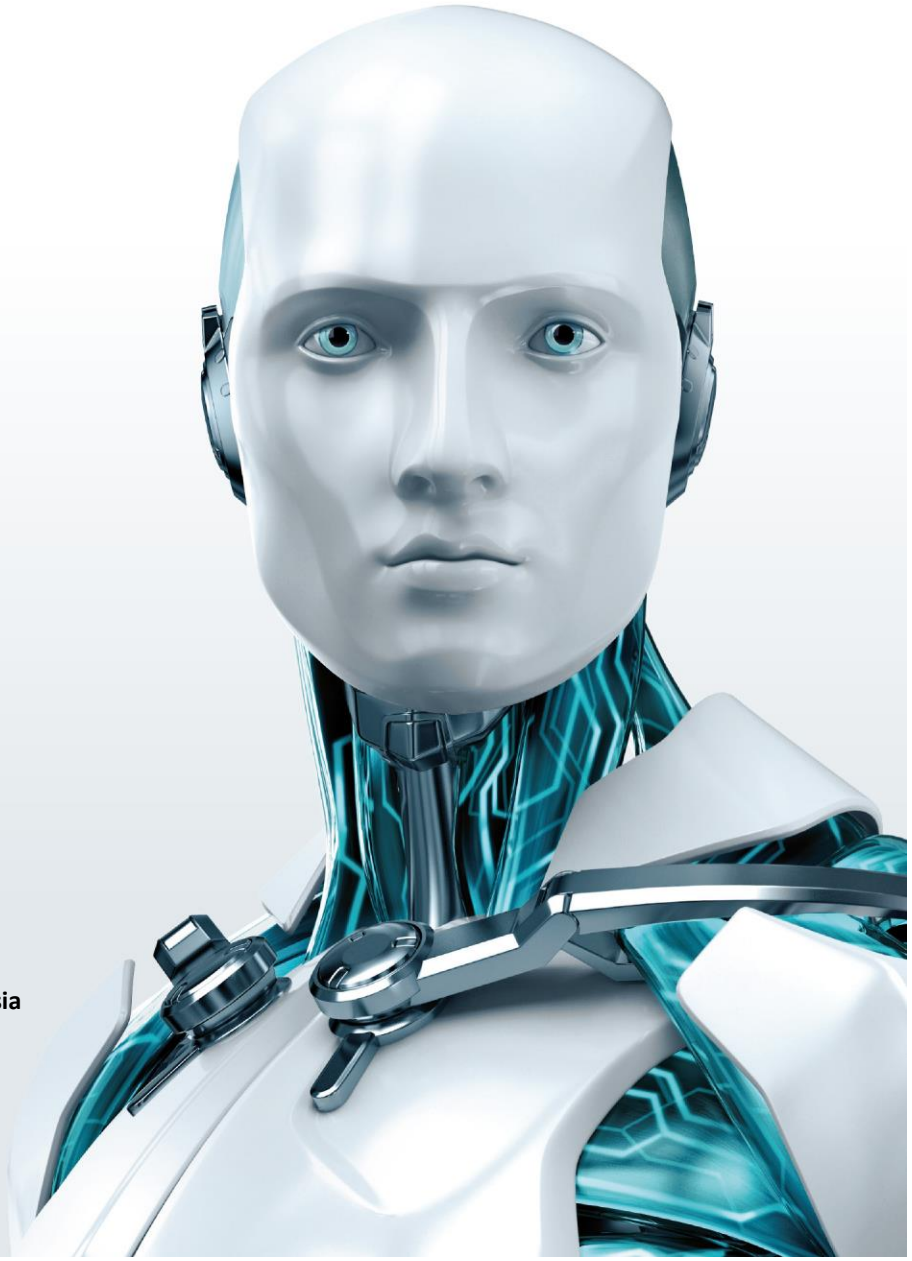
# Explotación de vulnerabilidades en Windows durante 2014



**Artem Baranov, Malware Researcher, ESET Rusia**

Enero de 2015

ENJOY SAFER TECHNOLOGY™

# Introducción

Decidimos escribir una nueva versión de nuestro informe anterior sobre las principales tendencias respecto a la explotación de vulnerabilidades de Windows y su mitigación a lo largo del 2014. En dicho informe, mencionábamos que los ataques 0-day fueron una de las tendencias principales en 2013 y además que los ciberdelincuentes habían desarrollado *exploits* 0-day específicos para realizar ataques dirigidos. Esta tendencia siguió progresando en el transcurso de 2014.

En esta edición anual del informe agregamos una sección especial con notas sobre Internet Explorer (IE). El 2014 fue realmente difícil para los usuarios de este navegador, dado que Microsoft (MS) solucionó el doble de vulnerabilidades de IE que en 2013. También agregamos información adicional sobre las técnicas de mitigación de *exploits* para usuarios de Windows y explicamos por qué no es tan fácil proteger el sistema operativo como parecería a primera vista.

# Información general

La Tabla 1, abajo, muestra información sobre vulnerabilidades reparadas para Internet Explorer (versiones 6 a 11). Las vulnerabilidades que fueron aprovechadas por los atacantes antes de que las revisiones correspondientes estuvieran disponibles (0-day) se resaltaron en rojo.

| Componente | Boletín | Tipo | Vulnerabilidad |
|---|---|---|---|
| **Internet Explorer** | MS14-010, MS14-012, MS14-018, MS14-021, MS14-029, MS14-035, MS14-037, MS14-051, MS14-052, MS14-056, MS14-065, MS14-080 | Ejecución remota de código(12) | CVE-2014-0267, CVE-2014-0268, CVE-2014-0269, CVE-2014-0270, CVE-2014-0271, CVE-2014-0272, CVE-2014-0273, CVE-2014-0274, CVE-2014-0275, CVE-2014-0276, CVE-2014-0277, CVE-2014-0278, CVE-2014-0279, CVE-2014-0280, CVE-2014-0281, CVE-2014-0283, CVE-2014-0284, CVE-2014-0285, CVE-2014-0286, CVE-2014-0287, CVE-2014-0288, CVE-2014-0289, CVE-2014-0290, CVE-2014-0293, CVE-2014-0297, CVE-2014-0298, CVE-2014-0299, CVE-2014-0302, CVE-2014-0303, CVE-2014-0304, CVE-2014-0305, CVE-2014-0306, CVE-2014-0307, CVE-2014-0308, CVE-2014-0309, CVE-2014-0311, CVE-2014-0312, CVE-2014-0313, CVE-2014-0314, CVE-2014-0321, **CVE-2014-0322**, **CVE-2014-0324**, CVE-2014-0235, CVE-2014-1751, CVE-2014-1752, CVE-2014-1753, CVE-2014-1755, CVE-2014-1760, **CVE-2014-1776**, CVE-2014-0310, **CVE-2014-1815**, CVE-2014-0282, CVE-2014-1762, CVE-2014-1764, CVE-2014-1766, CVE-2014-1769, CVE-2014-1770, CVE-2014-1771, CVE-2014-1772, CVE-2014-1773, CVE-2014-1774, CVE-2014-1775, CVE-2014-1777, CVE-2014-1778, CVE-2014-1779, CVE-2014-1780, CVE-2014-1781, CVE-2014-1782, CVE-2014-1783, CVE-2014-1784, CVE-2014-1785, CVE-2014-1786, CVE-2014-1788, CVE-2014-1789, CVE-2014-1790, CVE-2014-1791, CVE-2014-1792, CVE-2014-1794, |

| | |
|---|---|
| | CVE-2014-1795, CVE-2014-1796, CVE-2014-1797, CVE-2014-1799, CVE-2014-1800, CVE-2014-1802, CVE-2014-1803, CVE-2014-1804, CVE-2014-1805, CVE-2014-2753, CVE-2014-2754, CVE-2014-2755, CVE-2014-2756, CVE-2014-2757, CVE-2014-2758, CVE-2014-2759, CVE-2014-2760, CVE 2014-2761, CVE-2014-2763, CVE-2014-2764, CVE-2014-2765, CVE-2014-2766, CVE-2014-2767, CVE-2014-2768, CVE-2014-2769, CVE-2014-2770, CVE-2014-2771, CVE-2014-2772, CVE-2014-2773, CVE-2014-2775, CVE-2014-2776, CVE-2014-2777, CVE-2014-1763, CVE-2014-1765, CVE-2014-2785, CVE-2014-2786, CVE-2014-2787, CVE-2014-2788, CVE-2014-2789, CVE-2014-2790, CVE-2014-2791, CVE-2014-2792, CVE-2014-2794, CVE-2014-2795, CVE-2014-2797, CVE-2014-2798, CVE-2014-2800, CVE-2014-2801, CVE-2014-2802, CVE-2014-2803, CVE-2014-2804, CVE-2014-2806, CVE-2014-2807, CVE-2014-2809, CVE-2014-2813, CVE-2014-2783, CVE-2014-2774, CVE-2014-2784, CVE-2014-2796, CVE-2014-2808, CVE-2014-2810, CVE-2014-2811, **CVE-2014-2817**, CVE-2014-2818, CVE-2014-2819, CVE-2014-2820, CVE-2014-2821, CVE-2014-2822, CVE-2014-2823, CVE-2014-2824, CVE-2014-2825, CVE-2014-2826, CVE-2014-2827, CVE-2014-4050, CVE-2014-4051, CVE-2014-4052, CVE-2014-4055, CVE-2014-4056, CVE-2014-4057, CVE-2014-4058, CVE-2014-4063, CVE-2014-4067, CVE-2014-4145, **CVE-2013-7331**, CVE-2014-2799, CVE-2014-4059, CVE-2014-4065, CVE-2014-4079, CVE-2014-4080, CVE-2014-4081, CVE-2014-4082, CVE-2014-4083, CVE-2014-4084, CVE-2014-4085, CVE-2014-4086, CVE-2014-4087, CVE-2014-4088, CVE-2014-4089, CVE-2014-4090, CVE-2014-4091, CVE-2014-4092, CVE-2014-4093, CVE-2014-4094, CVE-2014-4095, CVE-2014-4096, CVE-2014-4097, CVE-2014-4098, CVE-2014-4099, CVE-2014-4100, CVE-2014-4101, CVE-2014-4102, CVE-2014-4103, CVE-2014-4104, CVE-2014-4105, CVE-2014-4106, CVE-2014-4107, CVE-2014-4108, CVE-2014-4109, CVE-2014-4110, CVE-2014-4111, **CVE-2014-4123**, CVE-2014-4124, CVE-2014-4126, CVE-2014-4127, CVE-2014-4128, CVE-2014-4129, CVE-2014-4130, CVE-2014-4132, CVE-2014-4133, CVE-2014-4134, CVE-2014-4137, CVE-2014-4138, CVE-2014-4140 (ASLR Bypass), CVE-2014-4141, CVE-2014-4143, CVE-2014-6323, CVE-2014-6337, CVE-2014-6339 (ASLR Bypass), CVE-2014-6340, CVE-2014-6341, CVE-2014-6342, CVE-2014-6343, CVE-2014-6344, CVE-2014-6345, CVE-2014-6346, CVE-2014-6347, CVE-2014-6348, CVE-2014-6349, CVE-2014-6350, CVE-2014-6351, CVE-2014-6353, CVE-2014-6327, CVE-2014-6328, CVE-2014-6329, CVE-2014-6330, CVE-2014-6366, CVE-2014-6368 (ASLR Bypass), CVE-2014-6369, CVE-2014-6373, CVE-2014-6374, CVE-2014-6375, CVE-2014-6376, CVE-2014-8966 |

*Tabla 1*

Analizaremos con mayor detalle el aprovechamiento de vulnerabilidades de Internet Explorer más adelante, en la sección "Internet Explorer".

La tabla muestra que, en 2014, Microsoft solucionó aproximadamente el doble de vulnerabilidades que en el año pasado. La siguiente Imagen 1 representa estas estadísticas en forma visual. Microsoft aún soporta la vieja

(y totalmente insegura) versión del navegador Internet Explorer 6. Esta versión aún se sigue distribuyendo con Windows Server 2003. El soporte para este navegador finalizará en 2015.

La Tabla 2 muestra las vulnerabilidades resueltas y las actualizaciones publicadas para diversos tipos de componentes de Windows. Combinamos todos los componentes de modo de usuario de Windows (UMC, por sus siglas en inglés) en la sección "UMC de Windows". Y como se puede ver, también hay varias vulnerabilidades utilizadas por los atacantes con *exploits* 0-day. Hasta las sesiones mínimas de Windows ejecutan muchos servicios, por lo tanto, los atacantes pueden utilizar potencialmente las vulnerabilidades de los servicios del sistema para abrirse paso a través de él.

| Componente | Boletín | Tipo | Vulnerabilidad |
|---|---|---|---|
| **Windows UMC (VBScript, Direct2D, MSXML, DirectShow, SAMR, File Handling/kernel32.dll, Shell handler/shell32.dll, Remote Desktop, Journal, On-Screen Keyboard, Media center/mcplayer.dll, Installer, Task Scheduler, OLE, Message Queuing, Schannel, Kerberos, Audio Service, IIS, IME (Japanese), GDI+/gdi32.dll, RPC/rpcrt4.dll, Graphics/ windowscodecs.dll** | MS14-011, MS14-007, MS14-005, MS14-013, MS14-016, MS14-027, MS14-030, MS14-033, MS14-038, MS14-039, MS14-041, MS14-043, MS14-049, MS14-054, MS14-060, MS14-062, MS14-064, MS14-066, MS14-067, MS14-068, MS14-071, MS14-074, MS14-076, MS14-078, MS14-036, MS14-047, MS14-084, MS14-085 | Ejecución remota de código(11), Fuga de información(3), Bypass de Opciones de Seguridad(4), Escalación de privilegios(9), Tampering(1) | CVE-2014-0271, CVE-2014-0263, **CVE-2014-0266**, CVE-2014-0301, CVE-2014-0317, CVE-2014-0315, **CVE-2014-1807**, CVE-2014-1816, CVE-2014-0296, CVE-2014-1824, CVE-2014-2781, CVE-2014-2780, CVE-2014-4060, CVE-2014-1814, CVE-2014-4074, **CVE-2014-4114**, CVE-2014-4971, CVE-2014-6332, **CVE-2014-6352**, CVE-2014-6321, CVE-2014-4118, **CVE-2014-6324**, CVE-2014-6322, CVE-2014-6318, CVE-2014-4078, CVE-2014-4077, CVE-2014-1818, CVE-2014-0316, CVE-2014-6363, CVE-2014-6355 |
| **Win32k** | MS14-003, MS14-015, MS14-045, MS14-058, MS14-079 | Escalación de privilegios(4), Denegación de Servicio(1) | CVE-2014-0262, CVE-2014-0300, CVE-2014-0323, CVE-2014-0318, CVE-2014-1819, **CVE-2014-4113**, **CVE-2014-4148**, CVE-2014-6317 |
| **KM drivers (ndproxy.sys, tcpip.sys, afd.sys, fastfat.sys)** | MS14-002, MS14-006, MS14-031, MS14-040, MS14-045, MS14-063, MS14-070 | Escalación de privilegios(5), Denegación de Servicio(2) | **CVE-2013-5065**, CVE-2014-0254, CVE-2014-1811, CVE-2014-1767, CVE-2014-4064, CVE-2014-4115, CVE-2014-4076 |
| **.NET Framework** | MS14-009, MS14-026, MS14-046, MS14-053, | Escalación de privilegios(3), | CVE-2014-0253, CVE-2014-0257, **CVE-2014-0295** (ASLR Bypass), CVE-2014- |

| | | |
|---|---|---|
| MS14-057, MS14-072 | Bypass de Opciones de Seguridad(1), Denegación de Servicio(1), Ejecución remota de código(1) | 1806, CVE-2014-4062 (ASLR Bypass), CVE-2014-4072, CVE-2014-4073, CVE-2014-4121, CVE-2014-4122 (ASLR Bypass), CVE-2014-4149 |

*Tabla 2: Vulnerabilidades y revisiones*

La Imagen 1 representa la cantidad de vulnerabilidades reparadas este año para una amplia gama de componentes.

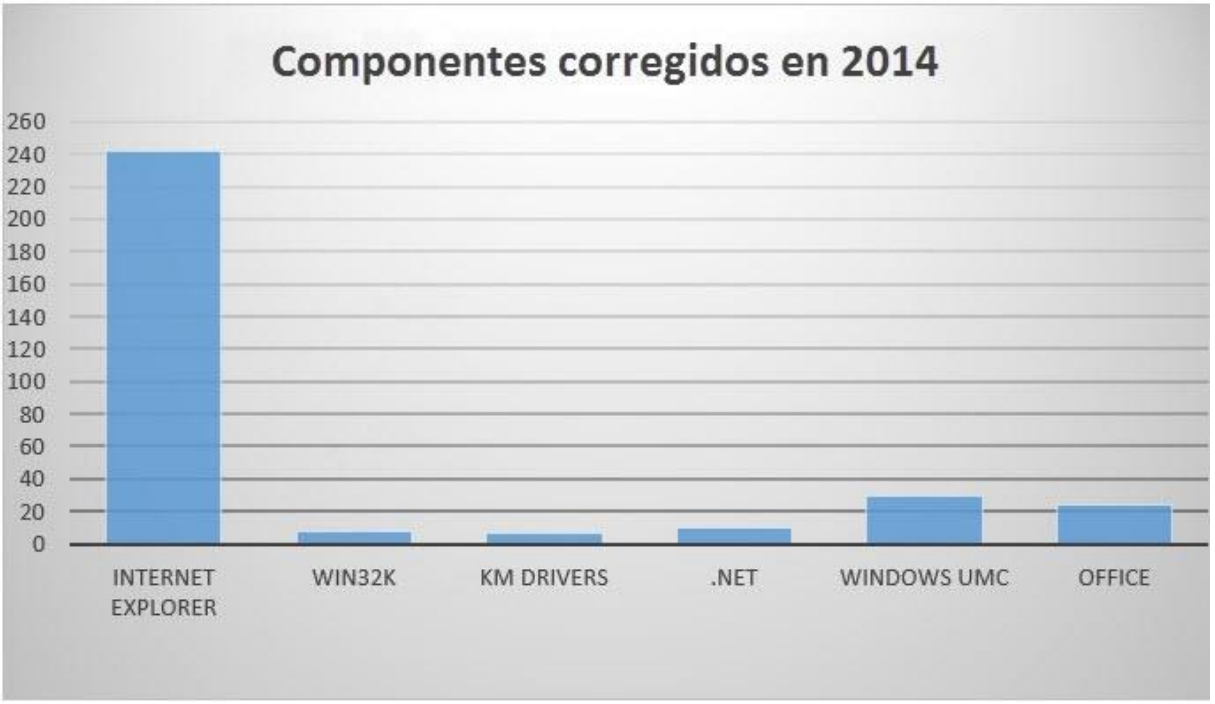



*Imagen 1*

Podemos ver que en 2014 se solucionaron una gran cantidad de vulnerabilidades en el navegador *web* Internet Explorer. Casi todas estas vulnerabilidades eran del tipo "ejecución remota de código" (RCE, por sus siglas en inglés). Esto significa que un atacante puede ejecutar código en forma remota en un entorno vulnerable, mediante la ayuda de una página *web* especialmente diseñada. Dicha página *web* puede contener un código especial, llamado *exploit*, para accionar una vulnerabilidad específica.

Normalmente, los atacantes usan esos *exploits* para instalar *malware* silenciosamente cuando detectan una

versión vulnerable de Windows. Este ataque es un ejemplo de una infección mediante una descarga desde una página *web* (*drive-by-download*) y es por eso que resaltamos ese tipo de aprovechamiento de vulnerabilidades como la tendencia principal en ataques a Internet Explorer, lo que se muestra abajo en la Imagen 2.

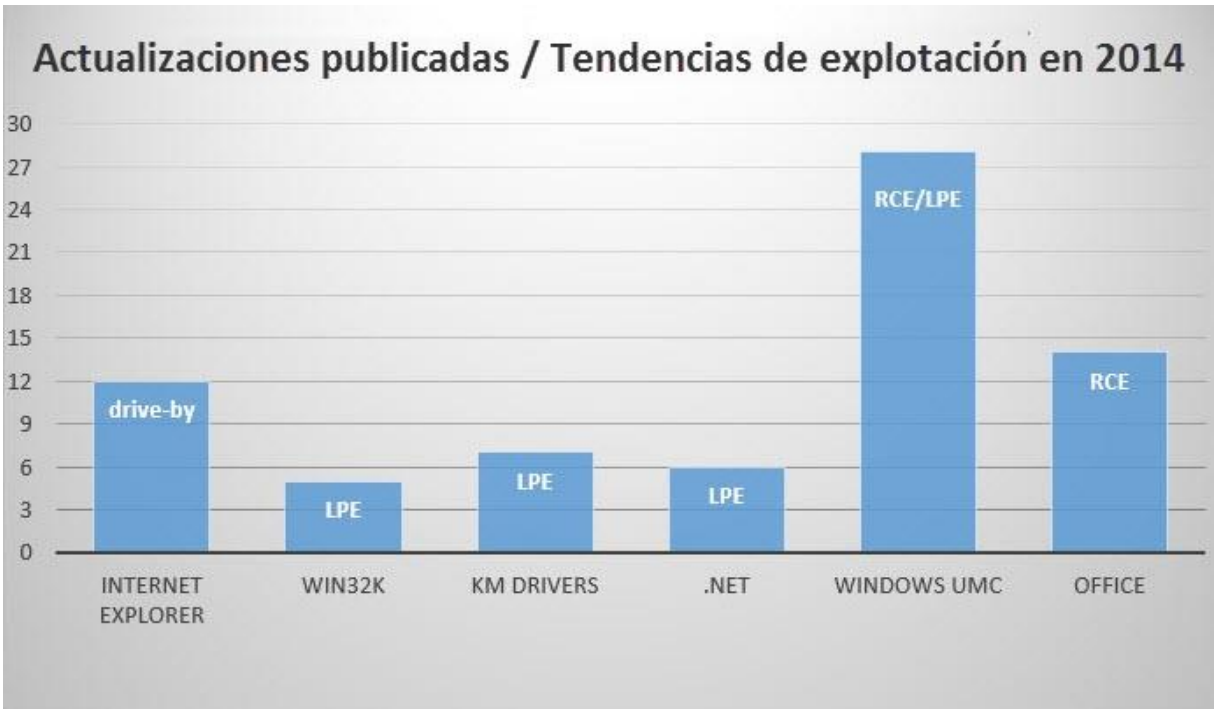



*Imagen 2*

Como mencionamos anteriormente, la infección por páginas *web* se refiere a la instalación silenciosa de *malware* a través de un *exploit* de ejecución remota de código (RCE). Hacemos la distinción entre vulnerabilidades RCE y por páginas *web* porque la última está relacionada a la instalación de *malware* desde un navegador, a diferencia de otros tipos de ejecución remota de código para los cuales, por ejemplo, se usan aplicaciones de Microsoft Office.

LPE significa en inglés *Local Privilege Escalation* (Escala de privilegios para usuarios locales) o lo que Microsoft llama *Elevation of Privilege*, EoP (Elevación de privilegios). Un atacante usa dichas vulnerabilidades para obtener el máximo nivel de acceso a cualquier recurso de Windows: por ejemplo, para trabajar bajo la cuenta *SYSTEM* que le da a un programa la capacidad de ejecutar código arbitrario en el modo *kernel* en versiones de Windows de 32 bits. Ambos tipos de ataque (infección por páginas *web* y LPE) se analizarán en mayor detalle más abajo, en la sección "Ataques de infección por páginas *web* y escala de privilegios para usuarios locales".

Podemos ver que el *driver* `win32k.sys` y otros controladores de Windows, resaltados en la columna llamada "KM drivers" (controladores del modo *kernel*), son componentes típicos que utilizan los atacantes para obtener los máximos privilegios dentro del sistema operativo. Los creadores de *malware* pueden emplear dichos *exploits* para evadir las restricciones integradas en Windows de manera que los atacantes puedan ejecutar el código de

modo *kernel* (también conocido como escape a las restricciones del modo de usuario). En otro escenario, un atacante puede usar dichos *exploits* en conjunto con los *exploits* RCE para evadir las restricciones del modo de *sandbox* del navegador *web*.

La comparación de la cantidad de vulnerabilidades resueltas en 2014 con las del año anterior (Imagen 3) es interesante e instructiva.

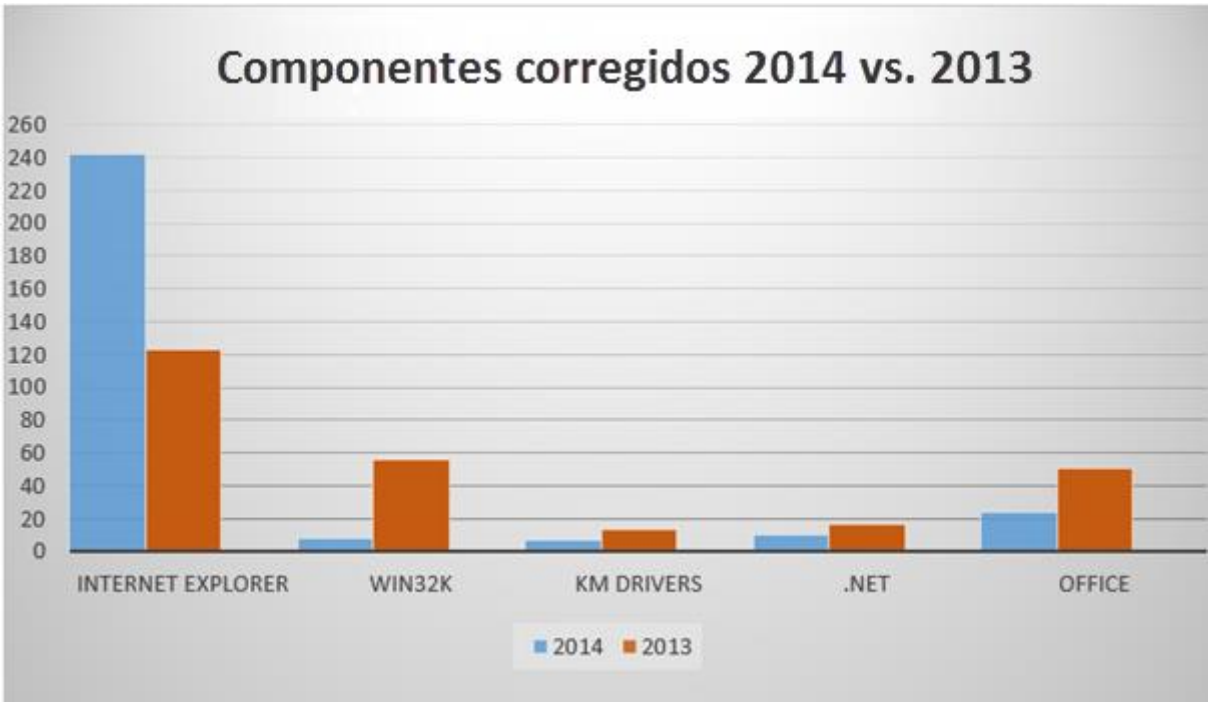



*Imagen 3*

Estas estadísticas nos muestran que en 2014 se resolvieron menos vulnerabilidades que en 2013 en todos los componentes y productos, con la excepción de Internet Explorer, para el cual hubo que lidiar con casi el doble de errores.

Las vulnerabilidades en Office también suelen ser el objetivo de ataques. Durante 2014, descubrimos varios ataques donde los atacantes utilizaron una vulnerabilidad en Microsoft Office y Windows para entregar software malicioso.

El Equipo de Investigación de ESET fue el primero en descubrir una notoria vulnerabilidad 0-day (CVE-2014-4114) en el administrador de paquetes OLE (packager.dll), que permitía la instalación de *malware* en el equipo de una víctima a través de una presentación de Microsoft PowerPoint especialmente diseñada. Mis colegas Robert Lipovsky y Anton Cherepanov hicieron un análisis detallado sobre una campaña maliciosa utilizada por los ciberdelincuentes para distribuir el *malware* BlackEnergy aprovechando esta vulnerabilidad.

# Explotación de vulnerabilidades

Durante el último año observamos muchas vulnerabilidades que fueron aprovechadas por los atacantes. Las tablas de la sección "Información general" ofrecen una visión real de las vulnerabilidades utilizadas en estos ataques. Nuestros analistas de *malware* están monitoreando de cerca esta situación y agregaron los *exploits* para estas vulnerabilidades a nuestras bases de datos de detección apenas las descubrieron como resultado de ataques a usuarios. Las siguientes tablas suministran información adicional sobre estas vulnerabilidades y las detecciones por el software de ESET creadas para los *exploits* correspondientes.

| CVE | Tipo | Componente | Vulnerabilidad | Parche | Bypass DEP&ASLR, otros |
|---|---|---|---|---|---|
| CVE-2014-0322 | Ejecución remota de código | Internet Explorer 10 | use-after-free | MS14-012 | Chequeo de ActionScript-heap-spray/ROP/EMET |
| CVE-2014-0502 | Ejecución remota de código | Flash Player | double-free | APSB14-07 | ActionScript-non-ASLR-hxds.dll/ROP<br>non-ASLR-msvcrt.dll/ROP<br>non-ASLR-msvcr71.dll/ROP |
| CVE-2014-1761 | Ejecución remota de código | Word 2003-2013 | memory-corruption | MS14-017 | non-ASLR-mscomctl.dll/ROP (<=Word 2010) |
| CVE-2014-1776 | Ejecución remota de código | Internet Explorer 6-11 | use-after-free | MS14-021 | ActionScript-heap-spray/ROP |
| CVE-2014-0515 | Ejecución remota de código | Flash Player | buffer-overflow | APSB14-13 | - |
| CVE-2014-0160 | Fuga de información | Windows 8.1 & RT 8.1 In-Box Junos Pulse Client (Juniper Networks) | Heartbleed | KB2962140 | - |
| | | Windows 2003 Server+ | | | |

| | | | | | |
|---|---|---|---|---|---|
| CVE-2014-4114 | Ejecución remota de código | OLE package manager (Packager.dll) | by design (bug) | MS14-060 | Instalación remota de malware vía .INF-file |
| CVE-2014-4113 | Escalación de privilegios | Win32k | integer-overflow | MS14-058 | Desreferencia de NULL pointer en x32<br><br>Desreferencia de wild pointer en x64 |
| CVE-2014-6352 | Ejecución remota de código | Windows Vista+<br><br>OLE package manager (Packager.dll) | MS14-060 fail | MS14-064 | Instalación de malware vía OLE-object malicioso |
| CVE-2014-6332 | Ejecución remota de código | Windows 2003 Server+<br><br>Windows OLE<br><br>(OleAut32.dll) | by design (bug) | MS14-064 | Bypasses (por diseño) DEP&ASLR |

*Tabla 3*

Aquí podemos ver *exploits* para Internet Explorer, el complemento para Adobe Flash Player, Microsoft Word y diversos componentes de Windows.

| Vulnerabilidad in-the-wild | Detección de ESET | Mes | Ataque dirigido |
|---|---|---|---|
| CVE-2014-0502 | SWF/Exploit.CVE-2014-0502 | Febrero | Sí |
| CVE-2014-1761 | Win32/Exploit.CVE-2014-1761 | Marzo | Sí |
| CVE-2014-1776 | Win32/Exploit.CVE-2014-1776<br>SWF/Exploit.CVE-2014-1776<br>JS/Exploit.Agent.NGS | Abril | Sí |
| CVE-2014-0515 | SWF/Exploit.CVE-2014-0515 | Abril | Sí |
| CVE-2014-4114 | Win32/Exploit.CVE-2014-4114 | Octubre | Sí |
| CVE-2014-4113 | Win64/Dianti.A<br>Win32/Dianti.A | Octubre | Sí |
| CVE-2014-6352 | Win32/Exploit.CVE-2014-6352.A | Octubre | Sí |
| CVE-2014-6332 | Win32/Exploit.CVE-2014-6332.A | Noviembre | No |

*Tabla 4*

La columna de la derecha (Ataque dirigidos) demuestra lo que ya mencionamos al comienzo de este artículo: los atacantes crean *exploits* 0-day para usar en ataques dirigidos a sectores específicos. La detección de la última vulnerabilidad de la lista, CVE-2014-6332, se agregó luego de que Microsoft corrigiera la vulnerabilidad, por lo que no es estrictamente 0-day.

Como hemos visto, Microsoft incorporó técnicas de mitigación de *exploits* a todas las nuevas versiones de Windows. Las más importantes y ya analizadas muchas veces en el pasado son la Prevención de Ejecución de Datos (DEP, por sus siglas en inglés) y la Selección Aleatoria del Diseño del Espacio de Direcciones (ASLR, por

sus siglas en inglés).

Para evadir la DEP, los atacantes recurrieron al uso de varios tipos de dispositivos de Programación Orientada al Retorno (ROP, por sus siglas en inglés) que podían ubicarse fácilmente en archivos DLL compilados sin soporte para ASLR. Estos dispositivos de ROP representan fragmentos de código que pueden ayudar a los atacantes a modificar la protección de páginas de memoria a través de *Shellcodes*. Descubrimos el uso de la ROP para evadir la DEP en un *exploit* de Adobe Flash Player: hay más detalles en la investigación publicada por mi colega Sébastien Duquette.

Otra tecnología bastante conocida para reducir la eficacia de *exploits* es ASLR (la selección aleatoria del diseño del espacio de direcciones). Constituye una forma de impedir el acceso mediante *Shellcodes*, ya que elige ubicaciones aleatorias para los datos de un programa, incluyendo las direcciones que utilizan los *Shellcodes* maliciosos. De esta forma, el atacante no puede predecir con certeza la dirección correcta en memoria donde se necesita colocar la *Shellcode* maliciosa, ni tampoco crear las condiciones favorables para usar las vulnerabilidades. Por lo tanto, la característica de seguridad de ASLR protege a los usuarios ante una amplia gama de vulnerabilidades y les genera problemas a los atacantes, lo que incrementa el costo total de desarrollo de un *exploit* confiable.

Lamentablemente, Windows y sus componentes (así como también .NET Framework o Microsoft Office) pueden contener archivos DLL no seguros heredados que no fueron compilados con opciones seguras. Estas bibliotecas ejecutables son muy útiles para los atacantes, dado que están ubicadas en direcciones de memoria predecibles. En la Tabla 5, abajo, podemos ver varias vulnerabilidades de este tipo, la mayoría sin protección: es decir, compiladas sin soporte para la ASLR.

| Vulnerabilidad (Bypass de ASLR) | Producto / Componente | Detalles |
|---|---|---|
| CVE-2014-0295 (MS14-009) | .NET Framework 2.0 SP2, .NET Framework 3.5.1 | Vulnerabilidad en vsavb7rt.dll Siendo explotada In-The-Wild (ITW) |
| CVE-2014-4062 (MS14-046) | .NET Framework 2.0 SP2 – 3.5.1 | - |
| CVE-2014-4122 (MS14-057) | .NET Framework 2.0 SP2 – 3.5.1 | - |

| | | |
|---|---|---|
| CVE-2014-0319<br>(MS14-014) | Silverlight 5 | - |
| CVE-2014-1809<br>(MS14-024) | Windows Common Control<br><br>MS Office 2007 - 2013 | Vulnerabilidad en mscomctl.ocx<br><br>Siendo explotada In-The-Wild (ITW) |
| CVE-2014-0316<br>(MS14-047) | Windows 7 – 8.1 / RT 8.1<br><br>Local RPC (LRPC) | Bypass implícito de ASLW mediante out-of-process memory-spray |
| CVE-2014-4140<br>(MS14-056) | Internet Explorer 9 - 11 | - |
| CVE-2014-6339<br>(MS14-065) | Internet Explorer 8-9 | - |

*Tabla 5*

La vulnerabilidad CVE-2014-0316 es particularmente interesante, ya que los atacantes pueden utilizarla para evadir la ASLR "en forma remota" (fuera de proceso) desde otro proceso mediante el envío de solicitudes de LRPC (del inglés, llamada a procedimiento remoto local) especialmente diseñadas. Los atacantes pueden usar esta vulnerabilidad en conjunto con una vulnerabilidad de RCE para facilitar el proceso de inserción al sistema desde el navegador.

# Ataques de infección por páginas web y escalación de privilegios para usuarios locales

En el presente, las infecciones por páginas *web* representan un tipo de ataque típico que usan los criminales para ejecutar códigos maliciosos en forma remota. Como ya sabemos, en general usan varios tipos de Exploit Kits (EK) para redirigir a los usuarios y poder habilitar la instalación de *malware*. En este contexto, los criminales pueden introducir contenido malicioso e infectar un sitio *web* legítimo, que redirigirá a los usuarios a una página de destino donde están presentes los *exploits*.

Un excelente ejemplo de cómo se puede implementar una infección por páginas *web* fue el que dieron mis colegas en el artículo titulado "Operación Windigo". Los criminales pueden usar cuentas robadas de servidores para infectar los sistemas Linux con archivos binarios maliciosos. También pueden distribuir dichos archivos

binarios como módulos adicionales, por ejemplo, para el *software* Apache. Una vez que se infectó el servidor, el *malware* consigue el control total de los sistemas de los visitantes que accedieron a los sitios *web* ejecutados en el servidor y así pueden inyectar código HTML malicioso en páginas legítimas. Este es uno de los casos que le permite a un atacante organizar infecciones por páginas *web*.

Durante el año pasado, también escribimos muchas veces sobre diversos grupos que utilizan las infecciones por páginas *web* como el vector primario para la propagación de *malware*. Por ejemplo, el grupo de espionaje Sednit usaba Exploit Kits para hacer infecciones por páginas *web* y colocar *software* malicioso en los sistemas de los usuarios corporativos de Europa Oriental. Este grupo utilizaba diversos *exploits* de Microsoft Internet Explorer para desplegar programas de puerta trasera en los equipos infectados.

Existen solo dos casos en que los atacantes suelen utilizar las vulnerabilidades de escalación de privilegios para usuarios locales (LPE) o la Elevación de privilegios (EoP). En el primero, un *exploit* usado para atacar mediante una infección por páginas *web* puede emplear dicha vulnerabilidad en conjunto con una vulnerabilidad de RCE para evadir el modo de sandbox del navegador. La *sandbox* no permite que la *Shellchode* instale *software* malicioso, por lo que los atacantes necesitan una vulnerabilidad adicional que les permita evadir esta restricción.

En el segundo caso, el *malware* puede usar una vulnerabilidad LPE para evadir las restricciones de modo de usuario de Windows e insertar código arbitrario en el modo *kernel* (Ring 0) en versiones de Windows de 32 bits. Uno de esos tipos de *malware* es el *bootkit*, que puede usar un *exploit* de LPE para cargar su propio controlador en la memoria. Mis colegas Eugene Rodionov, Aleksander Matrosov y David Harley analizaron los *bootkits* en su presentación de VB2014 Bootkits: pasado, presente y futuro.

Este año observamos otro *exploit* de LPE (CVE-2014-4113) con una notable técnica para ejecutar código arbitrario en el modo *kernel* con la ayuda de una desreferencia de puntero NULL (inicializado con valor igual a cero), detectado por ESET como *Win32/Dianti.A* y *Win64/Dianti.A*. Un *exploit* aprovecha un error en la función *win32k!xxxHandleMenuMessages* del controlador `win32k.sys`. Además, también puede funcionar en una versión de Windows de 64 bits por su desreferencia de puntero WILD (no inicializado en ningún valor).

```
1 signed int __usercall fnPrepareExploitation@<eax>(int Addr@<esi>)
2 {
3   int pWin32ThreadInfoStruct; // ebx@1
4   HANDLE v2; // eax@7
5   int v4; // [sp+4h] [bp-4h]@7
6
7   pWin32ThreadInfoStruct = fnCallPtiCurrentAndGetTebWin32ThreadInfo();
8   if ( !pWin32ThreadInfoStruct )
9   {
10    GetLastError();
11    fnLogMessage("[%d] Failed, %08X\n", 25);
12  }
13  if ( !ZwAllocateVirtualMemory )
14  {
15    GetLastError();
16    fnLogMessage("[%d] Failed, %08X\n", 31);
17  }
18  v4 = 0x2000;
19  *(_DWORD *)Addr = 1;
20  v2 = GetCurrentProcess();
21  if ( ZwAllocateVirtualMemory(v2, Addr, 0, &v4, 1060864, 64) )
22  {
23    GetLastError();
24    fnLogMessage("[%d] Failed, %08X\n", 46);
25  }
26  fnFillMaliciousTagWnd(pWin32ThreadInfoStruct);
27  return 1;
28 }
```

*Imagen 4*

En la captura de pantalla anterior se puede ver la función del *exploit* que genera respuestas para aprovechar la vulnerabilidad CVE-2014-4113 (*Win32/Dianti.A*). Este es un ejemplo típico del uso de asignación de memoria en una página nula (en versiones de Windows de 32 bits). El código malicioso asigna memoria en esta página, donde se almacena una estructura win32k especialmente diseñada llamada win32k!tagwnd.

Esta estructura contiene un puntero a una función de devolución de llamada a la que el código legítimo en el modo *kernel* llama durante el aprovechamiento de la vulnerabilidad. También llama a la función especial *fnCallPtiCurrentAndGetTebWin32ThreadInfo* para recuperar un puntero legítimo que se utilizará para inicializar el componente malicioso *win32k!tagwnd*. El formato de esta estructura se muestra en la siguiente imagen (Windows 7 de 64 bits).

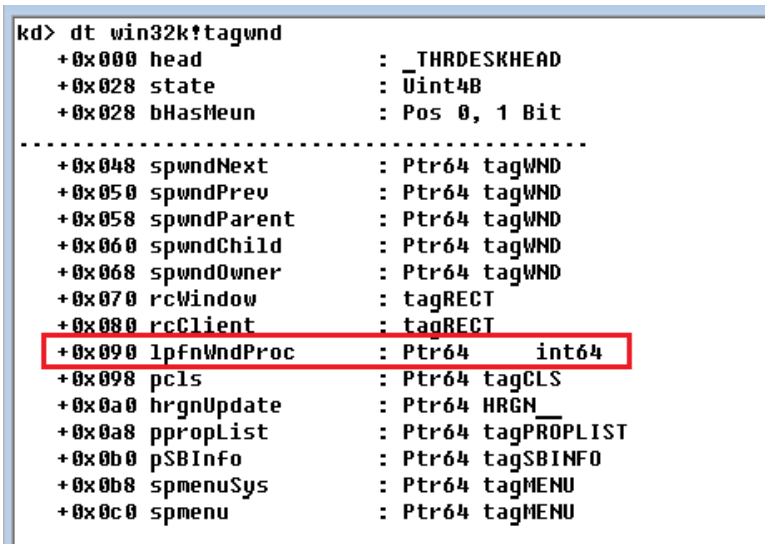



*Imagen 5*

Como ya habíamos mencionado en el informe del año pasado "Aprovechamiento de vulnerabilidades de Windows en 2013", Microsoft agregó una funcionalidad de seguridad para Windows 8 que impide la asignación de memoria en las páginas nulas. Para los usuarios de Windows 7, esta característica de seguridad comenzó a estar disponible con la actualización MS13-031. Además, las versiones de 64 bits de Windows 8 usan otra funcionalidad de seguridad llamada Prevención de ejecución en modo de supervisor (Supervisor Mode Execution Prevention, SMEP). La SMEP impide la ejecución de las páginas de memoria del modo de usuario con código del modo de *kernel*.

Como podemos ver en la tabla de la sección "Información general", este año, Microsoft resolvió muy pocas vulnerabilidades para `win32k.sys`. Pero de todos modos, este controlador tuvo muchas vulnerabilidades que ya se resolvieron hace unos años. Esta situación fue posible probablemente porque el código `win32k.sys` antes estaba ubicado en archivos DLL en modo de usuario.

Previamente a la aparición de Windows NT 4.0, cuando se incluía este controlador como parte del sistema operativo, el subsistema completo de la GUI se ubicaba en bibliotecas y procesos especiales de modo de usuario. Pero en Windows NT 4.0, parte de este código se fusionaba con el modo de *kernel* y el controlador `win32k.sys`. Por eso es posible que contenga tantas vulnerabilidades, ya que algunos fragmentos de código se tomaron directamente desde las bibliotecas de modo de usuario, quizá sin siquiera hacer controles adicionales.

# Internet Explorer

En 2014, Microsoft resolvió una cantidad muy importante de vulnerabilidades para su navegador. Estas fallas eran del tipo RCE, y suelen ser utilizadas por los atacantes para instalar silenciosamente diversos tipos de

*malware*. Lo que es más, siete de estas vulnerabilidades fueron aprovechadas antes de que existiera revisión alguna (*in-the-wild*). Las versiones más nuevas de Internet Explorer cuentan con características especiales de seguridad que ayudan al usuario a mitigar estos tipos de ataques. Observemos los detalles.

En primer lugar, deberíamos mencionar el Modo protegido mejorado (EPM, por sus siglas en inglés), que se incorporó con la versión de Internet Explorer 10 (IE10). El EPM es una *sandbox* completa para el navegador y cuenta con la capacidad de aislar las pestañas activas del navegador desde los recursos de Windows, como las *sandboxes* para aplicaciones que también están presentes en casi todos los sistemas operativos modernos, incluyendo Mac OS X e iOS de Apple, así como Android de Google.

El EPM solo es soportado por Windows 8+, ya que está relacionado a un mecanismo de *kernel* especial de Windows llamado AppContainer. Es fácil entender que AppContainer es una extensión del mecanismo de Nivel de integridad (o *caja de arena parcial*) conocido como Modo protegido, que fue incorporado en Internet Explorer 7, y que se lanzó en el año 2006. Hay que recordar que IE10+ EPM se basa en las características de *kernel* incorporadas en Windows, por lo que es imposible usar esta funcionalidad de mitigación en versiones de Windows anteriores a Windows 8, ya que AppContainer no cuenta con soporte en versiones más viejas. La única excepción es Windows 7 de 64 bits, donde el EMP activa los procesos de 64 bits para el navegador.

Como hemos observado en los últimos dos años, los atacantes pueden aprovechar las DLL que no usan la ASLR para crear *exploits* más resistentes o "para evadir la ASLR en forma predeterminada". En IE10, Microsoft introdujo una opción especial llamada ForceASLR. Al igual que el EPM, ForceASLR se basa en las innovaciones del *kernel* de Windows, presentes en Windows 8 en forma predeterminada o en Windows 7 si se aplicó la actualización especial KB2639308. Esta opción aplica la ASLR en forma predeterminada en todos los archivos DLL, que se cargan en el contexto de los procesos del navegador, incluso cuando dichos módulos no estaban compilados con soporte para la ASLR (opción `/DYNAMICBASE`). Esta opción corresponde a la opción *ASLR obligatoria* de EMET.

Analizaremos EMET con mayor detalle en la sección "Mitigadores". De hecho, se puede usar la opción ForceASLR de Windows 8+ para indicarle a un cargador de archivos ejecutables que debe aplicar la opción ASLR en forma predeterminada para todos los DLL que se carguen en un proceso específico.

Hay que recordar que Internet Explorer (incluyendo la última versión, IE11) funciona en forma predeterminada como un programa de 32 bits, incluso en sistemas de 64 bits. Es necesario activar manualmente la opción correspondiente (Habilitar procesos de 64 bits para el Modo de protección mejorado). Hay más información sobre esta opción en las publicaciones Aprovechamiento de vulnerabilidades de Windows en 2013 y Protección ante *exploits* para Microsoft Windows.

El uso de IE en la configuración predeterminada reduce significativamente su resistencia al aprovechamiento de vulnerabilidades. En un espacio de direcciones de 32 bits, es considerablemente más fácil evadir la ASLR con la técnica para facilitar la ejecución de código arbitrario conocida como *heap spraying*, mientras que en espacios

de direcciones de 64 bits, esto resulta casi imposible.

Como podemos ver en la Tabla 3, arriba, y según el análisis de nuestro informe anterior, las vulnerabilidades que usan un objeto previamente liberado (*use-after-free* o UAF) son comunes no solo para IE, sino también para muchos componentes de Windows. En una situación normal, el navegador funciona con memoria según los siguientes pasos:

1) El navegador asigna un bloque de memoria en la pila de memoria (*heap memory*)
2) El navegador usa el bloque de memoria
3) El navegador libera el bloque de memoria

Si el código del navegador contiene una vulnerabilidad de UAF, se modifica la conducta normal y, tras la eliminación del bloque correspondiente, se referencia otra vez.

1) El navegador asigna un bloque de memoria en la pila de memoria (*heap memory*)
2) El navegador usa el bloque de memoria
3) El navegador libera el bloque de memoria
4) El navegador hace referencia al bloque liberado en forma reiterada

En consecuencia, un *exploit* puede utilizar esta situación para sus propios propósitos.

1) El navegador asigna un bloque de memoria en la pila de memoria (*heap memory*)
2) El navegador usa el bloque de memoria
3) El navegador libera el bloque de memoria
4) El usuario visita una página *web* con el *exploit*
5) El *exploit* utiliza la técnica *heap spraying* (para evadir la ASLR) y llena los bloques asignados con la *Shellcode* o con direcciones especiales requeridas para accionar la vulnerabilidad
6) El *exploit* crea condiciones especiales para el navegador con el objetivo de forzarlo para que haga referencia a un puntero no válido, que ya fue validado en el paso 4
7) El navegador se refiere en forma reiterada al bloque liberado anteriormente, que ya es válido, y acciona la vulnerabilidad. A continuación, el código malicioso obtiene el control, antes de ejecutar los dispositivos de ROP para evadir la DEP, por ejemplo, mediante *ntdll!NtProtectVirtualMemory*.

Para proteger el código del navegador que potencialmente no está a salvo de dichas vulnerabilidades, Microsoft incorporó una medida de protección especial llamada Anti-UAF con algoritmo de liberación de montón de memoria aislado y diferido. Es fácil comprender que la liberación diferida de bloques de montón de memoria es una solución interesante, ya que, desde el punto de vista de Windows, el bloque de memoria liberado aún sigue ocupado, por lo que un atacante no podrá apropiarse de él para su propio uso.

En la Tabla 6, abajo, se pueden ver las actualizaciones que introdujeron estos mitigadores anti-UAF. Lo mismo se

aplica al heap de memoria aislada, ya que su misión es aislar del *stack* la asignación de memoria para los objetos del navegador críticos para la seguridad, desde donde los atacantes pueden encontrar los datos necesarios de una manera más rápida o simplemente utilizarlos sin mayor esfuerzo con el fin de explotar una vulnerabilidad.

| Innovación | Comienzo en | Descripción |
|---|---|---|
| Enhanced Protected Mode | IE10 | Para Windows 7 x64 habilitar las dirección virtuales de 64 bits para los procesos de las Tabs.<br><br>Para Windows 8+ x32/x64 habilitar el AppContainer para Sandboxing |
| ForceASLR | IE10 (Windows 8+ or Windows 7 w/ KB2639308) | Fuerza la aplicación de ASLR para todos los módulos cargados por el navegador |
| 64-bit tabs | IE11 | Para Windows 7+ x64 habilitar el espacio de direcciones virtuales para los procesos de las Tabs |
| Protected Heap & Delay Free (Anti-UAF) | IE11 (MS14-035, MS14-037) | Introduce prácticas de mitigación para exploits que disparan vulnerabilidades del tipo use-after-free |
| Out-of-date ActiveX control blocking | IE 8-11 on Windows 7 SP1+<br><br>Oracle Java & MS Silverlight Plugins (KB2991000) | Bloquea la carga de componentes de Oracle Java y MS Silverlight plugins para Internet Explorer |

*Tabla 6*

Otra característica de seguridad de IE incorporada este año se llama Bloqueo de control ActiveX obsoleto. Como lo dice su nombre, esta característica se especializa en bloquear las versiones obsoletas de varios complementos del navegador. Hoy en día, esta funcionalidad puede bloquear las versiones obsoletas de los complementos de Oracle Java y Microsoft Silverlight. Esta característica es capaz de bloquear una amplia gama de *exploits*, según el aprovechamiento malicioso de las versiones viejas de los complementos vulnerables. La funcionalidad se puso a disponibilidad de los clientes junto con las actualizaciones del segundo martes de agosto de 2014.

Lamentablemente, es posible evadir la característica de seguridad mencionada de *sandbox* completa (EPM) para IE11 con algunas tretas especiales. Como lo demostraron los investigadores del equipo de Project Zero de Google, el navegador presenta debilidades en su modelo de seguridad. Como lo informó Google, los atacantes pueden usar la vulnerabilidad de escape de la *sandbox* CVE-2014-6350 (MS14-065) para evadir esta importante

característica de seguridad. La *sandbox* de IE usa medidas basadas en un proceso agente especial que se puede abrir para las operaciones de lectura de memoria y, potencialmente, se puede usar para la divulgación de información sobre los datos internos de la *sandbox* y el aprovechamiento subsecuente de la vulnerabilidad.

Este proceso agente puede suministrar acceso especial a los recursos de Windows para las pestañas (procesos) que están en modo *sandbox*. Otro método de aprovechamiento de vulnerabilidades, también mencionado por los investigadores de Google, se basa en un objeto de sección especial utilizado por todos los procesos en el árbol de procesos activos de IE para compartir entre ellos las configuraciones necesarias.

# Mitigadores

Más arriba mencionamos diversas técnicas para la mitigación de *exploit*s. Ahora analizaremos otra característica de seguridad opcional y cómo puede usarse para fortalecer la protección ante *exploit*s. En 2014, Microsoft lanzó una nueva versión de su famoso kit de herramientas de seguridad EMET (la versión actual es 5.1). Para quien no esté familiarizado con esta herramienta, puede encontrar información al respecto en el sitio de soporte de Microsoft.

En la nueva versión de EMET, Microsoft incorporó algunas características de seguridad muy útiles. Estas funcionalidades se llaman Reducción de la superficie de ataque (Attack Surface Reduction, ASR) y Filtrado de tablas de exportación de direcciones Plus (Export Address Table Filtering Plus, EAF+). La ASR es similar a la opción de IE llamada *Bloqueo de control ActiveX obsoleto*, pero es capaz de cubrir una gama mucho más amplia de *exploits*. Así como la opción mencionada de IE solo puede bloquear la carga de los complementos obsoletos al espacio de direcciones de los procesos, la ASR puede bloquear la carga de *todos* los módulos especificados a los espacios de direcciones de los siguientes procesos: Microsoft Internet Explorer, Excel, Word y PowerPoint.

En forma predeterminada, la ASR bloquea la carga de los módulos que suelen ser utilizados por los atacantes para realizar infecciones por páginas *web* a estos procesos, incluyendo los complementos de Oracle Java (`npjpi*.dll`, `jp2iexp.dll`), Vector Markup Language DLL (`vgx.dll`) (SA 2963983) y Flash Player (`flash*.ocx`). Por lo tanto, si se usa esta opción (que ya viene habilitada en forma predeterminada) para estos programas, no se podrá ejecutar el contenido correspondiente. El grupo específico de módulos de bloqueo depende de las aplicaciones específicas. Esta opción funciona para algunas zonas de IE que no son de confianza, de modo que en la Zona de Intranet, el usuario puede ver el contenido correcto. La Tabla 7 abajo muestra los módulos bloqueados por la ASR, que no pueden cargarse en el contexto del proceso correspondiente.

| Proceso | Opción ASR |
|---|---|

| | |
|---|---|
| Internet Explorer (iexplore.exe) | npjpi*.dll;jp2iexp.dll;vgx.dll;msxml4*.dll;wshom.ocx;scrrun.dll |
| PowerPoint (powerpnt.exe) | flash*.ocx |
| Word (winword.exe) | flash*.ocx |
| Excel (excel.exe) | flash*.ocx |

*Tabla 7*

Una opción más reciente de EMET se llama EAF+. Mejora la opción existente llamada EAF y puede funcionar independientemente de la versión anterior. Como ya sabemos, al *Shellcode* de los *exploits* le interesa recuperar las direcciones de varias funciones exportadas desde los módulos del sistema como `ntdll.dll` durante el tiempo de ejecución mediante el análisis de la tabla de exportación de direcciones (*export address table* o EAT) de dichos módulos. La opción de EAF normal bloquea los intentos de la *Shellcode* para obtener acceso de escritura y poder ver las páginas de memoria donde está ubicada la EAT. Esto es específico para las EAT de `kernel32.dll` y `ntdll.dll`. El EAF+ extiende la protección de la EAT y también protege la EAT de `kernelbase.dll`.

Además, esta nueva opción también puede mitigar técnicas especiales de una *Shellcode*, que involucran el uso de dispositivos de ROP desde bibliotecas que ya se sabe que evaden la opción EAF original. Esto significa que el acceso de un *exploit* a la EAT se llevará a cabo utilizando un código conocido, que no es malicioso en sí y que representa dispositivos de ROP específicos de bibliotecas legítimas. El EAF+ puede controlar esta situación con el agregado de una característica especial, que le permite al kit EMET reconocer dichos tipos de ataques. Además de impedir que todo código desconocido acceda a la EAT, también bloquea los intentos de acceso del código de bibliotecas legítimas que podrían estar siendo usadas por atacantes para explorar la EAT. En forma predeterminada, la opción EAT+ está habilitada en los procesos de Internet Explorer y Adobe Reader.

La Tabla 8 a continuación muestra una lista de módulos específicos: EMET bloqueará su código para que no pueda acceder a la EAT de kernel32, ntdll, y kernelbase.

| Proceso | Opción EAF+ |
|---|---|

| | |
|---|---|
| Internet Explorer (iexplore.exe) | mshtml.dll;flash*.ocx;jscript*.dll;vbscript.dll;vgx.dll |
| Adobe Acrobat (acrobat.exe, acrord32.exe) | AcroRd32.dll;Acrofx32.dll;AcroForm.api |

*Tabla 8*

En realidad, en la práctica, las técnicas de mitigación mencionadas arriba funcionan con configuraciones de Windows especiales. La Tabla 9, abajo, demuestra que a veces es difícil comprender qué opción puede resultar útil ante un ataque específico. Observemos la vulnerabilidad CVE-2014-1776 como ejemplo.

| Opción | ¿Efectiva? | Detalles |
|---|---|---|
| Enhanced Protected Mode IE10 & IE11 (EPM) on Windows 7 x32 | No | No sirve por diseño |
| Enhanced Protected Mode IE10 & IE11 (EPM) on Windows 8+ x32 | No | El Aislamiento del AppContainer Sanbox es insuficiente para bloquear las acciones del exploit (Sirve en conjunto con tabs de 64 bits) |
| Enhanced Protected Mode IE10 (EPM) on Windows 8+ x64 | No | IE10 no contiene la opción de Tabs en 64 bits |
| Enhanced Protected Mode IE10 (EPM) on Windows 7 x64 | Sí | En lugar del AppContainer, habilitar las direcciones de memoria virtual de 64-bits para las tabs del navegador |
| Enhanced Protected Mode IE10 (EPM) on Windows 7+ x64 & 64-bit tabs | Sí | Utilizado por malware en ActionScript que utiliza Heap Spray desde un objeto de Flash Player no es efectiva en direcciones de 64-bits |
| EMET 5 ASR | Sí | Bloquea la caga de VGX.dll & Flash (.ocx) en el proceso de IE para la zona de <Internet> |
| EMET 5 EAF+ | Sí | No le permite a una Shellcode obtener acceso a las páginas de memoria con exports de ntdll.dll |

| | | |
|---|---|---|
| EMET 5, 4.x Heap Spray | Sí | Bloquea ataques de Heap Spray en ActionScript utlizados por un Exploit (bypass de ASLR) |
| EMET 5, 4.x ROP (StackPivot, Caller, MemProt) | Sí | Solo con la opción de Deep Hooks habilitada |

*Tabla 9*

Otra opción interesante de EMET 5.1 que viene activada en forma predeterminada es Deep Hooks. Por ejemplo, esta opción fue útil para bloquear el notable *exploit 0-day* que aprovechaba la vulnerabilidad CVE-2014-1776. Se dan más detalles al respecto en el artículo Más detalles sobre el aviso de seguridad 2963983 0-day para IE.La opción Deep Hooks le permite a EMET monitorear con mayor detalle diversas operaciones mediante un enlace adicional de la API de Windows y la API Interna (ntdll). Se pueden encontrar ejemplo de dichos enlaces en la Tabla 10, abajo.

| | Deep Hooks activada | Deep Hooks desactivada |
|---|---|---|
| **Funciones de memoria** | *kernel32!VirtualAlloc* | *ntdll!NtAllocateVirtualMemory* |
| | | *kernelbase!VirtualAlloc* |
| | *kernel32!VirtualAllocEx* | *ntdll!NtAllocateVirtualMemory* |
| | | *kernelbase!VirtualAllocEx* |
| | *kernel32!VirtualProtect* | *ntdll!NtProtectVirtualMemory* |
| | | *kernelbase!VirtualProtect* |
| | *kernel32!VirtualProtectEx* | *ntdll!NtProtectVirtualMemory* |
| | | *kernelbase!VirtualProtectEx* |

*Tabla 10*

Como podemos ver en la Tabla 10, la opción Deep Hooks activa el enlace de las funciones del nivel inferior de Windows que llamaron desde la API controlada. Por ejemplo, si se habilita esta opción, EMET no solo controlará la API *kernel32!VirtualAlloc*, sino también las funciones *ntdll!NtAllocateVirtualMemory* y *kernelbase!VirtualAlloc*. La siguiente captura de pantalla muestra estas funcionalidades de seguridad de EMET en su interfaz GUI.

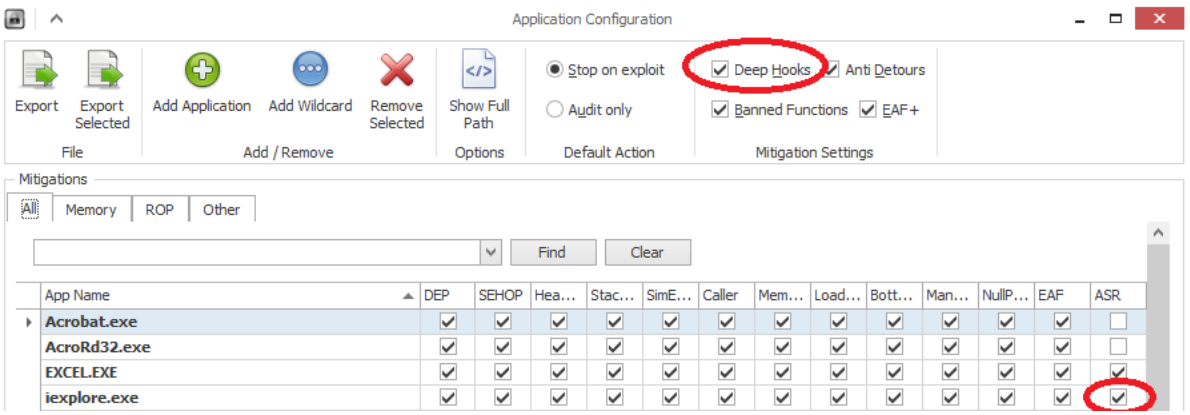



*Imagen 6*

# En vez de una conclusión

En definitiva, las mejoras en los mitigadores contra *exploits* del tipo RCE generan un incremento de costos para el desarrollo de *exploits*. Queda demostrado por el siguiente diagrama. De esta manera, los atacantes necesitan invertir más dinero para investigar nuevas vulnerabilidades que los ayuden a evadir las funcionalidades de seguridad contra *exploits*. Como lo demostramos en el presente informe, hoy en día necesitan un grupo de dos o incluso tres para abrirse paso en el sistema y obtener el control completo del equipo.

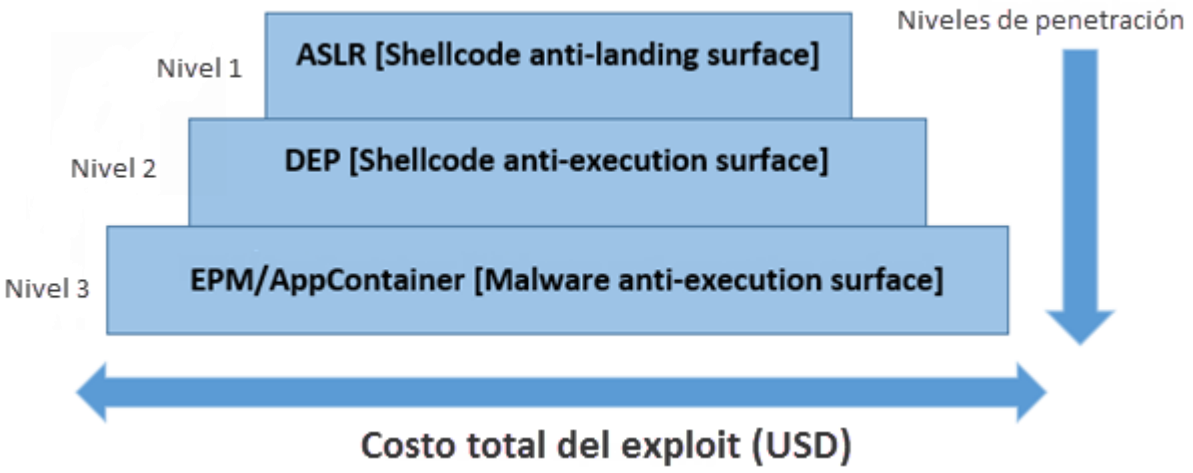
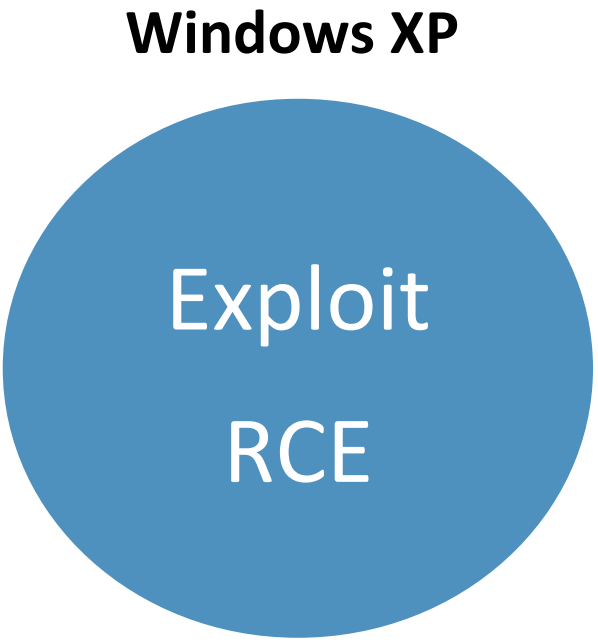
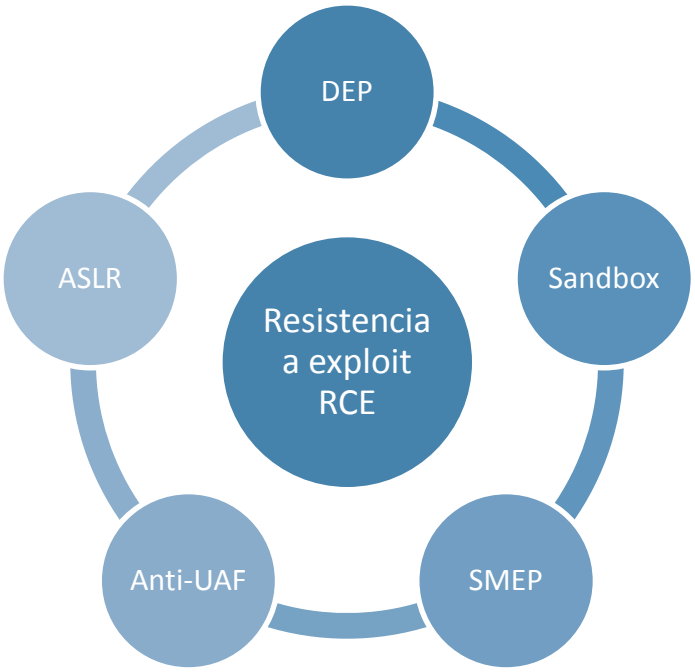



*Imagen 7*

Lamentablemente, muchos usuarios aún utilizan versiones no seguras de Windows, como Windows XP. Estas versiones de Windows no incluyen las características de seguridad modernas contra *exploits* aquí descritas y el usuario debe comprender que al utilizar versiones viejas está exponiendo el sistema a un riesgo significativo. En las siguientes imágenes se puede ver la comparación de los *exploits* RCE con Windows XP y Windows 8.1. Alcanza con decir que el usuario al menos debería pensarlo dos veces antes de seguir usando Windows XP.



# Windows 8.1




Para el próximo año podemos predecir que los ataques de infección por páginas *web* seguirán siendo el mecanismo principal para aprovechar vulnerabilidades y entregar códigos maliciosos. Dada la complejidad significativa y en crecimiento constante del desarrollo de los *exploits*, también podemos predecir que los ingenieros especialistas los seguirán desarrollando para utilizarlos en ataques dirigidos a sectores específicos.