IDA Data Displays

Author. Ahmad AlFareed
Section Reverse Engineering - Tools
rETKit

The Principal IDA Displays

مبدئيا IDA تقوم بانشاء 7 display windows . المرئية ثلاثة (Disassembly , Output & Function).

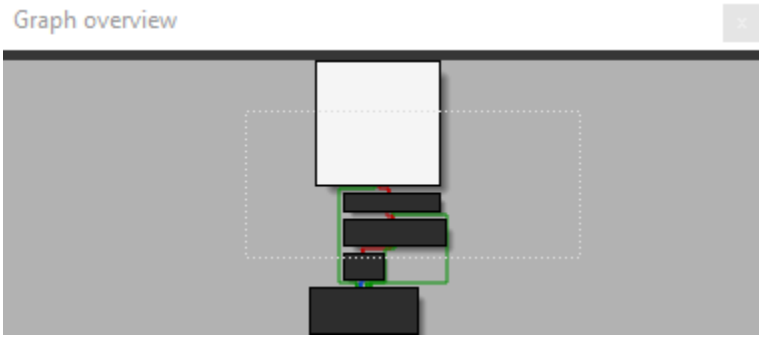
The Disassembly Window

ايضا تعرف بـ IDA-View وستكون الـ disassembly window هي اداتك الاساسية للمعالجة والتحليل. وبناء على ذالك منطقيا يجب انك تفهم ما يتم تقديمه في هاذي النافذة وفهم ماذا يجري. يوجد عندك تنسيقان لهاذي النافذة الـ -graph based view و text-oriented listing view قمنا بشرحها. معظم المهندسين الذين يستخدمون IDA يميلون الى نظام عرض واحد ويفضلونه على الاخر لانه يوجد ناس تفضل تدفق معين.

IDA Graph View

توضح هاذي الصورة وظيفة بسيطة معروضة بـ graph view حيث ان يتم تقسيم الوظيفة الى كتل اساسية حتى تتمكن من تصور (control flow) من الكتل.

ستلاحظ IDA تستخدم اسهما ملونة (احمر واخضر) لتمييز انواع المختلفة من التدفقات بين كل الوظيفة تولد الكتل الاساسية التي تنتهي بقفزة مشروطة تدفقين محتملين اعتمادا على الحالة التي يتم اختبارها ان كان يوجد سهم الحافة (edge arrow) نعم يوجد branch وان لا يوجد edge arrow لا يوجد branch . الاحمر يدل افتراضيا على انه لن يقفز هناك والاخضر افتراضيا يدخل انه سيقفز الى تلك الكتلة. ان كان Basic blocks التي تنتهي بحافة واحدة او يلحق بها Block واحدة فقط تسمى (Normal edge) وتكون زرقاء افتراضيا للاشارة الى الكتلة التالية التي سيتم تنفيذها. في الـ graph mode تعرض IDA وظيفة واحدة في كل مرة يمكنك عمل زوم على الوظيفة من خلال العجلة في الماوس ذهابا وايابا (العجلة + CTFL). الوظائف الكبيرة او المعقدة قد تسبب تشويش لـ graph view بشكل كبير مما يجعل التنقل به صعبا في هاذي الحالة يمكنك استخدام Graph Overview .

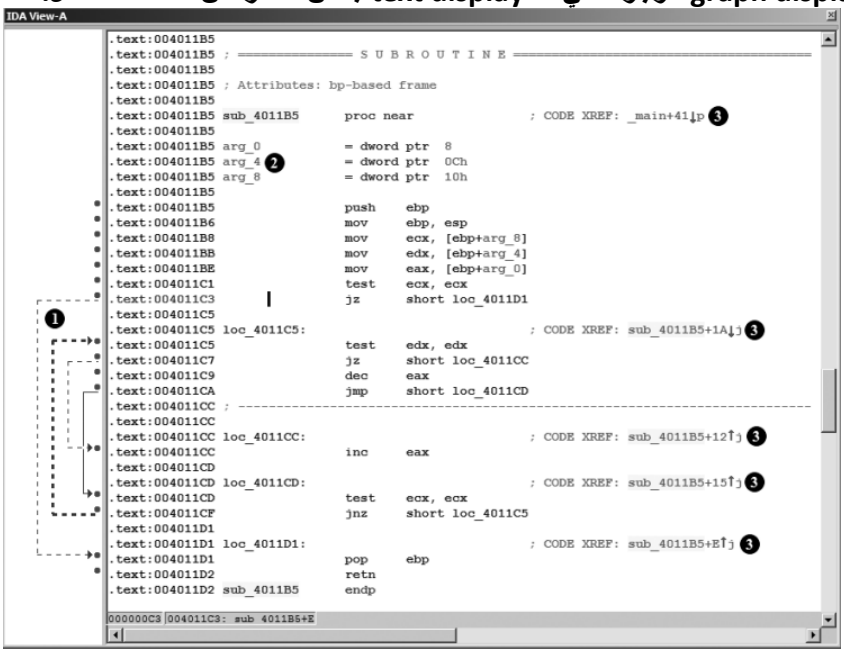


Creating additional disassembly windows

يمكنك انشاء عدة نوافذ من IDA-View X عن طريق (Views->Open Subviews->Disassembly).

IDA Text View

الـ text display يعرض كـ نص في قائمة تفكيك الكاملة للبرنامج بدلا من وظيفة واحدة فقط في الـ graph mode جميع المعلومات في الـ graph display موجودة في الـ text display بغض النظر عن اختلاف الشكل.



يتم تقديم الشكل كـ linear fashion مع عرض العناوين الافتراضية (Virtual Addresses) يتم عرضها كـ [SECTION NAME]:[VIRTUAL ADDRESS] كـ .text:004011C1 .

1. عند الجزء الايسر يسمى arrows window ويستخدم لتصوير التدفق غير الخطي داخل الوظيفة. الـ Solid arrows تمثل الـ unconditional jumps بينما الـ dashed arrows تمثل الـ conditional jumps .
2. الـ declarations في رقم 2 موجودة ايضا باعلى الـ Block في graph view هذا يعطي layout على الـ function's stack frame . تقوم IDA بحساب الـ Stack عن طريق اجراء تحليل تفصيلي لسلوك الـ stack pointer او اي stack frame pointer يستخدم في الـ Functions .
3. الـ comments الموجودة هي crossreferences في هاذي الحالة نرى الـ code cross references على عكس الـ data crossreferences والتي تشير الى تعليمة البرنامج اخرى تنقل التحكم الى الموقع الذي يحتوي على cross-reference comment سنقوم شرحها بتفصيل قريبا.
4.

The Functions Window

يتم استخدام الـ Functions window لسرد كل الوظائف الذي تعرفت عليها الـ IDA في قاعدة البيانات يكون كشكل :

```
main_0          .text    0000000140011880  0000003A        000000F8                R.....B;
```

يشير هذا السطر الى وظيفة الـ main موجودة في segment يسمى text. عند عنوان الافتراضي 140011880 ويبلغ طوله 3A و R تدل على ((returns to the caller (R) ويستخدم سجل EBP (B) لرجوع الى local variables . يتم استخدام الـ Flags لوصف الوظيفة مثل B او R يمكنك مراجعة جميع الـ Flags في IDA Help .

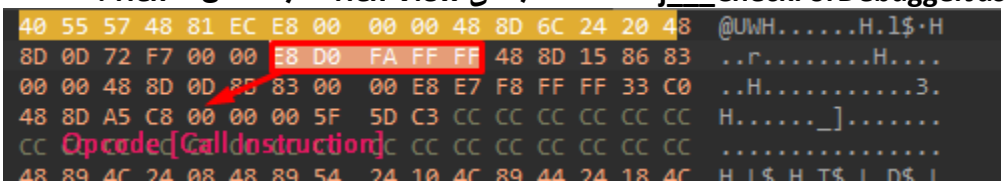
The Output Window

الـ Output window تعمل بمثابة output console الخاصة بـ IDA وهي المكان المناسب للبحث عن معلومات حول المهام التي تقوم بها IDA . على سبيل المثال عند فتح ملف الثنائي لاول مره يتم انشاء رسائل للاشارة الى مرحلة التحليل وتظهر العديد والعديد مثل يمكن الكتابة بها سكربتات او الـ Outputs الخاصة بـ Plugins معينة .

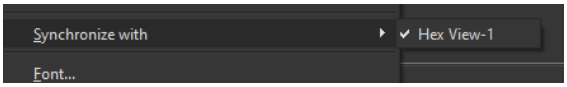
Secondary IDA Displays

The Hex View Window

الـ Hex View يعد تسمية خاطئة في هذة الحالة حيث يتكون الـ IDA Hex View لعرض مجموعة متنوعة من التنسيقات ويمكن استخدامها كـ hex editor . افتراضيا الـ Hex View يعطي hex dump لمحتوى البرنامج مع 16 بايت لكل سطر ASCII معروض على الجنب. يمكنك فتح عدة نوافذ من Hex View كباقي النوافذ الاخرى. افتراضيا ايضا تتم مزامنة Hex window الرئيسية بـ disassembly window الرئيسية. وايضا في كل نقطة تكون بها في الـ Disassembly بفضل المزامنة تكون نفس النقطة موجودة في Hex-View على سبيل المثال نحن عند نقطة " call j___CheckForDebuggerJustMyCode " عند الذهاب الى Hex-View سنجدها لكن كـ Hex .



المزامنة :



The Exports Window

الـ Exports window تسرد لك الـ entry points في الملف. تتضمن هاذي النقطة execution entry point كما هو محدد في الـ header section . او اي وظائف يصدرها هذا الملف لتستخدمها ملفات اخرى. توجد الوظائف المصدرة بشكل شائع في ملفات مثل Windows DLL الـ Exported entries تتكون من الاسم و virtual address و ordinal number . اما بالنسبة للـ executable files في الـ Exports window على الاقل يجب ان يكون ادخال واحد وهو الـ entry point .

Ordinal	Address	Name
[main entry]	0000000140011271	start

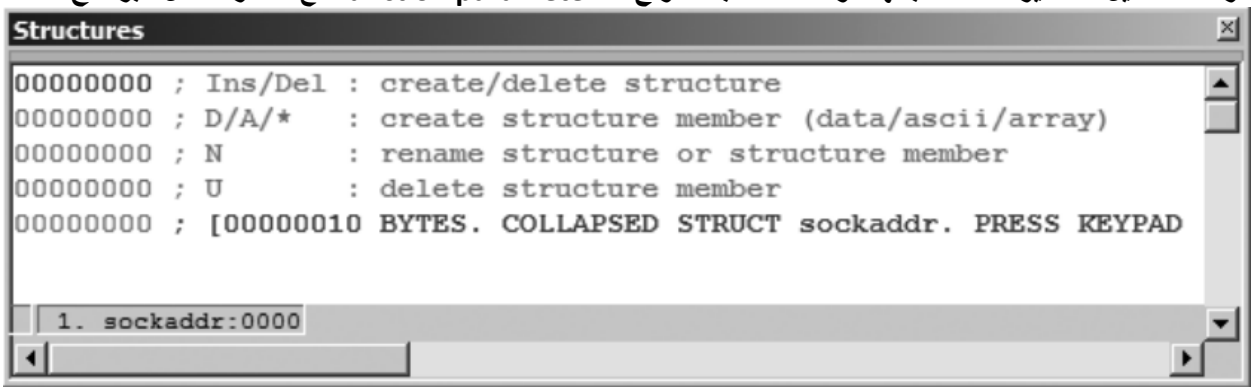
The Imports Window

الـ Imports window هو عكس الـ Exports window هو يسرد جميع الوظائف التي يتم استيرادها بواسطة الـ Binary الذي تم تحليلة. تكون الـ Imports windows عندما يستخدم البرنامج shared libraries . الثنائيات المرتبطة بشكل ثابت (Statically linked) ليس لها اي تبعيات خارجية وبالتالي لا يوجد Imports . كل شئ بـ Imports window يوجد به اسم الوظيفة واسم المكتبة الذي تحتوي على هاذي الوظيفة. نظرا لان التعليمات البرمجية الخاصة بالوظيفة المستورة موجودة في الـ shared library . فان مع كل كلشئ موجود في import window هو virtual address مع علاقة بـ import table .

0000000140020000	GetStartupInfoW	KERNEL32

The Structures Window

يتم استخدام الـ Structures window لعرض تخطيط اي بنية من البنيات المعقدة مثل الـ C structs او Unions . اثناء مرحلة التحليل تستشير IDA مكتباتها الواسعة لمطابقة انواع الـ function parameter مع الذاكرة داخل البرنامج.



الصورة هاذي توضح Structures Window تعتقد IDA انه يتم استخدام socketaddr structure . يوجد العديد من الاسباب التي جعلت IDA تتوصل الى هذا الاستنتاج ممكن ان يكون احد هاذي الاسباب لاحظت استدعاء وظيفة connect لانشاء اتصال بالشبكة. يؤدي النقر المزدوج على اي Structure الى قيام IDA بتوسيع البنية وهذا يسمح برؤية التخطيط التفصيلي للبنية بما في ذالك الـ field names . الاستخدامات الرئيسية لهاذي النافذة توفير مرجع جاهز لـ standard data structures و تزويدك بوسيلة لانشاء هياكل بيانات خاصة بك لاستخدامها كـ memory layout templates عندما تكشف custom data structures داخل البرنامج.

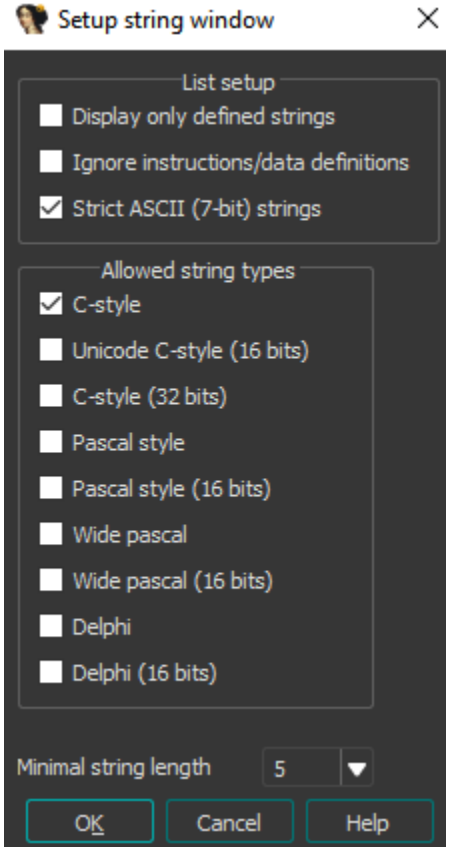
The Enums Window

الـ Enums window تشبه Structures window . عندما تكتشف IDA نوع standard enumerated datatype سيتم ادراجها في هاذي الـ Window . يمكنك جعل عمليات التفكيك الخاصة بك اكثر قابلية للقراءة باستخدام enums بدلا من الـ integer constants .

Tertiary IDA Displays

The Strings Window

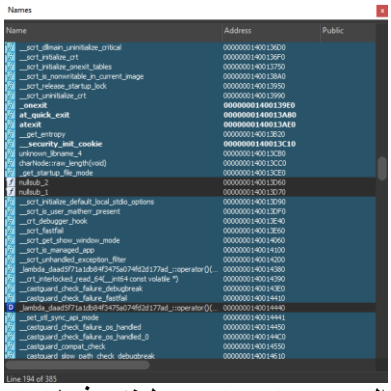
هي اداة مساعدة في IDA . الغرض من Strings window هو عرض قائمة السلاسل المستخرجة من الـ Binary بالاضافة الى العنوان الذي توجد به كل سلسلة. النقر المزدوج الى اي سلسلة ينقلك الى موقعها . عند استخدامها مع cross-references الـ Strings window توفر وسيلة لاكتشاف العديد من المواضيع في الثنائي وتحليله على سبيل المثال يمكن ان يكون رابط او registry key معين. تقوم بفتح هاذي النافذة فقط عند انتهاء IDA من التحليل. الـ default string type المستخدم في IDA هو C-style ASCII string .



اذا كنت متوقع مواجهة اي شيء غير c-style يمكن اختياره مثل Unicode strings . يوجد خيار Display only defined strings هذا الخيار يقيد لعرض عناصر named string data فقط التي تم انشاؤها من قبل المطور او IDA . ويوجد خيار اخر Ignore instructions/data definitions يؤدي هذا الخيار الى بالبحث عن سلاسل عبر التعليمات والـ data definitions الموجودة. يسمح هذا الخيار برؤية السلاسل التي قد تكون متواجدة بجزء من التعليمات لملف الثنائي وتم تحويلها عن طريق الخطأ الى تعليمات او رؤية سلاسل داخل البيانات التي قد تم تنسيقها كشيء اخر غير السلسلة مثل الـ arrays . وايضا هذا الخيار سيؤدي الى انشاء سلاسل غير مرغوب (junk strings) فيها .

The Names Window

الـ Names window توفر ملخص بجميع الاسماء العامة الموجودة في الـ Binary . الاسم ليس اكثر من وصف رمزي يعطى للـ virtual address . تستمد IDA في البداية قائمة الاسماء من الـ symbol-table و signature analysis اثناء التحميل الاولي للملف. يمكن فرز الاسماء ابجديا او تصاعديا او حسب Virtual Addresses او تنازليا . قائمة الاسماء هي مفيدة في التنقل من موقع الى موقع اخر في البرنامج.



النقر المزدوج على اي اسم ينقلك الى موقعه فورا. الـ Displayed names ملونه و فيها letter coded وكلهم يشير الى شيء معين.

F

هي regular function هاذي الوظيفة لا تعترف بها IDA كـ library functions .

L

هي library function . تعترف IDA على وظائف المكتبة من خلال استخدام signature matching algorithms . اذا لم يكن هناك signature سيتم تصنيفها كمكتبة عادية

I

هي imported name وهو في الغالب اسم وظيفة مستورد من shared library . الفرق بين و library function هو انه لا يوجد code للـ imported name بينما الـ library function سيكون مفكوك اثناء مرحلة التحليل الاولى وستكون مفككة اثناء تنقلك بين disassembler .

C

هو Named code هذه هي مواقع التعليمات (instruction locations) التي لا تعتبرها IDA جزء من اي function .

D

هي Data بيانات عادة ما تتمثل لمواقع البيانات الـ global variables .

A

String data . هو موقع البيانات المرجعي الذي يحتوي على string .

IDA تقوم بتوليد اسماء لجميع المواقع التي يتم الرجوع اليها مباشرة اما كـ كود (مثل call or branches) او كـ Data (مثل read & write). اذا كان يوجد اسم للمراجع هاي في الـ symbol table اداة IDA ستعتمد هاذي الاسماء. في حال عدم توفر symbol table تقوم IDA بانشاء اسم افتراضي لاستخدامة في عملية التفكيك . اي موقع مسمى بدمج الـ virtual address مع prefix وبالنهاية يشير الى نوع الموقع . الـ prefixes الاكثير شيوعا :

sub_xxxxx - subroutine
loc_xxxxx - instruction location
byte_xxxxx - 8-bit data
word_xxxxx - 16-bit data at location
dword_xxxxx - 32-bit data at location
unk_xxxxx - Data of unknown size at location

The Segments Window

ملاحظة ان الـ segments في IDA في غالب الوقت تسمى sections عند مناقشة الـ structure of binary files . لا تخلط بين الـ segments بـ (segmented memory architecture) . الـ Information المقدمة تتضمن segment name و بداية ونهاية الـ addresses و permission flags . المقصود بـ عناوين البداية ونهاية الـ (virtual address range) .

Segments

Drag this title to dock somewhere else

Name	Start	End	R	W	X	D	L	Align	Base	Type	Class	AD	es	ss	ds	fs	gs
.textbss	0000000140001000	0000000140011000	R	W	X	.	L	para	0001	public	CODE	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.text	0000000140011000	0000000140019000	R	.	X	.	L	para	0002	public	CODE	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.rdata	0000000140019000	000000014001C000	R	.	.	.	L	para	0003	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.data	000000014001C000	000000014001D000	R	W	.	.	L	para	0004	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.pdata	000000014001D000	0000000140020000	R	.	.	.	L	para	0005	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.idata	0000000140020000	00000001400203A8	R	.	.	.	L	para	0006	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.idata	00000001400203A8	0000000140021000	R	.	.	.	L	para	0006	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
.msvcjmc	0000000140021000	0000000140022000	R	W	.	.	L	para	0007	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF
_00cfg	0000000140022000	0000000140023000	R	.	.	.	L	para	0008	public	DATA	64	0000	0000	0004	FFFFF...	FFFFFFFFFFFFFFFF

ملاحظة يمكن استخدام segment names – nonstandard مثل (UPX1).

The Signatures Window

تستخدم IDA extensive library of signatures لتحديد blocks of code معروفة. الـ Signatures يتم استخدامها لتحديد -compiler generated startup sequences في محاولة تحديد الـ compiler الذي قام بانشاء الـ Binary . وايضا تستخدم لتصنيف بين الوظائف مثل library functions مدرجة من الـ Compiler او functions added نتيجة الـ static linking . عندما IDA تحدد library functions افضل لك.

Signatures

Drag this title to dock somewhere else

File	State	#func	Library name
vc64_14	Applied	104	Microsoft VisualC v14 64bit runtime
vc64ucrt	Applied	5	Microsoft VisualC 64bit universal runtime
vc64seh.sig	Applied	0	SEH for vc64 7-14

The Type Libraries Window

تمثل الـ Type libraries بانواع البيانات المحددة مسبقا. ونماذج الوظائف الاولية المأخوذة من header files تتضمن ايضا الـ compilers الاكثر شيوعا. من خلال معالجة الـ header files اداة IDA تفهم الـ datatypes التي تتوقعها الوظائف ويمكنها ايضا وضع تعليقات على ذالك. يتم جمع المعلومات هاذي كلها من til .

The Function Calls Window

في اي برنامج ممكن ان function تستدعي او تستدعى من دوال اخرى. من البسيط والمهم انشاء رسم بياني يبين العلاقات بين callers & callees . يطلق عليه function call graph او function call tree .

The Problems Window

تقوم بسرد لك اي صعوبات واجهتها IDA في تفكيك الملف وكيف اختارت التعامل مع تلك الصعوبات .