What Happens In Windows 7 Stays In Windows 7

Joseph Moti & Marion Marschalek

POC2013

Agenda

- Vulnerabilities
- Automated Vulnerability Search
- An Approach
- A Solution as Proof of Concept
- Demo ;)
- Whats next?

Who are we

Joseph Moti

Moti Joseph has been involved in computer security. In the last few years he has been working on reverse engineering exploit code and developing security products.
Moti is a former speaker at Black Hat 2007, USA, CONF2009, Poland Warsaw, POC 2009, South Korea, ShakaCon 2009, USA, CONF2010, Poland Karkow, POC 2010, South Korea, CHINA 2011 at Shanghai Jiao Tong University , NOPcon 2012 and SysCan2010 Taiwan,Taipe

Marion Marschalek

Marion Marschalek is currently employed at IKARUS Security Software GmbH based in Vienna, Austria. She is working as Malware Analyst and in Incident Response. Marion was a former speaker at Defcon21 in Las Vegas and Hack.lu2013 in Luxembourg.
In March this year Marion won the Female Reverse Engineering Challenge 2013, organized by RE professional Halvar Flake.

Intro

Got a bug in your software?

Can I haz it??

Schneier on security.

Vulnerabilities are **software mistakes** in specification and design, but mostly mistakes **in programming**. Any large software package will have thousands of mistakes. Once discovered, they can be **used to attack systems**. This is the point of security patching: eliminating known vulnerabilities. But many systems don't get patched, so the **Internet is filled with known, exploitable vulnerabilities**.

How to find vulnerabilities?

- Application Penetration Testing
- Fuzzing
- Reverse Engineering
- Source Code Review
- Or.. Being more advanced:
  - Tracking software bugs, introducing bugs into software, reversing security patches

Who is interested in finding them?

- Hackers – its fun to break stuff, usually
- Software Companies – should find them (and fix them..)
- Criminals – use them for illegal stuff
- Governments – use them for semi-legal stuff
- Media – gets good news out of (exploited) vulnerabilities

And why automate it?

- It‘s faster!!
  - The hacker – can break more
  - The software company – can fix faster
  - Criminals – can make more money
  - Governments – can … [SECRET]
  - Media – has more to write about

The Approach

What happens in Windows7 stays in Windows7…

Win7

quartz.dll

```
xor     eax, eax
inc     eax
shl     eax, cl
...
shl     eax, 2
push    eax          ; cb
call    ds:__imp__CoTaskMemAlloc@4
```

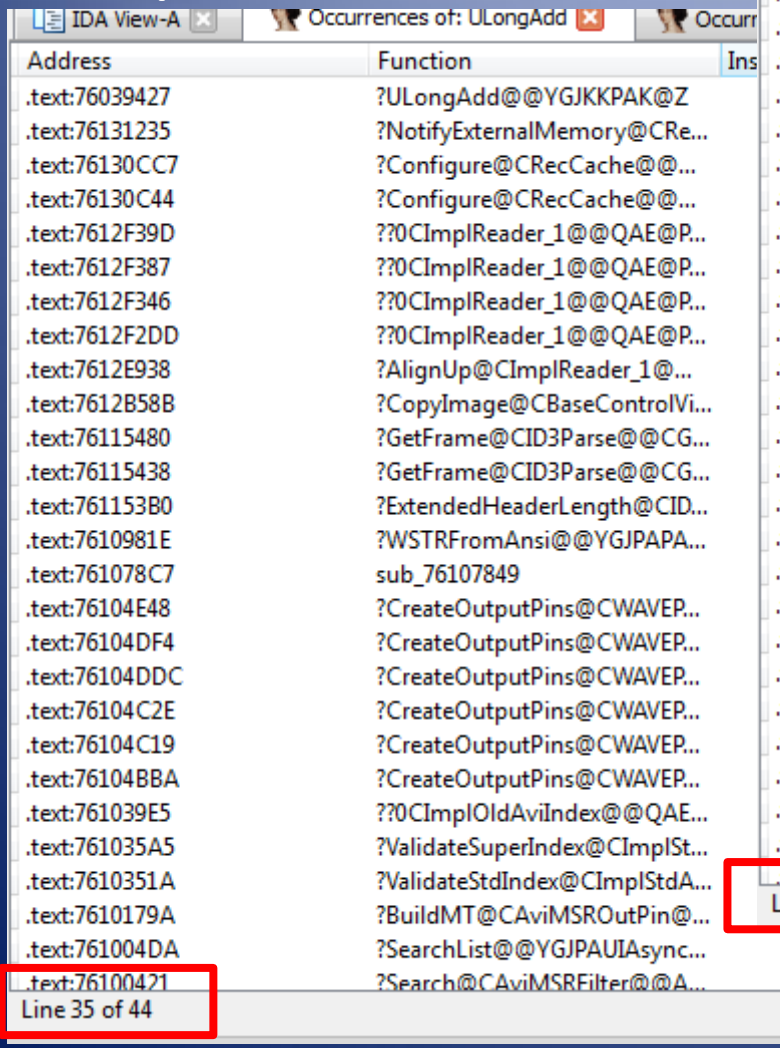
Win8

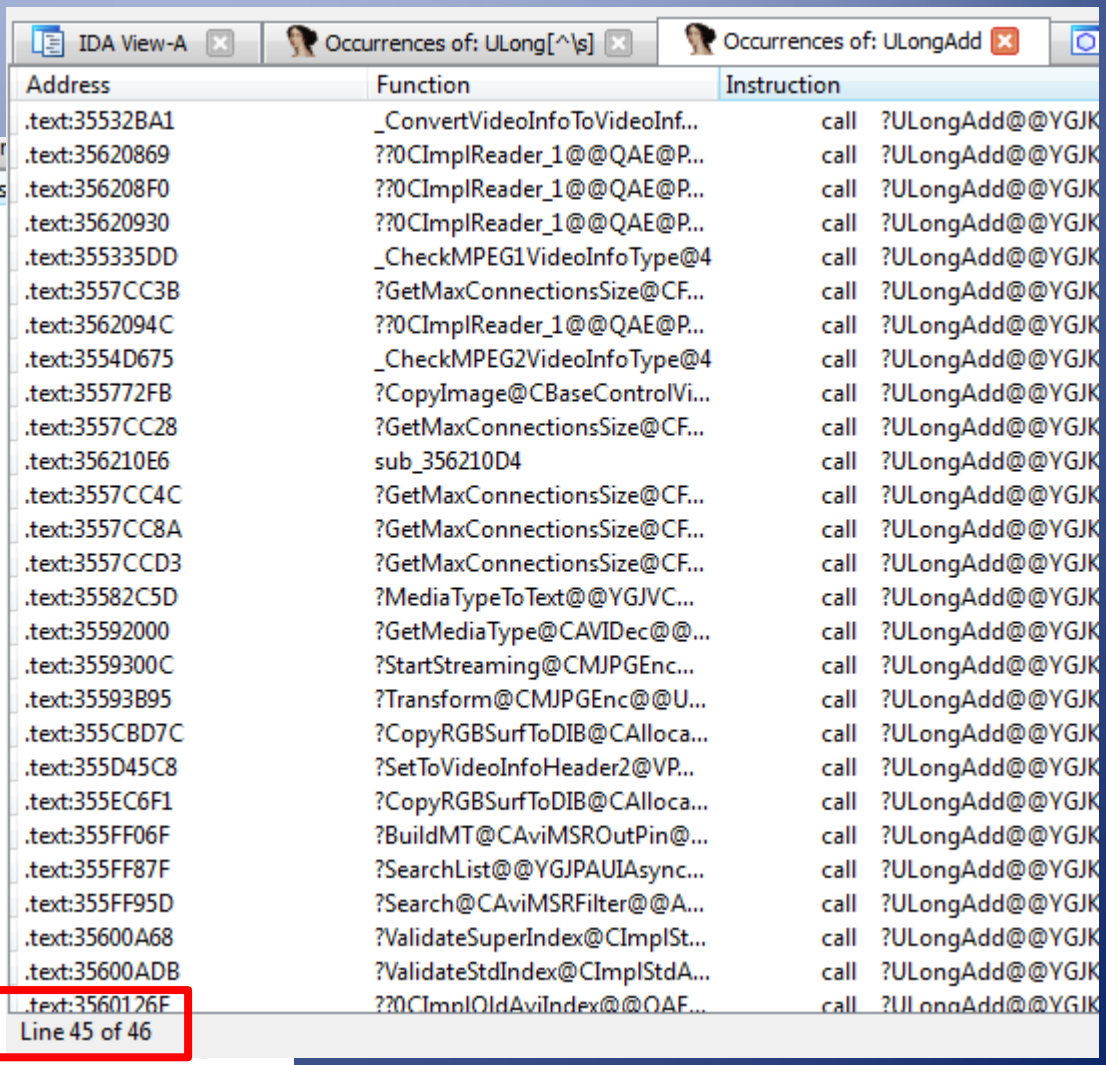
quartz.dll

```
lea     ecx, [ebp+cb]
push    ecx
push    4
push    eax
mov     [esi], eax
call    ?ULongMult@@YGJKKPAK@Z
test    eax, eax
...
push    [ebp+cb]     ; cb
call    ds:__imp__CoTaskMemAlloc@4
```

Counting Function Calls

Win7 quartz.lib





Win8 quartz.lib

Spot The Patch

Win7 quartz.lib

```
cmp     cx, 8
jbe     short loc_7609B8EC
mov     eax, 80040220h
jmp     short loc_7609B933
-----------------------------------------

                        ; CODE XREF:
xor     eax, eax
inc     eax
shl     eax, cl
push    esi
mov     esi, [ebp+arg_0]
push    edi
mov     [esi], eax
shl     eax, 2
push    eax             ; cb
call    ds:__imp__CoTaskMemAlloc@4 ;
mov     edi, [ebp+arg_4]
mov     [edi], eax
test    eax, eax
jnz     short loc_7609B914
and     [esi], eax
```

Win8 quartz.lib

```
lea     ecx, [ebp+cb]
push    ecx             ; unsigned __int3
push    4               ; int
push    eax             ; int
mov     [esi], eax
call    ?ULongMult@@YGJKKPAK@Z ; ULongMul
test    eax, eax
jns     short loc_355B131C
mov     eax, 80070216h
jmp     short loc_355B1351
-----------------------------------------

                        ; CODE XREF: COve
push    [ebp+cb]        ; cb
call    ds:__imp__CoTaskMemAlloc@4 ; CoTa
mov     [edi], eax
test    eax, eax
jnz     short loc_355B1334
and     [esi], eax
mov     eax, 8007000Eh
jmp     short loc_355B1351
```

Intsafe.h & Strsafe.h

- Searching for security patches:
  - Type Conversion
  - Safe Math Functions
  - Buffer Boundary Checks on Strings
- Set of 130 Signatures of ‚Safe Functions‘

‚Safe Functions‘

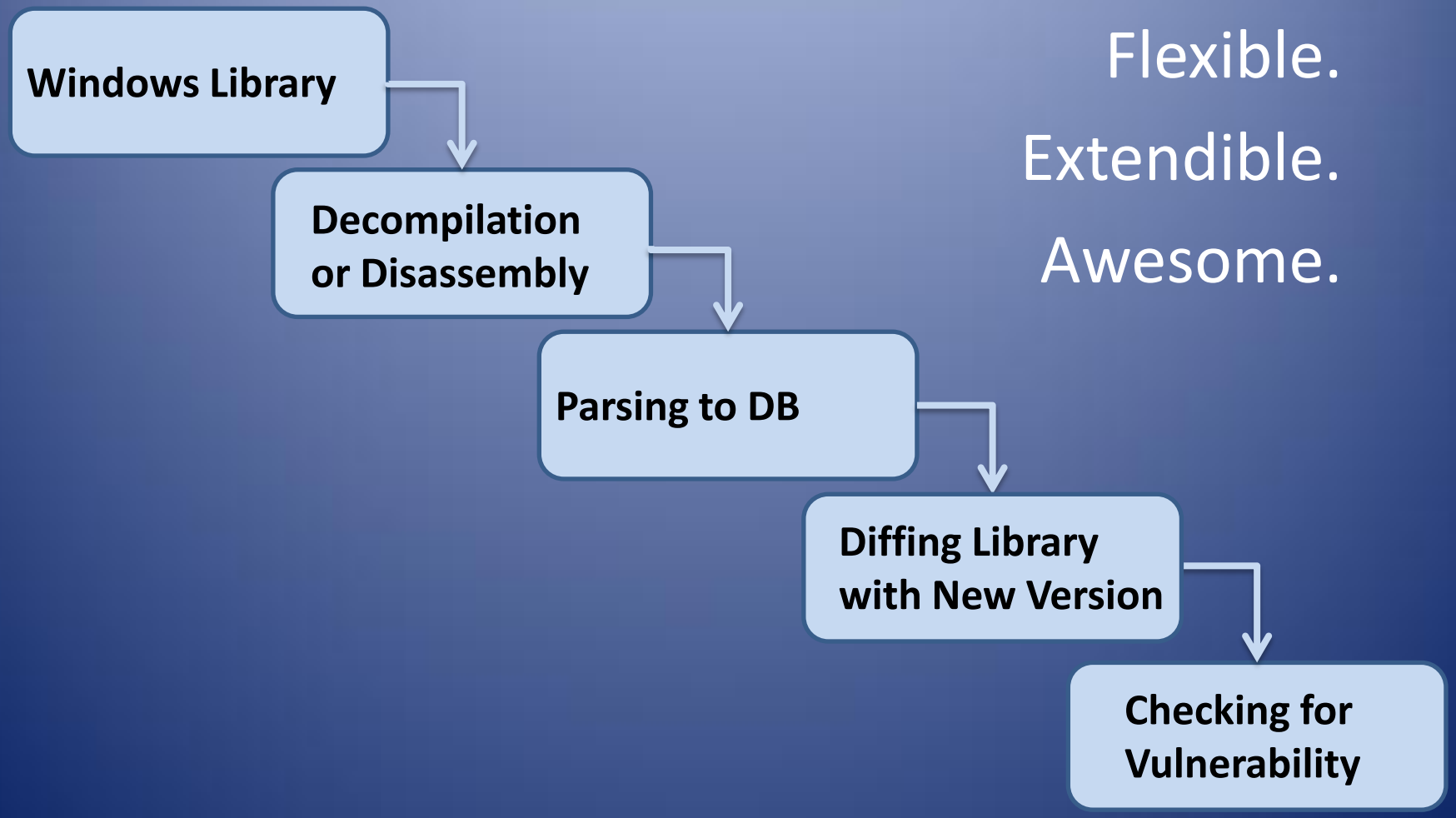
UInt8ToInt8
UInt8ToChar
ByteToInt8
ByteToChar
ShortToInt8
ShortToUChar
ShortToChar
UShortToUInt8
UShortToShort
IntToInt8
IntToUChar
IntToChar

UInt8Add
UShortAdd
UIntAdd
ULongAdd
SizeTAdd
ULongLongAdd
UInt8Sub
UShortSub
UIntSub
ULongSub
SizeTSub
ULongLongSub

StringCbGets
StringCbGetsEx
StringCbLength
StringCbPrintf
StringCbPrintfEx
StringCbVPrintf
StringCbVPrintfEx
StringCchCat
StringCchCatEx
StringCchCatN
StringCchCatNEx
StringCchCopy

... and many many more ....

The Approach



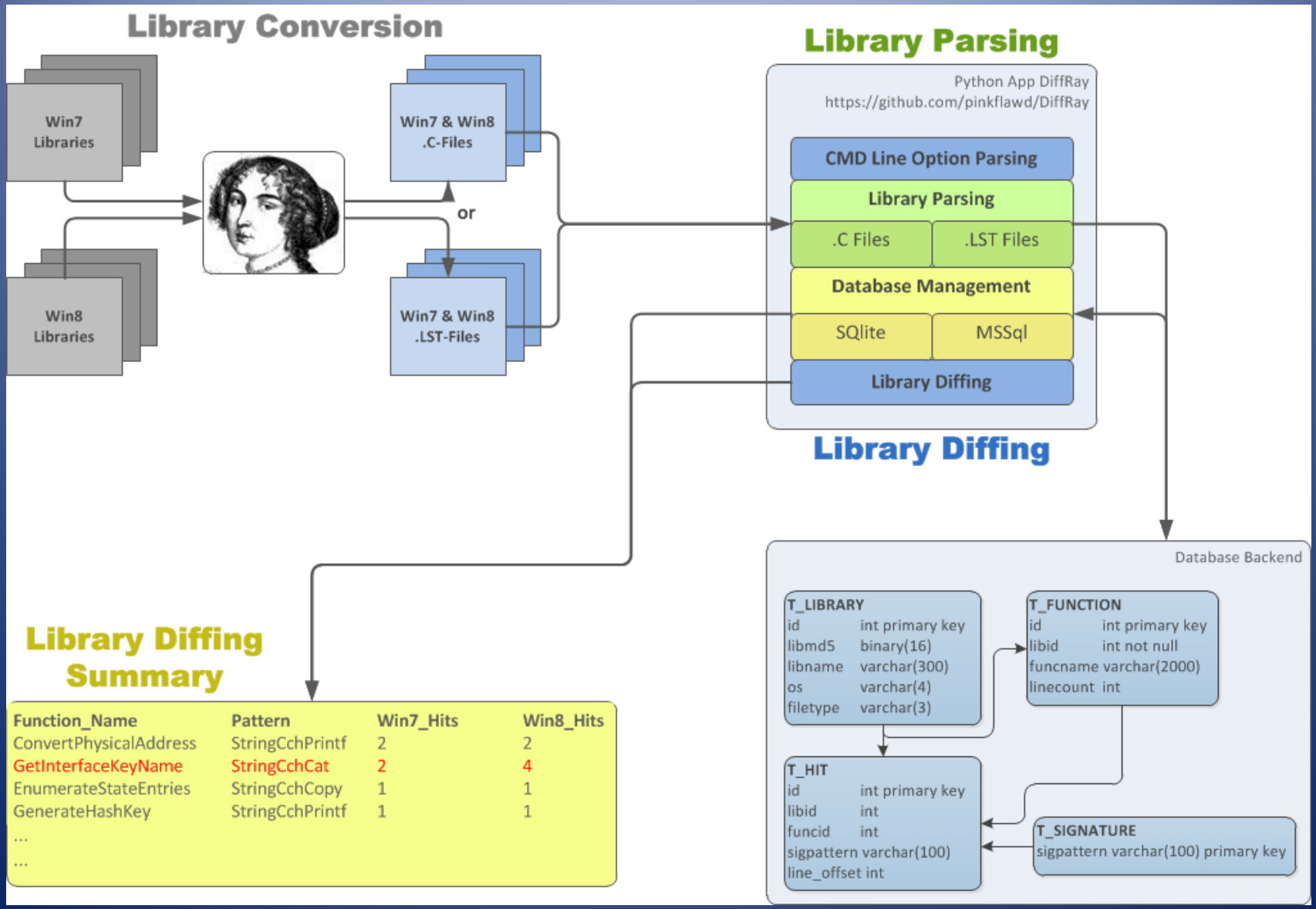
Flexible.

Extendible.

Awesome.

The Solution

Pretty, eh?


Library Conversion
Win7 Libraries
Win8 Libraries
Win7 & Win8 .C-Files
Win7 & Win8 .LST-Files
or
Library Parsing
Python App DiffRay
https://github.com/pinkflawd/DiffRay
CMD Line Option Parsing
Library Parsing
.C Files
.LST Files
Database Management
SQlite
MSSql
Library Diffing
Library Diffing
Library Diffing Summary
Function_Name Pattern Win7_Hits Win8_Hits
ConvertPhysicalAddress StringCchPrintf 2 2
GetInterfaceKeyName StringCchCat 2 4
EnumerateStateEntries StringCchCopy 1 1
GenerateHashKey StringCchPrintf 1 1
...
...
Database Backend
T_LIBRARY
id int primary key
libmd5 binary(16)
libname varchar(300)
os varchar(4)
filetype varchar(3)
T_FUNCTION
id int primary key
libid int not null
funcname varchar(2000)
linecount int
T_HIT
id int primary key
libid int
funcid int
sigpattern varchar(100)
line_offset int
T_SIGNATURE
sigpattern varchar(100) primary key

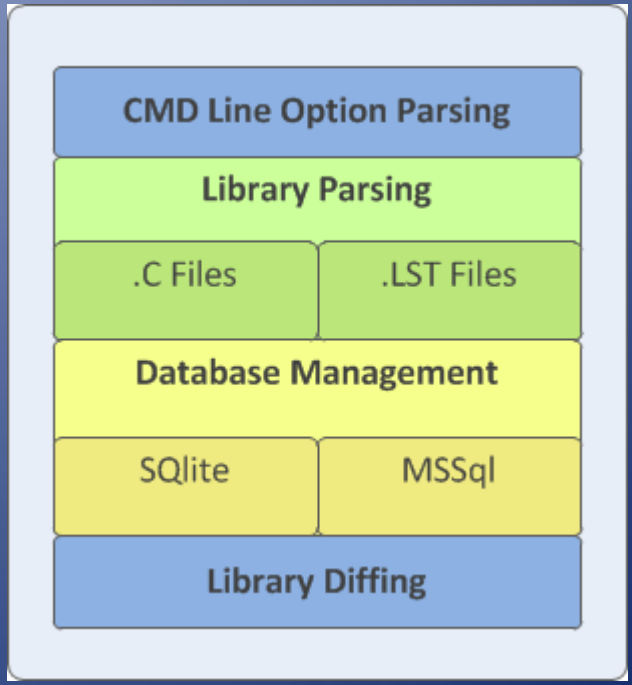
Getting the .C

- Library Conversion using IDA Pro

- means: .dll -> .idb -> .c

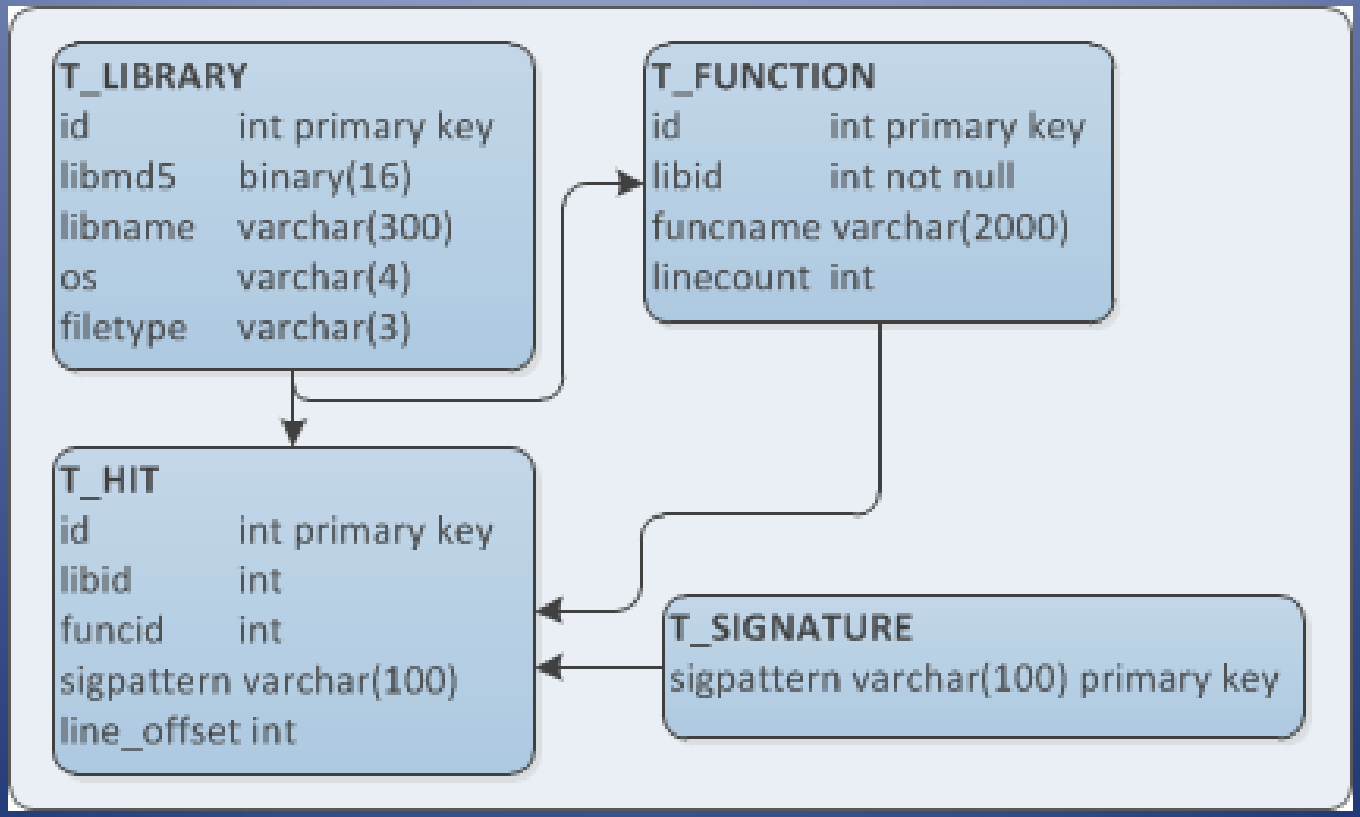
Library Parsing

- DiffRay on *https://github.com/pinkflawd/DiffRay*
- Parses a library / directory of libraries
- Manages libraries , functions and signature hits
- Diff libraries functionwise
  - Based on library ID or library name pattern



The Database

- MSSql or SQLite (for now)



Diff it!

- Compare libraries on a function basis
- Extract hits per function per signature

Function_Name	Pattern	Win7_Hits	Win8_Hits
ConvertPhysicalAddress	StringCchPrintf	2	2
GetInterfaceKeyName	StringCchCat	2	4
EnumerateStateEntries	StringCchCopy	1	1
GenerateHashKey	StringCchPrintf	1	1
…			
…			

DiffRay HowTo: Configuration

- **signatures.conf** – whatever symbols you‘re searching for
- **logger.conf** – logging output and formatting, details to be found at *http://docs.python.org/2/howto/logging.html*
- **mssql.conf** – MSSql access credentials

DiffRay HowTo: Parsing

Maintenance:

```
python [dir]\src\Main.py --create-scheme --update-sigs

python [dir]\src\Main.py --parse [library_path]
                         --os [Win7|Win8] --type [C|LST]

python [dir]\src\Main.py --dirparse [directory_path]
                         --os [Win7|Win8] --type [C|LST]

python [dir]\src\Main.py --flushall
```

Switches:

```
--backend [mssql|sqlite]
--no-flush
```

DiffRay HowTo: Diffing

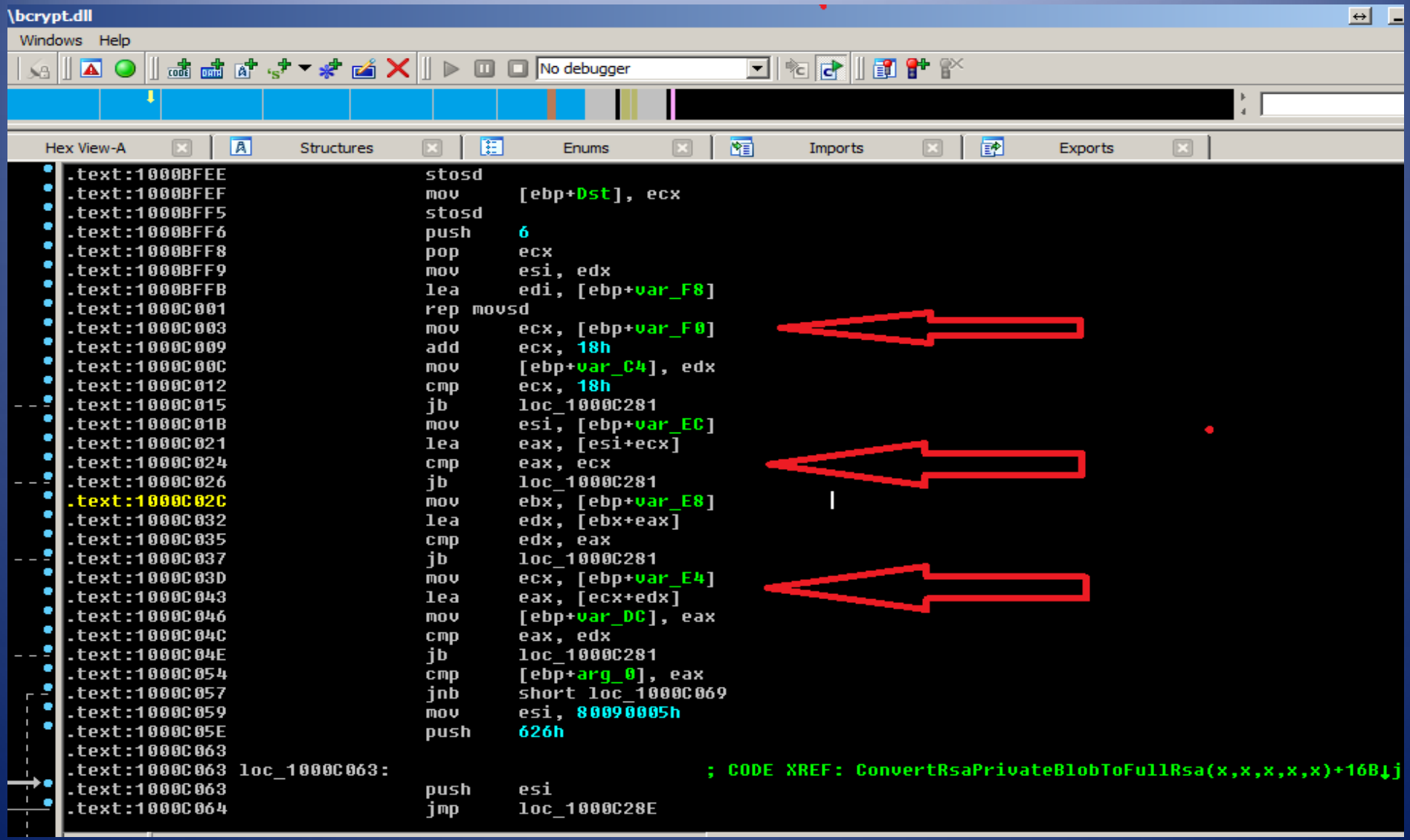
Info Output & Diffing:

```
python [dir]\src\Main.py --search_libs [libname_pattern]

python [dir]\src\Main.py --lib_all_info [lib_id]

python [dir]\src\Main.py --diff
                         --lib_1 [win7lib] --lib_2 [win8lib]

python [dir]\src\Main.py --diff_byname [libname_pattern]
```

Demo time!

Findings

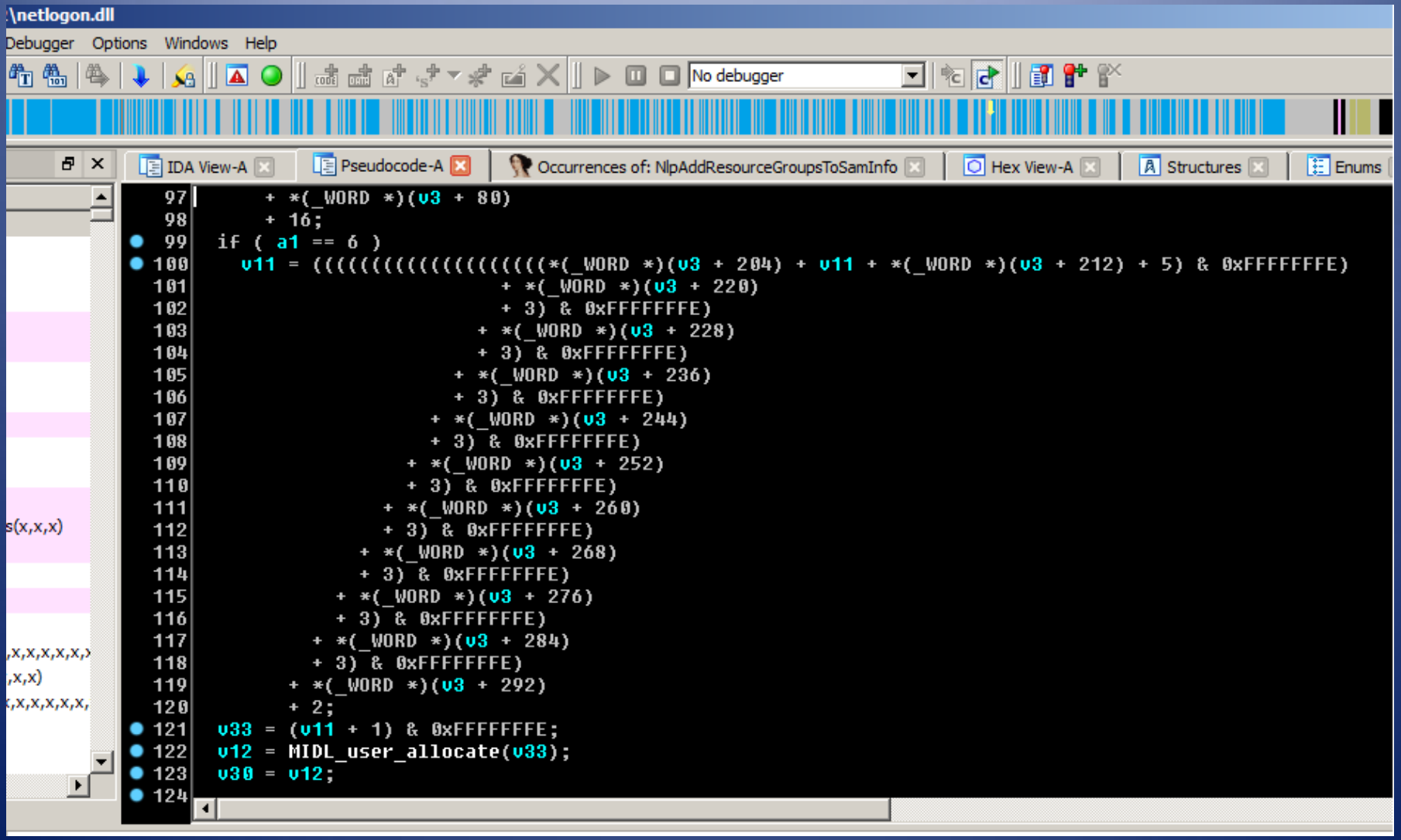
Windows 8
bcrypt.dll!ConvertRsaPrivateBlobToFullRsa



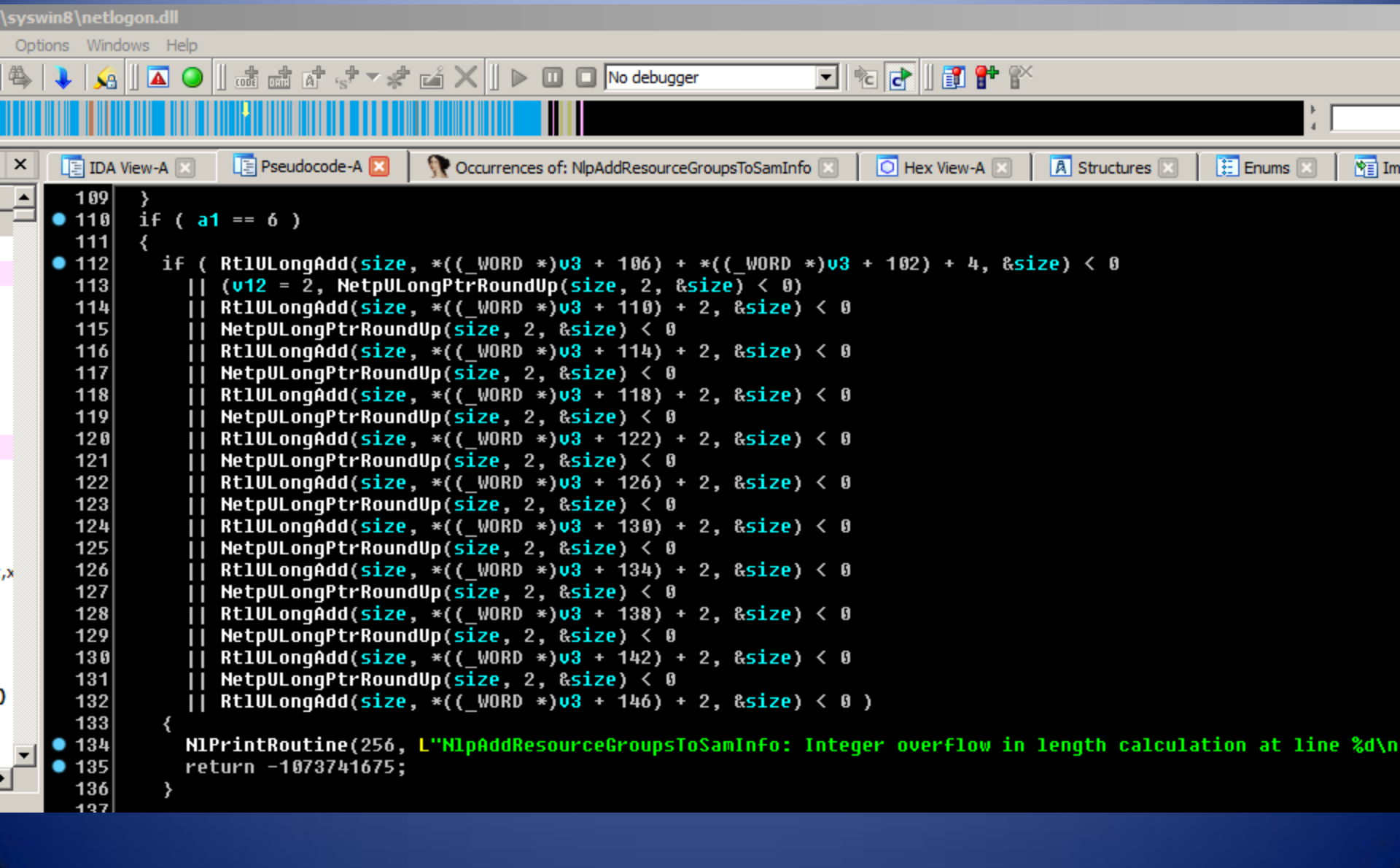
Windows 7 (ULongAdd)
bcrypt.dll!ConvertRsaPrivateBlobToFullRsa

```
.text:6D80C94E                push    18h
.text:6D80C950                pop     eax
.text:6D80C951                lea     edi, [ebp+var_2C]
.text:6D80C954                rep movsd
.text:6D80C956                lea     ecx, [ebp+var_4]
.text:6D80C959                push    ecx
.text:6D80C95A                push    [ebp+var_24]
.text:6D80C95D                mov     [ebp+var_4], eax
.text:6D80C960                push    eax
.text:6D80C961                call    _ULongAdd@12    ; ULongAdd(x,x,x)
.text:6D80C966                test    eax, eax
.text:6D80C968                jl      loc_6D80CB1E
.text:6D80C96E                mov     esi, [ebp+var_20]
.text:6D80C971                lea     eax, [ebp+var_4]
.text:6D80C974                push    eax
.text:6D80C975                push    esi
.text:6D80C976                push    [ebp+var_4]
.text:6D80C979                call    _ULongAdd@12    ; ULongAdd(x,x,x)
.text:6D80C97E                test    eax, eax
.text:6D80C980                jl      loc_6D80CB1E
.text:6D80C986                mov     ebx, [ebp+var_1C]
.text:6D80C989                lea     eax, [ebp+var_4]
.text:6D80C98C                push    eax
.text:6D80C98D                push    ebx
.text:6D80C98E                push    [ebp+var_4]
.text:6D80C991                call    _ULongAdd@12    ; ULongAdd(x,x,x)
.text:6D80C996                test    eax, eax
.text:6D80C998                jl      loc_6D80CB1E
.text:6D80C99E                mov     edi, [ebp+var_18]
.text:6D80C9A1                lea     eax, [ebp+var_4]
.text:6D80C9A4                push    eax
.text:6D80C9A5                push    edi
.text:6D80C9A6                push    [ebp+var_4]
.text:6D80C9A9                call    _ULongAdd@12    ; ULongAdd(x,x,x)
.text:6D80C9AE                test    eax, eax
.text:6D80C9B0                jl      loc_6D80CB1E
000BD4E 6D80C94E: ConvertRsaPrivateBlobToFullRsa(x,x,x,x,x)+14
```

Windows 7
netlogon.dll! NlpAddResourceGroupsToSamInfo



Windows 8 (ULongAdd)
netlogon.dll! NlpAddResourceGroupsToSamInfo



Windows 7
Integer overflow

```
.text:06D161E0 ; Attributes: bp-based frame
.text:06D161E0
.text:06D161E0 ; int __stdcall _EscapeField(LPCWSTR psz, int)
.text:06D161E0 ?_EscapeField@@YGJPBGPAPAG@Z proc near  ; CODE XREF: SHGetParsingNameFromProperty
.text:06D161E0
.text:06D161E0 var_C           = dword ptr -0Ch
.text:06D161E0 uBytes          = dword ptr -8
.text:06D161E0 var_4           = dword ptr -4
.text:06D161E0 psz             = dword ptr  8
.text:06D161E0 arg_4           = dword ptr  0Ch
.text:06D161E0
.text:06D161E0                 mov     edi, edi
.text:06D161E2                 push    ebp
.text:06D161E3                 mov     ebp, esp
.text:06D161E5                 mov     eax, [ebp+arg_4]
.text:06D161E8                 and     dword ptr [eax], 0
.text:06D161EB                 sub     esp, 0Ch
.text:06D161EE                 push    edi
.text:06D161EF                 push    [ebp+psz]       ; lpString
.text:06D161F2                 call    ds:__imp__lstrlenW@4 ; lstrlenW(x)
.text:06D161F8                 mov     edi, eax
.text:06D161FA                 imul    edi, 3
.text:06D161FD                 inc     edi
.text:06D161FE                 cmp     edi, 20000h
.text:06D16204                 ja      loc_6D16307
.text:06D1620A                 and     [ebp+uBytes], 0
.text:06D1620E                 lea     eax, [ebp+uBytes]
.text:06D16211                 push    eax             ; uBytes
.text:06D16212                 push    edi             ; int
.text:06D16213                 call    ??$LocalAllocArray@G@@YGJIPAPAG@Z ; LocalAllocArray<ushor
.text:06D16218                 mov     [ebp+var_4], eax
```

Windows 8 /ULongLongToUint Patched!

```
; Attributes: bp-based frame

; int __stdcall _EscapeField(LPCWSTR psz, int)
?_EscapeField@@YGJPBGPAPAG@Z proc near  ; CODE XREF: SHGetParsingNameFromPropertyS

var_C           = dword ptr -0Ch
var_8           = dword ptr -8
uBytes          = dword ptr -4
psz             = dword ptr  8
arg_4           = dword ptr  0Ch

                mov     edi, edi
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+arg_4]
                sub     esp, 0Ch
                push    ebx
                mov     ebx, [ebp+psz]
                push    esi
                xor     esi, esi
                push    edi
                mov     [eax], esi
                lea     eax, [ebp+var_8]
                push    eax             ; unsigned int *
                push    ebx             ; lpString
                call    _lstrlenW@4     ; lstrlenW(x)
                push    3
                pop     ecx
                mul     ecx
                push    edx
                push    eax             ; unsigned __int64
                call    ?ULongLongToUInt@@YGJ_KPAI@Z ; ULongLongToUInt(unsigned __
                test    eax, eax
```

Whats Next

- Possible Extensions
  - Win8, we‘re coming!!
  - Extended signatures
- Improvements
  - Transparent DB library
  - Integration of components
- Known issues
  - Duplicate hits, false positives, slooow, output is not handy

Happy Diffing.