



THALIUM

Symless

An IDA reverse assistant

R. Midget – B. Verstraeten
June 9th 2023

About us

B. Verstraeten

- Reverser @ [Thalium](#)



Royal Midget

- Ex Reverser @ [Thalium](#)
- Now Reverser @ [Synacktiv](#)



Summary

- Introduction to Symless
- Application on a SSTIC challenge
- Symless under the hood
- Live Demo on a Real-Life example

Symless - Introduction

Symless – Introduction

What is Symless ?

- **A Python IDA Pro Plugin**
- **Symbol-less**

What is its purpose ?

- **Make the reverser's life easier by automating some of their tasks**

How ?

- **Automatic structure** buildings
- **Propagating** type information in the database to generate **xrefs**
- **Improving** human-readability of **decompiler** output



Example on SSTIC 2022 binary challenge

SSTIC 2022 Challenge example

Binary is

- ELF x64 with symbols (functions names)
- Simple FTP Server implemented in C

Vulnerability research

- Looking at dangerous API like **read**
 - Is there an OOB ?

Illustrate symless features while checking this potential vulnerability

How to check if there is a vulnerability ?

```
nb_bytes_read = Read(fd_src, destination_buffer, nb_max_bytes_to_read)
```

```
void __fastcall __noreturn handleClientFTPServer(char *arg1__)  
{  
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
```

```
    retaddr = sign_pointer(retaddr, v10);
```

```
    arg1 = arg1_;
```

```
    v1 = *(_DWORD *) (arg1_ + 4);
```

```
    v2 = welcome_msg;
```

```
    v3 = strlen(welcome_msg);
```

```
    write(v1, v2, v3);
```

```
    while ( 1 )
```

Fd_src

Dest_buf Nb_max_bytes

```
{  
    v12 = read(*(_DWORD *) (arg1_ + 4), (void *) (arg1_ + 24), 0x400uLL);  
    *(_BYTE *) (arg1_ + v12 + 24) = 0;  
    v4 = arg1;  
    v5 = (__int64 (__fastcall *) (__int64)) auth_pointer(*(_QWORD *) (arg1_ + 1096), 0LL);  
    v11 = v5(v4);  
    v8 = (void (__fastcall *) (__int64, __int64)) auth_pointer(*(_QWORD *) (arg1_ + 1104), 0LL);  
    v8(arg1, v11);  
}
```

Write byte 0 at max index
1024 of (arg1 + 24)

How to know if there is an OOB Write ?

What is the consequence of the OOB ?

Does the attacker control these data ?

How to check if there is a vulnerability without Symless ?

```
void __fastcall __noreturn handleClientFTPServer(char *arg1__)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    retaddr = sign_pointer(retaddr, v10);
    arg1 = arg1__;
    v1 = *(_DWORD *) (arg1 + 4);
    v2 = welcome_msg;
    v3 = strlen(welcome_msg);
    write(v1, v2, v3);
    while ( 1 )
    {
        v12 = read(*(_DWORD *) (arg1 + 4), (void *) (arg1 + 24), 0x400uLL);
        *(_BYTE *) (arg1 + v12 + 24) = 0;
        v4 = arg1;
        v5 = (__int64 (__fastcall *) (__int64))auth_pointer(*(_QWORD *) (arg1 + 1096), 0LL);
        v11 = v5(v4);
        v8 = (void (__fastcall *) (__int64, __int64))auth_pointer(*(_QWORD *) (arg1 + 1104), 0LL);
        v8(arg1, v11);
    }
}
```

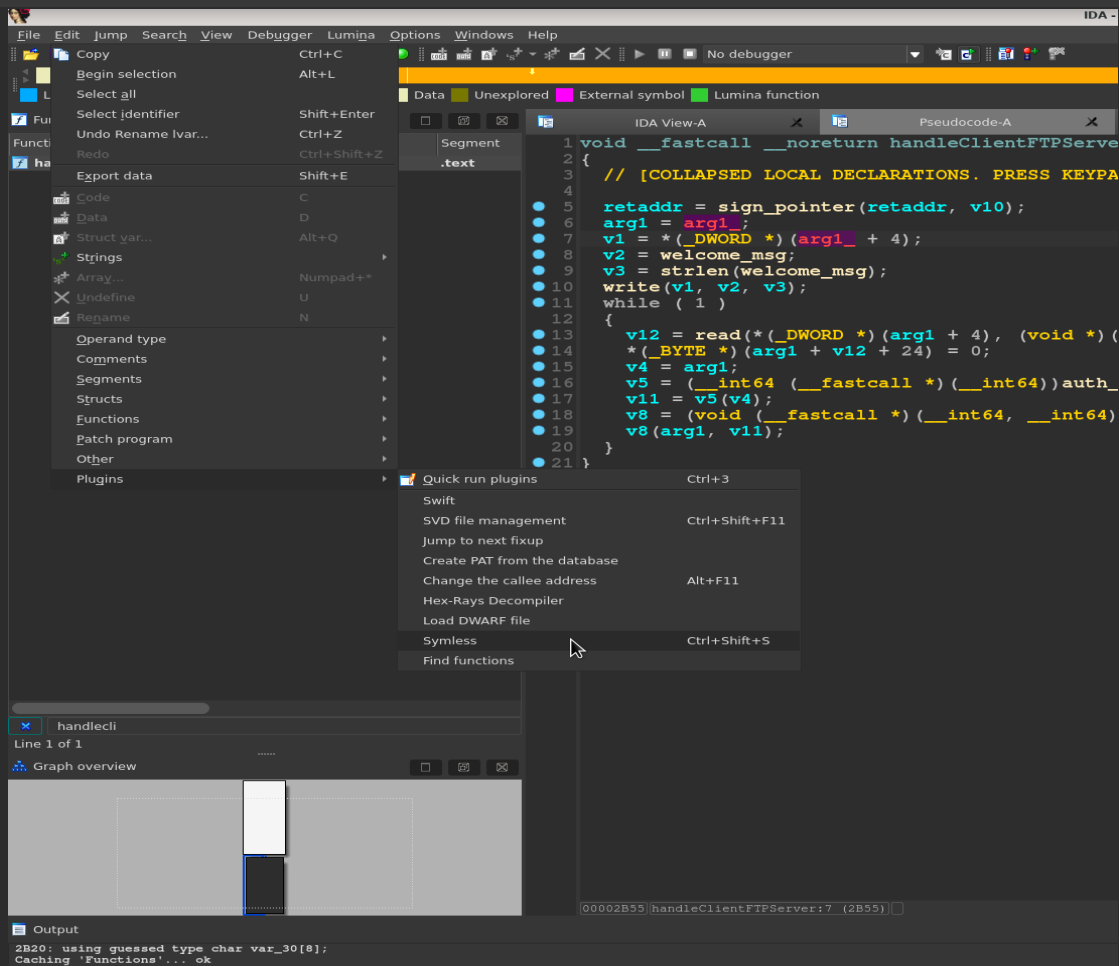
Find the constructor of object arg1

- How ? Follow backward from caller to caller until finding the origin
- Then ? Create the structure and fields associated

Find all the methods that use the object arg1

- How ? Follow backward and forward the propagation of the object in callers and callees
- Then ? Understand usage of all of the fields

Symless application



Feature : Overall enhancement of decompiler output

Before Symless

```
void __fastcall __noreturn handleClientFTPServer(char *arg1_)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    retaddr = sign_pointer(retaddr, v10);
    arg1 = arg1_;
    v1 = *(_DWORD *) (arg1_ + 4);
    v2 = welcome_msg;
    v3 = strlen(welcome_msg);
    write(v1, v2, v3);
    while ( 1 )
    {
        v12 = read(*(_DWORD *) (arg1 + 4), (void *) (arg1 + 24), 0x400uLL);
        *(_BYTE *) (arg1 + v12 + 24) = 0;
        v4 = arg1;
        v5 = (__int64 (__fastcall *) (__int64))auth_pointer(*(_QWORD *) (arg1 + 1096), 0LL);
        v11 = v5(v4);
        v8 = (void (__fastcall *) (__int64, __int64))auth_pointer(*(_QWORD *) (arg1 + 1104), 0LL);
        v8(arg1, v11);
    }
}
```

Comparison before / after applying Symless on the database

Feature : Overall enhancement of decompiler output

After Symless

```
void __fastcall __noreturn handleClientFTPServer(struc_newFTPServer *newFtpServer__)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    retaddr = sign_pointer(retaddr, v10);
    newFtpServer = newFtpServer_;
    v1 = newFtpServer->field_00000004;
    v2 = welcome_msg;
    v3 = strlen(welcome_msg);
    write(v1, v2, v3);
    while ( 1 )
    {
        v12 = read(newFtpServer->field_00000004, &newFtpServer->field_00000018, 0x400uLL);
        *((_BYTE *)&newFtpServer->field_00000018 + v12) = 0;
        v4 = newFtpServer;
        v5 = (__int64 (__fastcall *) (struc_newFTPServer *))auth_pointer(newFtpServer->parseCommandFTPServer, 0LL);
        v11 = v5(v4);
        v8 = (void (__fastcall *) (struc_newFTPServer *, __int64))auth_pointer(newFtpServer->handleCommandFTPServer, 0LL);
        v8(newFtpServer, v11);
    }
}
```

Comparison before / after applying Symless on the database

Feature : Automatic structures building

Structure FTPServer

- 30 fields
- 161 xrefs created

Minimum 200 mouse clicks saved

- Eco-friendly

```
00000000 ; Allocated at: 0x2710
00000000 struct_newFTPServer struct ; (sizeof=0x4E0, mappedto_30)
00000000 field_00000000 dd ? ; XREF: newFTPServer+5D/w
00000000 ; startFTPServer+3E/r ...
00000004 field_00000004 dd ? ; XREF: startFTPServer+49/w
00000004 ; handleClientFTPServer+35/r ...
00000008 field_00000008 dd ? ; XREF: parseCommandFTPServer+C2/r
00000008 ; parseCommandFTPServer+DB/r ...
0000000C padd_0000000C dd ?
00000010 field_00000010 dq ? ; XREF: destructorFTPServer+4A/r
00000010 ; destructorFTPServer+55/r ...
00000018 field_00000018 dq ? ; XREF: handleClientFTPServer+63/o
00000018 ; parseCommandFTPServer+51/o ...
00000020 padd_00000020 db 1016 dup(?)
00000418 field_00000418 db ? ; XREF: getUsernameFTPServer+49/r
00000418 ; canExecCmdFTPServer+4C/r ...
00000419 padd_00000419 db 7 dup(?)
00000420 a500UnknownComm dq ? ; XREF: handleCommandFTPServer+83/w
00000420 ; handleCommandFTPServer+285/w ...
00000428 field_00000428 db ? ; XREF: handleCommandFTPServer+2F9/r
00000428 ; handleCommandFTPServer+316/w ...
00000429 field_00000429 db ? ; XREF: parseCommandFTPServer+B5/r
00000429 ; handleCommandFTPServer+292/r ...
0000042A padd_0000042A db 6 dup(?)
00000430 dword_0 dq ? ; XREF: getUsernameFTPServer+2E/r
00000430 ; getUsernameFTPServer+56/r ...
00000438 startFTPServer dq ? ; XREF: main+46/r
00000438 ; newFTPServer+71/w
00000440 handleClientFTPServer dq ? ; XREF: newFTPServer+8A/w
00000440 ; startFTPServer+50/r
00000448 parseCommandFTPServer dq ? ; XREF: newFTPServer+A3/w
00000448 ; handleClientFTPServer+85/r
00000450 handleCommandFTPServer dq ? ; XREF: newFTPServer+BC/w
00000450 ; handleClientFTPServer+A4/r
00000458 getUsernameFTPServer dq ? ; XREF: newFTPServer+D5/w
00000458 ; parseCommandFTPServer+96/r
00000460 canExecCmdFTPServer dq ? ; XREF: newFTPServer+EE/w
00000460 ; handleCommandFTPServer+39/r
00000468 getPermsFTPServer dq ? ; XREF: newFTPServer+107/w
00000468 ; handleRetrFTPServer+DC/r
00000470 destructorFTPServer dq ? ; XREF: newFTPServer+120/w
00000470 ; handleQuitFTPServer+85/r
00000478 handleUserFTPServer dq ? ; XREF: newFTPServer+139/w
00000478 ; handleCommandFTPServer+B6/r
00000480 handlePassFTPServer dq ? ; XREF: newFTPServer+152/w
00000480 ; handleCommandFTPServer+DD/r
00000488 handleTypeFTPServer dq ? ; XREF: newFTPServer+16B/w
00000488 ; handleCommandFTPServer+104/r
```

How to know if there is an OOB Write ?

```
void __fastcall __noreturn handleClientFTPServer(struc_newFTPServer *newFtpServer__)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    retaddr = sign_pointer(retaddr, v10);
    newFtpServer = newFtpServer_;
    v1 = newFtpServer->field_00000004;
    v2 = welcome_msg;
    v3 = strlen(welcome_msg);
    write(v1, v2, v3);
    while ( 1 )
    {
        v12 = read(newFtpServer->field_00000004, &newFtpServer->field_00000018, 0x400uLL);
        *((_BYTE *)&newFtpServer->field_00000018 + v12) = 0;
        v4 = newFtpServer;
        v5 = (__int64 (__fastcall *) (struc_newFTPServer *)) auth_pointer(newFtpServer->parseCommandFTPServer, 0LL);
        v11 = v5(v4);
        v8 = (void (__fastcall *) (struc_newFTPServer *, __int64)) auth_pointer(newFtpServer->handleCommandFTPServer, 0LL);
        v8(newFtpServer, v11);
    }
}
```

Write up to 0x400 bytes

Write byte 0 at index
Between 0 and 1024
of field_018

Feature : Automatic Structure buildings

Does size(field_018) greater than 1025/0x401 ?

Size is 1024/0x400

```
00000000 ; Allocated at: 0x2710
00000000 struc_newFTPServer struc ; (sizeof=0x4E0, mappedto_30)
00000000 listening_socket dd ? ; XREF: newFTPServer+5D/w
00000000 ; startFTPServer+3E/r ...
00000004 field_00000004 dd ? ; XREF: startFTPServer+49/w
00000004 ; handleClientFTPServer+35/r ...
00000008 field_00000008 dd ? ; XREF: parseCommandFTPServer+C2/r
00000008 ; parseCommandFTPServer+DB/r ...
0000000C padd__0000000c dd ?
00000010 field_00000010 dq ? ; XREF: destructorFTPServer+4A/r
00000010 ; destructorFTPServer+55/r ...
00000018 field_00000018 dq ? ; XREF: handleClientFTPServer+63/o
00000018 ; parseCommandFTPServer+51/o ...
00000020 padd__00000020 db 1016 dup(?)
00000418 field_00000418 db ? ; XREF: getUsernameFTPServer+49/r
00000418 ; canExecCmdFTPServer+4C/r ...
00000419 padd__00000419 db 7 dup(?)
00000420 a500UnknownComm dq ? ; XREF: handleCommandFTPServer+83/w
00000420 ; handleCommandFTPServer+285/w ...
00000428 field_00000428 db ? ; XREF: handleCommandFTPServer+2F9/r
00000428 ; handleCommandFTPServer+316/w ...
00000429 field_00000429 db ? ; XREF: parseCommandFTPServer+B5/r
00000429 ; handleCommandFTPServer+292/r ...
0000042A padd__0000042a db 6 dup(?)
```

➔ Write Null-Byte Off By One in field_0418

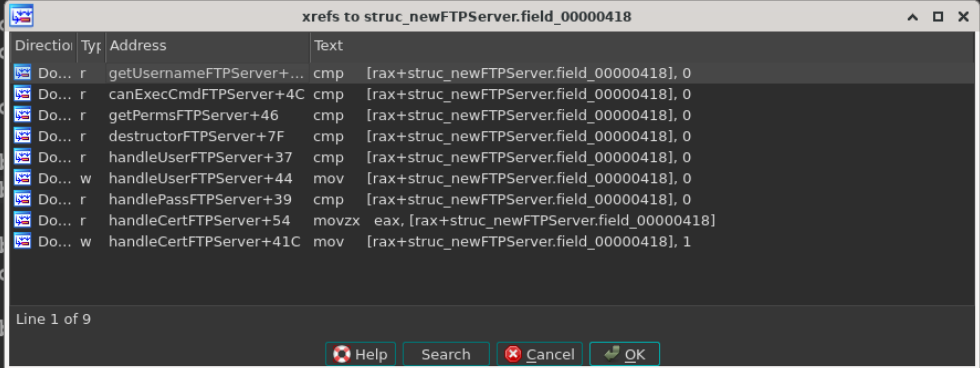
Feature : Xrefs on structures fields : field_0418

```
00000004 ; handleClientFTPServer+35/r ...
00000008 field_00000008 dd ? ; XREF: parseCommandFTPServer+C2/r
00000008 ; parseCommandFTPServer+DB/r
0000000C padd_0000000c dd ?
00000010 field_00000010 dd ?
00000018 field_00000018 dd ?
00000018 field_00000018 dd ?
00000020 padd_00000020 dd ?
00000418 field_00000418 dd ?
00000418 field_00000418 dd ?
00000419 padd_00000419 dd ?
00000420 a500UnknownComm dd ?
00000420 a500UnknownComm dd ?
00000428 field_00000428 dd ?
00000428 field_00000428 dd ?
00000429 field_00000429 db ? ; XREF: parseCommandFTPServer+B5/r

struc_newFTPServer *handleCertFTPServer()
{
    ...
}

struc_newFTPServer *handleUserFTPServer()
{
    ...
}

mov [rax+struc_newFTPServer.field_00000418], 1 mov [rax+struc_newFTPServer.field_00000418], 0
```



Flag indicating CERT or USER type

- Possible to confuse CERT to USER by nullifying field_0000418

Are data controlled by an attacker ?

```
void __fastcall __noreturn handleClientFTPServer(struc_newFTPServer *newFtpServer__)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    retaddr = sign_pointer(retaddr, v10);
    newFtpServer = newFtpServer_;
    v1 = newFtpServer->field_00000004;
    v2 = welcome_msg;
    v3 = strlen(welcome_msg);
    write(v1, v2, v3);
    while ( 1 )
    {
        v12 = read(newFtpServer->field_00000004, &newFtpServer->field_00000018, 0x400uLL);
        *((_BYTE *)&newFtpServer->field_00000018 + v12) = 0;
        v4 = newFtpServer;
        v5 = (__int64 (__fastcall *) (struc_newFTPServer *))auth_pointer(newFtpServer->parseCommandFTPServer, 0LL);
        v11 = v5(v4);
        v8 = (void (__fastcall *) (struc_newFTPServer *, __int64))auth_pointer(newFtpServer->handleCommandFTPServer, 0LL);
        v8(newFtpServer, v11);
    }
}
```

Feature : Xrefs on structures fields

```
00000000 ; Allocated at: 0x2710
00000000 struct_newFTPServer struc ; (sizeof=0x4E0, mappedto_30)
00000000 listening_socket dd ? ; XREF: newFTPSever
00000000 ; startFTPServer+3F
00000004 field_00000004 dd ? ; XREF: startFTPSer
00000004 ; handleClientFTPSe
00000008 field_00000008 dd ? ; XREF: parseComman
00000008 ; parseCommandFTPSe
00000000
00000000
00000001
00000001 Up w startFTPServer+49 mov [rcx+struct_newFTPServer.field_00000004], eax
00000001 Up r handleClientFTPServer+35 mov ebx, [rax+struct_newFTPServer.field_00000004]
00000001 Up r handleClientFTPServer+5C mov edi, [rax+struct_newFTPServer.field_00000004]; fd
00000001 Up r handleCommandFTPServ... mov r14d, [rax+struct_newFTPServer.field_00000004]
00000001 Up r destructorFTPServer+33 mov edi, [rax+struct_newFTPServer.field_00000004]; fd
00000001 Up r handlePasvFTPServer+71 mov edi, [rax+struct_newFTPServer.field_00000004]; fd
00000001 Up r handleListFTPServer+D5 mov r14d, [rax+struct_newFTPServer.field_00000004]
00000001 Up r handleRetrFTPServer+1B2 mov r14d, [rax+struct_newFTPServer.field_00000004]
00000001 Up r handleQuitFTPServer+54 mov r14d, [rax+struct_newFTPServer.field_00000004]
00000002
00000002
00000002
00000002
00000002
00000002
00000002
00000002
00000002
00000002
00000002
```

xrefs to struct_newFTPServer.field_00000004

Direction	Type	Address	Text
Up	w	startFTPServer+49	mov [rcx+struct_newFTPServer.field_00000004], eax
Up	r	handleClientFTPServer+35	mov ebx, [rax+struct_newFTPServer.field_00000004]
Up	r	handleClientFTPServer+5C	mov edi, [rax+struct_newFTPServer.field_00000004]; fd
Up	r	handleCommandFTPServ...	mov r14d, [rax+struct_newFTPServer.field_00000004]
Up	r	destructorFTPServer+33	mov edi, [rax+struct_newFTPServer.field_00000004]; fd
Do...	r	handlePasvFTPServer+71	mov edi, [rax+struct_newFTPServer.field_00000004]; fd
Do...	r	handleListFTPServer+D5	mov r14d, [rax+struct_newFTPServer.field_00000004]
Do...	r	handleRetrFTPServer+1B2	mov r14d, [rax+struct_newFTPServer.field_00000004]
Do...	r	handleQuitFTPServer+54	mov r14d, [rax+struct_newFTPServer.field_00000004]

Line 1 of 9

[Help] [Search] [Cancel] [OK]

```
newFtpServer->field_00000004 = accept_conn();
```

```
00000000 ; Allocated at: 0x2710
00000000 struct_newFTPServer struct
00000000 listening_socket dd ?
00000000
00000004 client_socket dd ?
00000004
```

Two thick, curved lines, one bright green and one white, sweep from the bottom right towards the top right of the slide.

How Symless works ?

Symless data flow

Symless **propagates** a structure from **disassembly**

Structure pointer →

Propagation ↓

```
lea rax, const CINet::`vftable'{for `IInternetProtocolEx'}
mov [rcx], rax
mov rbx, rcx
lea rax, const CINetHttp::`vftable'{for `IWinInetHttpInfo'}
mov [rcx+88h], rbp
mov [rcx+8], rax
mov rsi, r8
lea rax, const CINetHttp::`vftable'{for `IWinInetCacheHints2'}
mov dword ptr [rcx+90h], 0Bh
mov [rcx+10h], rax
mov rdi, rdx
lea rax, const CINetHttpS::`vftable'{for `IInternetThreadSwitch'}
mov [rcx+98h], rbp
mov [rcx+18h], rax
```

Gather accessed fields

Recognize **virtual table** loading

Data flow entries

Where to apply this propagation ?

- Before structures initialisation

Symless uses two types of entry points

- Memory allocation for structures
- Constructor of C++ classes (from **virtual tables**)

```
mov    ecx, 68h ; 'h' ; size
mov    edi, 8007000Eh
call   malloc_0
mov    rbx, rax ← Entry
test   rax, rax
jz     loc_1800782BF
xor     edx, edx
mov    rcx, rax
lea    r8d, [rdx+68h]
call   memset_0
```

Conflicts resolution

Conflicts have to be resolved before getting decent results

- Structure **inner conflicts**
- **Conflicts between structures**
- **Duplicates** need to be **merged**

Conflicts resolution is **the most complicated part** of the process

- E.g., HexRaysPyTools asks ~~to~~ the reverser

Symless handles them automatically

- Using multiple **heuristics**
- No user interaction is required

The Symless Experience

Symless aims to build a **complete backbone** for each structure

- By identifying all its **fields**
- By linking classes and their **virtual tables**

Finding relevant names & types is left to the user

- This is done **along the reversing process**

Symless can use some symbols to rename the structures

The Symless Experience - simplified

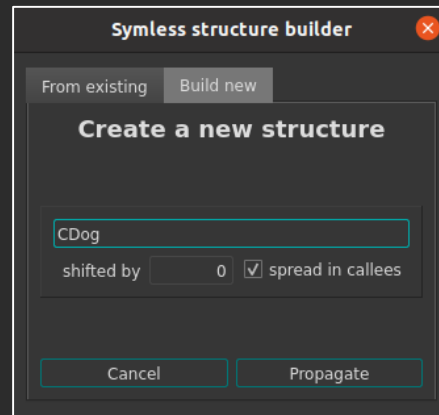
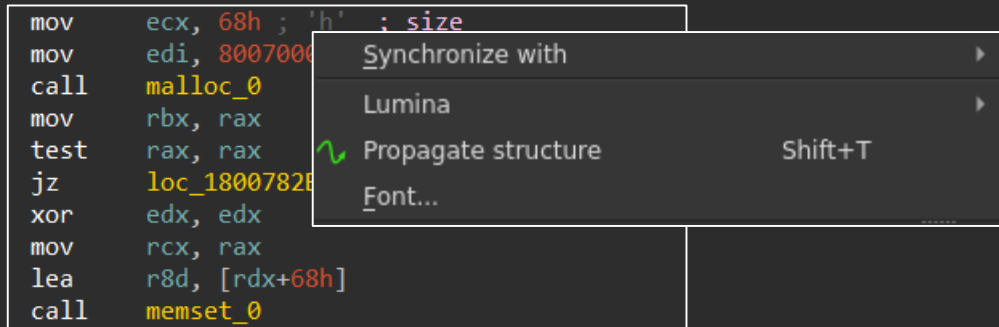


Symless modes

Pre-Analysis mode

```
> python .\symless.py .\urlmon.dll
Using IDA installation: "C:\Program Files\IDA Pro 7.7"
Creating IDA database from binary C:\Users\bapti\Desktop\symless\urlmon.dll
Running IDA script..
* IDAT : C:\Program Files\IDA Pro 7.7\idat64.exe
* Script: C:\Users\bapti\Desktop\symless\symless.py ("--config", "C:\Users\bapti\Desktop\symless\urlmon.dll", "--prefix", "ajdqtb")
* Base : C:\Users\bapti\Desktop\symless\urlmon.dll.i64
* Logs : C:\Users\bapti\AppData\Local\Temp\urlmon_tcx5_fz7.log
```

Interactive plugin mode



Live demo

Conclusion

Symless features

- Automatic **structures building**
- **Xrefs on structures fields**
- Overall **enhancement of decompiler output**
- And others..



<https://github.com/thalium/symless>

Support

- x64 & x86 binaries
- IDA 7 Pro +



<https://thalium.re/>

Any questions ?