spooler SubSystem ... Microsoft Corp...	
spoolsv.exe	1428			
PortReporter.exe	1512		VMware Tools Service VMware, Inc.	
VMwareService.exe	1688		Application Layer Gat... Microsoft Corp...	
alg.exe	740		LSA Shell (Export Ve... Microsoft Corp...	
lsass.exe	1896		Windows Explorer Microsoft Corp...	
explorer.exe	244		Generic Host Proces... Microsoft Corp...	
svchost.exe				

Figura 3-5: Process Explorer examina el malware svchost.exe

El Explorador de procesos muestra cinco columnas: Proceso (el nombre del proceso), PID (el identificador del proceso), CPU (uso de la CPU), Descripción y Nombre de la empresa. La vista se actualiza cada segundo. De forma predeterminada, los servicios se resaltan en rosa, los procesos en azul, los nuevos procesos en verde y los procesos finalizados en rojo. Los resaltados en verde y rojo son temporales y se eliminan después de que el proceso se haya iniciado o finalizado. Al analizar malware, observe la ventana del Explorador de procesos para ver si hay cambios o nuevos procesos y asegúrese de investigarlos a fondo.

El Explorador de procesos puede mostrar bastante información para cada proceso.

Por ejemplo, cuando la ventana de visualización de información de DLL está activa, puede hacer clic en un proceso para ver todas las DLL que cargó en la memoria. Puede cambiar la ventana de visualización de DLL a la ventana de identificadores, que muestra todos los identificadores que tiene el proceso, incluidos los identificadores de archivos, los mutex, los eventos, etc.

La ventana Propiedades que se muestra en la Figura 3-6 se abre al hacer doble clic en el nombre de un proceso. Esta ventana puede proporcionar información particularmente útil sobre el malware en cuestión. La pestaña Subprocesos muestra todos los subprocessos activos, la pestaña TCP/IP muestra las conexiones activas o los puertos en los que el proceso está escuchando y la pestaña Imagen (abierta en la figura) muestra la ruta en el disco al ejecutable.

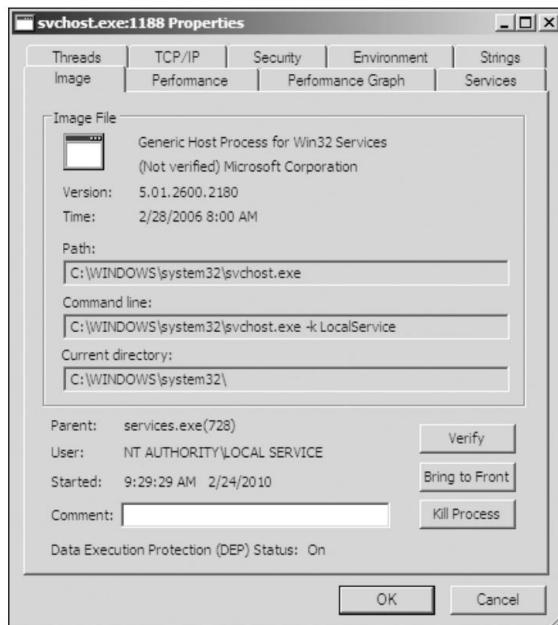


Figura 3-6: La ventana Propiedades, pestaña Imagen

Usando la opción de verificación

Una característica particularmente útil de Process Explorer es el botón Verificar en la pestaña Imagen. Haga clic en este botón para verificar que la imagen en el disco es, de hecho, el binario firmado por Microsoft. Debido a que Microsoft utiliza firmas digitales para la mayoría de sus ejecutables principales, cuando Process Explorer verifica que una firma es válida, puede estar seguro de que el archivo es realmente el ejecutable de Microsoft. Esta característica es particularmente útil para verificar que el archivo de Windows en el disco no se haya dañado; el malware a menudo reemplaza los archivos auténticos de Windows con los suyos propios en un intento de ocultarse.

El botón Verificar verifica la imagen en el disco en lugar de en la memoria, y es inútil si un atacante utiliza el reemplazo de procesos, que implica ejecutar un proceso en el sistema y sobrescribir su espacio de memoria con un ejecutable malicioso. El reemplazo de procesos proporciona al malware los mismos privilegios.

como el proceso que reemplaza, de modo que el malware parece ejecutarse como un proceso legítimo, pero deja una huella: la imagen en la memoria será diferente de la imagen en el disco. Por ejemplo, en la Figura 3-6, el proceso svchost.exe está verificado, pero en realidad es malware. Analizaremos el reemplazo de procesos con más detalle en el Capítulo 12.

Comparación de cadenas

Una forma de reconocer el reemplazo de procesos es utilizar la pestaña Cadenas en la ventana Propiedades del proceso para comparar las cadenas contenidas en el ejecutable del disco (imagen) con las cadenas en la memoria para ese mismo ejecutable que se ejecuta en la memoria. Puede alternar entre estas vistas de cadenas utilizando los botones en la esquina inferior izquierda, como se muestra en la Figura 3-7. Si las dos listas de cadenas son drásticamente diferentes, es posible que se haya producido un reemplazo de procesos. Esta discrepancia de cadenas se muestra en la Figura 3-7. Por ejemplo, la cadena FAVORITES.DAT aparece varias veces en la mitad derecha de la figura (svchost.exe en la memoria), pero no se puede encontrar en la mitad izquierda de la figura (svchost.exe en el disco).

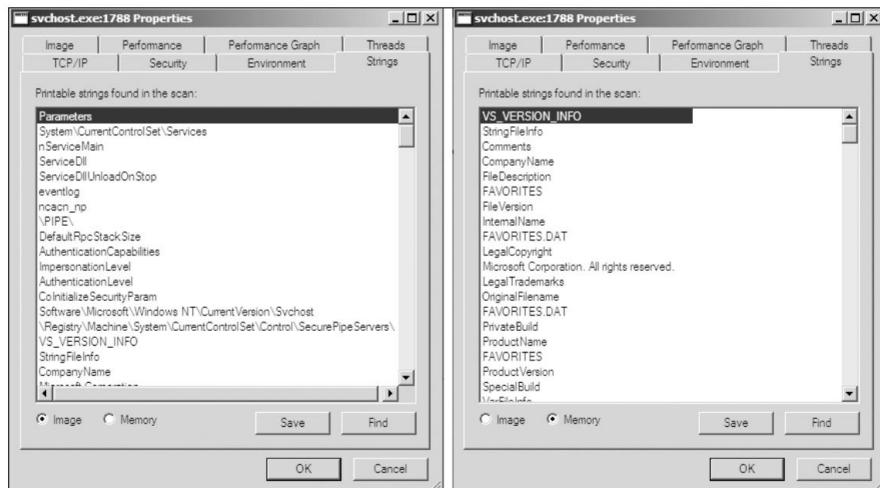


Figura 3-7: La pestaña Cadenas del Explorador de procesos muestra cadenas en el disco (izquierda) versus cadenas en la memoria (derecha) para svchost.exe activo.

Uso de Dependency Walker

El Explorador de procesos le permite iniciar Depends.exe (Dependency Walker) en un proceso en ejecución haciendo clic derecho en el nombre de un proceso y seleccionando Iniciar Depends. También le permite buscar un identificador o DLL eligiendo Buscar Busque el controlador o DLL.

La opción Buscar DLL es especialmente útil cuando se encuentra una DLL maliciosa en el disco y se desea saber si algún proceso en ejecución utiliza esa DLL. El botón Verificar verifica el archivo EXE en el disco, pero no todas las DLL cargadas durante el tiempo de ejecución. Para determinar si una DLL se carga en un proceso después del tiempo de carga, se puede comparar la lista de DLL en el Explorador de procesos con las importaciones que se muestran en Dependency Walker.

Análisis de documentos maliciosos

También puede utilizar Process Explorer para analizar documentos maliciosos, como archivos PDF y documentos de Word. Una forma rápida de determinar si un documento es malicioso es abrir Process Explorer y, a continuación, abrir el documento sospechoso de ser malicioso. Si el documento inicia algún proceso, debería verlo en Process Explorer y poder localizar el malware en el disco a través de la pestaña Imagen de la ventana Propiedades.

NOTA Abrir un documento malicioso mientras se utilizan herramientas de supervisión puede ser una forma rápida de determinar si un documento es malicioso; sin embargo, tendrá éxito si ejecuta solo versiones vulnerables del visor de documentos. En la práctica, es mejor utilizar versiones de la aplicación de visualización que no tengan parches intencionados para garantizar que la explotación sea exitosa. La forma más fácil de hacerlo es con varias instantáneas de su máquina virtual de análisis, cada una con versiones antiguas de visores de documentos como Adobe Reader y Microsoft Word.

Comparación de instantáneas de registro con Regshot

Regshot (que se muestra en la Figura 3-8) es una herramienta de comparación de registros de código abierto que le permite tomar y comparar dos instantáneas de registro.

Para utilizar Regshot para analizar malware, simplemente haga la primera captura haciendo clic en el botón 1st Shot y luego ejecute el malware y espere a que termine de realizar cambios en el sistema. A continuación, haga la segunda captura haciendo clic en el botón 2nd Shot . Por último, haga clic en el botón Comparar para comparar las dos capturas.

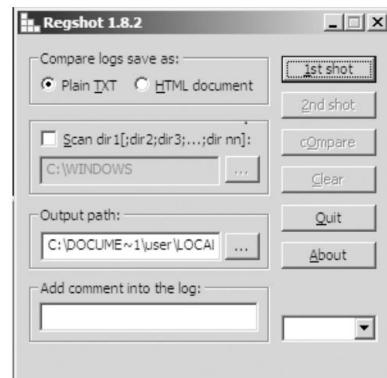


Figura 3-8: Ventana de captura de registro

El listado 3-1 muestra un subconjunto de los resultados generados por Regshot durante el análisis de malware. Se tomaron instantáneas del registro antes y después de ejecutar el spyware ckr.exe.

Toma de registro
Comentarios:
Fecha y hora: <fecha>
Computadora: MALWAREANALYSIS
Nombre de usuario: nombre de usuario

Claves añadidas: 0

```

-----
Valores añadidos:3
-----
HKLM\SOFTWARE\Microsoft\Windows\Versión actual\Ejecutar\ckr:C:\WINDOWS\system32\
ckr.exe
...
...

-----
Valores modificados:2
-----
HKLM\SOFTWARE\Microsoft\Criptografía\RNG\Semilla: 00 43 7C 25 9C 68 DE 59 C6 C8
9D C3 1D E6 DC 87 1C 3A C4 E4 D9 0A B1 BA C1 FB 80 EB 83 25 74 C4 C5 E2 2F CE
4E E8 AC C8 49 E8 E8 10 3F 13 F6 A1 72 92 28 8A 01 3A 16 52 86 36 12 3C C7 EB
5F 99 19 1D 80 8C 8E BD 58 3A DB 18 06 3D 14 8F 22 A4
...
...

-----
Cambios totales:5
-----
```

Listado 3-1: Resultados de comparación de Regshot

Como puede ver, ckr.exe crea un valor en HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run como mecanismo de persistencia . Una cierta cantidad de ruido Es típico en estos resultados, porque la semilla del generador de números aleatorios se actualiza constantemente en el registro.

Al igual que con procmon, su análisis de estos resultados requiere un escaneo del paciente. para encontrar pepitas de interés.

Falsificando una red

El malware suele enviar señales y, finalmente, comunicarse con un servidor de comando y control, como analizaremos en profundidad en el Capítulo 14. Puede crear una red falsa y obtener rápidamente indicadores de red, sin necesidad de conectarse a Internet. Estos indicadores pueden incluir nombres DNS, direcciones IP y firmas de paquetes.

Para falsificar una red con éxito, debe evitar que el malware se dé cuenta de que se está ejecutando en un entorno virtualizado. (Consulte el Capítulo 2 para obtener información sobre cómo configurar redes virtuales con VMware). Si combina las herramientas que se analizan aquí con una configuración de red de máquinas virtuales sólida, aumentará enormemente sus posibilidades de éxito.

Uso de ApateDNS

ApateDNS, una herramienta gratuita de Mandiant (www.mandiant.com/products/research/mandiant_apatedns/download) es la forma más rápida de ver las solicitudes DNS realizadas por malware. ApateDNS falsifica las respuestas DNS a una dirección IP especificada por el usuario escuchando en el puerto UDP 53 de la máquina local. Responde a las solicitudes DNS con la respuesta DNS configurada en una dirección IP que usted especifique. ApateDNS puede mostrar los resultados hexadecimales y ASCII de todas las solicitudes que recibe.

Para utilizar ApateDNS, configure la dirección IP que desea enviar en las respuestas DNS en y seleccione la interfaz en . A continuación, presione el botón Iniciar servidor ; esto iniciará automáticamente el servidor DNS y cambiará la configuración de DNS a localhost. A continuación, ejecute su malware y observe cómo aparecen las solicitudes DNS en la ventana de ApateDNS. Por ejemplo, en la Figura 3-9, redirigimos las solicitudes DNS realizadas por el malware conocido como RShell. Vemos que la información DNS se solicita para `evil.malwar3.com` y que esa solicitud se realizó a las 13:22:08.

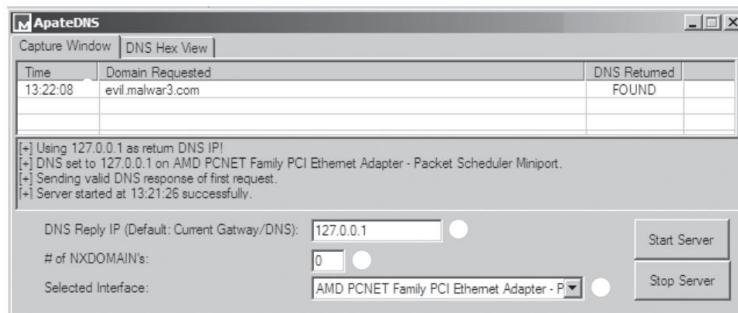


Figura 3-9: ApateDNS responde a una solicitud para `evil.malwar3.com`

En el ejemplo que se muestra en la figura, redirigimos las solicitudes DNS a 127.0.0.1 (localhost), pero es posible que desee cambiar esta dirección para que apunte a algo externo, como un servidor web falso que se ejecuta en una máquina virtual Linux. Debido a que la dirección IP será diferente a la de su máquina virtual de análisis de malware de Windows, asegúrese de ingresar la dirección IP adecuada antes de iniciar el servidor. De manera predeterminada, ApateDNS utilizará la puerta de enlace actual o la configuración DNS actual para insertarla en las respuestas DNS.

Puede detectar dominios adicionales utilizados por una muestra de malware mediante el uso de la opción de dominio inexistente (NXDOMAIN) en . El malware suele recorrer los diferentes dominios que ha almacenado si no se encuentran el primer o el segundo dominio. El uso de esta opción NXDOMAIN puede engañar al malware para que le proporcione dominios adicionales que tiene en su configuración.

Monitoreo con Netcat

Netcat, la “navaja suiza TCP/IP”, se puede utilizar tanto en conexiones entrantes como salientes para escanear puertos, crear túneles, utilizar servidores proxy, reenviar puertos y mucho más. En modo de escucha, Netcat actúa como un servidor, mientras que en modo de conexión actúa como un cliente. Netcat toma datos de la entrada estándar para transmitirlos a través de la red. Todos los datos que recibe se muestran en la pantalla a través de la salida estándar.

Veamos cómo se puede utilizar Netcat para analizar el RShell de malware de la Figura 3-9. Mediante ApateDNS, redirigimos la solicitud DNS a `evil.malwar3.com` a nuestro host local. Suponiendo que el malware se está enviando a través del puerto 80 (una opción común), podemos usar Netcat para escuchar las conexiones antes de ejecutar el malware.

El malware suele utilizar el puerto 80 o 443 (tráfico HTTP o HTTPS, respectivamente), ya que estos puertos no suelen estar bloqueados ni monitoreados como conexiones salientes. El listado 3-2 muestra un ejemplo.

```
C:\> nc -l -p 80
POST /cq/frame.htm HTTP/1.1
Anfitrón: www.google.com
Agente de usuario: Mozilla/5.0 (Windows; Windows NT 5.1; TWFsd2FyZUh1bnRlcg==; rv:1.38)

Aceptar: texto/html, aplicación
Aceptar idioma: en-US, en;q=
Aceptar codificación: gzip, deflate
Mantenerse con vida: 300
Tipo de contenido: aplicación/x-form-urlencoded
Longitud del contenido

Microsoft Windows XP [Versión 5.1.2600]
(C) Derechos de autor 1985-2001 Microsoft Corp.

Z:\Malware>
```

Listado 3-2: Ejemplo de Netcat escuchando en el puerto 80

El comando Netcat (nc) muestra las opciones necesarias para escuchar en un puerto. El indicador `-l` significa escuchar y `-p` (con un número de puerto) especifica el puerto en el que se debe escuchar. El malware se conecta a nuestro receptor Netcat porque estamos usando ApateDNS para la redirección. Como puede ver, RShell es un shell inverso , pero no proporciona el shell de inmediato. La conexión de red aparece primero como una solicitud HTTP POST a www.google.com , datos POST falsos que probablemente RShell inserta para ofuscar su shell inverso, porque los analistas de red con frecuencia solo miran el inicio de una sesión.

Detección de paquetes con Wireshark

Wireshark es un rastreador de código abierto, una herramienta de captura de paquetes que intercepta y registra el tráfico de red. Wireshark ofrece visualización, análisis del flujo de paquetes y análisis en profundidad de paquetes individuales.

Al igual que muchas de las herramientas que se analizan en este libro, Wireshark se puede utilizar tanto para el bien como para el mal. Se puede utilizar para analizar redes internas y el uso de la red, depurar problemas de aplicaciones y estudiar protocolos en acción. Pero también se puede utilizar para rastrear contraseñas, aplicar ingeniería inversa a protocolos de red, robar información confidencial y escuchar las conversaciones en línea en la cafetería de tu barrio.

La pantalla de Wireshark tiene cuatro partes, como se muestra en la Figura 3-10:

El cuadro Filtro se utiliza para filtrar los paquetes mostrados. La lista de paquetes muestra todos los paquetes que satisfacen el filtro de visualización. La ventana de detalles del paquete muestra el contenido del paquete seleccionado actualmente (en este caso, el paquete 47).

La ventana hexadecimal muestra el contenido hexadecimal del paquete actual. La ventana hexadecimal está vinculada con la ventana de detalles del paquete y resaltarán los campos que seleccione.

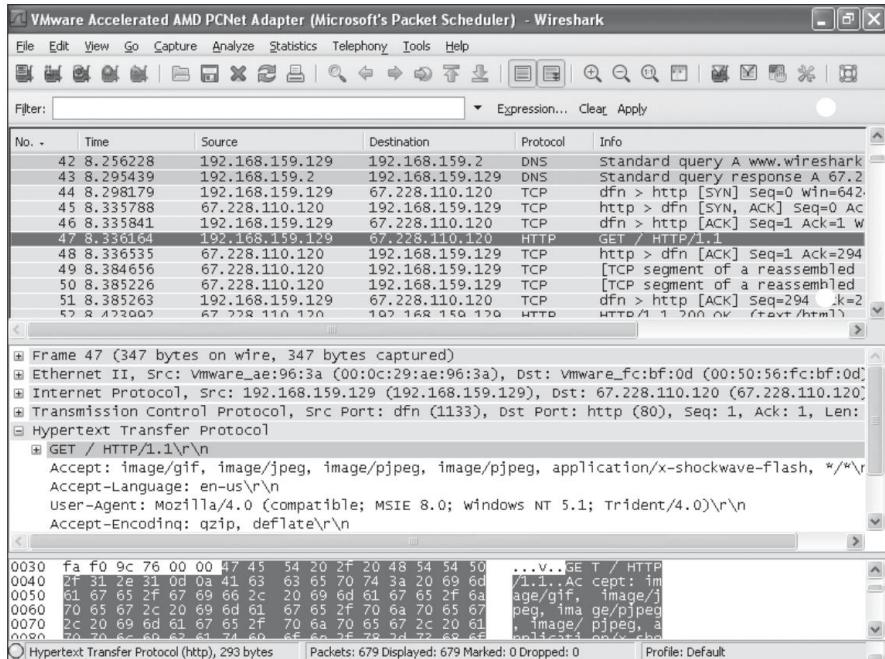


Figura 3-10: Ejemplo de DNS y HTTP de Wireshark

Para utilizar Wireshark y ver el contenido de una sesión TCP, haga clic con el botón derecho en cualquier paquete TCP y seleccione Seguir flujo TCP. Como puede ver en la Figura 3-11, ambos extremos de la conversación se muestran en el orden de la sesión, con diferentes colores que muestran cada lado de la conexión.

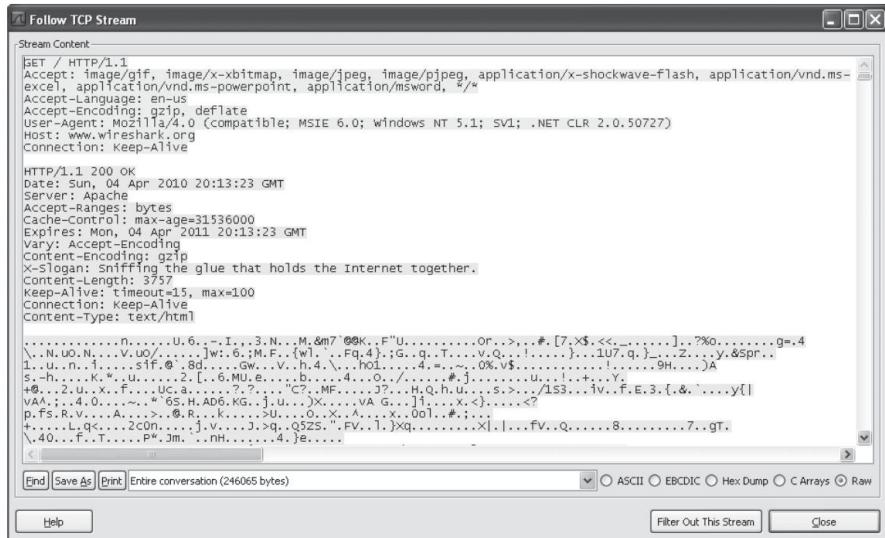


Figura 3-11: Ventana Seguir flujo TCP de Wireshark

Para capturar paquetes, seleccione Capturar → Interfaces y seleccione la interfaz que desea utilizar para recopilar paquetes. Las opciones incluyen el uso del modo promiscuo o la configuración de un filtro de captura.

ADVERTENCIA Se sabe que Wireshark tiene muchas vulnerabilidades de seguridad, así que asegúrese de ejecutarlo en un entorno seguro.

Wireshark puede ayudar a comprender cómo el malware realiza la comunicación en la red al rastrear los paquetes a medida que se comunica. Para utilizar Wireshark con este fin, conéctese a Internet o simule una conexión a Internet y, a continuación, inicie la captura de paquetes de Wireshark y ejecute el malware. (Puede utilizar Netcat para simular una conexión a Internet).

El capítulo 14 analiza el análisis del protocolo y usos adicionales de Wireshark con más detalle.

Usando INetSim

INetSim es un paquete de software gratuito basado en Linux que permite simular servicios comunes de Internet. La forma más sencilla de ejecutar INetSim si su sistema operativo base es Microsoft Windows es instalarlo en una máquina virtual Linux y configurarlo en la misma red virtual que su máquina virtual de análisis de malware.

INetSim es la mejor herramienta gratuita para proporcionar servicios falsos, lo que le permite analizar el comportamiento de la red de muestras de malware desconocidas mediante la emulación de servicios como HTTP, HTTPS, FTP, IRC, DNS, SMTP y otros. El Listado 3-3 muestra todos los servicios que INetSim emula de forma predeterminada, todos los cuales (incluidos los puertos predeterminados utilizados) se muestran aquí cuando se inicia el programa.

- * dns 53/udp/tcp - iniciado (PID 9992)
- * http 80/tcp - iniciado (PID 9993)
- * https 443/tcp - iniciado (PID 9994)
- smtp 25/tcp - iniciado (PID 9995)
- * irc 6667/tcp - iniciado (PID 10002)
- * smtps 465/tcp - iniciado (PID 9996)
- * ntp 123/udp - iniciado (PID 10003)
- pop3 110/tcp - iniciado (PID 9997)
- * dedo 79/tcp - iniciado (PID 10004)
- * syslog 514/udp - iniciado (PID 10006)
- * tftp 69/udp - iniciado (PID 10001)
- pop3s 995/tcp - iniciado (PID 9998)
- * tiempo 37/tcp - iniciado (PID 10007)
- * ftp 21/tcp - iniciado (PID 9999)
- * ident 113/tcp - iniciado (PID 10005)
- * hora 37/udp - iniciado (PID 10008)
- * ftps 990/tcp - iniciado (PID 10000)
- * día 13/tcp - iniciado (PID 10009)
- * día 13/udp - iniciado (PID 10010)
- * echo 7/tcp - iniciado (PID 10011)
- * echo 7/udp - iniciado (PID 10012)
- * descartar 9/udp - iniciado (PID 10014)

```
* descartar 9/tcp - iniciado (PID 10013)
* cita 17/tcp - iniciado (PID 10015)
* quotd 17/udp - iniciado (PID 10016)
* chargen 19/tcp - iniciado (PID 10017)
* dummy 1/udp - iniciado (PID 10020)
* chargen 19/udp - iniciado (PID 10018)
* dummy 1/tcp - iniciado (PID 10019)
```

Listado 3-3: Servicios emulados predeterminados de INetSim

INetSim hace todo lo posible por parecerse a un servidor real y tiene muchas funciones fácilmente configurables para garantizar el éxito. Por ejemplo, de forma predeterminada, devuelve el banner del servidor web Microsoft IIS si se escanea.

Algunas de las mejores características de INetSim están integradas en su simulación de servidores HTTP y HTTPS. Por ejemplo, INetSim puede servir casi cualquier archivo solicitado. Por ejemplo, si un programa malicioso solicita un JPEG de un sitio web para continuar su funcionamiento, INetSim responderá con un JPEG con el formato correcto. Aunque esa imagen podría no ser el archivo que el malware está buscando, el servidor no devuelve un error 404 ni ningún otro error, y su respuesta, incluso si es incorrecta, puede mantener el malware en funcionamiento.

INetSim también puede registrar todas las solicitudes y conexiones entrantes, lo que le resultará especialmente útil para determinar si el malware está conectado a un servicio estándar o para ver las solicitudes que está realizando. Además, INetSim es extremadamente configurable. Por ejemplo, puede configurar la página o el elemento que se devuelve después de una solicitud, de modo que si se da cuenta de que el malware en cuestión está buscando una página web en particular antes de continuar con su ejecución, puede proporcionar esa página. También puedes modificar el puerto en el que escuchan varios servicios, lo que puede ser útil si el malware utiliza puertos no estándar.

Y como INetSim está diseñado teniendo en cuenta el análisis de malware, ofrece muchas funciones únicas, como su servicio Dummy, una función que registra todos los datos recibidos del cliente, independientemente del puerto. El servicio Dummy es muy útil para capturar todo el tráfico enviado desde el cliente a puertos que no están vinculados a ningún otro módulo de servicio. Puede usarlo para registrar todos los puertos a los que se conecta el malware y los datos correspondientes que se envían. Al menos se completará el protocolo de enlace TCP y se podrán recopilar datos adicionales.

Herramientas dinámicas básicas en la práctica

Todas las herramientas que se analizan en este capítulo se pueden utilizar en conjunto para maximizar la cantidad de información obtenida durante el análisis dinámico. En esta sección, analizaremos todas las herramientas que se analizan en el capítulo y presentaremos una configuración de muestra para el análisis de malware. Su configuración podría incluir lo siguiente:

1. Ejecutar procmon y configurar un filtro en el nombre ejecutable del malware y borrar todos los eventos justo antes de ejecutarlo.
2. Iniciar el Explorador de procesos.
3. Recopilación de una primera instantánea del registro mediante Regshot.

4. Configura tu red virtual a tu gusto usando INetSim y

Servidor de archivos ApateDNS.

5. Configuración del registro del tráfico de red mediante Wireshark.

La figura 3-12 muestra un diagrama de una red virtual que se puede configurar para el análisis de malware. Esta red virtual contiene dos hosts: la máquina virtual de Windows para el análisis de malware y la máquina virtual de Linux que ejecuta INetSim.

La máquina virtual Linux está escuchando en muchos puertos, incluidos HTTPS, FTP y HTTP, mediante el uso de INetSim. La máquina virtual Windows está escuchando en el puerto 53 las solicitudes DNS mediante el uso de ApateDNS. El servidor DNS para la máquina virtual Windows se ha configurado como host local (127.0.0.1). ApateDNS está configurado para redirigirlo a la máquina virtual Linux (192.168.117.169).

Si intenta navegar a un sitio web mediante la máquina virtual de Windows, ApateDNS resolverá la solicitud DNS y lo redireccionará a la máquina virtual de Linux. Luego, el navegador realizará una solicitud GET a través del puerto 80 al servidor INetSim que escucha en ese puerto en la máquina virtual de Linux.



Figura 3-12: Ejemplo de una red virtual

Veamos cómo funcionaría esta configuración en la práctica examinando el malware msts.exe. Completamos nuestra configuración inicial y luego ejecutamos msts.exe en nuestra máquina virtual de análisis de malware. Después de un tiempo, detenemos la captura de eventos con procmon y ejecutamos una segunda instantánea con Regshot. En este punto, comenzamos el análisis de la siguiente manera:

1. Examine ApateDNS para ver si se realizaron solicitudes DNS. Como se muestra en la Figura 3-13, observamos que el malware realizó una solicitud DNS para www.malwareanalysisbook.com.

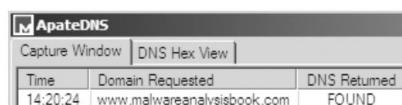


Figura 3-13: Solicitud de ApateDNS para www.malwareanalysisbook.com

- Revise los resultados de procmón para ver si se han realizado modificaciones en el sistema de archivos.
- Los resultados de procmón que se muestran en la Figura 3-14, vemos CreateFile y WriteFile (números de secuencia 141 y 142) operaciones para C:\WINDOWS\system32\winhlp2.exe. Después de una investigación más profunda, comparamos winhlp2.exe con msts.exe y vemos que son idénticos. Concluimos que el malware se copia a sí mismo en esa ubicación.

Sequence	Time...	Process Name	PID	Operation	Path	Result	Detail
141	4.55....	msts.exe	1800	CreateFile	C:\WINDOWS\system32\winhlp2.exe	SUCCESS	Desired Access: Generic Write...
142	4.55....	msts.exe	1800	WriteFile	C:\WINDOWS\system32\winhlp2.exe	SUCCESS	Offset: 0, Length: 7.168
143	4.55....	msts.exe	1800	CloseFile	C:\WINDOWS\system32\winhlp2.exe	SUCCESS	

Figura 3-14: Salida de Procmon con el filtro msts.exe configurado

- Compare las dos instantáneas tomadas con Regshot para identificar cambios.

Al revisar los resultados de Regshot, que se muestran a continuación, vemos que el malware instaló el valor de registro de ejecución automática winhlp en HKLM\SOFTWARE\Microsoft\Ubicación de Windows\CurrentVersion\Run . Los datos escritos en ese valor son donde el malware se copió a sí mismo (C:\WINDOWS\system32\winhlp2.exe), y ese binario recién copiado se ejecutará al reiniciar el sistema.

Valores añadidos:

HKLM\SOFTWARE\Microsoft\Windows\Versión actual\Ejecutar\winhlp: C:\WINDOWS\system32\winhlp2.exe

- Utilice Process Explorer para examinar el proceso y determinar si crea mutex o escucha conexiones entrantes. La salida de Process Explorer en la Figura 3-15 muestra que msts.exe crea un mutex (también conocido como mutante) llamado Evil . Analizamos los mutex en profundidad en el Capítulo 7, pero debe saber que es probable que msts.exe haya creado el mutex para garantizar que solo se esté ejecutando una versión del malware a la vez. Los mutex pueden proporcionar una excelente huella digital para el malware si son lo suficientemente únicos.

- Revise los registros de INetSim para ver si hay solicitudes y conexiones intentadas en Servicios estándar. La primera línea de los registros de INetSim (que se muestra a continuación) nos indica que el malware se comunica a través del puerto 443, aunque no con el protocolo Secure Sockets Layer (SSL) estándar, como se muestra a continuación en los errores informados en .

```
[2010-X] [15013] [https 443/tcp 15199] [192.168.117.128:1043] conectar
[2010-X] [15013] [https 443/tcp 15199] [192.168.117.128:1043]
Error al configurar SSL: el intento de aceptación de SSL falló con un error desconocido
Error:140760FC:Rutinas SSL:SSL23_GET_CLIENT_HELLO:protocolo desconocido
[2010-X] [15013] [https 443/tcp 15199] [192.168.117.128:1043] desconectar
```

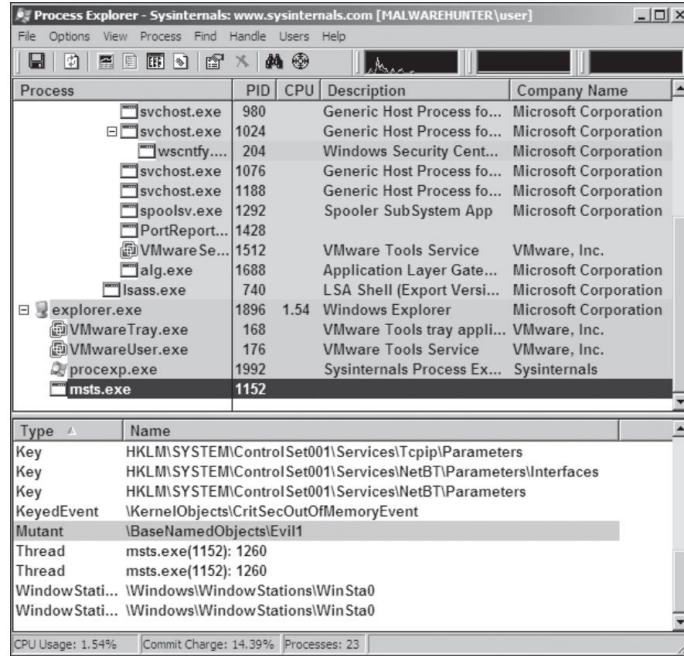


Figura 3-15: Examen de Process Explorer de un proceso msts.exe activo

6. Revise la captura de Wireshark para ver el tráfico de red generado por el malware. Al usar INetSim mientras capturamos con Wireshark, podemos capturar el protocolo de enlace TCP y los paquetes de datos iniciales enviados por el malware. El contenido del flujo TCP enviado a través del puerto 443, como se muestra en la Figura 3-16, muestra datos ASCII aleatorios, que a menudo son indicativos de un protocolo personalizado. Cuando esto sucede, lo mejor es ejecutar el malware varias veces más para buscar consistencia en los paquetes iniciales de la conexión. (La información resultante podría utilizarse para redactar una firma basada en red, habilidades que exploramos en el Capítulo 14.)

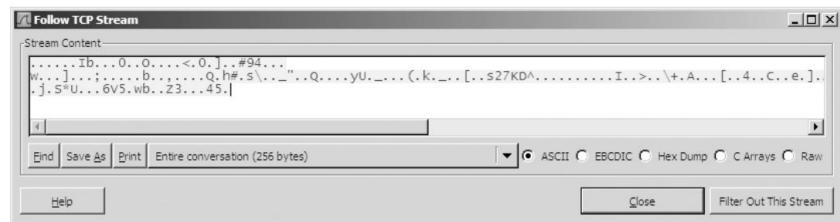


Figura 3-16: Wireshark muestra el protocolo de red personalizado

Conclusión

El análisis dinámico básico del malware puede ayudar y confirmar los hallazgos del análisis estático básico. La mayoría de las herramientas descritas en este capítulo son gratuitas y fáciles de usar, y brindan una cantidad considerable de detalles.

Sin embargo, las técnicas básicas de análisis dinámico tienen sus deficiencias, por lo que no nos detendremos aquí. Por ejemplo, para comprender completamente el componente de red en msts.exe , deberá aplicar ingeniería inversa al protocolo para determinar la mejor manera de continuar con el análisis. El siguiente paso es realizar técnicas avanzadas de análisis estático con desensamblado y disección a nivel binario, lo que se analiza en el próximo capítulo.

LABORATORIOS

Laboratorio 3-1

Analice el malware encontrado en el archivo Lab03-01.exe utilizando herramientas básicas de análisis dinámico.

Preguntas

1. ¿Cuáles son las importaciones y cadenas de este malware?
2. ¿Cuáles son los indicadores basados en el host del malware?
3. ¿Existen firmas útiles basadas en la red para este malware? Si es así, ¿cuáles son?

Laboratorio 3-2

Analice el malware encontrado en el archivo Lab03-02.dll utilizando herramientas básicas de análisis dinámico.

Preguntas

1. ¿Cómo se puede conseguir que este malware se instale?
2. ¿Cómo lograrías que este malware se ejecutara después de la instalación?
3. ¿Cómo puedo encontrar el proceso bajo el cual se ejecuta este malware?
4. ¿Qué filtros podrías configurar para utilizar procmon para recopilar información?
5. ¿Cuáles son los indicadores basados en el host del malware?
6. ¿Existen firmas basadas en red útiles para este malware?

Laboratorio 3-3

Ejecute el malware que se encuentra en el archivo Lab03-03.exe mientras lo monitorea utilizando herramientas básicas de análisis dinámico en un entorno seguro.

Preguntas

1. ¿Qué observas al monitorear este malware con Process?
¿Explorador?
2. ¿Puedes identificar alguna modificación de la memoria viva?
3. ¿Cuáles son los indicadores basados en el host del malware?
4. ¿Cuál es el propósito de este programa?

Laboratorio 3-4

Analice el malware encontrado en el archivo Lab03-04.exe utilizando herramientas básicas de análisis dinámico. (Este programa se analiza con más detalle en los laboratorios del Capítulo 9).

Preguntas

1. ¿Qué sucede cuando ejecuta este archivo?
2. ¿Qué está provocando el bloqueo en el análisis dinámico?
3. ¿Existen otras formas de ejecutar este programa?

PARTE 2

ANÁLISIS ESTÁTICO AVANZADO

4

CURSO INTENSIVO DE X86 DESMONTAJE

Como se analizó en capítulos anteriores, los métodos básicos de análisis de malware estático y dinámico son buenos para la clasificación inicial, pero no brindan suficiente información para analizar el malware por completo.

Las técnicas estáticas básicas son como mirar el exterior de un cuerpo durante una Autopsia. Puede utilizar el análisis estático para extraer algunas conclusiones preliminares, pero se requiere un análisis más profundo para obtener toda la historia. Por ejemplo, puede descubrir que se importa una función en particular, pero no sabrá cómo se utiliza o si se utiliza en absoluto.

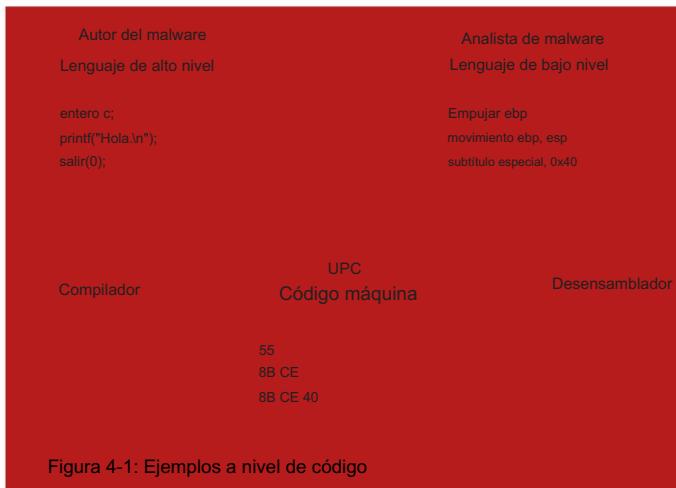
Las técnicas dinámicas básicas también tienen sus defectos. Por ejemplo, el análisis dinámico básico puede indicarle cómo responde el malware en cuestión cuando recibe un paquete especialmente diseñado, pero sólo podrá conocer el formato de ese paquete si investiga más a fondo. Ahí es donde entra en juego el desensamblado, como aprenderá en este capítulo.

El desmontaje es una habilidad especializada que puede resultar abrumadora para quienes recién comienzan a programar. Pero no se desanime: este capítulo le brindará una comprensión básica del desmontaje para que pueda comenzar con el pie derecho.

Niveles de abstracción

En la arquitectura informática tradicional, un sistema informático puede representarse como varios niveles de abstracción que crean una forma de ocultar los detalles de implementación. Por ejemplo, se puede ejecutar el sistema operativo Windows en muchos tipos de hardware diferentes, porque el hardware subyacente está abstracto del sistema operativo.

La figura 4-1 muestra los tres niveles de codificación involucrados en el análisis de malware. Los autores de malware crean programas en un lenguaje de alto nivel y utilizan un compilador para generar código de máquina que la CPU ejecutará. Por el contrario, los analistas de malware y los ingenieros inversos operan en un lenguaje de bajo nivel; utilizamos un desensamblador para generar código ensamblador que podemos leer y analizar para averiguar cómo funciona un programa.



La figura 4-1 muestra un modelo simplificado, pero los sistemas informáticos se describen generalmente con los siguientes seis niveles de abstracción diferentes. Enumeramos estos niveles comenzando desde abajo. Los niveles de abstracción más altos se ubican cerca de la parte superior con conceptos más específicos debajo, de modo que cuanto más abajo se llegue, menos portátil será el nivel entre los sistemas informáticos.

Hardware El nivel de hardware, el único nivel físico, consta de circuitos eléctricos que implementan combinaciones complejas de operadores lógicos como las puertas XOR, AND, OR y NOT, conocidas como lógica digital. Debido a su naturaleza física, el hardware no puede ser manipulado fácilmente por software.

Microcódigo El nivel de microcódigo también se conoce como firmware. El microcódigo opera únicamente en el circuito exacto para el que fue diseñado. Contiene microinstrucciones que se traducen desde el nivel superior de código de máquina para proporcionar una forma de interactuar con el hardware. Cuando realizamos un análisis de malware, normalmente no nos preocupamos por el microcódigo porque suele ser específico del hardware de la computadora para el que fue escrito.

Código de máquina El nivel de código de máquina consta de códigos de operación, dígitos hexadecimales que le indican al procesador lo que desea que haga. El código de máquina se implementa típicamente con varias instrucciones de microcódigo para que el hardware subyacente pueda ejecutar el código. El código de máquina se crea cuando se compila un programa de computadora escrito en un lenguaje de alto nivel.

Lenguajes de bajo nivel Un lenguaje de bajo nivel es una versión legible por humanos del conjunto de instrucciones de una arquitectura informática. El lenguaje de bajo nivel más común es el lenguaje ensamblador. Los analistas de malware operan en el nivel de lenguajes de bajo nivel porque el código de máquina es demasiado difícil de comprender para un humano. Usamos un desensamblador para generar texto en lenguaje de bajo nivel, que consta de mnemotecnias simples como mov y jmp. Existen muchos dialectos diferentes del lenguaje ensamblador y exploraremos cada uno de ellos por separado.

NOTA: El ensamblaje es el lenguaje de más alto nivel que se puede recuperar de manera confiable y consistente del código de máquina cuando el código fuente del lenguaje de alto nivel no está disponible.

Lenguajes de alto nivel La mayoría de los programadores informáticos trabajan en el nivel de lenguajes de alto nivel. Los lenguajes de alto nivel proporcionan una fuerte abstracción del nivel de máquina y facilitan el uso de la lógica de programación y los mecanismos de control de flujo. Los lenguajes de alto nivel incluyen C, C++ y otros. Estos lenguajes suelen convertirse en código de máquina mediante un compilador a través de un proceso conocido como compilación.

Lenguajes interpretados Los lenguajes interpretados se encuentran en el nivel superior. Muchos programadores utilizan lenguajes interpretados como C#, Perl, .NET y Java. El código en este nivel no se compila en código de máquina, sino que se traduce en código de bytes. El código de bytes es una representación intermedia que es específica del lenguaje de programación. El código de bytes se ejecuta dentro de un intérprete, que es un programa que traduce el código de bytes en código de máquina ejecutable sobre la marcha en tiempo de ejecución. Un intérprete proporciona un nivel automático de abstracción en comparación con el código compilado tradicional, porque puede manejar errores y la gestión de memoria por sí solo, independientemente del sistema operativo.

Ingeniería inversa

Cuando el malware se almacena en un disco, normalmente se encuentra en formato binario a nivel de código de máquina. Como se ha comentado, el código de máquina es el formato de código que la computadora puede ejecutar de forma rápida y eficaz. Cuando desensamblamos el malware (como se muestra en la Figura 4-1), tomamos el binario del malware como entrada y generamos código en lenguaje ensamblador como salida, normalmente con un desensamblador. (El Capítulo 5 analiza el desensamblador más popular, IDA Pro).

El lenguaje ensamblador es en realidad una clase de lenguajes. Cada dialecto ensamblador se utiliza normalmente para programar una única familia de microprocesadores, como x86, x64, SPARC, PowerPC, MIPS y ARM. x86 es, con diferencia, la arquitectura más popular.