# **Reverse engineering Heroes 3 - Basics**

By Raistlin, put together by Jakub Grzana Last update: 17.02.2021

# **Prerequisites:**

- IDA Pro (interactive disassembler by Hex-Rays) which is NOT free program!
- IDA Database for Heroes 3 SoD 3.59 by GrayFace, ERA HD
- Debugger OllyDbg LINK
- Russian SoD Executable LINK Password: GoogleDoesNotLikeExe

Note: IDA database for 3.59 and ERA\_HD are NOT interchangeable. You must use 3.59 for 3.59. Although version for ERA is still useful, to somehow figure out what's going on, due to better documentation.

#### Introduction

This tutorial will explain only basics of using IDA on example of changing values hardcoded in game memory. It is just first step for real reverse engineering – but important one.

IDA is a disassembler for computer software which generates assembly language source code from machine-executable code IDA performs automatic code analysis, using cross-references between code sections, knowledge of parameters of API calls, and other information. However, IDA itself will not bring you so much, you need the database, that was created and updated by developers during last 20 years.

To start, get a copy of IDA Pro, download database and debugger. After installation of IDA, you can open database Heroes3f.idb just by double-clicking on it. IDA allows you to understand what is going on inside exec - it has named variables, functions etc.

Debugger can be used, among other things, to find precise addresses of constants you found in IDA. After installation, you must start instance of Debugger, then "open" executable of Heroes 3 SoD inside this debugger. Any SoD version should do, but I think WoG is based on russian version, so I encourage to use russian. (i don't have any proofs for that tho)

Here is proof that there're different versions of SoD exec – it is clearly visible in main menu, I don't know if the differences are meaningful though.



#### **Basic assembler instructions**

mov dest, source – copy value from source to dest jmp – jumps to some place in code (pretty much like goto in C) call name – call function with given name/address push val – push value on stack pop var – pop value from stack, save it in var

eax, ebx ... - registers of processor. You can store values or pointers in tchem mov eax, ebx - copies value stored in ebx to eax mov eax, [ebx] - copies value from memory to eax (ebx is pointer)

Note: since Heroes 3 is 32bit executable, registers are 32bit=4bytes aswell.

#### Usage of stack in functions:

```
(eax = 5, ebx = 8, esi = 0x446752)
push eax
push ebx
push esi
call 00678453
```

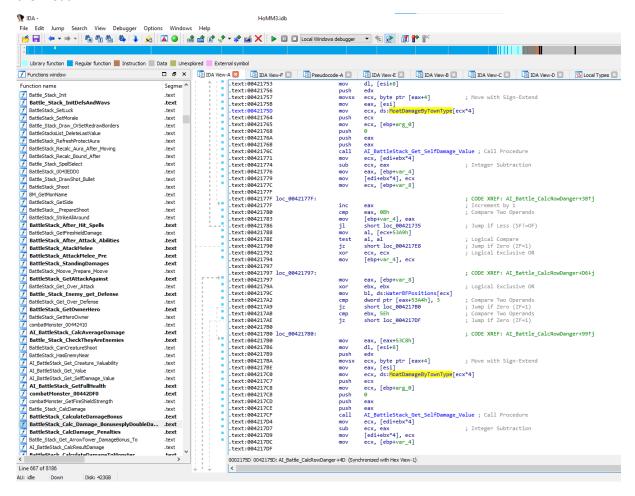
To get access to the arguments in the function use the registr esp. The first argument is [esp], the second - [esp+4], the third - [esp+8]

If you need more informations, there're many tutorials on the internet, for example this one.

# Example 1 – Remove double moat from Fortress

Imagine, that you want to disable double moat in the Fortress, and you do not know which function sets it. The first thing you have to do is open your IDA and start searching. Press alt+T and enter the simplest word: Moat.

When you did it you did it and pressed "Ok", you see a random function, which does something with the moat



Use **Tab** key to get into pseudocode, representing what's going on here — you will see that it's something you don't need.

```
靡 IDA -
                                                                                                                                                                                                                                                                                                                           HoMM3.idb
  File Edit Jump Search View Debugger Options Windows Help
    Library function Regular function Instruction Data Unexplored External symbol
                                                                                                                                                                                                                                                                               □ & × □ IDA View-A □
   Function name
                                                                                                                                                                                                Segmer ^
 Function name

Battle_Stack_Init

Battle_Stack_InitDefsAndWavs

BattleStack_SetLuck

BattleStack_SetMorale

BattleStack_SetMorale

BattleStack_Draw_OrSetRedrawBorders

BattleStack_Stack_Draw_OrSetRedrawBorders

BattleStack_RefereiProtectAura

BattleStack_RefereiProtectAura

BattleStack_RefereiProtectAura

BattleStack_RefereiProtectAura

BattleStack_Recale_Bound_After

BattleStack_SetSetSetSet

BattleStack_Details

Battle
                                                                                                                                                                                                                                                                    int v4; // eax
signed int v5; // eax
                                                                                                                                                                                                  .text
                                                                                                                                                                                                  .text
                                                                                                                                                                                                                                                                   int v9; // ebx
_BattleMgr_ *v1; // [esp+Ch] [ebp-8h]
signed int v12; // [esp+10h] [ebp-4h]
signed int v13; // [esp+10h] [ebp-4h]
      f Battle_Stack_SpellSelect
                                                                                                                                                                                                                                                                   LOBYTE(v4) = this->HasMoat;
      f BattleStack_0043EDD0
f Battle_Stack_DrawShot_Bullet
                                                                                                                                                                                                                                                                   v11 = this;
if ( v4 )
                                                                                                                                                                                                                                                                  {
  v5 = 0;
  v12 = 0;
  do
  {
    f BattleStack_Shoot
f BM_GetMonName
f BattleStack_GetSide
f BattleStack_PrepareShoot
                                                                                                                                                                                                  .text
                                                                                                                                                                                                                                                                                  v6 = WallBFPositions[v5];
if ( this->SiegeKind == 3 || v6 != 95 )
       f BattleStack StrikeAllAround
            BattleStack_StrikeAllAround
BattleStack_After_Hit_Spells
BattleStack_GetFireshieldDamage
BattleStack_After_Attack_Abilities
BattleStack_AtackMelee
BattleStack_AttackMelee_Pre
                                                                                                                                                                                                                                                                                  17 (add - 2007)

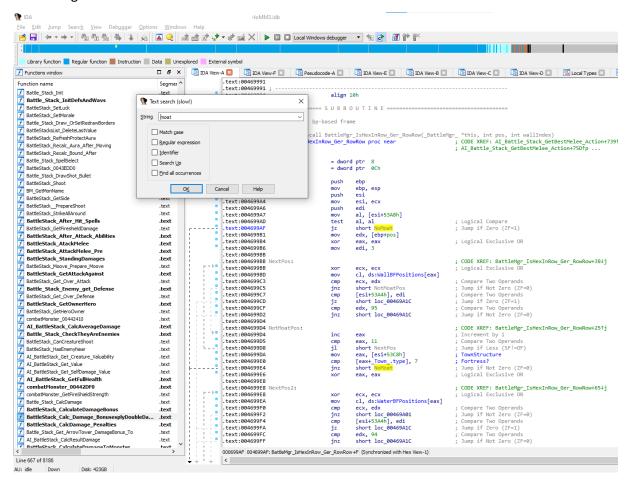
V7 = *(a3 + 4 * v6)

- AI_BattleStack_Get_SelfDamage_Value(a2, *a4, *a4, 0, MoatDamageByTownType[this->Town->type], *(a4 + 8));
                                                                                                                                                                                                                                                                                           v5 = v12;
*(a3 + 4 * v6) = v7;
this = v11;
                                                                                                                                                                                                .text
.text
.text
.text
.text
              BattleStack StandingDamages
             BattleStack_Moove_Preparingles
BattleStack_Moove_Preparingles
BattleStack_GetAttackAgainst
BattleStack_Get_Over_Attack
Battle_Stack_Enemy_get_Defense
                                                                                                                                                                                                                                                                         BattleStack_Get_Over_Defense
             BattleStack GetOwnerHero
                                                                                                                                                                                                .text
.text
.text
.text
.text
                                                                                                                                                                                                                                                                                  F Battle_Stack_CheckTheyAreEnemies
            BattleStack CanCreatureShoot

    BattleStack_CanCreatureShoot
    BattleStack_HasEnemyNear
    AI_BattleStack_Get_Creature_Valuability
    AI_BattleStack_Get_Value
    AI_BattleStack_Get_SelfDamage_Value
    AI_BattleStack_GetFullHealth

                                                                                                                                                                                                                                                                                                {  \forall 4 = AI\_BattleStack\_Get\_SelfDamage\_Value(a2, *a4, *a4, \theta, MoatDamageByTownType[v11->Town->type], *(a4 + 8)); \\ \forall 6 = v4; \\ \forall 8 = v13; 
                                                                                                                                                                                                  .text
                                                                                                                                                                                                  .text
                                                                                                                                                                                                .text
.text
.text
.text
                                                                                                                                                                                                                                                                                         }
v13 = ++v8;
             combatMonster_00442DF0
                                                                                                                                                                                                                                                                                    }
while ( v8 < 11 );
  Journey and the state of the st
                                                                                                                                                                                                                                                                   }
return v4;
 Line 667 of 8186
                                                                                                                                                                                                                                                             0002175D AI_Battle_CalcRowDanger:25 (42175D)
AU: idle Down Disk: 423GB
```

Use **Tab** again to go back into IDA view. After a few attempts more, you will find something more interesting



## Press Tab again and – bingo! This is the right function

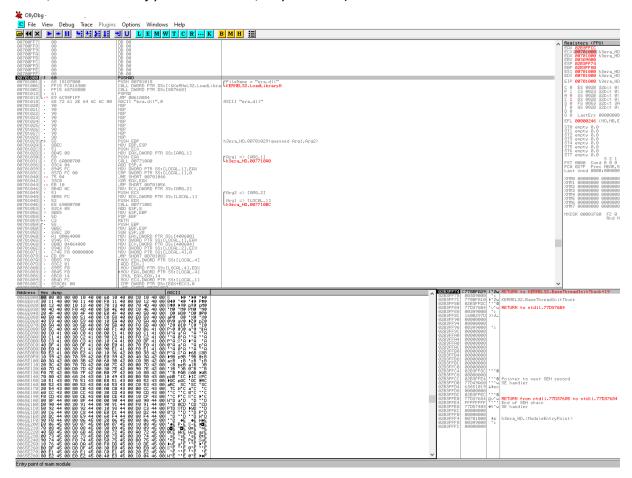
```
IDA View-A 🖂 📭 IDA View-F 🖾 📳 Pseudocode-A 🚨 📭 IDA View-E 🖾 📳 IDA View-B 🖾 📳
   1 char this call BattleMgr IsHexInRow Ger RowRow(int this, int pos, int wallIndex)
   2 {
   3
      int v3; // eax
   4
     char result; // al
   5
   6
      if (*(this + 21416))
   7
      {
8
        v3 = 0;
9
        while ( WallBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 95 )
  10
11
          if ( ++ \vee 3 >= 11 )
  12
            if ( *(*(this + 21448) + offsetof(_Town_, type)) != 7 )
13
14
             goto NoMoat;
15
            v3 = 0;
16
            while ( WaterBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 94 )
  17
18
              if ( ++ v3 >= 11 )
               goto NoMoat;
19
  20
21
            break;
  22
          }
  23
24
        if ( wallIndex )
          *wallIndex = v3;
25
26
        result = 1;
  27
  28
      else
  29
     {
  30 NoMoat:
31
        if ( wallIndex )
32
          *wallIndex = -1;
33
        result = 0;
  34
     }
35
     return result;
36 }
```

#### Here is the check

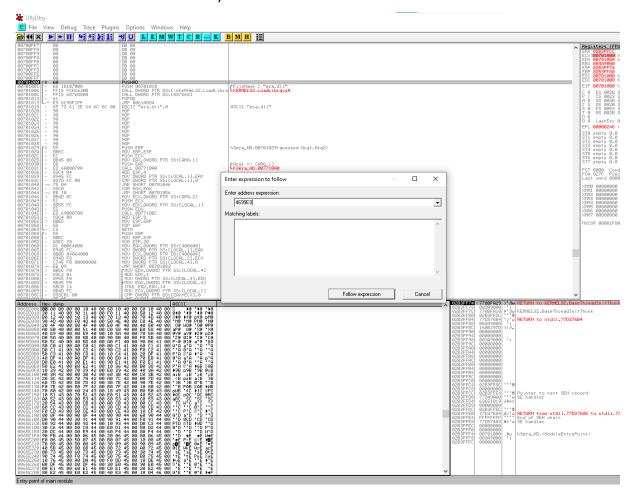
```
IDA View-A 🖂 📙 IDA View-F 🔣 🖳 Pseudocode-A 🔼 📙 IDA View-E 🖾 📜 IDA View-B 🗵
    1 char thiscall BattleMgr IsHexInRow Ger RowRow(int this, int pos, int wallIndex)
   2 {
   3
       int v3; // eax
      char result; // al
   5
   6
       if (*(this + 21416))
       {
   8
         v3 = 0;
         while ( WallBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 95 )
   9
  10
  11
            if ( ++ v3 >= 11 )
  12
            {
              if ( *(*(this + 21448) +
13
                                         offsetof(_Town_, type)) != 7 )
14
                goto NoMoat;
              v3 = 0;
15
16
              while ( WaterBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 94 )
  17
18
                if ( ++ v3 >= 11 )
19
                  goto NoMoat;
  20
 21
              break;
  22
           }
  23
          if ( wallIndex )
  24
25
           *wallIndex = v3;
26
         result = 1;
  27
  28
       else
  29
       {
  30 NoMoat:
31
         if ( wallIndex )
32
           *wallIndex = -1;
33
         result = 0;
  34
       }
35
       return result;
36 }
Mouse-click on it, then press Tab once again. You will get this:
 .text:004699D5
                                                                 ; Compare Two Operands
                               cmp
 .text:004699D8
                               j1
                                      short NextPos
                                                                 ; Jump if Less (SF!=OF)
 .text:004699DA
                                      eax, [esi+53C8h]
                                                                 ; TownStructure
                               mov
                                      [eax+_Town_.type], 7
short NoMoat
 .text:004699E
                                                                 ; Fortress?
                               cmp
 .text:004699E4
                               jnz
                                                                 ; Jump if Not Zero (ZF=0)
 .text:004699E6
                                      eax, eax
                                                                 ; Logical Exclusive OR
                               xor
```

Select and copy address of comparison (cmp asm instruction)

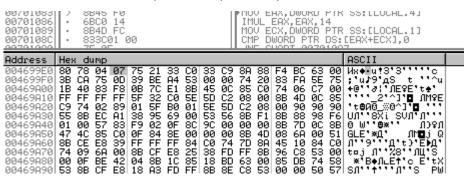
We're done in IDA at this point. Open (olly) debugger->File->Open->Select SoD executable. Click on hex dump frame (it is at the bottom of the window) (Note: here h3era HD.exe is used – doesn't change much, but don't do it if you work on 3.59, only SoD then)



#### Press ctrl+G and enter the adress you have found in IDA:

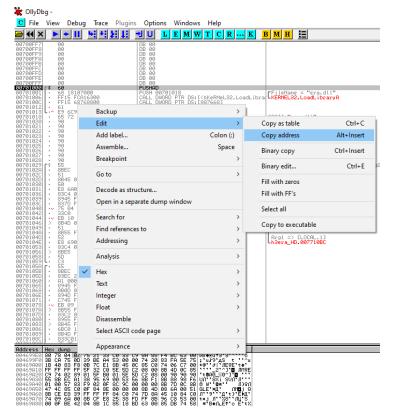


Here, at this address, you can find data determining assembly instruction and it's parameters. Find town number of fortress (which is 7 (Format T) –also remember values here are in hex)



On this example, we can see that Town type used in comparison (which is hardcoded) is stored in byte on address 0x4699E3 (every two hex values represent one byte) Making the comparison to always return false would be more complicated, so easiest solution is to change that hardcoded value to some non-existant town, for example Town=99.

## Copy address by rightclick on the value -> edit -> copy address



Now you can use it. There're many ways to change value in memory in WoG 3.59: by Lua mem, by ERM UN:C, by using Copiers array... but let's stick to easiest solution for now.

\*(Byte \*)0x4699E3 = 99;

Done! This will remove damage taken from first row of moat in fortress – it will still stop when stepped upon though.

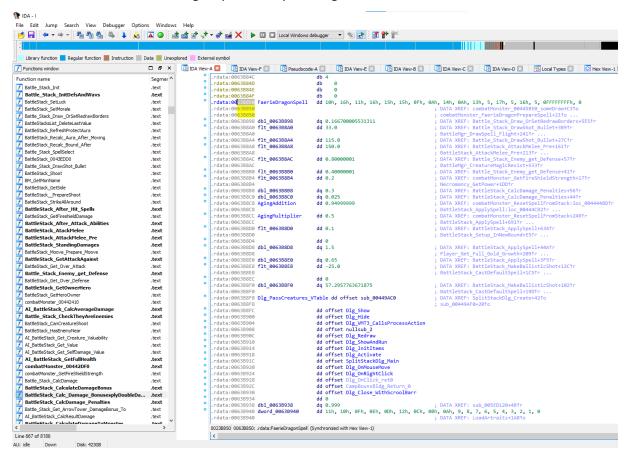
# Example 2 – Make Faerie Dragon to spell "Magic arrow" instead of "Chain Lightning"

Note: works only for h3era\_HD.exe, still nice for education, but don't try to recreate in 3.59.

Just start searching (alt+T) and find what you need

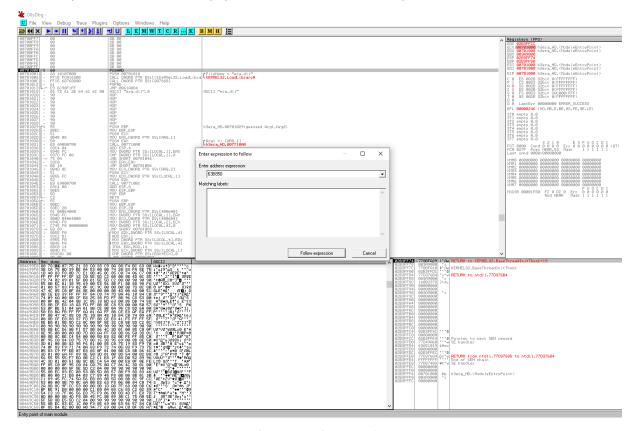
```
.text:00445C96
                             text:00445C96 loc 00445C96:
                                                                                                                                      ; CODE XREF: combatMonster 00445BE0 someDraw+A01j
                                                                                         eax, [edi]
ebx, esi
.text
                             text:00445C96
.text
                                                                                         [ebp+arg_4], eax
                                                                              mov
Text search (slow!)
                                                                       ×
                                                                                                                                      ; CODE XREF: combatMonster_00445BE0_someDraw+9E1j
; combatMonster_00445BE0_someDraw+B41j
String faerie
                                                                                         esi, [ebp+arg_0]
    Match case
                                                                                                                                      ; CODE XREF: combatMonster_00445BE0_someDraw+7F<sup>†</sup>j
; combatMonster_00445BE0_someDraw+8F<sup>†</sup>j
; Add
    Regular expression
                                                                                        edi, 4
edi, offset FaerieDragonSpell
short loc_00445C58
ecx, [ebp+arg_4]
edi, [ebp+arg_4]
edi, [ebp+arg_0], ecx
     Identifier
                                                                                                                                      ; Compare Two Operands
; Jump if Less (SF!=OF)
                                                                             cmp
jl
    Search Up
     Find all occurrences
             OK Cancel Help
                                                                                         esi, ecx
                           l.text:00445CB6 loc 00445CB6:
.text
                                                                                                                                      ; CODE XREF: combatMonster_00445BE0_someDraw+6Cfj
                                                                                        ecx, [ebp+var_8]
edi
                             .text:00445CB6
.text:00445CB9
.text:00445CBA
                                                                             mov
push
.text
                                                                              push
call
.text
                                                                                         BattleStack MakeBallisticShot : Call Procedure
                             .text:00445CBB
                            .text:00445CC0
.text:00445CC1
.text:00445CC2
                                                                                         edi
.text
                             .text:00445CC3
                                                                                         esp, ebp
                             .text:00445CC5
                                                                                                                                      : Return Near from Procedure
```

double click on the "FaerieDragonSpell", and you will go to the offset

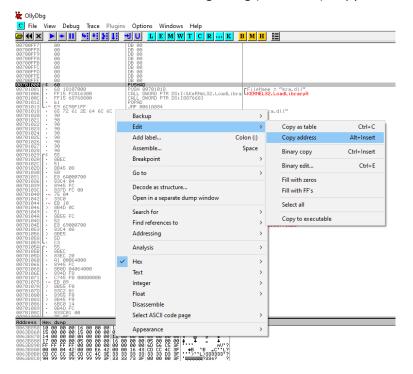


Select and copy the address

#### Go to Olly, click on the Hex Dump, press ctrl+G and follow the expression



the value 0x13=19 is the chain lightning (Format SP), copy it's address and do what you want



\*(int \*)0x63B878 = 0x0F; //replace Chain Lightning with Magic Arrow