Video game hacking:

Cheat Engine Workshop

Lucas PARSY

"It's pronounced Aine jean"

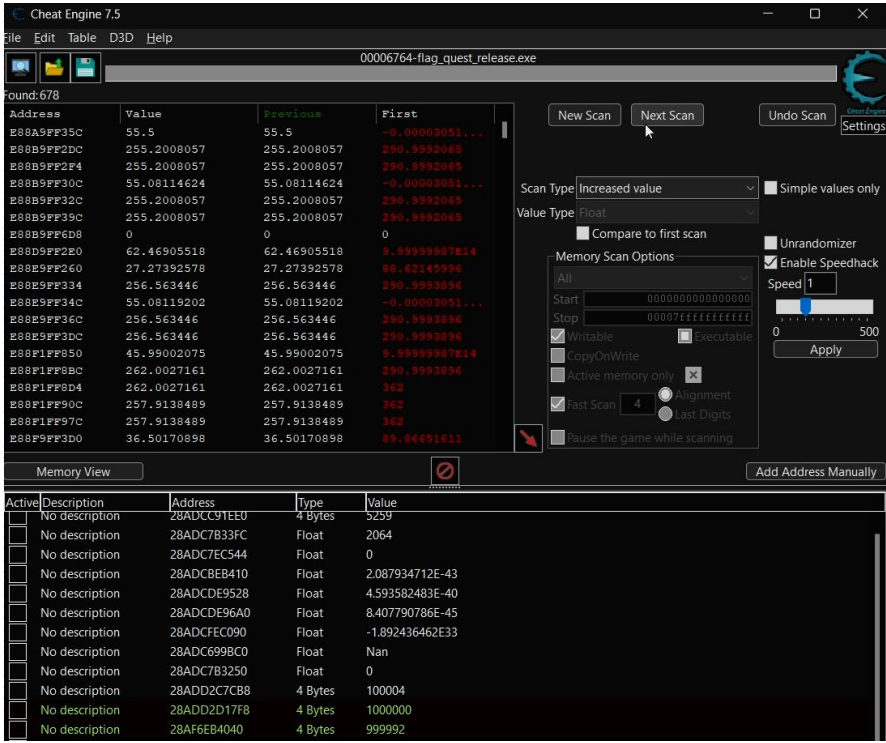
Resources

- Cheat Engine: cheatengine.org
- My game, Flag Quest:
  See QR code
  Game, writeup &more
- USB key with everything!

tuxlu.fr/talk_vghacking

Cheat Engine

- Memory scanner and debugger



Cheat Engine

- Memory scanner and debugger
- **AutoAssembly and LUA scripting**

```
Hook :
retGetGamePlayers_o:
readmem( retGetGamePlayers, 6 )
mov [LocalPlayer],rax
mov rcx, [rax+30]
test rcx,rcx
je short @f
    mov [OakPlayerController],rcx
    mov rcx, [rcx+488]
    test rcx, rcx
        je short @f
        mov rcx, [rax+30]
        mov rcx, [rcx+1988]
        test rcx,rcx
        je short @f
            mov [OakDeveloperPerks],rcx
            test byte ptr [rcx+C8],40
            jne short @f
                or byte ptr [rcx+C8],40

@@:
jmp retGetGamePlayers+6
```

Cheat Engine

- Memory scanner and debugger
- **AutoAssembly and LUA scripting**

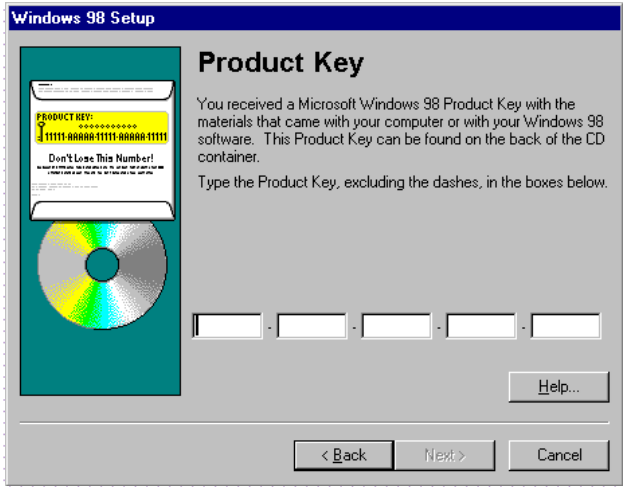
```
function AOBScanAA(script, symbol)
  local success,disableInfo = autoAssemble(script)
  if not success then return nil, disableInfo end -- disab
  local addr = getAddress(symbol)
  autoAssemble(script, disableInfo) -- disable script and
  return addr, 'success'
end

function AOBScanRegion(bytestr, start, stop)
  local script = ([[
  [ENABLE]
  aobscanregion(luaAOBScanRegionSymbol,%X,%X,%s)
  registersymbol(luaAOBScanRegionSymbol)
  [DISABLE]
  unregistersymbol(luaAOBScanRegionSymbol)
  ]]):format(getAddress(start), getAddress(stop), bytestr)
  return AOBScanAA(script, 'luaAOBScanRegionSymbol')
end

function AOBScanModule(bytestr, module)
  local script = ([[
  [ENABLE]
  aobscanmodule(luaAOBScanModuleSymbol,%s,%s)
  registersymbol(luaAOBScanModuleSymbol)
  [DISABLE]
  unregistersymbol(luaAOBScanModuleSymbol)
  ]]):format(module, bytestr)
  return AOBScanAA(script,'luaAOBScanModuleSymbol')
end
```

Cheat Engine

- Memory scanner and debugger
- AutoAssembly and LUA scripting
- **GUI ‘trainer’ generator**

Cheat Engine

- Memory scanner and debugger
- AutoAssembly and LUA scripting
- GUI ‘trainer’ generator
- **Not limited to video games**



Finding values

COINS : 0

New Scan | Next Scan

Value:

Hex 0

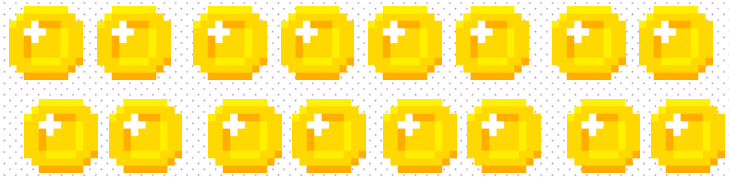
Scan Type Exact Value

Found: 153,518,055 (shown: 10,000,000)

Address	Value	Previous
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0
MSVCP...	0	0

Finding values

COINS : 4



New Scan | Next Scan

Value: 4

Hex

Scan Type: Exact Value

Found: 113

Address	Value	Previous
1318B...	0	4
1318B...	0	4
13193...	4	4
13193...	4	4
13193...	4	4
13193...	4	4
13193...	4	4
15C0A...	4	4

Finding values

COINS : 20

New Scan | Next Scan

Value: 20

Hex

Found:4

Address	Value	Previous
15C0D...	20	20
15C0D...	20	20
15C0D...	20	20
15C0E...	20	20

Finding values

COINS : 4000

Activ	Description	Address	Type	Value
☐	coins	00000000		
☐	coins	15C0DD859784	4 Bytes	20
☐	coins	15C0DD859AC4	4 Bytes	20
☐	coins	15C0EC83FD84	4 Bytes	20
☐	coins	15C0DD859C84	4 Bytes	20

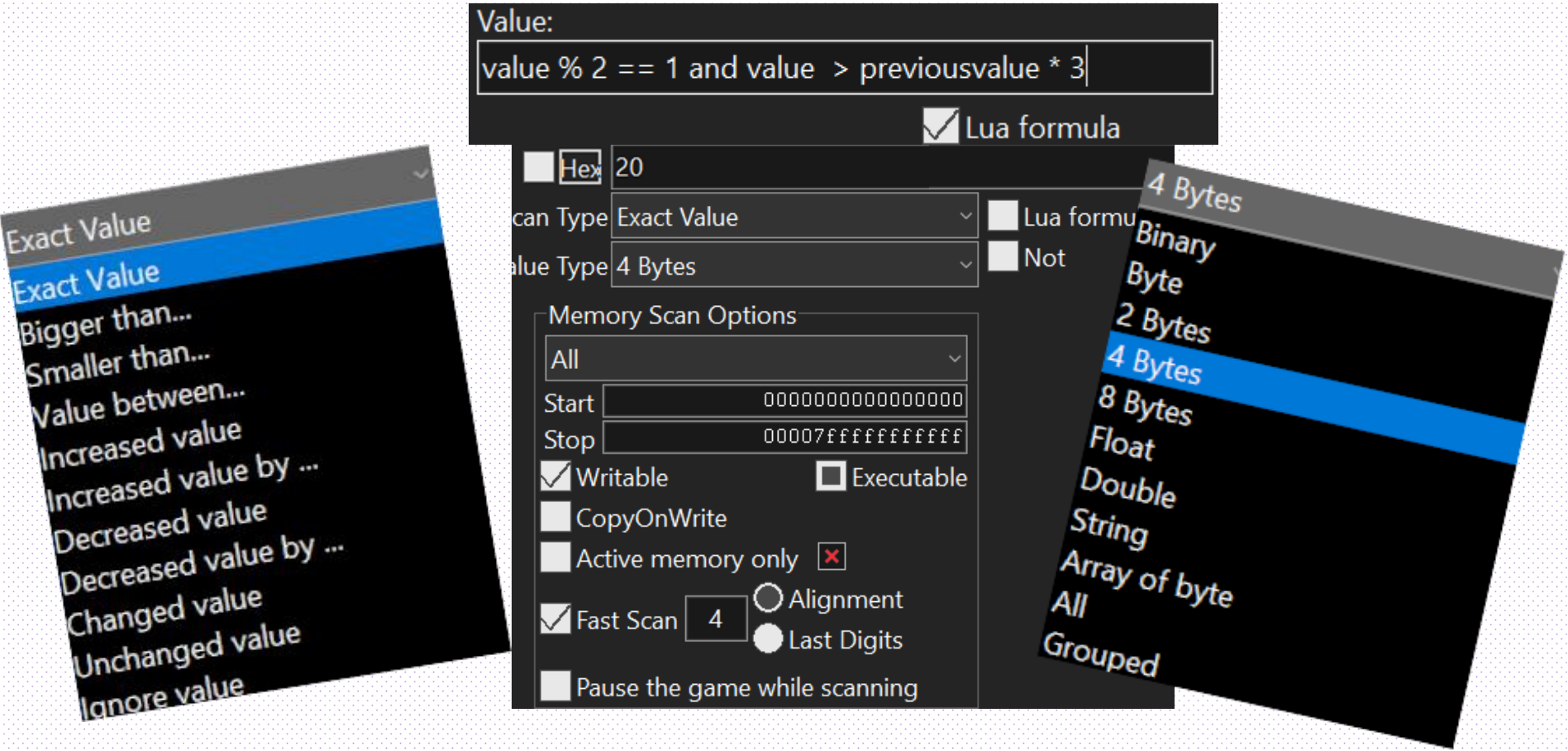
Change Value

what value to change this to?

4000

OK

Finding values: advanced features



Memory viewer

Description	Address	Type	Value
health	15C0DD85978	4 Bytes	1337

Memory View

```
Protect:Read/Write  AllocationBase=15C0DD40000 Base=
address     80 81 82 83 84 85 86 87 89ABCDEF01234567
15C0DD85978 39 05 00 00 64 00 00 00 9...d...  ...\...
15C0DD85988 C8 00 00 00 16 00 00 00  ........  .... ..
15C0DD85998 80 51 69 29 5C 01 00 00  Qi)\....  ........
15C0DD859A8 B0 6A 74 0E 5C 01 00 00  jt.\....  ........
15C0DD859B8 C6 17 00 00 00 20 00 00  .... ..  Qm)\...
15C0DD859C8 13 37 00 00 00 01 00 00 .7......  B..\...
```

```
struct Player
{
    int health   = 1337;
    int ???      = ???;
    int ???      = ???;
}
```

Memory viewer

Description	Address	Type	Value
health	15C0DD85978	4 Bytes	1337

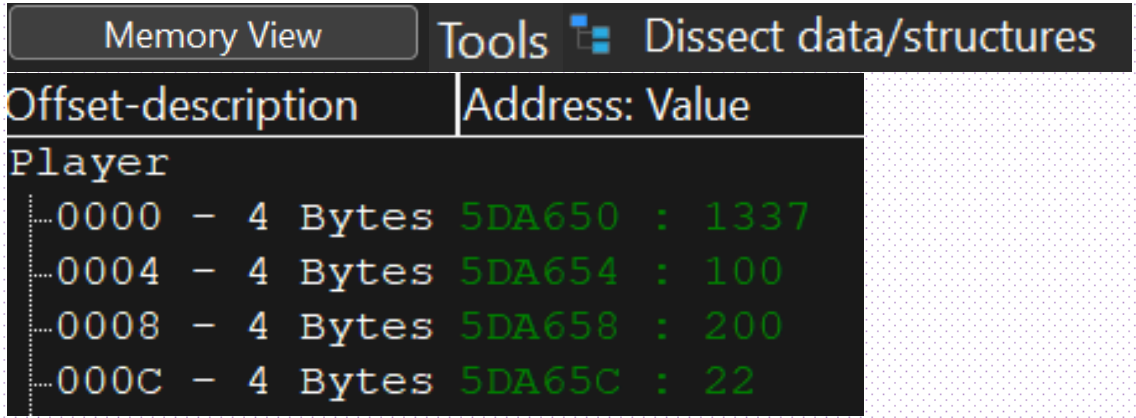
Memory View | Display Type › • 4 Byte decimal

```
Protect:Read/Write   AllocationBase=15C0DD40000 Base=
address      78          7C          89ABCDEF01234567
15C0DD85978 1337         100         9...d...  ...\...
15C0DD85988 200          22           .......  .... ..
15C0DD85998 694768000    348          Qi)\............
15C0DD859A8 242510512    348          jt.\...  .......
15C0DD859B8 6086         8192          .... ..  Qm)\...
15C0DD859C8 14099        256         .7......  B..\...
```

```
struct Player
{
    int health   = 1337;
    int strength = 100;
    int defense  = 200;
}
```

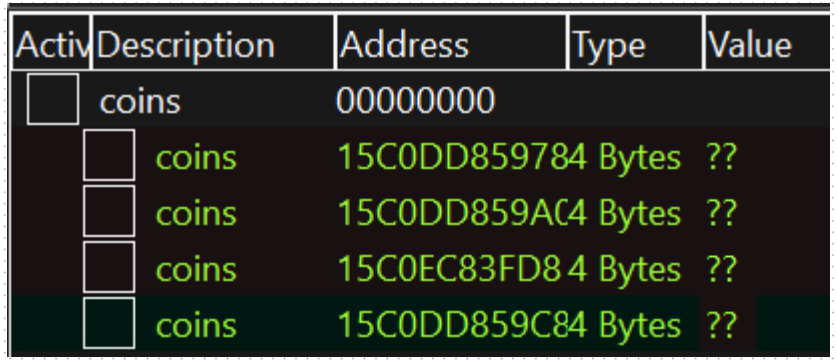
Data structures

Description	Address	Type	Value
health	15C0DD85978	4 Bytes	1337




```
struct Player
{
    int health   = 1337;
    int strength = 100;
    int defense  = 200;
}
```

Persisting memory addresses



Activ	Description	Address	Type	Value
☐	coins	00000000		
☐	coins	15C0DD859784	Bytes	??
☐	coins	15C0DD859AC4	Bytes	??
☐	coins	15C0EC83FD84	Bytes	??
☐	coins	15C0DD859C84	Bytes	??

Reload game
Lose everything!

Persisting memory addresses

Solution: search *potentially* static addresses
search for all code that points to the address

coins 2216AF30F68 4 Bytes 58

Pointer scan for this address

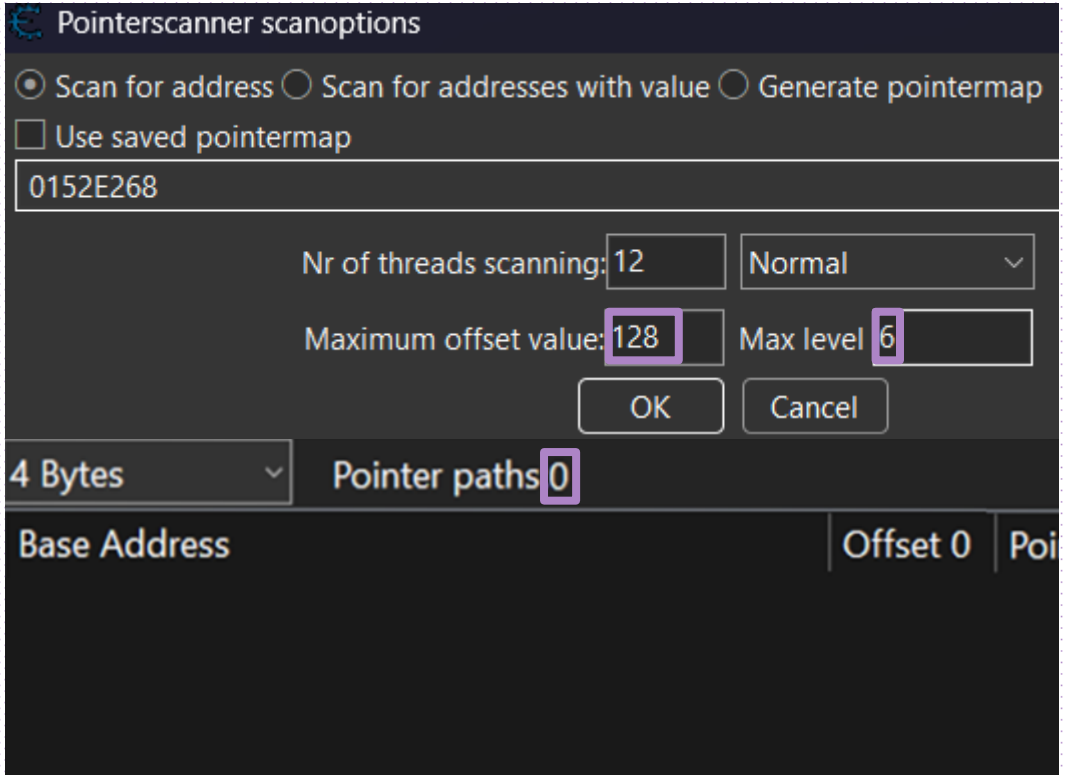
Recursive scan

```
op [(Address - 1) + 01]
op [(Address - 2) + 02]
op [(Address - 3) + 03]
…
```

```
7FF7BB2E2724 - 48 03 41 08  - add rax,[rcx+08]      RCX=000002216AF30F60
7FF7BB165E68 - 49 89 44 24 08  - mov [r12+08],rax   R12=000002216AF30F60
7FF7BB1B9D40 - 49 8B 55 08  - mov rdx,[r13+08]      R13=000002216AF30F60
```

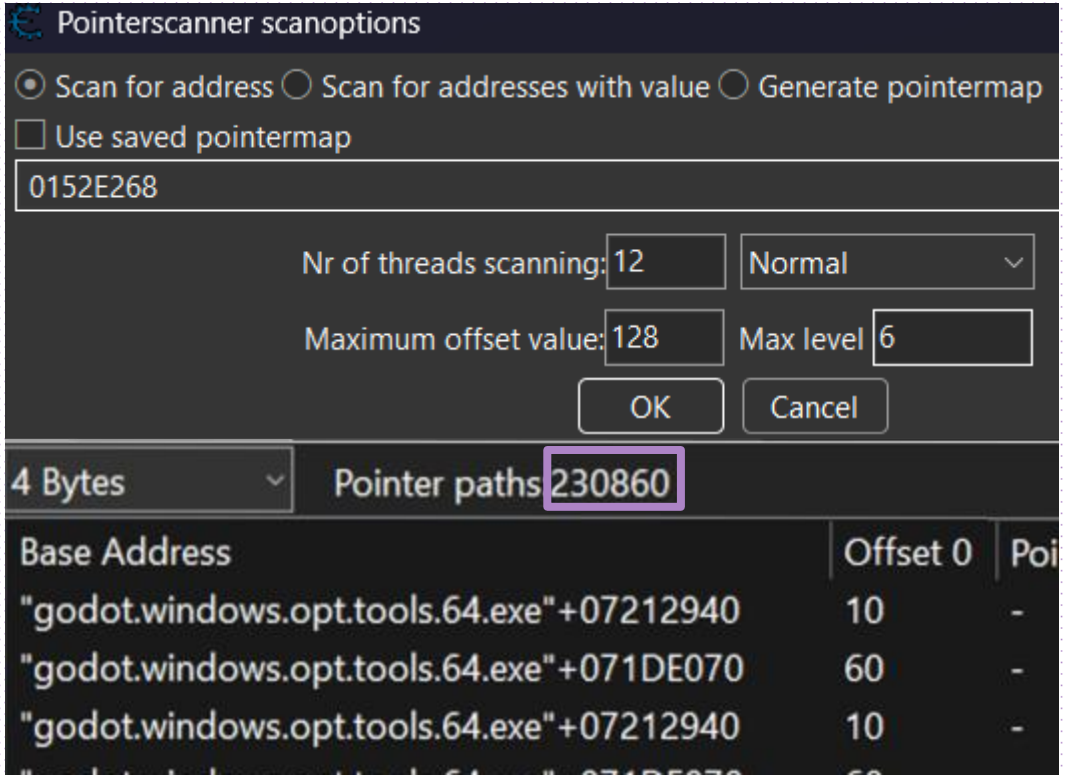
Persisting memory addresses

Problem 1: no results :/



Persisting memory addresses

Problem 2: too many results!



Persisting memory addresses

solution: rescan, and compare results

Generate pointermap → Pointer scan for this address

Filename	Address
pointermap_coins5.scandata	248C053F6E8
pointermap_coins3.scandata	1D847EDC898
pointermap_coins1.scandata	247DF1D46D8

4 Bytes | Pointer paths 1

Base Address	Offset 0	Points to:
"godot.windows.opt.tools.64.exe"+0717F820	3B8	2216AF30F68 = 206

coins	2216AF30F68	4 Bytes 74
pointerscan result	P->2216AF30F68	4 Bytes 74

What if we don't search a value?

How to search for a condition?

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

What if we don't search a value?

How to search for a condition?:
code filter

Memory View | Tools | Code Filter

Addresses executed since last filter operation:0

Has been executed | Has not been executed | Start | Stop

Load address list: From Trace | From Disassembler | From File

Address List (46093)

Address	Executed
Tutorial-i386.exe.text+1BB5	No
Tutorial-i386.exe.text+1BBA	No
Tutorial-i386.exe.text+1BBF	No
Tutorial-i386.exe.text+1BC8	No

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

What if we don't search a value?

How to search for a condition?:
code filter

Addresses executed since last filter operation:1791

Has been executed

Has not been executed

Start | Stop

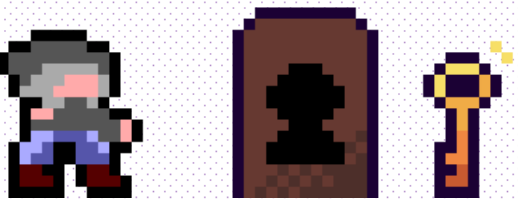
Address List (44302)

Address	Executed
Tutorial-i386.exe.text+1BB5	No
Tutorial-i386.exe.text+1BBA	No
Tutorial-i386.exe.text+1BBF	No
Tutorial-i386.exe.text+1BC8	Yes

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

What if we don't search a value?

How to search for a condition?:
code filter



Addresses executed since last filter operation: 595

Has been executed

Has not been executed

Start | Stop

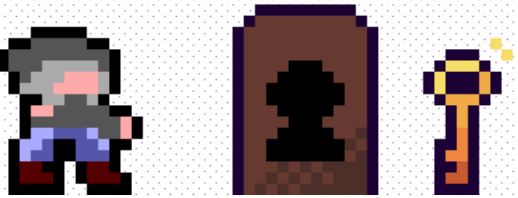
Address List (595)

Address	Executed
Tutorial-i386.exe.text+1BB5	No
Tutorial-i386.exe.text+1BBA	No
Tutorial-i386.exe.text+1BBF	No
Tutorial-i386.exe.text+1BC8	Yes

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

What if we don't search a value?

How to search for a condition?:
code filter



Addresses executed since last filter operation: 1

Has been executed

Has not been executed

Start | Stop

Address List (1)

Address	Executed
Tutorial-i386.exe.text+1BB5	Yes

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

Instruction patching

ASM, help!

```
74 02          je    Tutorial-i386.exe.text+26687
EB 49          jmp   Tutorial-i386.exe.text+266D0
A1 B0666500    mov   eax,[Tutorial-i386.exe+2566B0]
3B 45 E8       cmp   eax,[ebp-18]
74 02          je    Tutorial-i386.exe.text+26693
EB 1F          jmp   Tutorial-i386.exe.text+266B2
C7 45 E8 000...mov   [ebp-18],00000000
6A 00          push  00
```

Instruction patching

ASM primer

je if ==	**jg** if >	**add** +=
jne if !==	**jl** if <	**sub** -=

mov x=y

nop do nothing

(padding)

Instruction patching

Replace the *has_key* condition

```
74 02          je      Tutorial-i386.exe.text+26687
EB 49          jmp     Tutorial-i386.exe.text+266D0
A1 B0666500    mov     eax,[Tutorial-i386.exe+2566B0]
3B 45 E8       cmp     eax,[ebp-18]
74 02          je      Tutorial-i386.exe.text+26693
EB 1F          jmp     Tutorial-i386.exe.text+266B2
C7 45 E8 000...mov     [ebp-18],00000000
6A 00          push    00
```

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

Instruction patching

Replace the *has_key* condition

```
74 02          je      Tutorial-i386.exe.text+26687
EB 49          jmp     Tutorial-i386.exe.text+266D0
A1 B0666500    mov     eax,[Tutorial-i386.exe+2566B0]
3B 45 E8       cmp     eax,[ebp-18]
74 02          je      Tutorial-i386.exe.text+26693
EB 1F          jmp     Tutorial-i386.exe.text+266B2
C7 45 E8 000...mov     [ebp-18],00000000
6A 00          push    00
```

```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if has_key:
            open_door()
    if button("down"):
        crouch()
```

Instruction patching

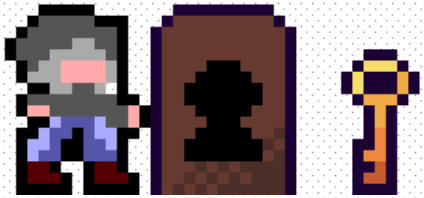
Replace the *has_key* condition


```
def player_move():
    if collision("coin"):
        coins += 1
    if collision("door"):
        if not has_key:
            open_door()
    if button("down"):
        crouch()
```

Instruction patching

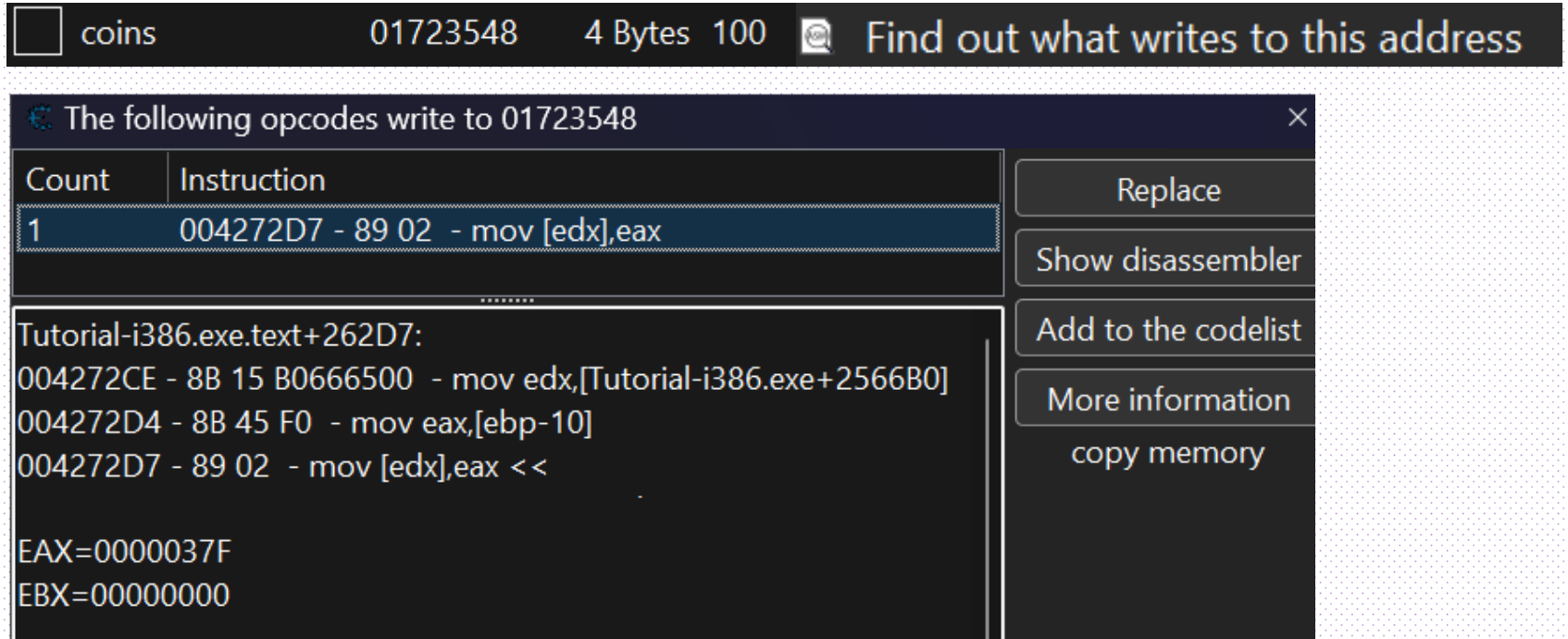
Replace the *has_key* condition

```
74 02          je    Tutorial-i386.exe.text+26687
EB 49          jmp   Tutorial-i386.exe.text+266D0
A1 B0666500    mov   eax,[Tutorial-i386.exe+2566B0]
3B 45 E8       cmp   eax,[ebp-18]
75 02          jne   Tutorial-i386.exe.text+26693
EB 1F          jmp   Tutorial-i386.exe.text+266B2
C7 45 E8 000...mov   [ebp-18],00000000
6A 00          push  00
```



Instruction patching

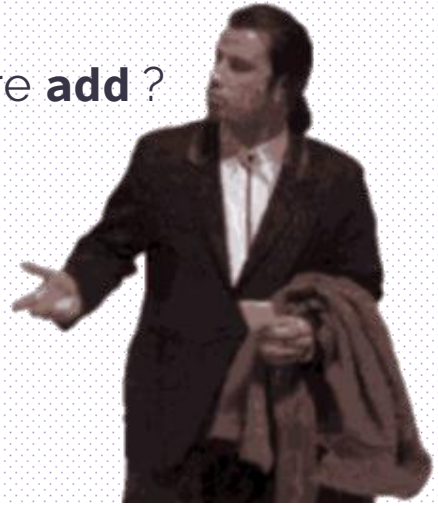
What if we could do it from known memory addresses?



Instruction patching: problems

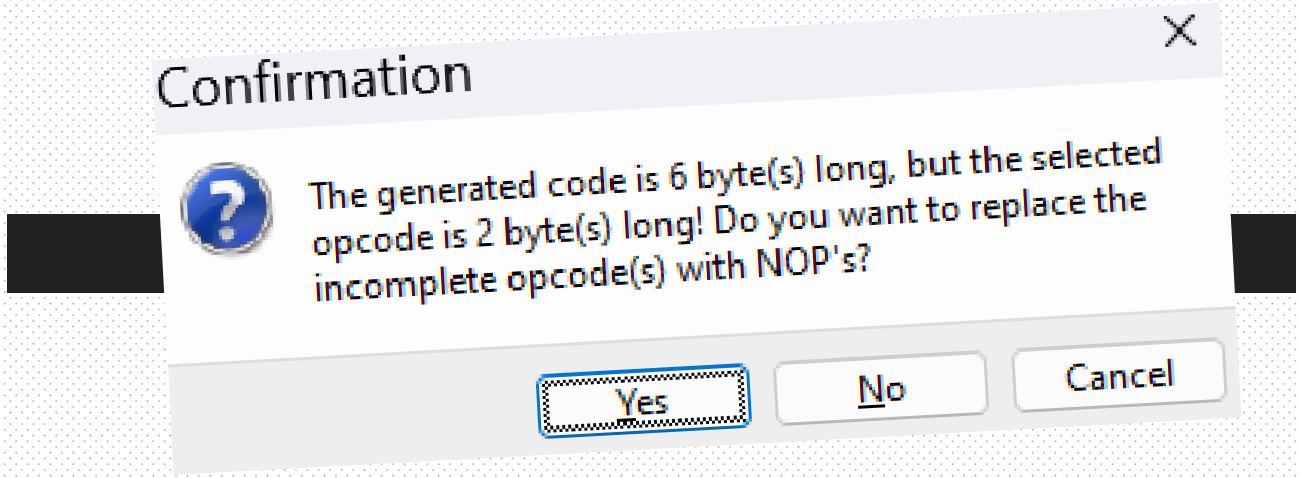
Less obvious instructions

```
mov     [edx],eax
```



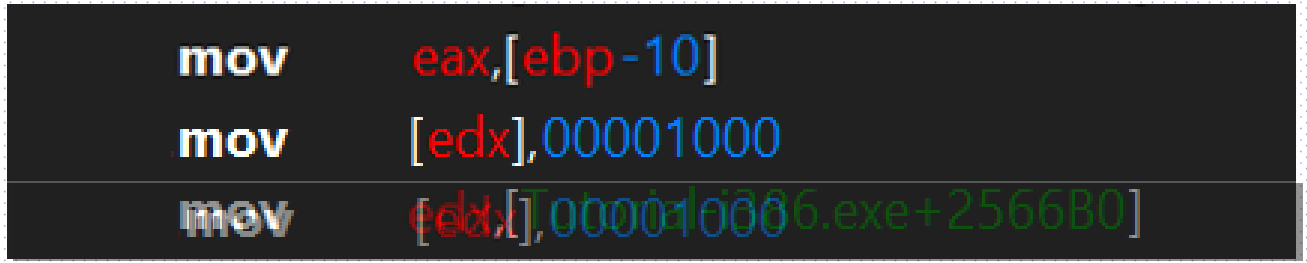
Instruction patching: problems

Instruction size clash



Instruction patching: problems

Instruction size clash
Where to add our code?



Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory
- Create complex methods
- Persisting and toggleable

```
[ENABLE]
alloc(newmem,2048)
label(return)
alloc(multiplier, 2)
registerSymbol(multiplier)



multiplier:
    dd (int)5

newmem:
mov eax, [edx]
add eax, [multiplier]
mov [edx],eax
mov eax, [Tutorial-i386.exe+2566B0]
jmp return

"Tutorial-i386.exe"+272D7:
jmp newmem
nop 2
return:

[DISABLE]
dealloc(multiplier)
unregisterSymbol(multiplier)

dealloc(newmem)
"Tutorial-i386.exe"+272D7:
db 89 02 A1 B0 66 65 00
```

Autoassemble

Solution: **auto assembler!**

- **Auto-allocate memory**

```
alloc(newmem,2048)
label(return)

newmem:
// your code here

jmp return

"Tutorial-i386.exe"+272D7: //original
jmp newmem                 address
nop 2
return:
```

Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory

  manages labels, variables…


```
alloc(newmem,2048)
label(return)
alloc(multiplier, 2)
registerSymbol(multiplier)


multiplier:
    dd (int)5

newmem:
mov eax, [edx]
add eax, [multiplier]


jmp return

"Tutorial-i386.exe"+272D7:
jmp newmem
nop 2
return:
```

Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory
- **Create complex methods**

```
alloc(newmem,2048)
label(return)
alloc(multiplier, 2)
registerSymbol(multiplier)


multiplier:
    dd (int)5

newmem:
mov eax, [edx] //coins += multiplier
add eax, [multiplier]
mov [edx],eax
mov eax, [Tutorial-i386.exe+2566B0]
jmp return

"Tutorial-i386.exe"+272D7:
jmp newmem
nop 2
return:
```

Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory
- Create complex methods
- **Persisting and toggleable**

```
newmem:
mov eax, [edx] //coins += multiplier
add eax, [multiplier]
mov [edx],eax
mov eax, [Tutorial-i386.exe+2566B0]
jmp return

"Tutorial-i386.exe"+272D7:
jmp newmem
nop 2
return:

[DISABLE]
dealloc(multiplier)
unregisterSymbol(multiplier)

dealloc(newmem)
"Tutorial-i386.exe"+272D7:
db 89 02 A1 B0 66 65 00
```

Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory
- Create complex methods
- **Persisting and toggleable**

```
newmem:
mov eax, [edx] //coins += multiplier
add eax, [multiplier]
mov [edx],eax
mov eax, [Tutorial-i386.exe+2566B0]
jmp return

"Tutorial-i386.exe"+272D7:
jmp newmem
nop 2
return:

[DISABLE]
dealloc(multiplier)
unregisterSymbol(multiplier)

dealloc(newmem)
"Tutorial-i386.exe"+272D7:
db 89 02 A1 B0 66 65 00
```

Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory
- Create complex methods
- **Persisting and toggleable**

```
[ENABLE]
alloc(newmem,2048)
registerSymbol(multiplier)
registerSymbol(INJECT)
aobscanmodule(INJECT,Tutorial-i386.exe,
89 02 A1 B0 66 65 00)

multiplier:
    dd (int)5

newmem:
mov eax, [edx] //coins += multiplier
add eax, [multiplier]
mov [edx],eax
mov eax, [Tutorial-i386.exe+2566B0]
jmp return

INJECT:
jmp newmem
nop 2
return:
```

Autoassemble

Solution: **auto assembler!**

- Auto-allocate memory
- Create complex methods
- **Persisting and toggleable**

```
[ENABLE]
alloc(newmem,2048)
registerSymbol(multiplier)
registerSymbol(INJECT)
aobscanmodule(INJECT,Tutorial-i386.exe,
89 02 A1 B0 66 65 00)

multiplier:
    dd (int)5

newmem:
mov eax, [edx] //coins += multiplier
add eax, [multiplier]
mov [edx],eax
mov eax, [Tutorial-i386.exe+2566B0]
jmp return

INJECT:
jmp newmem
nop 2
return:

[DISABLE]
dealloc(multiplier)
unregisterSymbol(multiplier)

dealloc(newmem)
INJECT:
db 89 02 A1 B0 66 65 00
unregistersymbol(INJECT)
```

Level 3: thinking 3D

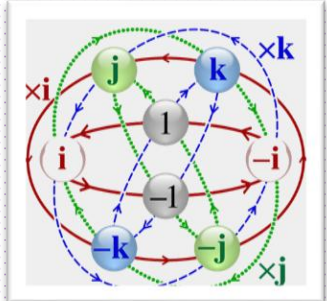
- Where 3D angles?
- Uses quaternions, rotation matrices and radian angles
- See Godot docs for more infos.

Memory View

2B4CC81B1AC	0.00	0.00	0.00	-6798912...	0.00	0.00
2B4CC81B1C4	0.00	0.00	0.00	-1.00 SX	0.00	0.00
2B4CC81B1DC	0.00	1.00 SY	0.00	0.00	0.00	-1.00 SZ ?
2B4CC81B1F4	-33.93	14.65	-20.00	-1.00	0.00	0.00
2B4CC81B20C	0.00	1.00	0.00	0.00	0.00	-1.00
2B4CC81B224	-20.20 X	6.64 Y	-27.89 Z	0.00	0.00	0.00
2B4CC81B23C	0.00	1.00	1.00	1.00	0.00	0.00

Properties

Basis	basis	`Basis(1, 0, 0, 0, 1, 0, 0, 0, 1)`
Vector3	origin	`Vector3(0, 0, 0)`



Finding shared instructions

Modify an instruction, suddenly:

- Bugs appear
- Making yourself invincible makes enemies invincible too.
- Game crashes

```
class Entity {
    int health = 100;

public:
    void takeDamage(int damage) {  }
};

class Player : public Entity {};
class Enemy : public Entity {};
```

```
player.takeDamage(100);
```

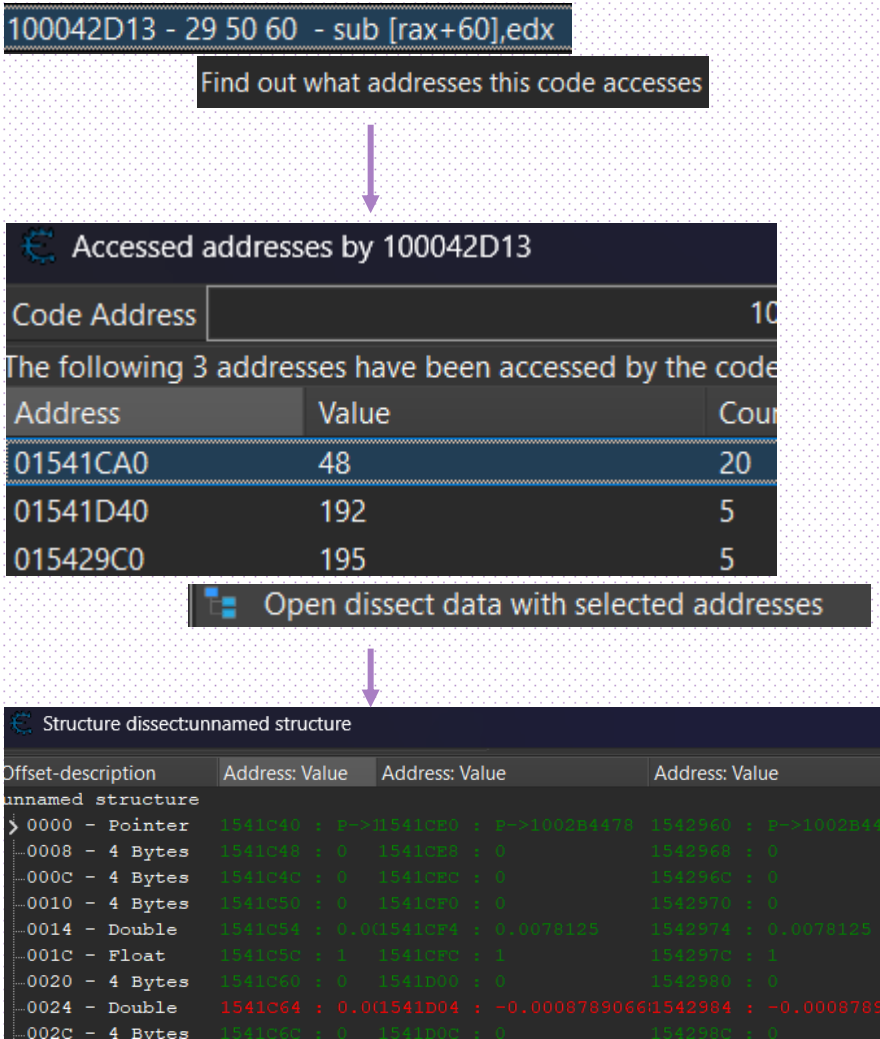
```
sub rax, (int)100
```

```
enemy.takeDamage(10);
```

Finding shared instructions

Solution 1: Dissect data/structures

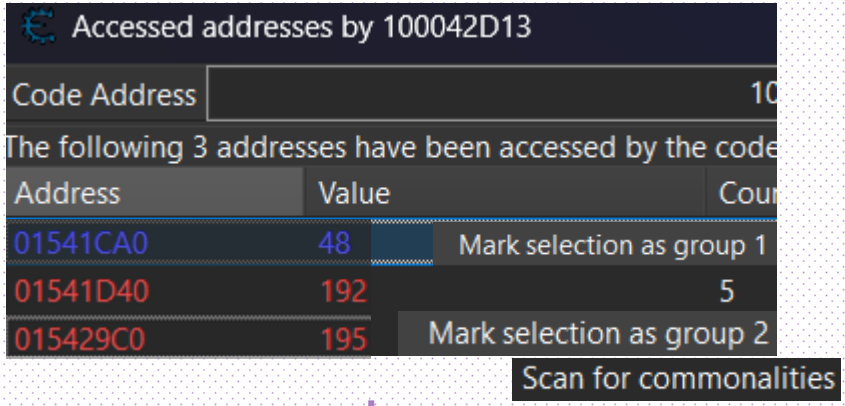
- "Find out what addresses this code access"
- Sellect all: “dissect data”.
- Compare value in groups

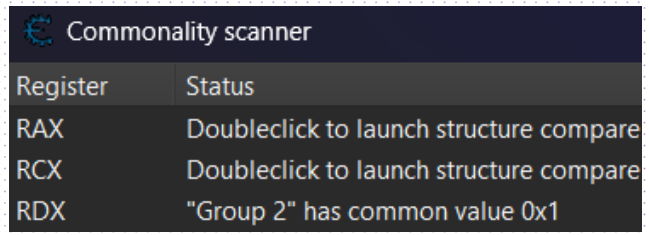


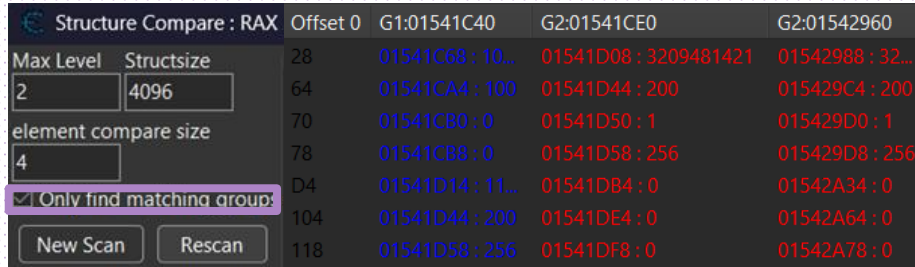
Finding shared instructions

Solution 2: Find commonalities between addresses

- Auto-find common values between different calls
- Compare structures with only unique distinct values



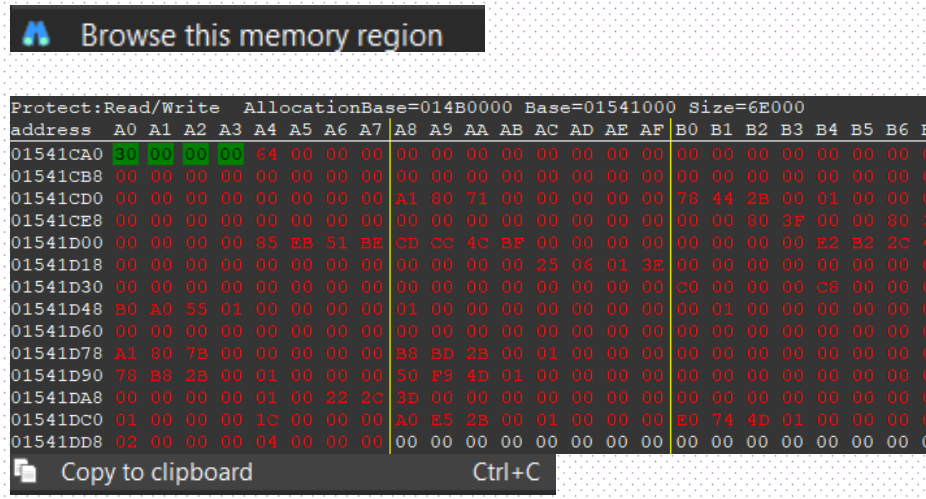




Finding shared instructions

Solution 3: manual search

- Some “static” values change only when relaunching the game
- “*Find commonalities*” does not work across game reboots
- Use some scripts to help scans




```
cat ./comm_test.txt
    80 F4 23 40 F6
    65 F4 8A 40 26
    68 F4 23 40 F6
python ./detect_commonalities_memory.py
./comm_test.txt 3
    -2 = F4
    0 = 40
```

LUA autoclicker

Code complex scripts with LUA

- Complete CE API
- Timers, memory scan, AutoAssembly integration
- GUI development

```
{$lua}
if syntaxcheck then return end
[ENABLE]


l2sh_counter = 0
l2sh_timer = createTimer(nil)

speedhack_setSpeed(500)
l2sh_timer.Enabled = false
l2sh_timer.Interval = 100 -- spam keys
every 0.1s


function l2SH_loop(v_timer)

  if ((l2sh_counter == -1) or
      (l2sh_counter == 750)) then
     l2sh_timer.Enabled = false
  -- stop after 75s or manual interrupt
     speedhack_setSpeed(1)
     return
  end

  if (getForegroundWindow() ~=
findWindow(null, "flag quest")) then
    return -- do not keypresses when
game window is not focused
  end
```

LUA autoclicker

Code complex scripts with LUA

- Complete CE API
- Timers, memory scan, AutoAssembly integration
- GUI development

```
function l2SH_loop(v_timer)

  if ((l2sh_counter == -1) or
      (l2sh_counter == 750)) then
     l2sh_timer.Enabled = false
  -- stop after 75s or manual interrupt
     speedhack_setSpeed(1)
     return
  end

  if (getForegroundWindow() ~=
findWindow(null, "flag quest")) then
    return -- do not keypresses when
game window is not focused
  end


  if ((l2sh_counter % 10) == 0) then
    doKeyPress(VK_SPACE)
    -- 1/10 space keypress
  else
      doKeyPress(VK_RIGHT)
  end
  l2sh_counter = l2sh_counter + 1
end


-- put this AFTER method definition
l2sh_timer.OnTimer = l2SH_loop
```

LUA autoclicker

Code complex scripts with LUA

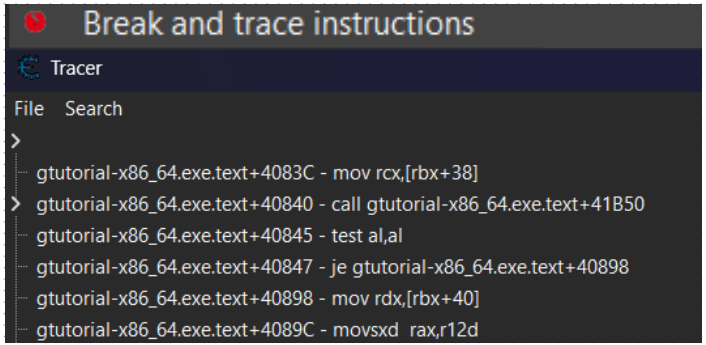
- Complete CE API
- Timers, memory scan, AutoAssembly integration
- GUI development

```
    speedhack_setSpeed(1)
    return
  end

  if (getForegroundWindow() ~=
findWindow(null, "flag quest")) then
    return -- do not keypresses when
game window is not focused
  end


  if ((l2sh_counter % 10) == 0) then
    doKeyPress(VK_SPACE)
    -- 1/10 space keypress
  else
      doKeyPress(VK_RIGHT)
  end
  l2sh_counter = l2sh_counter + 1
end


-- put this AFTER method definition
l2sh_timer.OnTimer = l2SH_loop
l2sh_timer.Enabled = true


[DISABLE]
speedhack_setSpeed(1)
l2sh_timer.destroy() --important!
```

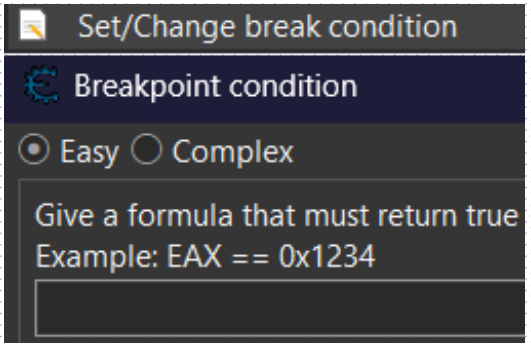
Nice CheatEngine features

- Debugger: ultra powerful

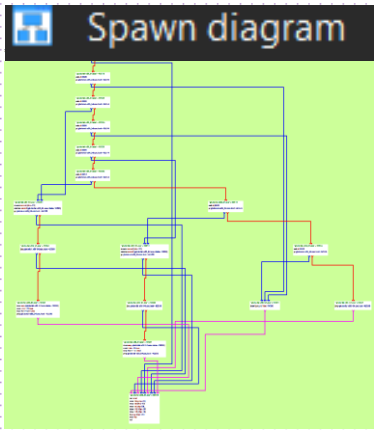
stack trace



conditional breakpoints

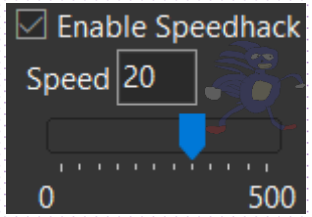


Flow charts

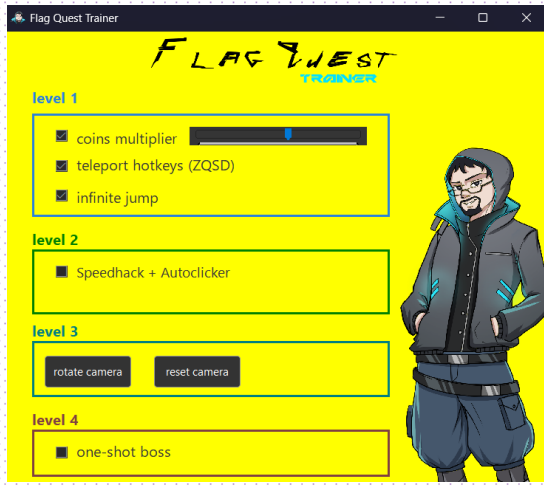


Nice and useless CheatEngine features

- Speedhack: it just works



- GUI generator: very powerful
  but long topic



- Unrandomizer: never works

- Mono Features: better use Unity Explorer directly

The best CheatEngine feature

- Luacode: put LUA inside your ASM
- Write complex code inside of ASM patch
- Have full LUA API access
- Rename register as variables

```
code:
  cmp r9,rax
  jne end


{$LUACODE pos=RBX}

local npos = pos+0x424 + 8 -- Z pos addr
if npos ~= getAddressSafe("lev3_z") then
  print("Updating all CE 3D position vars")

  local names = {"z", "y", "x", "sz2",
"sz1", "sz0", "sy2", "sy1", "sy0", "sx2",
"sx1", "sx0"}

    for _, name in ipairs(names) do
      registerSymbol("lev3_"..name, npos)
      -- going backward in memory
      npos = npos - 4
    end
end
{$ASM}


end:
  movups [rbx+00000420],xmm2 // original
instruction for setting sz2
  jmp return
```

The best worst CheatEngine feature

- Luacode: put LUA inside your ASM

- Bugged at the moment
  Need to add a space after your comments lines at the start of your cheat

- Not optimized

- May crash your game

- Consider `{$CCODE}`
  write C inside your ASM

Common Cheat Engine problems 1

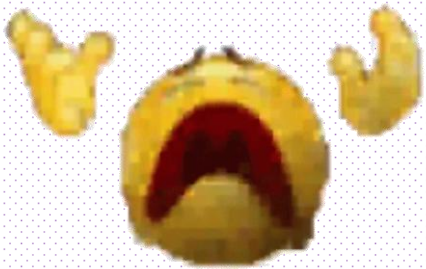
- I changed a value and my game crashed!
  -> *welcome to CE. It will do that.*

- My “find what writes/access” address search has no result
  -> *you already have a scan running, but the window is hidden behind another.*

- I added My AutoAssembly cheat to the codelist, but I can’t enable it
  -> *you clicked on* Execute *and it already patched the game. Relaunch the game.*

Common Cheat Engine problems 2

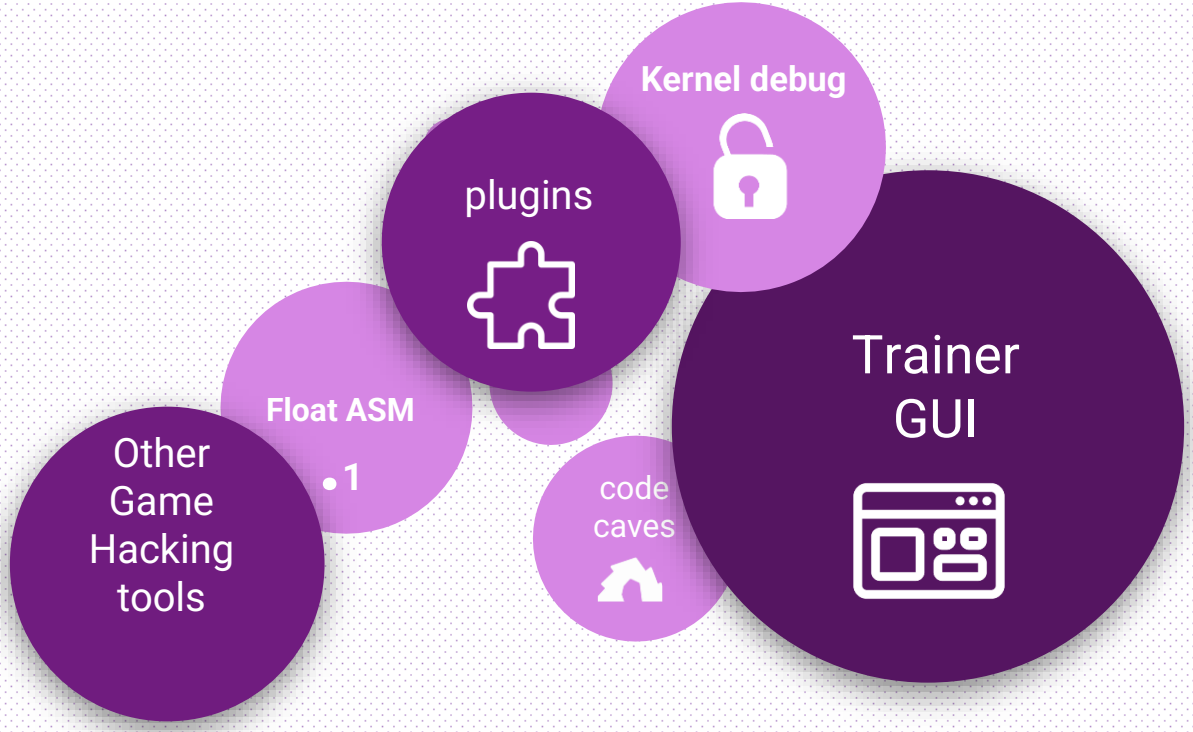
- I did a scan, saved my results in a file.
  Redid a scan, saved to the same file.
  Now the scan doesn't work.
  -> *Just don't do that*

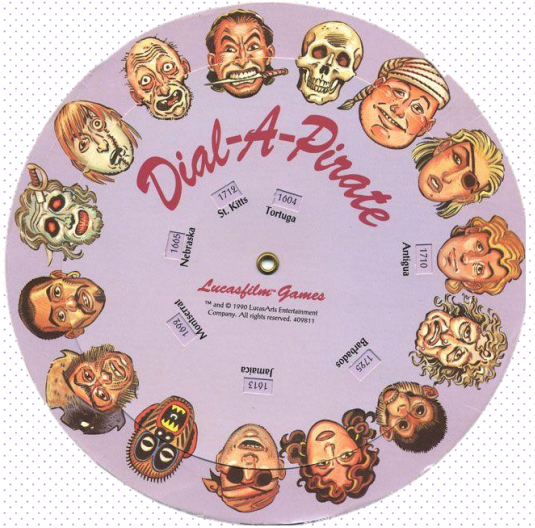


- My LUA script runs twice!
  -> *you forgot to add* `if syntaxcheck then return end`

  I launched a LUA script that never terminates, modified it and restarted it. Now I have bugs.
  -> *Relaunch Cheat Engine :/*

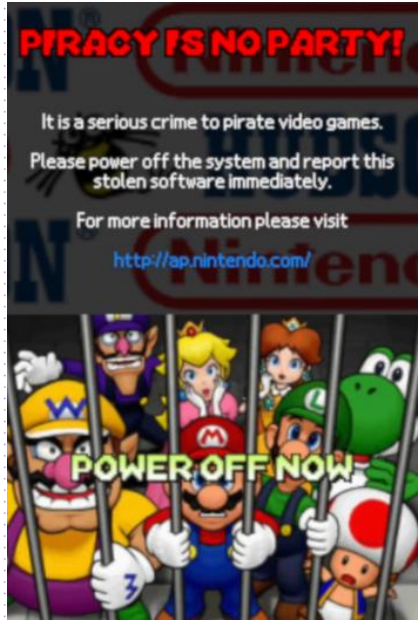
Next steps


Kernel debug
plugins
Float ASM
• 1
Other
Game
Hacking
tools
code
caves
Trainer
GUI



tuxlu.fr/talk_vghacking

tuxlu.fr/talk_vghacking



thank you.