

how i
met your

pointer

Hijacking client software for fuzz and profit



OVERVIEW

- Introduction.
- Fuzzing 101. *yawn*
- The need for a different approach.
 - Abusing the client.
- A possible implementation. Boyka.
- **EXPERIMENT.**
- Conclusion.

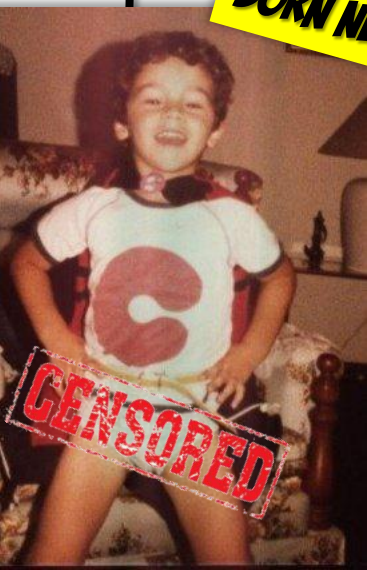


\$ WHOAMI

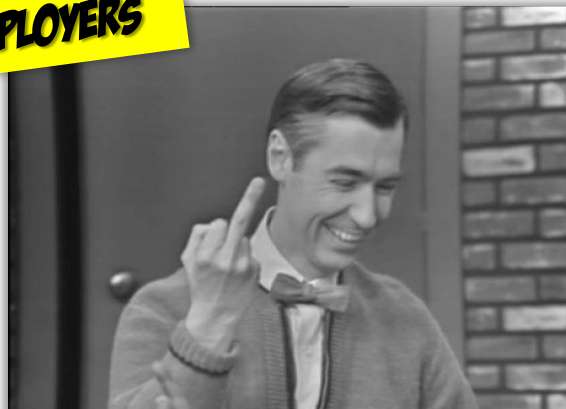
PARTICLE PHYSICIST



BORN NERD



**PROSPECTIVE
EMPLOYERS**



FK THAT SH*T**



\$ WHOAMI

LICHTBILDAUSWEIS
IDENTITY CARD / CARTE IDENTITE

9C3XVP8V

Name / Suriname / Nom
OF NORWAY
Vorname / Given name / Prénoms
PRIME MINISTER
Geburtsort / Place of birth / Lieu de naissance

0

getOriginal / Nationality / Nationalité
NORWEGEN
Gültig bis / Date of expiry / Date d'expiration
28.12.2017



LET'S BE CLEAR

THIS WORK IS SHIT

LET'S BE CLEAR

**THIS WORK IS  SAY
"ESSENTIALLY FLAWED"**

WHAT THIS IS ABOUT

- Interesting approach to software testing
- Touching things you are not supposed to
- Breaking stuff (if you're lucky!)
- Multiple references to pop culture
 - and chocolate!



Look for the
ChocoQuiz icon



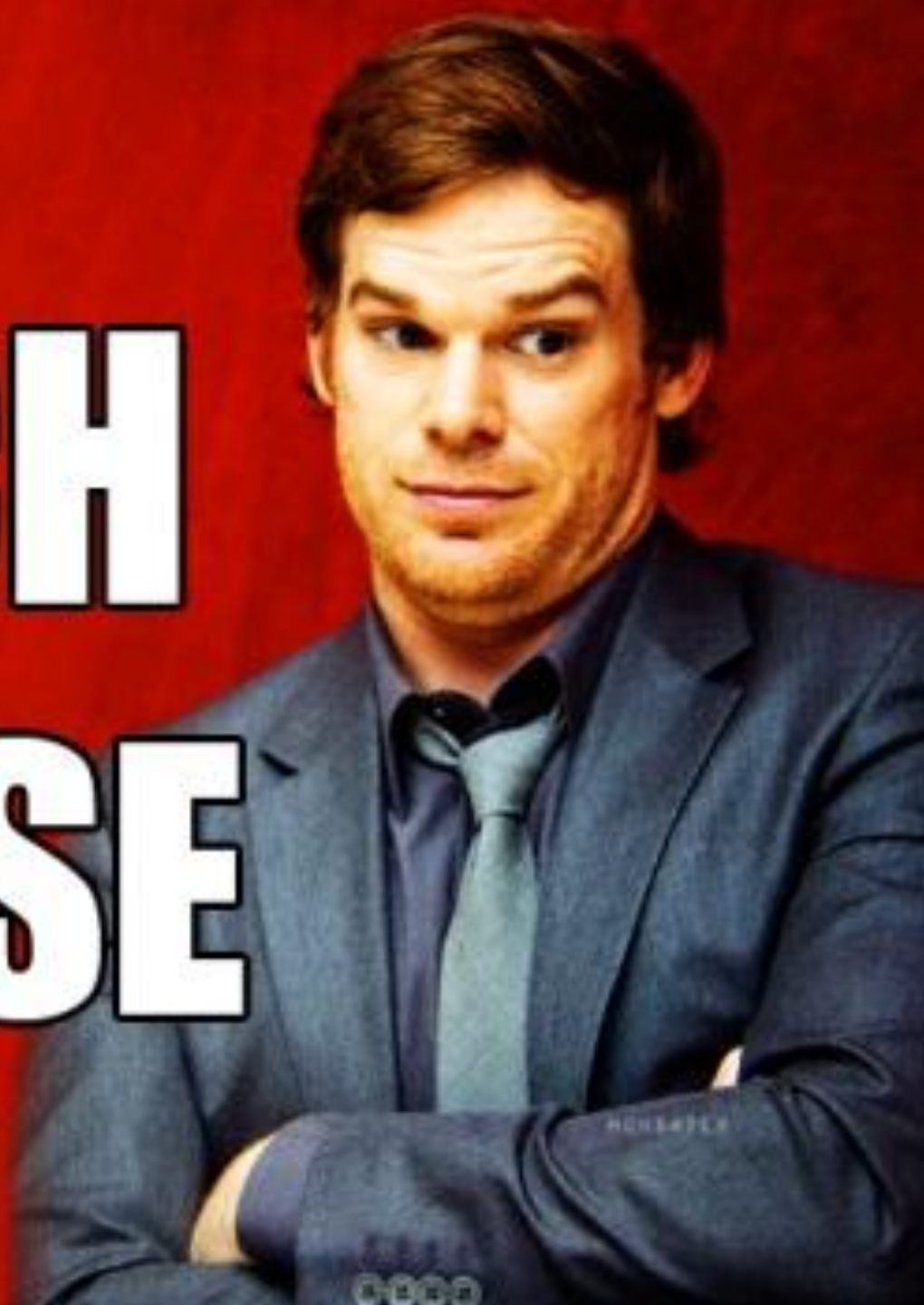
DISCLAIMER

**FOR EDUCATIONAL PURPOSES
ONLY...**



Fuzzing 101

**BITCH
PLEASE**



**Fuzzing is like violence:
if it doesn't solve your problems,
you are not using enough of it.**



BRUTE FORCE

MONSTER CONFIGURATION

```
1  from sulley import *
2
3  s_initialize("user")
4  s_static("USER")
5  s_delim(" ")
6  s_string("yomama")
7  s_static("\r\n")
8
9  s_initialize("pass")
10 s_static("PASS")
11 s_delim(" ")
12 s_string("issofat")
13 s_static("\r\n")
14
15 s_initialize("cwd")
16 s_static("CWD")
17 s_delim(" ")
18 s_string("c: ")
19 s_static("\r\n")
20
21 s_initialize("delete")
22 s_static("DELETE")
23 s_delim(" ")
24 s_string("\\test.txt")
25 s_static("\r\n")
```

```
from sulley import *
from requests import ftp # this is our ftp.py file

sess = sessions.session(session_filename="audits/freefloatftp.session")
target = sessions.target("192.168.1.11", 21)
target.netmon = pedrpc.client("192.168.1.11", 26001) # NetMonitor (packets)
target.procmon = pedrpc.client("192.168.1.11", 26002) # ProcMonitor (crashes)
target.procmon_options = { "proc_name" : "FTPServer.exe" }

sess.add_target(target)

sess.connect(s_get("user"))
sess.connect(s_get("user"), s_get("pass"))

sess.connect(s_get("pass"), s_get("cwd"))
sess.connect(s_get("pass"), s_get("delete"))
sess.connect(s_get("pass"), s_get("mdtm"))
sess.connect(s_get("pass"), s_get("mkd"))

sess.fuzz()
```

ftp.py - protocol

ftp_session.py

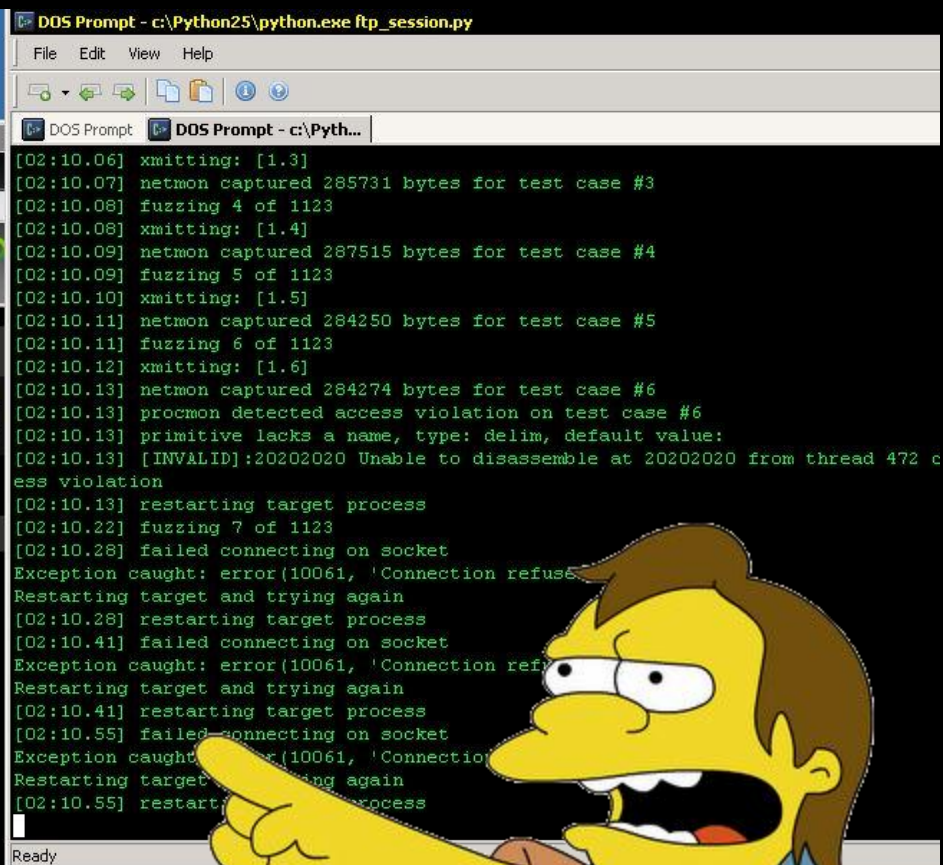
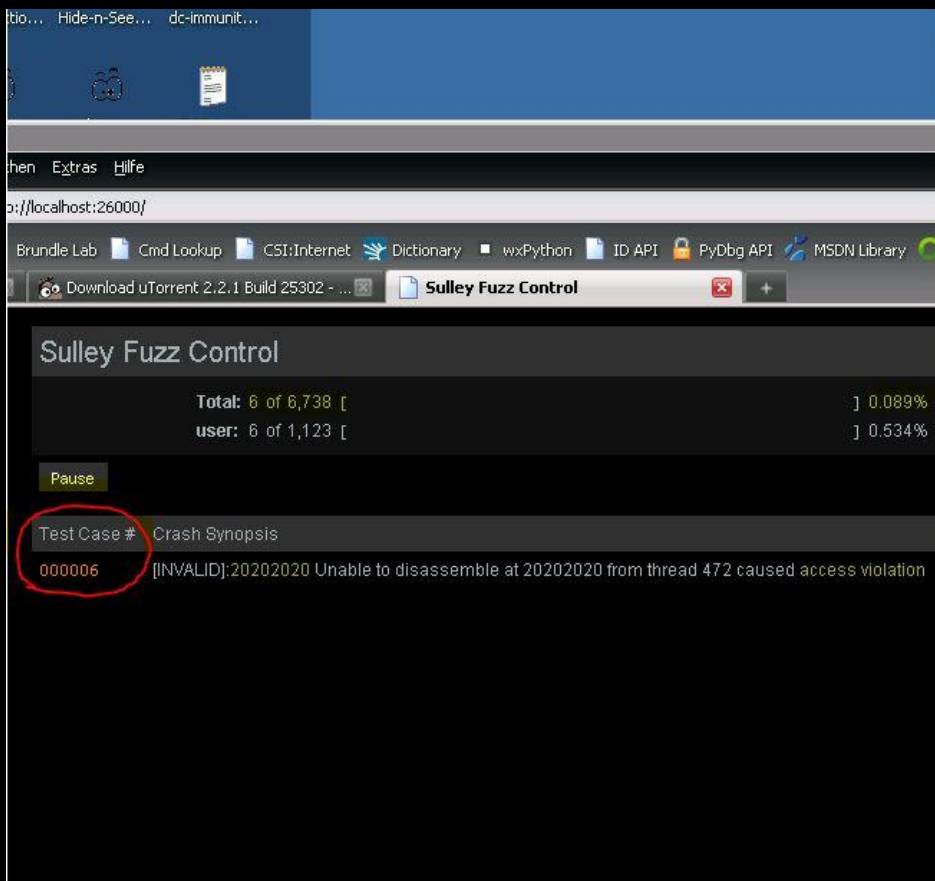


And Now



I Wait

CRASH! BOOM! BANG! HAHA!



PRECISE CRASH INFORMATION

```
http://localhost:26000/view_crash/6 +
[INVALID]:20202020 Unable to disassemble at 20202020 from thread 472 caused access violation
when attempting to read from 0x20202020

CONTEXT DUMP
EIP: 20202020 Unable to disassemble at 20202020
EAX: 00000216 ( 534) -> N/A
EBX: 00000002 ( 2) -> N/A
ECX: 0014d3c0 ( 1364928) -> F unt authority\systemzA (heap)
EDX: 7c90e514 (2089870612) -> N/A
EDI: 003b19d5 ( 3873237) -> (heap)
ESI: 0040a44e ( 4236366) -> N/A
EBP: 003b1298 ( 3871384) -> N/A
ESP: 00b2fc2c ( 11729964) ->
+00: 20202020 ( 538976288) -> N/A
+04: 20202020 ( 538976288) -> N/A
+08: 20202020 ( 538976288) -> N/A
+0c: 20202020 ( 538976288) -> N/A
+10: 20202020 ( 538976288) -> N/A
+14: 20202020 ( 538976288) -> N/A

disasm around:
0x20202020 Unable to disassemble

SEH unwind:
fffffff -> kernel32.dll:7c839ad8 push ebp
```



THERE'S ALWAYS A BUT



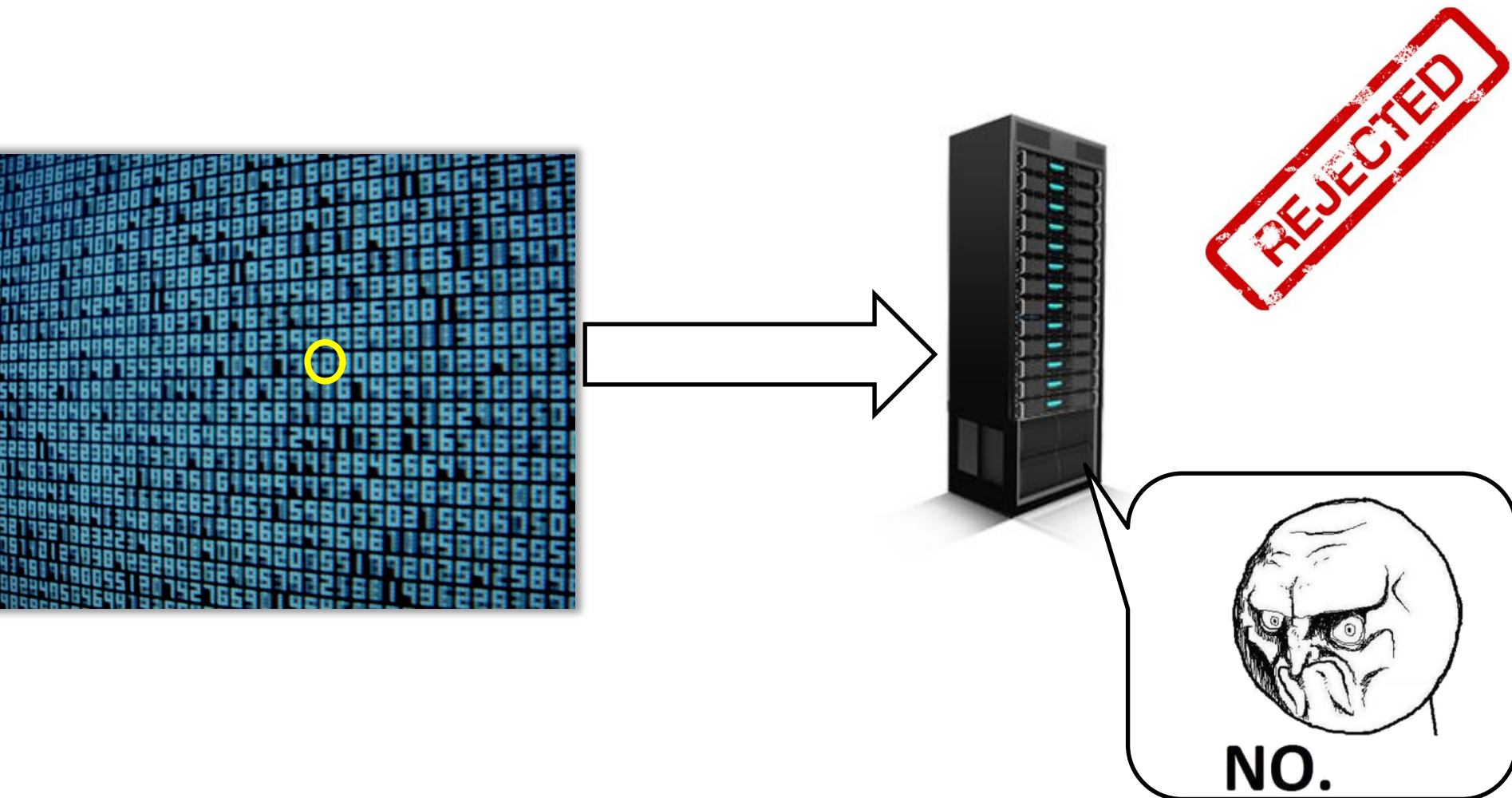
That's all very nice.

**But what if I *don't*
know the protocol?**

**There is NO
documentation at all.
:((((**



*I CAN ALWAYS TRY **DUMB** FUZZING!*



THINK ABOUT CHECKSUMS...

Packet

Data

Checksum

Checksum = SHA1(Data)

SHA1: 160 bits

$P(\text{right}) = 1/2^{160} \approx 1/10^{48}$

$10^{48} = 1\text{k} \cdot 1\text{T} \cdot 1\text{T} \cdot 1\text{T} \cdot 1\text{T} \cdot 1\text{T}$

Dumb Fuzzing...



F*CK!



IS EVERYTHING LOST ?

**FAIL
HARDER**

**THINK
WRONG**

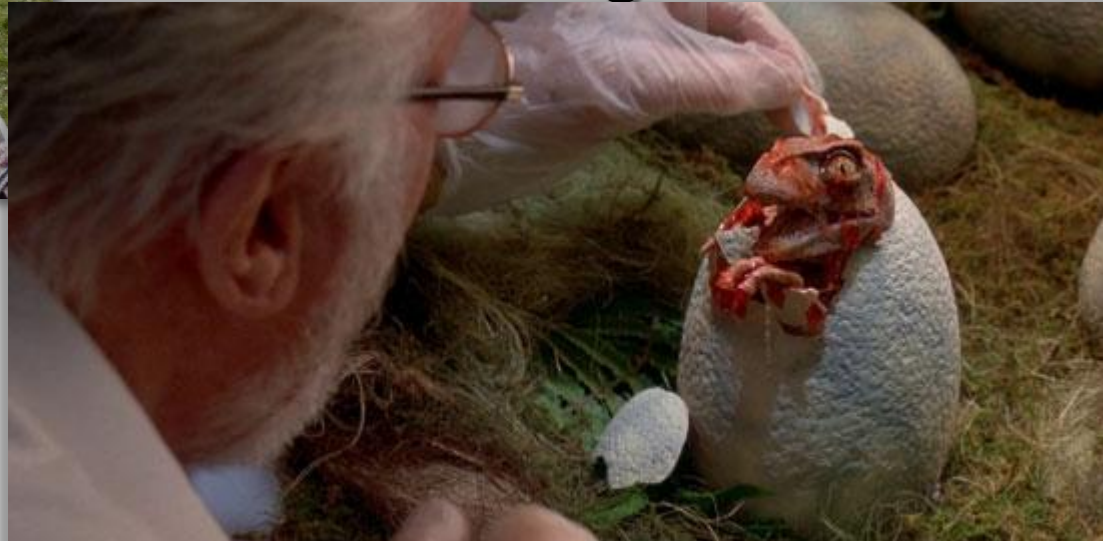
**The need for
a *different*
approach**



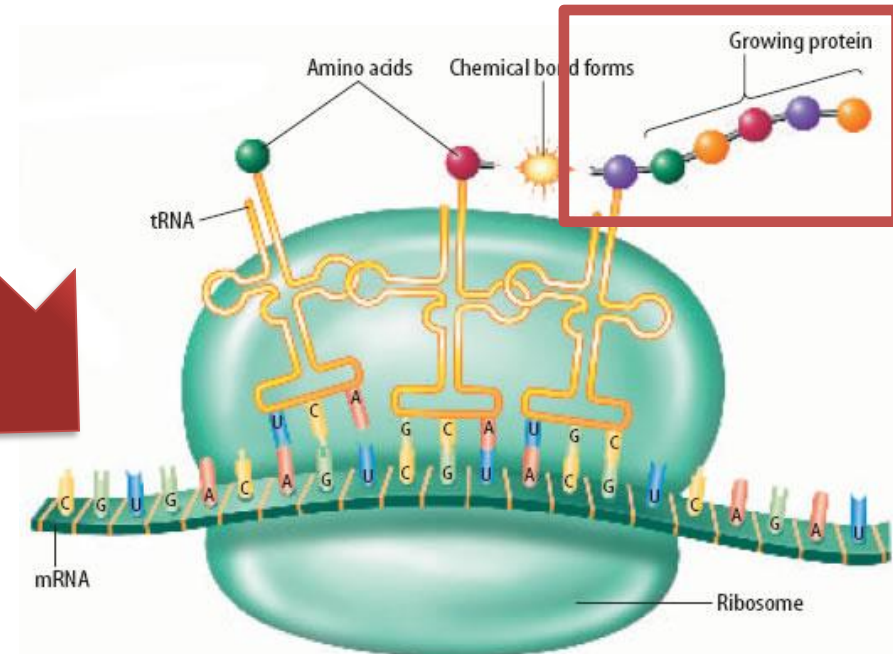
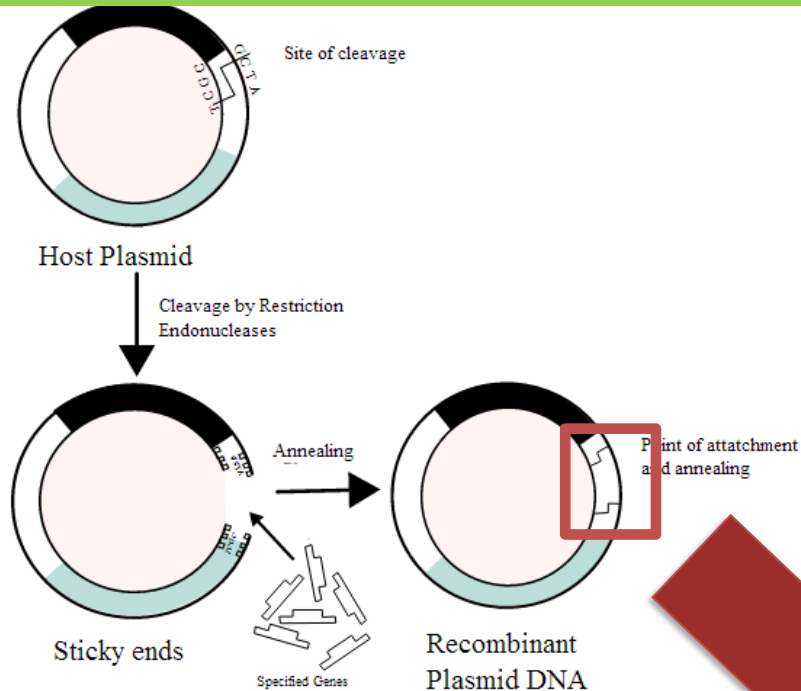
THEN ALONG CAME *my wife*



- **Biochemist**
- Works doing protein... something
- I suspect she really works doing...



BIOTECH WILL SAVE THE WORLD OR KILL US ALL



At the ribosome, the RNA's message is translated into a specific protein.





In a nutshell...





Functions window

Function name

MapHDC(int)
MapHGDI OBJ(int)
MapHIMAGELIST(int)
MapHMENU(int)
MapHWND(int)
_ArithmeticSender01
_callSender02
_callSender03
_callSender04
_callSender05
_sendCaller01
_sendCaller02
_sendCaller03
getSystemCP(int)
j_\$LN12_12
j_\$LN13_27

Line 2370 of 8806

Graph overview

IDA View-A

Hex View-A

Structures

```
fild [esp+110h+len]
mov [esi+4Ch], edi
fstcw [esp+110h+var_FE]
fmul ds:dbl_5B32E8
movzx eax, [esp+110h+var_FE]
or eax, 0C00h
mov dword ptr [esp+110h+var_FC], eax
fadd qword ptr [esi+50h]
fst qword ptr [esi+50h]
fldcw word ptr [esp+110h+var_FC]
fistp [esp+110h+var_FC]
mov ecx, dword ptr [esp+110h+var_FC]
fldcw [esp+110h+var_FE]
fldz
cmp ecx, 0FFFFFFFh
jnb short float_arithmetic
```

```
fst qword ptr [esi+50h]
```

```
float_arithmetic:
fld qword ptr [esi+58h]
fstcw [esp+110h+var_FE]
movzx eax, [esp+110h+var_FE]
or eax, 0C00h
mov dword ptr [esp+110h+var_FC], eax
fldcw word ptr [esp+110h+var_FC]
fistp [esp+110h+var_FC]
mov edx, dword ptr [esp+110h+var_FC]
fldcw [esp+110h+var_FE]
cmp edx, 0FFFFFFFh
jnb short loc_40DB9A
```

```
fistp qword ptr [esi+58h]
jmp short loc_40DB9C
```

```
loc_40DB9A:
fstp st
```

100.00% | {240,901} | {212,592} | 0000CFBA | 0040DBBA: cgp ArithmeticSender01:loc 40DBBA

Output window

MILF initialized

cgp_ArithmeticSender01

cgp_sendCaller01

sub_40DAA0

SendMess

send

__time64

SIMPLE ARGUMENTS

```
; int __stdcall cgp_ArithmeticSender01(char *buf, int len, int)
cgp_ArithmeticSender01 proc near

var_FE= word ptr -0FEh
var_FC= qword ptr -0FCh
Dest= dword ptr -0F4h
var_4= dword ptr -4
buf= dword ptr 4
len= dword ptr 8
arg_8= dword ptr 0Ch

sub     esp, 100h
mov     eax, stackCookie
xor     eax, esp
mov     [esp+100h+var_4], eax
push    ebp
mov     ebp, [esp+104h+buf]
```



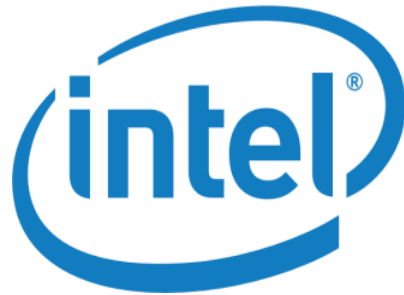
A still from the TV show 'The Big Bang Theory' featuring Sheldon Cooper and Leonard Hofstadter. They are in a bar, looking up with expressions of intense excitement. Sheldon, on the left, is wearing a dark suit and tie. Leonard, on the right, is wearing a grey sweatshirt with 'WESLEYAN' printed on it and has his mouth wide open in a shout. The background is dark with some colorful bar lights.

IT GETS EXCITING

Detours

= userland hooking

= amazing stuff



= dynamic binary
instrumentation

= AWESOME stuff !!!

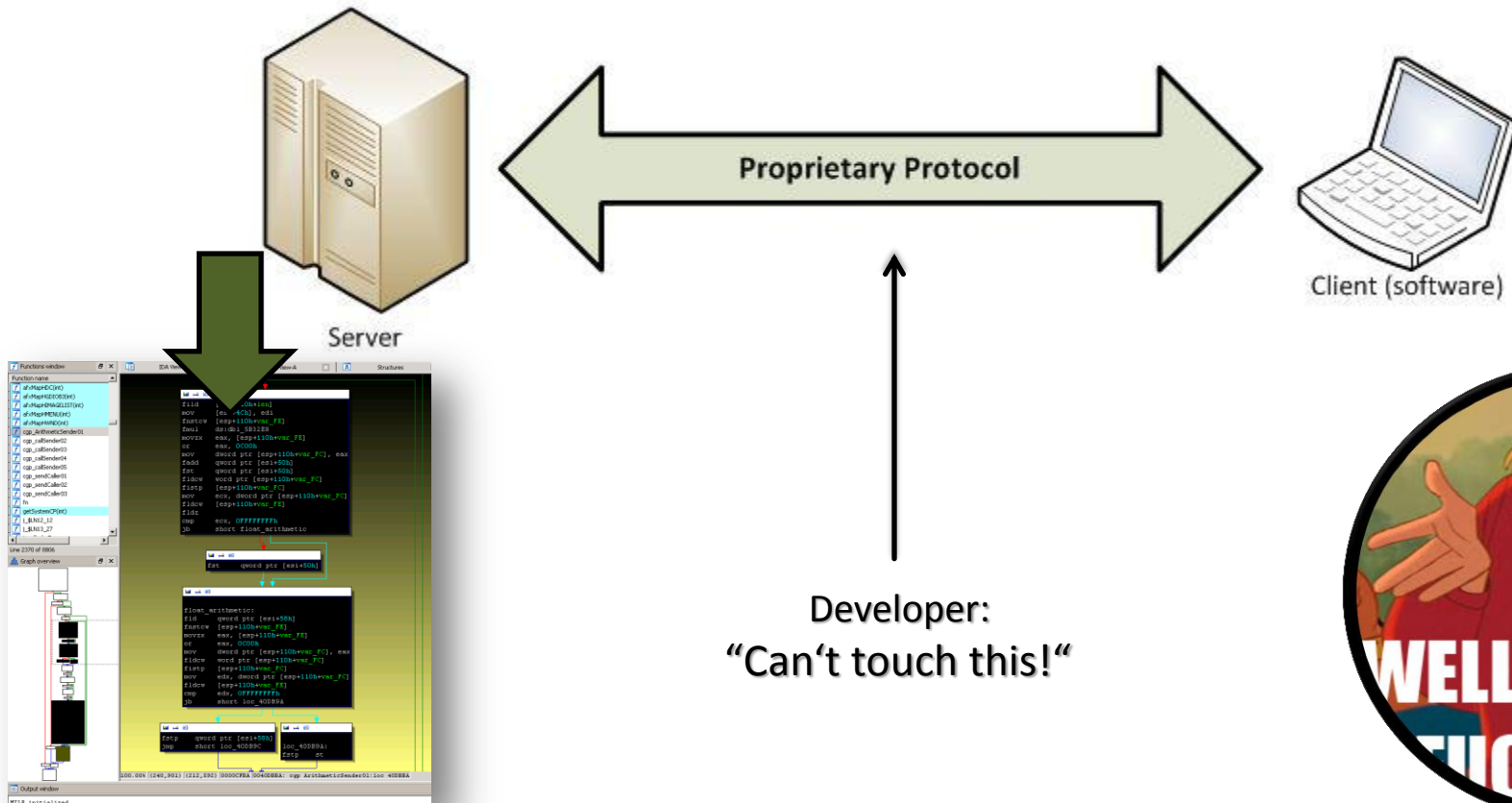


MICROSOFT DETOURS

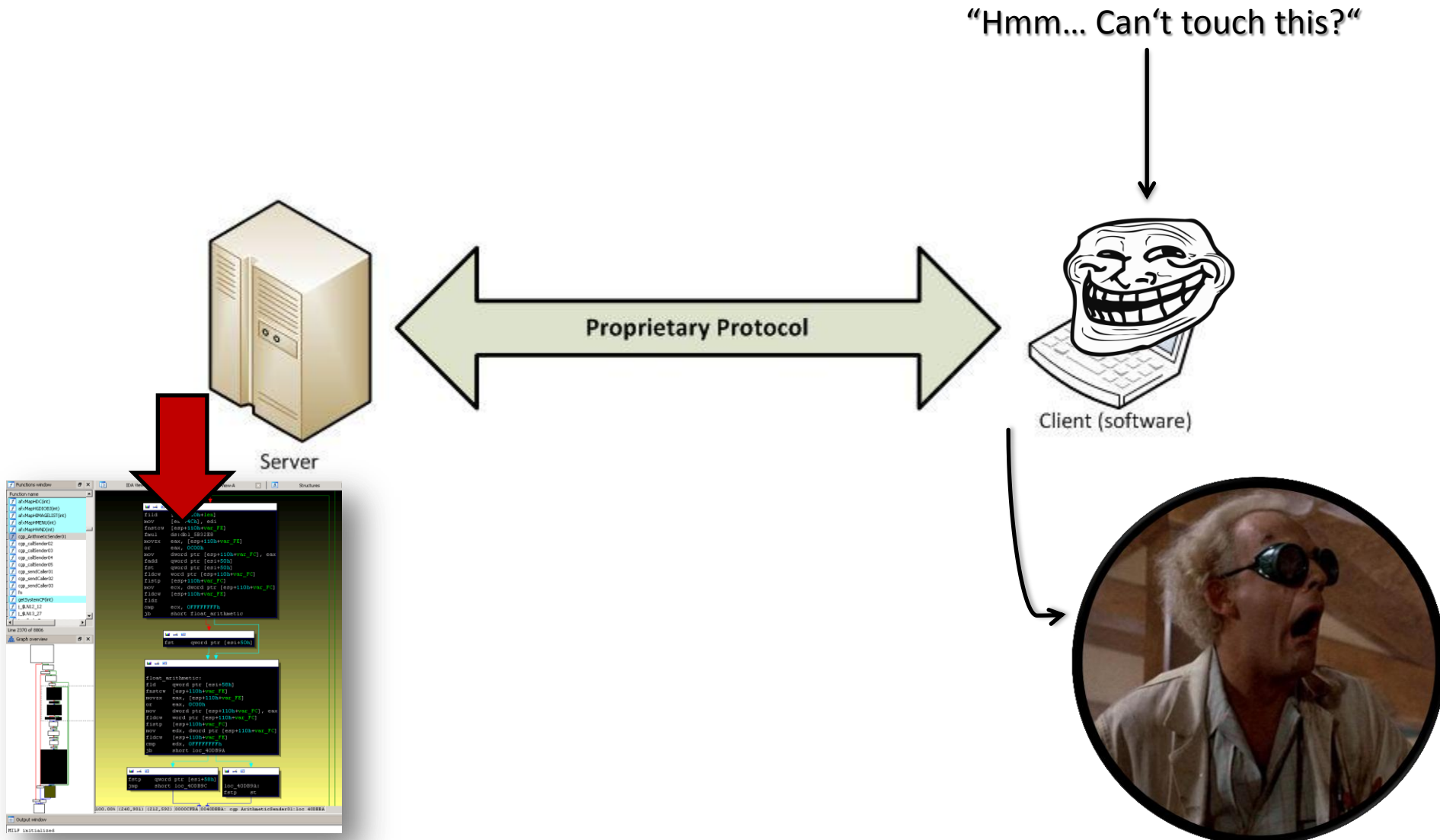
- **Library** for intercepting arbitrary Win32 binary functions.
- Interception code is applied dynamically **at runtime**.
- **Replaces the first few instructions** of the target function
- with an **unconditional jump** to the detour function.
- **Replace or extend** the target function.



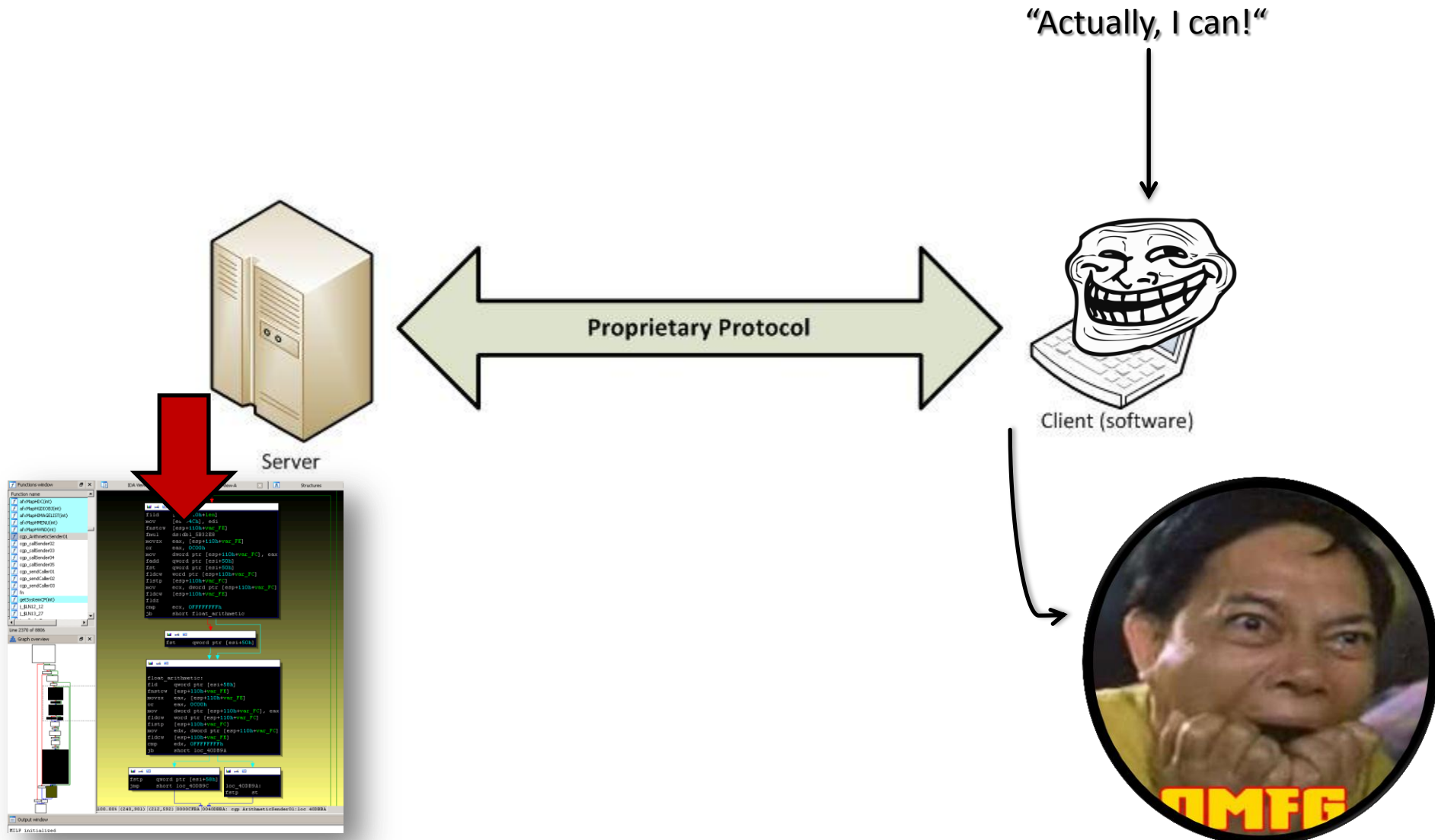
WHAT CAN POSSIBLY GO WRONG?



WHAT CAN POSSIBLY GO WRONG?



WHAT CAN POSSIBLY *GO WRONG*?



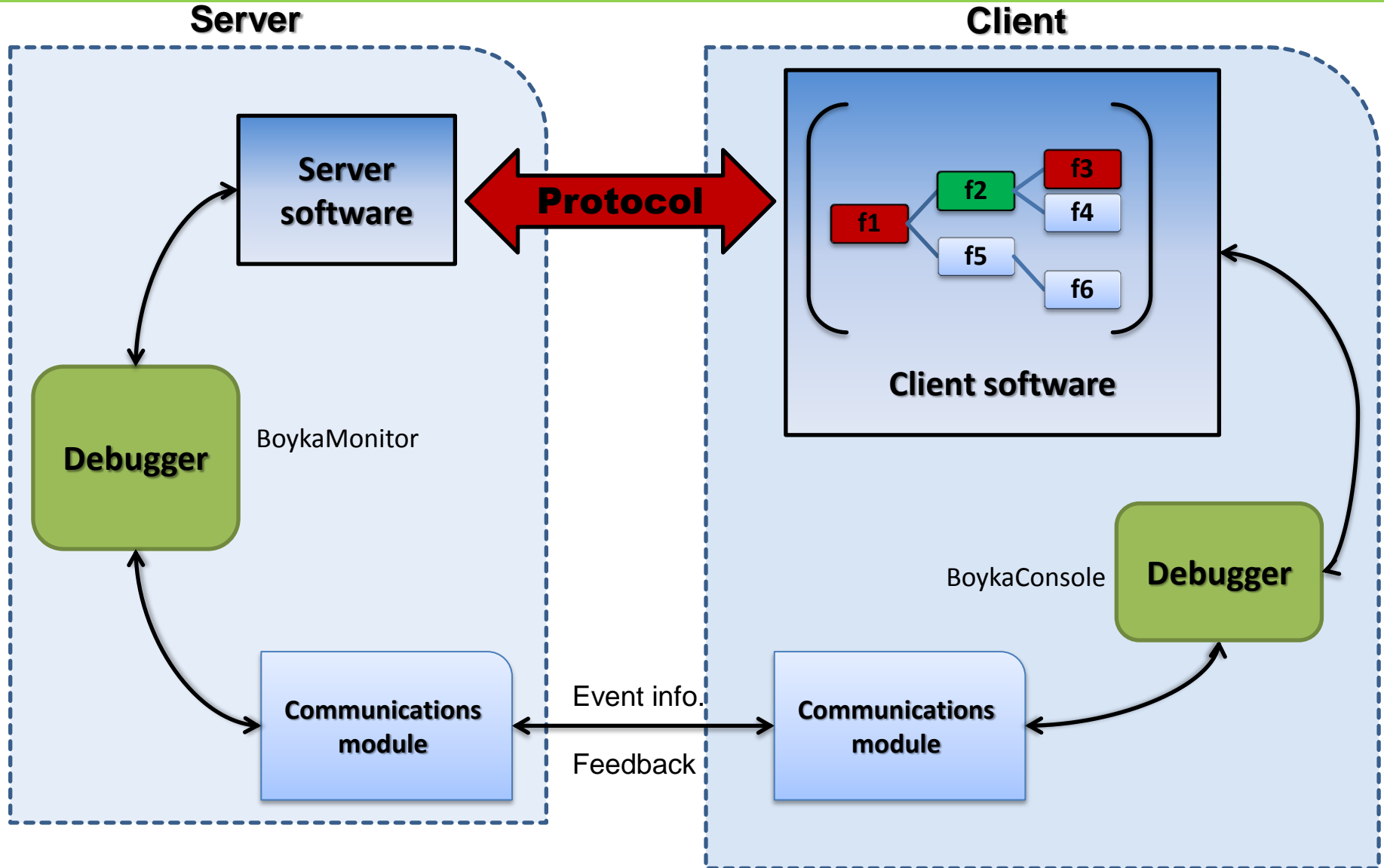
LONG STORY SHORT...



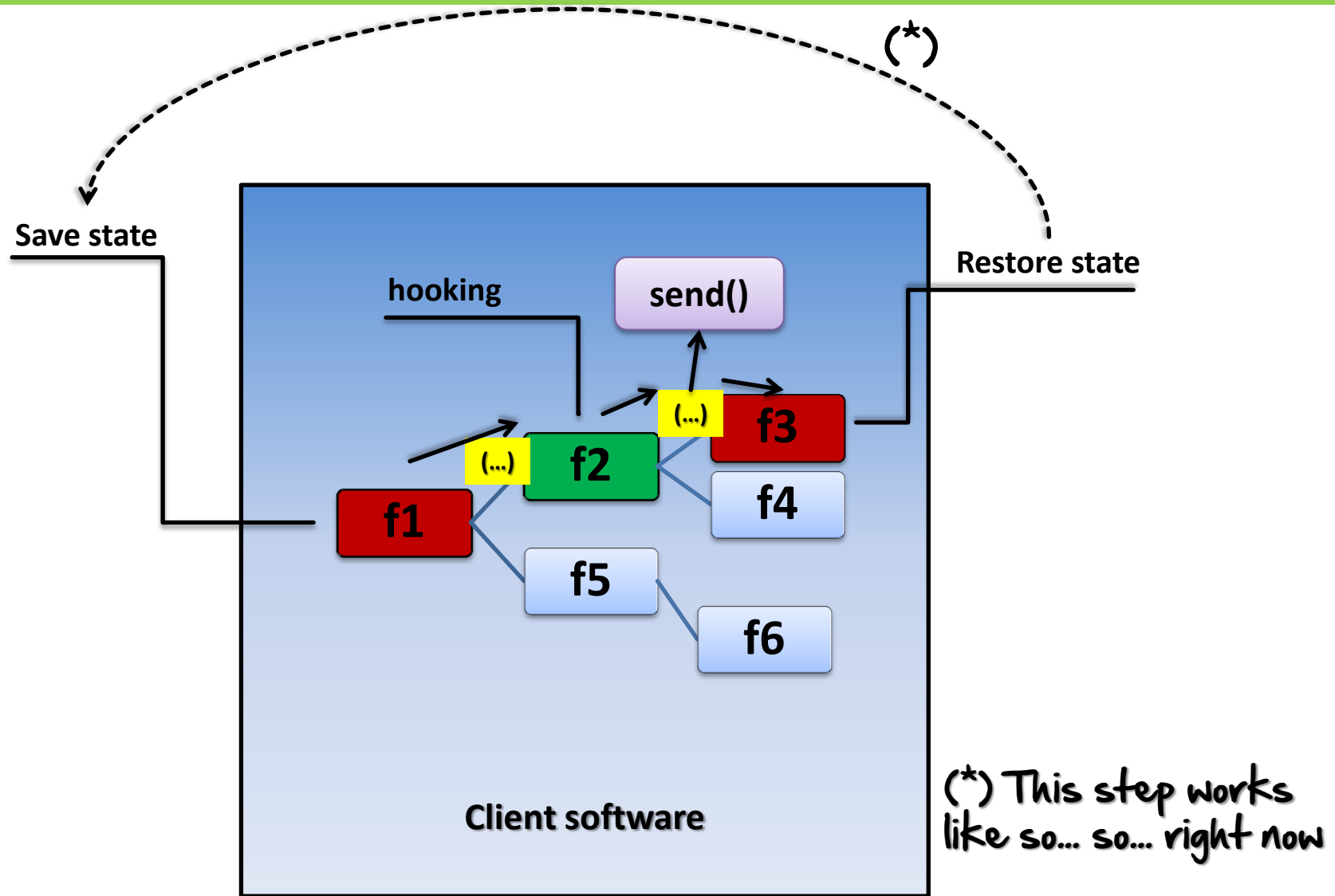


PLUMBING TIME

OVERVIEW (FROM A MILLION MILES AWAY)



OVERVIEW (FROM A THOUSAND MILES AWAY)



NOT SURE IF THIS MAKES SENSE



**OR I'M STARTING TO BELIEVE MY OWN
BULLSHIT**



THE CHALLENGE

- I can "inject" some data into the server
 - By hijacking client execution at certain points
 - ...
 - ... aha...
-
- **Which. Points. Do. I. Use. ?!?!?!?**

Carlos,
help me!

There are
TOO MANY
Addresses
!!!!!!!





Anyone getting dizzy?

**BRACE YOURSELVES,
SHAMELESS
AUTOPROMOTION IS
COMING.**



M*LF & PIN TRACER

beta





SOME COOL FEATURES

- Mark dangerous functions
- Find immediate compares
- Mark switches
- Show paths between functions
- Find File IO
- Find Network IO
- Find Allocations
- Find dangerous „size params“
- Create IDA (connection) graphs
- Create „custom viewers“
- etc.

Output window

```
Python>net_dict = ia.locate_net_io()
Python>for caller, imp_list in net_dict.iteritems():
Python>    print GetFunctionName(caller)
Python>    for l in imp_list:
Python>        print "    -", Name(l)
Python>
sub_43E300
- WSASend
- WSAGetLastError
sub_43DE81
- WSASetLastError
sub_43DF03
- WSASetLastError
sub_469916
- recvfrom
sub_43FADB
- WSAWaitForMultipleObjects
sub_42650D
- WSAGetLastError
sub_469A0E
- WSASendTo
sub_43FB0F
- WSAGetLastError
- WSAEnumNetworkCards
sub_427210
- WinMain@16
sub_433309
- WSAGetLastError
sub_469003
- WSAGetLastError
sub_443614
- WSAWaitForMultipleObjects
```

Output window

```
Python>imm = ia.imm_compares(gc)
Python>for k,v in imm.iteritems():
Python>    print hex(k), "cmp %s, %s" % (v[0], v[1])
Python>
0x43bb81 cmp dword ptr [eax], 0
0x45a621 cmp [ebp+var_1], 0
0x46cac4 cmp eax, 2746h
0x43baa7 cmp dword_518184, 0
0x43b140 cmp bl, 2
0x43b140 cmp [ebp+var_8], 0FFFFFFFh
0x43b140 cmp byte ptr [ebx], 3Ah
0x43b140 cmp esi, 0FFFFFFFh
0x43b140 cmp eax, 2738h
0x43b140 cmp word ptr [eax+8], 2
0x43b140 cmp byte ptr [esi+0Ch], 0
0x43b140 cmp cl, 2
0x43b140 cmp [ebp+var_1], 0
0x43b140 cmp word ptr [eax+0Ah], 4
0x43b140 cmp byte ptr [edi], 20h
0x43b140 cmp [ebp+var_1], 0
0x43b140 cmp [ebp+var_1], 0
0x43b140 cmp edi, 0FFFFFFFh
0x43b140 cmp byte ptr [edi], 5Bh
```

IDA - S:\Possible_Victims\utorrent\utorrent.idb (utorrent.exe)

File Edit Jump Search View Debugger Options Windows Help

Hot spots (Allocs)

Hex View-A

IDA View-A

Output window

```
Python>ia.locate_allocs(interactive = True)
{4597632: [5034408], 4735362: [5034764], 4903
[5035368], 4359315: [5034408], 4649519: [5034
[5035348, 5035368], 4442917: [5035364, 503642
[5035364, 5035368], 4756775: [5034404], 44077
4710700: [5035424], 4413869: [5035364, 503536
4365488: [5034536], 4407486: [5035368], 44109
[5035348, 5035368], 4351685: [5034408], 43095
[5034892], 4783696: [5035364, 5035368], 47831
4369490: [5034072, 5034076], 4735315: [503476
4430299: [5035368], 4309725: [5034436, 503443
4433505: [5035368], 4437604: [5034408], 44100
4835052: [5034892], 4836845: [5034740, 503491
5034736], 4784113: [5035364, 5035368], 444030
[5035368], 4355710: [5034408], 4426879: [5036
```


IDASCOPE

ArithLog Rating: ☒ Exclude Zeroing

Basic Blocks size: ☒ Looped Blocks only

Allowed calls: ☐ Group by Functions

28 blocks from a total of 13292 blocks matched with the above settings.

	Address ▾	Name	Block Address	# Instr	Arithmetic/Logic Rating	▴
13	0x433c80	CRC32	0x433f20	8	75.00	
14	0x433c80	CRC32	0x433cc0	116	71.55	
15	0x433c80	CRC32	0x433ed0	19	68.42	
16	0x428791	XorChainEncrypt	0x4287a5	6	50.00	
17	0x4286bb	Base64Decode	0x428767	9	33.33	
18	0x428520	Rc4	0x42854b	19	36.84	
19	0x427b37	DecryptString	0x427b4d	12	33.33	
20	0x427b01	StringEncrypt_...	0x427b16	9	33.33	
21	0x42633c	MersenneTwister	0x4263a2	14	57.14	
22	0x42633c	MersenneTwister	0x426355	14	57.14	
23	0x426307	MersenneTwist...	0x426315	11	54.55	



<http://pnx-tf.blogspot.com/>

DIFFERENTIAL DEBUGGING

- Hook every function -> log hits.
- 1st run. Exercise as many functionality as possible.
- 2nd run. Focus on the interesting feature.
- Compare both -> filter out.

Function_1
GUI_stuff
Windows_stuff
Function_2
Thread_sync
Function_3
[...]

Function_1
GUI_stuff
Windows_stuff
Login_stuff
Thread_sync
Encryption_stuff
[...]



DIFFERENTIAL DEBUGGING

- Hook every function -> log hits.
- 1st run. Exercise as many functionality as possible.
- 2nd run. Focus on the interesting feature.
- Compare both -> filter out.

Function_1
GUI_stuff
Windows_stuff
Function_2
Thread_sync
Function_3
[...]

Function_1
GUI_stuff
Windows_stuff
Login_stuff
Thread_sync
Encryption_stuff
[...]

Login_stuff
Encryption_stuff
[...]



A man with a beard and safety glasses is welding a large, complex metal structure in a workshop. He is wearing a green long-sleeved shirt and is focused on his work. Sparks are flying from the welding point. The structure appears to be a large, multi-chambered metal component, possibly a part of a weapon or industrial machinery. The workshop has a wooden floor and various tools and equipment are visible in the background.

BUILD YOUR WEAPON

EPIC ASS KICKING



WAKE UP!

You're gonna miss
the good stuff!!!

FINDING POSSIBLE WEAK SPOTS

IDA - S:\Soft

File Edit Jump

Functions window

Function name

- f j__abort
- f j__assemblePacket
- f j__atexit
- f j__atoi

Awesome Client

Login

Type your password here

Login

IDA View-A

```
call ds:__imp__GlobalAlloc@8 ; GlobalAlloc(x,x)
cmp esi, esp
call j__RTC_CheckEsp
mov [ebp+login], eax
mov eax, [ebp+len]
add eax, 1
mov esi, esp
push eax ; cchMax
mov ecx, [ebp+login]
push ecx ; lpString
push 3E8h ; nIDDlgItem
mov edx, [ebp+hwnd]
push edx ; hDlg
call ds:__imp__GetDlgItemTextA@16 ; GetDlgItemTextA(x,x,x,x)
cmp esi, esp
call j__RTC_CheckEsp
mov ecx, [ebp+len]
push eax ; len
mov ecx, [ebp+szPacket]
push ecx ; szPacket
mov edx, [ebp+login]
push edx ; login
call j__assemblePacket
add esp, 0Ch
mov [ebp+szEncPacket], eax
mov eax, [ebp+szEncPacket]
push eax ; encryptedString
call j__sendTheShit
add esp, 4
mov esi, esp
mov eax, [ebp+login]
push eax ; hMem
call ds:__imp__GlobalFree@4 ; GlobalFree(x)
cmp esi, esp
call j__RTC_CheckEsp
mov eax, [ebp+szPacket]
push eax ; pUserData
call j__free
```

Diagram illustrating a control flow graph (CFG) with nodes and edges, likely representing the execution flow of the assembly code shown.

FINDING POSSIBLE WEAK SPOTS

IDA - S:\Software\c0c

File Edit Jump Search View Debugger Options Windows Help

Functions window

Function name

- f j_abort
- f j_assemblePacket
- f j_atexit
- f j_atoi
- f j_atol
- f j_encryptLogin
- f j_exit
- f j_fclose
- f j_fflush
- f j_fputc
- f j_free
- f j_get_crtdouble_arg
- f j_get_int64_arg
- f j_get_int_arg
- f j_get_short_arg
- f j_is_wctype
- f j_isleadbyte
- f j_iswalnum
- f j_iswalph
- f j_iswascii
- f j_iswcntrl

Line 1656 of 1906

Graph overview

IDA View-A

Hex View-A

Structures

Enums

```
; Attributes: bp-based frame

; char *__cdecl assemblePacket(char *login, char *szPacket, unsigned int len)
__assemblePacket proc near

var_F8= byte ptr -0F8h
szSecretString= byte ptr -34h
szPktLen= byte ptr -20h
delimiter= dword ptr -0Ch
var_4= dword ptr -4
login= dword ptr 8
szPacket= dword ptr 0Ch
len= dword ptr 10h

push    ebp
mov     ebp, esp
sub     esp, 0F8h
push    ebx
push    esi
push    edi
lea     edi, [ebp+var_F8]
mov     ecx, 3Eh
mov     eax, 0CCCCCCCCh
rep stosd
mov     eax, __security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
mov     [ebp+delimiter], offset a:" "
mov     eax, dword ptr ds:a
mov     dword ptr [ebp+szSe
mov     ecx, dword ptr ds:a
mov     dword ptr [ebp+szSe
mov     edx, dword ptr ds:a
mov     dword ptr [ebp+szSecretString+8], edx
lea     eax, [ebp+szSecretString]
push    eax
```

Keep walking.
Nothing to see here...

CHEATING...

Client:

- Calculates login length
- Appends the length (in ASCII) to the login string.
- Appends a “custom” string
- *Encrypts* everything

Server: Length value (**from client**)
used to ***malloc()*** & ***strcpy()***



STAND BACK



I'M GOING TO TRY
SCIENCE



WHERE TO GO FROM HERE

- Better static / dynamic analysis
 - Automatization
 - Heuristic based
- Save / restore snapshot
 - Full emulation (Thx @pleed_ !)
 - Qemu-dbi?

EVERYTHING IS ONLINE

You can **lulz** at my code at:

<https://github.com/carlosgprado/Boyka>

@m0n0sapiens

carlos.g.prado@gmail.com

**IT'S
BEEN
LOVELY
BUT
I HAVE TO
SCREAM
NOW**

