



RIVERSIDE RESEARCH INSTITUTE

DCM Utility

A Data/Code Miner utilizing
Microsoft Detours

Jason Raber, Team Lead - Reverse Engineer

Overview

- The Problem: Program Reconstruction
- MS Detours - How it works
- DCM
- Demonstration DCM
- DCMParser
- FuncGen - IDA Pro Plug-in
- Summary
- Future Work

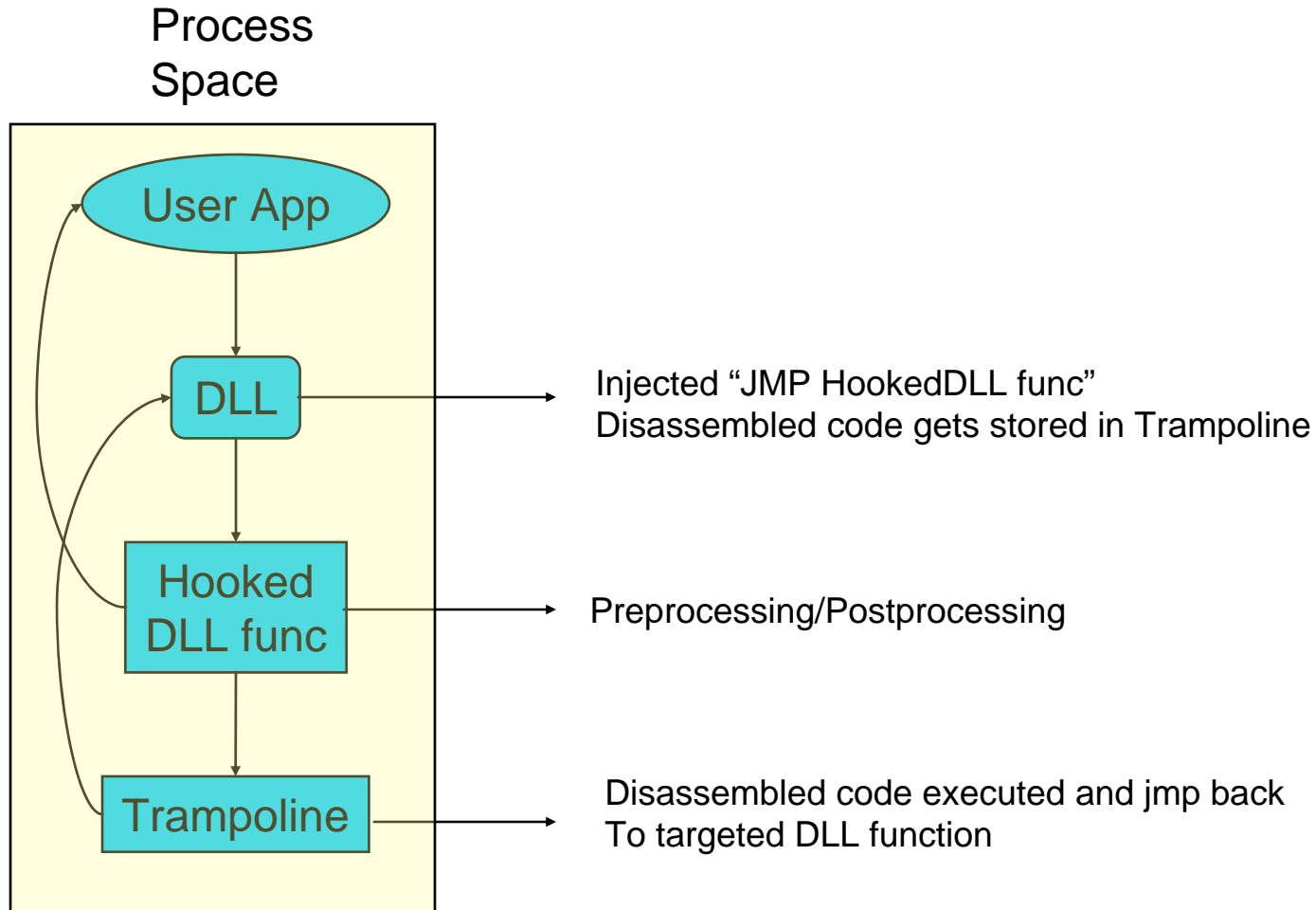
The Problem: Program Reconstruction

- Static analysis using IDA Pro can be a tedious process of running code through a debugger and annotating the disassembly in IDA.
- Decompiler of sorts
 - Why not recreate C function call representative function calls?

Overview

- The Problem: Program Reconstruction
- MS Detours - How it works
- DCM
- Demonstration DCM
- DCMParser
- FuncGen - IDA Pro Plug-in
- Summary
- Future Work

Detours: How it works



Overview

- The Problem: Program Reconstruction
- MS Detours - How it works
- DCM
- Demonstration DCM
- FuncGen - IDA Pro Plug-in
- Summary
- Future Work

DCM

- DCM facilitates:
 - Data flow
 - Code flow
 - Order of execution
 - Translation machine code to C code

How it Works - Sample APP

```
int foo4(int my_var1, int my_var2)
{
    int sum = my_var1 + my_var2;

    printf("foo4\n");
    return sum;
}

void foo3(char *str) { printf("foo3 %s\n", str); }
void foo2(int my_var) { printf("foo2 %X\n", my_var); }
void foo1(void) { printf("foo1\n"); }

void nested(int my_var, char *str)
{
    printf("nested\n");

    foo2(my_var);
    foo3(str);
}

int main(void)
{
    Sleep(100);

    printf("main");
    foo1();

    _asm {
        mov eax, 0xABCD
    }

    foo2(0xBEEF);
    foo3("MyString");
    nested(0xDEAD, "nested");

    int ret = foo4(1,2);
    printf("Return value for foo4 is = %d\n", ret);

    return 0;
}
```



How it Works - Disassembly

```
.text:004017F0 ; int __cdecl main(int argc,const char **argv,const char *
.text:004017F0 main          proc near          ; CODE XREF: __tma:
.text:004017F0
.text:004017F0 var_4          = dword ptr -4
.text:004017F0 argc          = dword ptr  8
.text:004017F0 argv         = dword ptr  0Ch
.text:004017F0 envp         = dword ptr  10h
.text:004017F0
* .text:004017F0          push     ebp
* .text:004017F1          mov     ebp, esp
* .text:004017F3          push     ecx
* .text:004017F4          push     64h          ; dwMilliseconds
* .text:004017F6          call    ds:__imp__Sleep@4 ; Sleep(x)
* .text:004017FC          push     offset aMain ; "main\n"
* .text:00401801          call    ds:__imp__printf
* .text:00401807          add     esp, 4
* .text:0040180A          call    foo1
* .text:0040180F          mov     eax, 0ABCDh
* .text:00401814          push     0BEEFh
* .text:00401819          call    foo2
* .text:0040181E          add     esp, 4
* .text:00401821          push     offset aMystring ; "MyString"
* .text:00401826          call    foo3
* .text:0040182B          add     esp, 4
* .text:0040182E          push     offset aNested ; "nested"
* .text:00401833          push     0DEADh
* .text:00401838          call    nested
* .text:0040183D          add     esp, 8
* .text:00401840          push     2
* .text:00401842          push     1
* .text:00401844          call    foo4
* .text:00401849          add     esp, 8
* .text:0040184C          mov     [ebp+var_4], eax
* .text:0040184F          mov     eax, [ebp+var_4]
* .text:00401852          push     eax
* .text:00401853          push     offset aReturnValueFor ; "Return v.
* .text:00401858          call    ds:__imp__printf
* .text:0040185E          add     esp, 8
* .text:00401861          xor     eax, eax
* .text:00401863          mov     esp, ebp
* .text:00401865          pop     ebp
* .text:00401866          retn
* .text:00401866          main          endp
```

All function calls are data mined

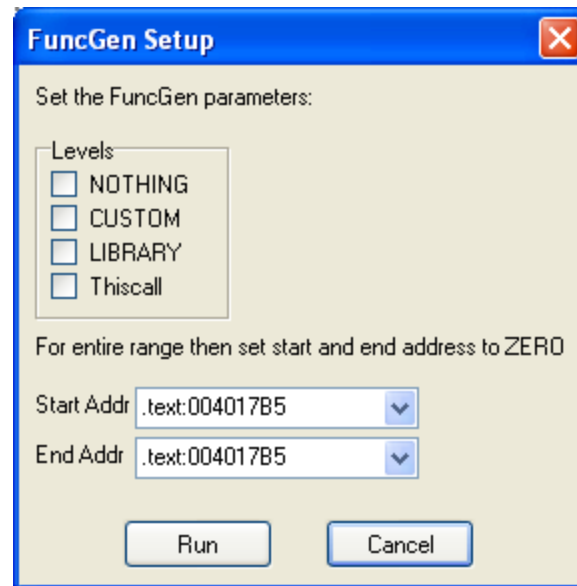


Overview

- The Problem: Program Reconstruction
- MS Detours - How it works
- DCM
- Demonstration DCM
- DCMParser
- FuncGen - IDA Pro Plug-in
- Summary
- Future Work

DM Demo

- Load DataMinerExample.idb
- Load IF.sln
- Run "FuncGen.plw"

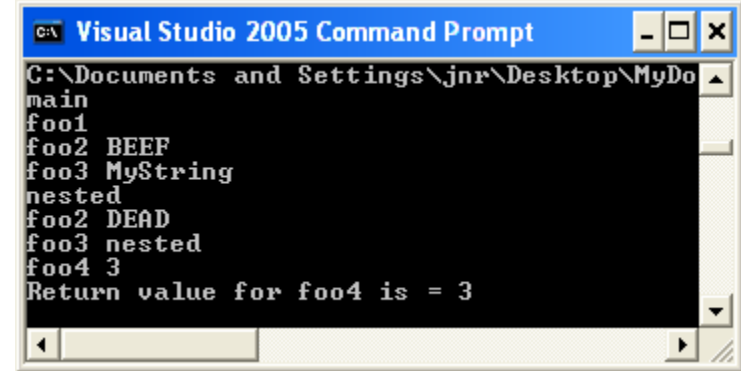


DCM Demo

- 'FuncGen' Library calls
 - Zero out start and end address
- Compile IF.sln
- > start syelogd.exe dm.txt
- > withdll /d:IF.dll dataMinerExample.exe
- 'FuncGen' Custom
 - Start: 0x401730 End: 0x401866
 - Run DM tool
- DCMParser

DCM - Results

```
---REGS--- AX: 3D32F0 BX: 0 CX: 781C37E4 DX: 0 BP: 13FFC0 DI: 4
[[[ 401169 ]]]
main()
  ---REGS--- AX: 5 BX: 0 CX: 7814238E DX: 781C3C58 BP: 13FF48 D
  [[[ 40180F ]]]
  foo1()
  foo1() -> void
  ---REGS--- AX: ABCD BX: 0 CX: 13FF6C DX: 100E3148 BP: 13FF48
  [[[ 40181E ]]]
  foo2(BEEF)
  foo2() -> void
  ---REGS--- AX: 7FFDD000 BX: 0 CX: 13FF6C DX: 100E3148 BP: 13F
  [[[ 40182B ]]]
  foo3(MyString)
  foo3() -> void
  ---REGS--- AX: 7FFDD000 BX: 0 CX: 13FF6C DX: 100E3148 BP: 13F
  [[[ 40183D ]]]
  nested(DEAD, nested)
    ---REGS--- AX: DEAD BX: 0 CX: 7814238E DX: 781C3C58 BP: 13F
    [[[ 4017DA ]]]
    foo2(DEAD)
    foo2() -> void
    ---REGS--- AX: 7FFDD000 BX: 0 CX: 40212C DX: 100E3148 BP: 1
    [[[ 4017E6 ]]]
    foo3(nested)
    foo3() -> void
  nested() -> void
  ---REGS--- AX: 7FFDD000 BX: 0 CX: 13FF6C DX: 100E3148 BP: 13F
  [[[ 401849 ]]]
  foo4(1, 2)
  foo4() -> 3
main() ->0
```



```
C:\ Visual Studio 2005 Command Prompt
C:\Documents and Settings\jnr\Desktop\MyDo
main
foo1
foo2 BEEF
foo3 MyString
nested
foo2 DEAD
foo3 nested
foo4 3
Return value for foo4 is = 3
```

Displays Register contents

Return address of func.
invocation

Parameters + Return values
are displayed

Change flow of program

- Can control return values
- Completely circumvent function calls

```
-----,-----,
foo3() -> void
nested() -> void
---REGS--- AX: 7FFDD000 BX: 0 CX: 13FF6C DX: 100E3148 BP: 13F
[[[ 401849 ]]]
foo4(1, 2)
foo4() -> 3
| main() ->0
```

Overview

- The Problem: Program Reconstruction
- MS Detours - How it works
- DCM
- Demonstration DCM
- **DCMParser**
- FuncGen - IDA Pro Plug-in
- Summary
- Future Work

DCMParser

```
IDA View-A
• .text:004017FC      push    offset aMain      ; "main\n"
• .text:00401801      call   ds:printf
• .text:00401807      add    esp, 4
• .text:0040180A      call   foo1                ;      foo1 (void)
• .text:0040180A      ;                                ;      foo1 () -> 5
• .text:0040180F      mov    eax, 0ABCDh
• .text:00401814      push  0BEEFh
• .text:00401819      call   foo2                ;      foo2 (BEEF)
• .text:00401819      ;                                ;      foo2 () -> A
• .text:0040181E      add    esp, 4
• .text:00401821      push  offset aMyString    ; "MyString"
• .text:00401826      call   foo3                ;      foo3 (402120)
• .text:00401826      ;                                ;      foo3 () -> E
• .text:0040182B      add    esp, 4
• .text:0040182E      push  offset aNested     ; "nested"
• .text:00401833      push  0DEADh
• .text:00401838      call   nested              ;      nested (DEAD, 40212C)
• .text:00401838      ;                                ;      nested () -> C
• .text:0040183D      add    esp, 8
• .text:00401840      push  2
• .text:00401842      push  1
• .text:00401844      call   foo4                ;      foo4 (1, 2)
• .text:00401844      ;                                ;      foo4 () -> 3
• .text:00401849      add    esp, 8
• .text:0040184C      mov    [ebp+var_4], eax

00000C4C  0040184C: _main+5C
```

Overview

- The Problem: Program Reconstruction
- MS Detours - How it works
- DCM
- Demonstration DCM
- DCMParser
- FuncGen - IDA Pro Plug-in
- Summary
- Future Work

IDA Pro Plug-in

- Plug-in allows the user to automatically generate a detailed list, in the form of a .cpp file
- Figure 1 shows what is generated by the plug-in

```
// Plug-in generates this declaration and
// set of calls for every function
static void (__cdecl * Real_func)(int,
    int) = NULL;
...
Real_func = (void (__cdecl *) (int,
    int)) 0x401750;
ATTACH(&(PVOID&)Real_func, Mine_func);
...
DETACH(&(PVOID&)Real_func, Mine_func);
```

Figure 1: Function hook

Summary

- Data/Code mining through our custom utility can aid the reverse engineer to construct a quick data and code flow analysis after just one run of the executable.

Contact

Jason Raber

Team Lead - Reverse Engineer

937-427-7085

jraber@rri-usa.org

