

#HITB2023AMS

<https://conference.hitb.org/>

HITB
2023
AMS

Bypassing Anti-Cheats & Hacking Competitive Games





Hello I'm...

Rohan Aggarwal @nahoragg

Founder at DefCore Security

Bug Bounties

Live Hacking Event(H1 & Intigriti)

Love playing games(Valorant, Elden Ring, etc)





Outline

01 - Cheats vs Anti-Cheats

02 - Game Hacking Basics

03 - Kernel Cheats

04 - External Hardware Cheats

05 - Demonstration



What kind of Game Hacking?



History of Game Hacking





1980 - Modding

```

00000000
00000000 B8 C0 07 8E D8 B8 00 90 8E C0 B9 00 01 29 F6 29 FF FC F3 A5 EA 19
00000016 00 00 90 8C C8 BA F4 3F 8E D8 8E C0 8E D0 89 D4 6A 00 0F A1 BB 78
0000002C 00 64 0F B5 37 89 D7 B9 06 00 FC 65 F3 A5 89 D7 C6 45 04 12 64 89
00000042 3F 64 8C 47 02 8C C8 8E E0 8E E8 30 E4 30 D2 CD 13 31 D2 B9 02 00
00000058 BB 00 02 B8 04 02 CD 13 73 12 50 E8 2F 01 89 E5 E8 34 01 58 30 D2
0000006E 30 E4 CD 13 EB DF 31 D2 B9 12 00 BB 00 0A B8 01 02 CD 13 73 0B B1
00000084 0F B8 01 02 CD 13 73 02 B1 09 2E 89 0E CE 01 B8 00 90 8E C0 B4 03
0000009A 30 FF CD 10 B9 09 00 BB 07 00 BD D0 01 B8 01 13 CD 10 B8 00 10 8E
000000B0 C0 E8 14 00 E8 04 01 E8 0A 01 E8 D8 00 EA 00 00 20 90 05 00 00 00
000000C6 00 00 8C C0 A9 FF 0F 75 FE 31 DB 8C C0 2D 00 10 3B 06 F4 01 76 01
000000DC C3 2E A1 CE 01 2B 06 C2 00 89 C1 C1 E1 09 01 D9 73 09 74 07 31 C0
000000F2 29 D8 81 E8 09 E8 34 00 89 C1 03 06 C2 00 2E 3B 06 CE 01 75 12 B8
00000108 01 00 2B 06 C4 00 75 04 FF 06 C6 00 A3 C4 00 31 C0 A3 C2 00 C1 E1
0000011E 09 01 CB 73 AE 8C C0 80 C4 10 8E C0 31 DB EB A3 60 60 B8 2E 0E BB
00000134 07 00 CD 10 61 8B 16 C6 00 8B 0E C2 00 41 88 D5 8B 16 C4 00 88 D6
0000014A 81 E2 00 01 B4 02 52 51 53 50 CD 13 72 05 83 C4 08 61 C3 50 E8 0C
00000160 00 30 E4 30 D2 CD 13 83 C4 0A 61 EB C1 B9 05 00 89 E5 51 E8 1F 00
00000176 3A 0E 05 00 73 0F B8 45 0E 28 C8 CD 10 B0 58 CD 10 B0 3A CD 10 83
0000018C C5 02 E8 0E 00 59 E2 DE C3 B8 0D 0E CD 10 B0 0A CD 10 C3 B9 04 00
000001A2 8B 56 00 C1 C2 04 B4 0E 88 D0 24 0F 04 30 3C 39 76 02 04 07 CD 10
000001B8 E2 EB C3 52 BA F2 03 30 C0 EE 5A C3 B4 01 B7 00 B9 00 20 CD 10 C3
000001CE 00 00 0D 0A 4C 6F 61 64 69 6E 67 00 00 00 00 00 00 00 00 00 00
000001E4 00 00 00 00 00 00 00 00 00 00 00 00 04 00 00 FB 2B 00 00 00 00
000001FA 00 00 00 00 55 AA FA B0 80 E6 70 B8 00 90 8E D8 8E C0 8E E0 8E D0
00000210 89 D4 0E 1F 0F 01 1E A2 00 0F 01 16 A8 00 BA 92 00 EC 3C FF 74 12
00000226 67 8A 64 24 04 84 E4 74 04 0C 02 EB 02 24 FD 24 FE EE E8 2B 00 B0
0000023C D1 E6 64 E8 24 00 B0 DF E6 60 E8 1D 00 B8 01 00 0F 01 F0 EB 00 B8
00000252 18 00 8E D8 8E C0 8E D0 8E 0E 8E E8 66 EA 00 00 01 00 10 00 E8 16
00000268 00 E4 64 3C FF 74 0F A8 01 74 07 E8 09 00 E4 60 EB EC A8 02 75 E8
0000027E C3 EB 00 C3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 7F
.....).).....
.....?.....j...x
.d..7.....e.....E..d.
7d.G.....0.0...l...
.....s.P./....4.X0.
0....l.....s..
.....s.....
0.....
.....u.l...-...;...v.
.....+.....s.t.l.
)....4.....;...u.
..+...u.....l.....
..s.....l.....`
...a.....A.....
.....RQSP..r...a.P.
.0.0.....a.....0...
:.s...E.(...X...:...
.....Y.....
.V.....$.0<9v.....
..R...0..Z.....
...Loading.....
.....+.....
...U....p.....
.....<.t.
g.d$.t...$.$.+..
..d.$.....
.....f.....
..d<.t...t.....`...u.
.....

```

1995 – First Aimbot



Early 2000 – Multiplayer Cheats



Early 2000 – AntiCheats



2000



2002



Popular Kernel AntiCheats



easyTM
ANTI-CHEAT





Anti-Cheats

Detect Cheats - Signatures

Discover Cheats - Reports, Manual

Prevent Cheats - Obfuscation, Sandbox



Features of Kernel AC

- Blocking / stripping of process handles in UM
- Detection of test signing
- Detection of usermode hooks
- Detection of injected modules
- Detection of manually mapped modules
- Detection of kernel drivers
- Detecting of traces of manually mapped drivers
- Detection of virtual machines and emulation



User Mode

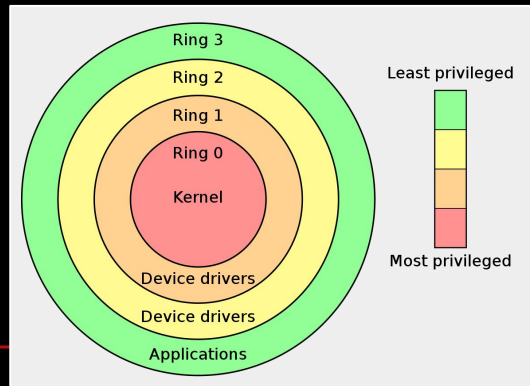
Kernel Mode

Restricted access to system resources

Private virtual space for each process

direct and unrestricted access to system resources

Single virtual space for whole kernel



#HITB2023AMS

<https://conference.hitb.org/>



Game Hacking Basics



Types of Cheats

Internal

Injected into the target process itself.

Complex to make(depends on engine)

Much Flexible and great performance

Prefered for Games with low-level or No AC

External

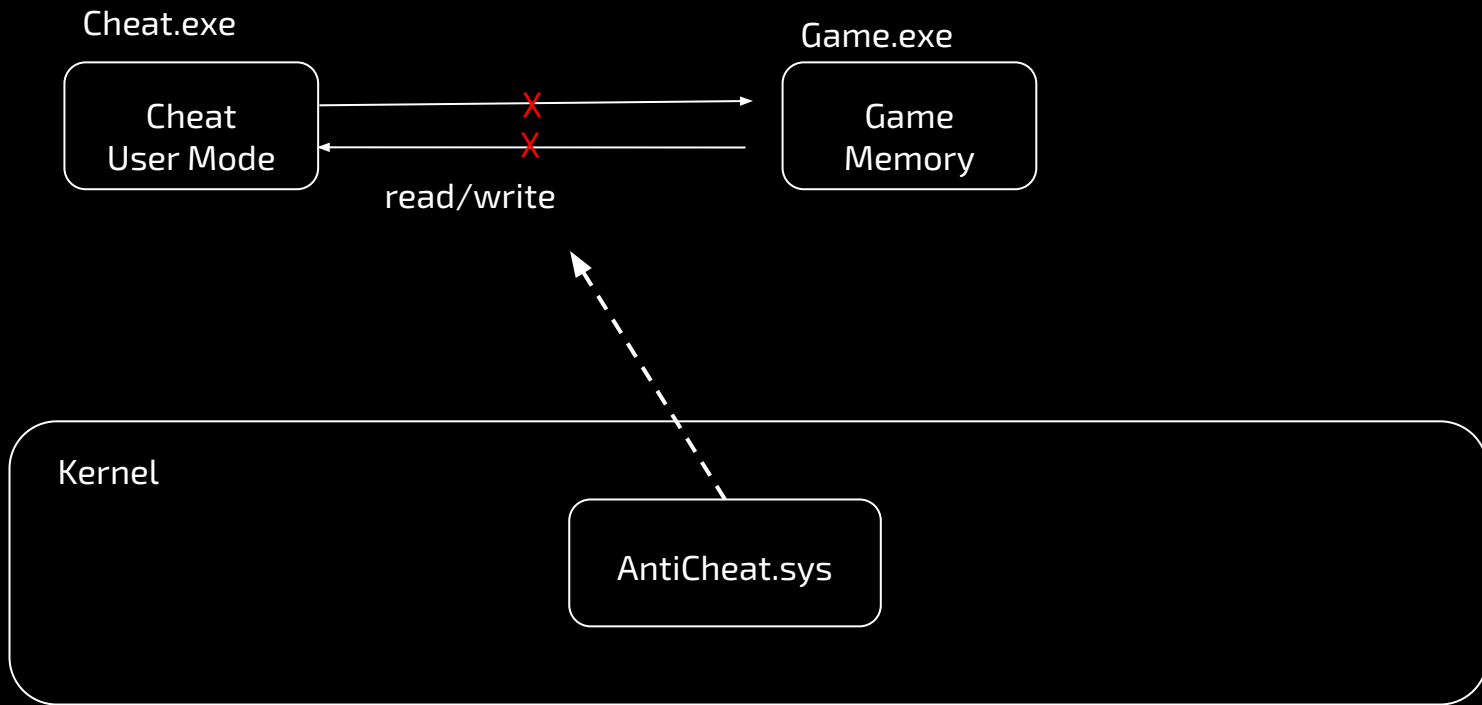
Have their own process that manipulates target process

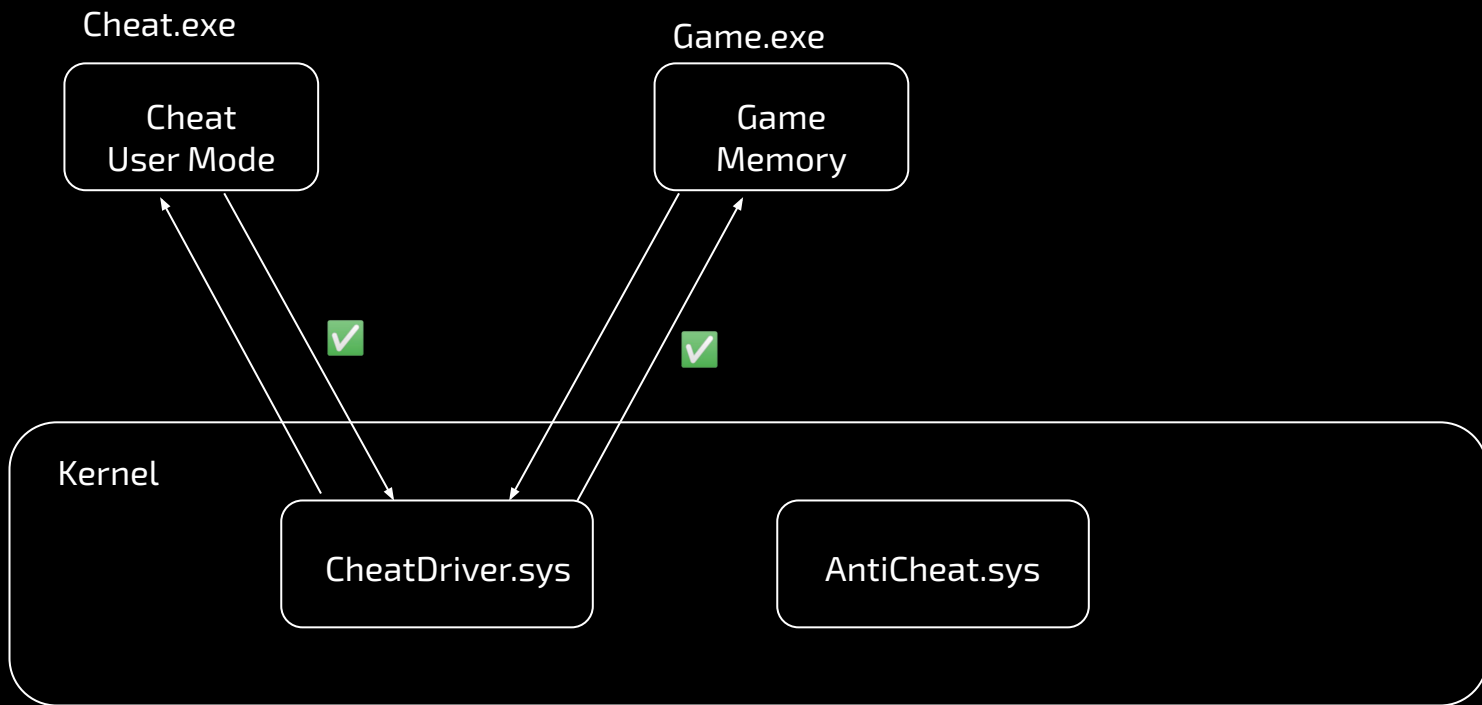
Easy to make(Any lang.)

Enough Flexibility and performance

Prefered for Games with Strong AC









Challenges

1. How to create driver?
2. How to load the driver into kernel?
3. How to communicate from user mode to kernel mode?
4. How I can make my driver Undetected?

#HITB2023AMS

<https://conference.hitb.org/>



Kernel Cheat Development



Cheat Development

1. Reversing

Reversing Game to find the offsets for required cheat.

2. Hooking

Hooking to system call function and placing our shell code for establishing communication b/w UM & KM.

3. Creating Driver

Creating Custom Driver that read/write to game memory from kernel.

4. Loading Driver

Loading the Driver into Kernel.

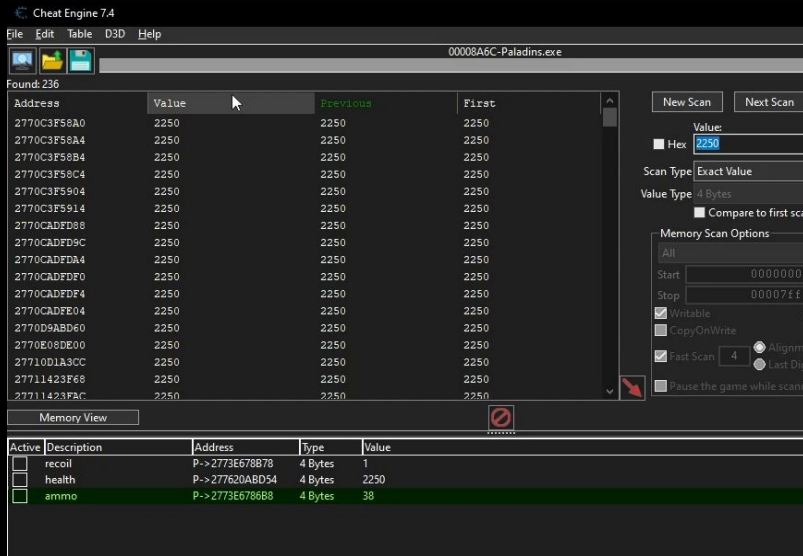
5. Creating User Mode

Creating User Mode that sends read/write request to kernel mode.

1. Reversing

Debugging(Cheat Engine)

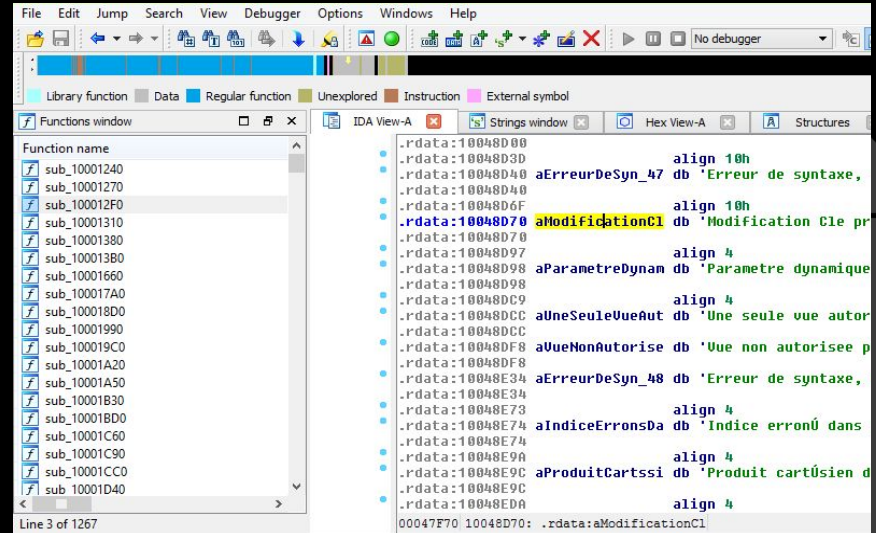
Disassembling(IDA pro)



Cheat Engine 7.4 interface showing a memory scan for the value 2250. The scan results table is visible, and the memory view shows the 'ammo' variable at address P->2773E67868 with a value of 38.

Address	Value	Previous	First
2770C3F58A0	2250	2250	2250
2770C3F58A4	2250	2250	2250
2770C3F58B4	2250	2250	2250
2770C3F58C4	2250	2250	2250
2770C3F5904	2250	2250	2250
2770C3F5914	2250	2250	2250
2770CAEFD88	2250	2250	2250
2770CAEFD9C	2250	2250	2250
2770CAEFDAA	2250	2250	2250
2770CAEFDFO	2250	2250	2250
2770CADDF4	2250	2250	2250
2770CADDF04	2250	2250	2250
2770D9ABD60	2250	2250	2250
2770E08DE00	2250	2250	2250
27710D1A3CC	2250	2250	2250
27711423F68	2250	2250	2250
27711423F8C	2250	2250	2250

Active	Description	Address	Type	Value
<input type="checkbox"/>	recoil	P->2773E678B78	4 Bytes	1
<input type="checkbox"/>	health	P->277620ABD54	4 Bytes	2250
<input checked="" type="checkbox"/>	ammo	P->2773E67868	4 Bytes	38



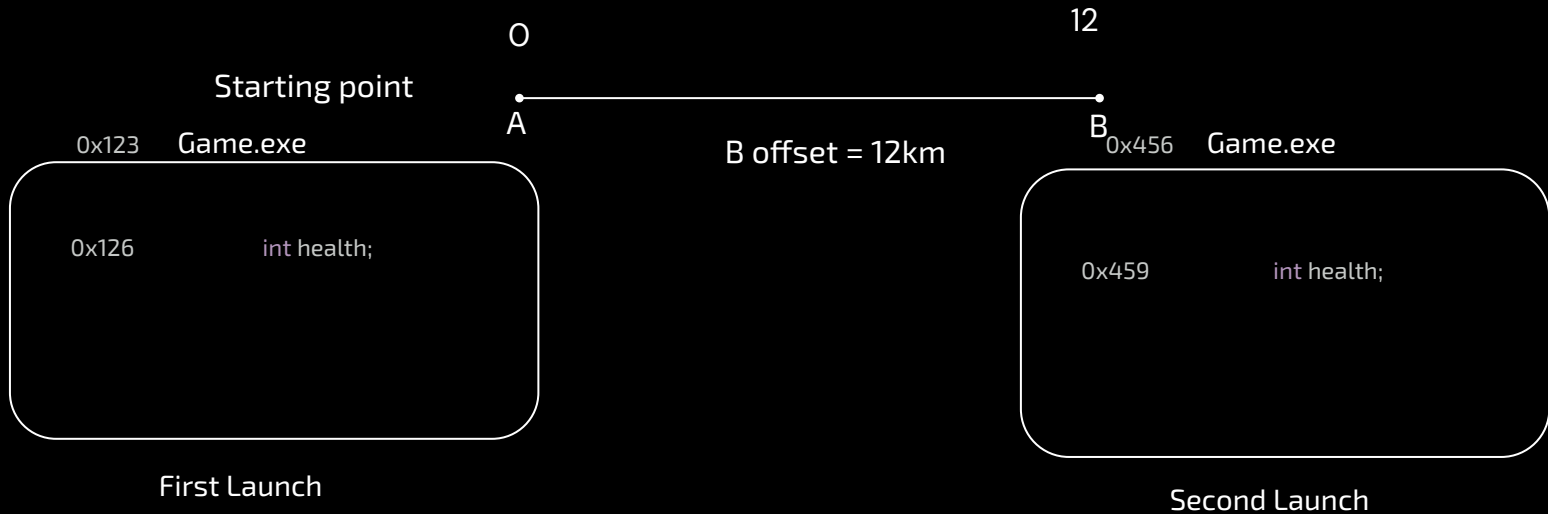
IDA Pro interface showing a disassembled assembly function. The function name is 'sub_10001D40' and the assembly code is visible, including instructions like 'align 10h', 'aErreurDeSyn_47 db 'Erreur de syntaxe,', and 'aModificationCl db 'Modification Cle pr'.

```

.rdata:10048D00
.rdata:10048D3D
.rdata:10048D40 aErreurDeSyn_47 db 'Erreur de syntaxe,
.rdata:10048D40
.rdata:10048D6F align 10h
.rdata:10048D70 aModificationCl db 'Modification Cle pr
.rdata:10048D70
.rdata:10048D97 align 4
.rdata:10048D98 aParametreDynam db 'Parametre dynamique
.rdata:10048D98
.rdata:10048DC9 align 4
.rdata:10048DC0 aUneSeuleVueAut db 'Une seule vue autor
.rdata:10048DCC
.rdata:10048DF8 aVueNonAutorise db 'Vue non autorisee p
.rdata:10048DF8
.rdata:10048E34 aErreurDeSyn_48 db 'Erreur de syntaxe,
.rdata:10048E34
.rdata:10048E73 align 4
.rdata:10048E74 aIndiceErronsDa db 'Indice errone dans
.rdata:10048E74
.rdata:10048E9A align 4
.rdata:10048E9C aProduitCartssi db 'Produit cartusien d
.rdata:10048E9C
.rdata:10048EDA align 4
00047E70 10048D70: .rdata:aModificationCl
  
```



What are offsets?



Health Offset from game.exe is same in both cases i.e. 0x3



What are offsets?

Game.exe

```
struct enemy { //0x0
    int health; //0x0
    int ammo; //0x4
    float x; //0x8
    float y; //0xC
    float z; //0x10
};

struct player { //0x10
    int health; //0x0
    int ammo; //0x4
    float x; //0x8
    float y; //0xC
    float z; //0x10
};
```

0x0 0x10



ammo = 40



*(ammoAdd) = 40



*(playerAdd + 0x4) = *(ammoAdd) = 40



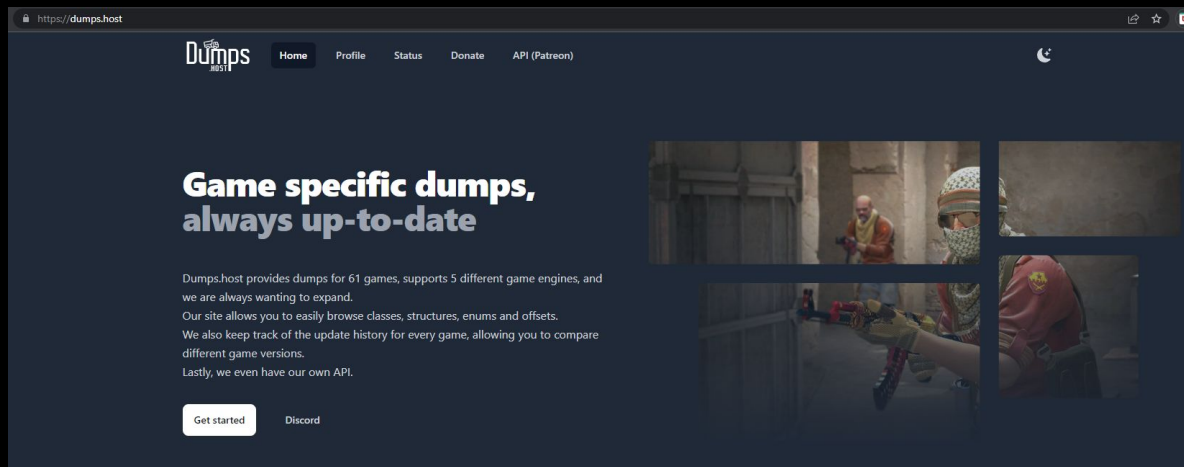
((GameAdd + 0x10) + 0x4) = *(playerAdd + 0x4)
= *(ammoAdd) = 40

Ammo Offset = Game.exe + 0x10 -> 0x4



Lazy Offsets

- <https://dumps.host/>
- <https://www.unknowncheats.me/>
- <https://github.com/>





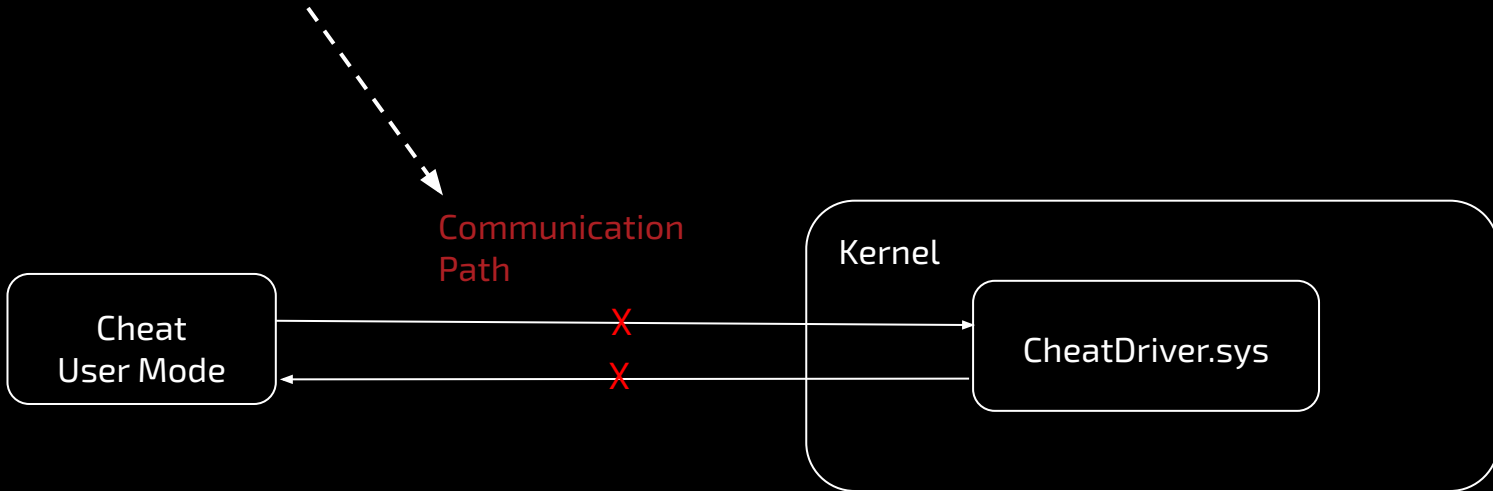
Hello World Driver

Requirements:

1. Visual Studio
2. Windows Driver Kit
3. WinDBG / DBGview

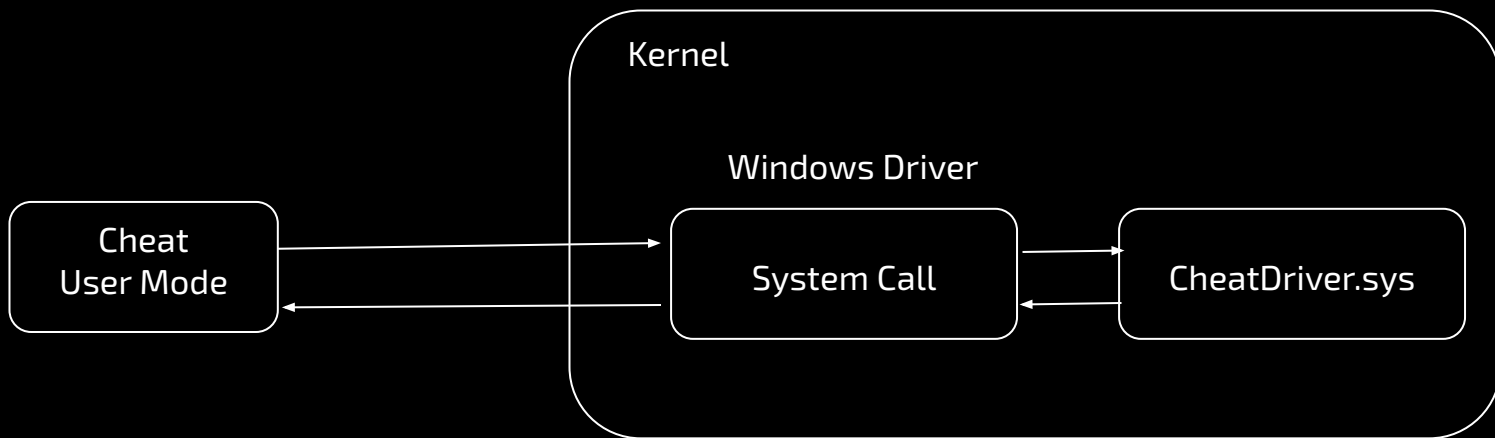


2. Hooking





2. Hooking



shell code:

```
Mov rax, CheatDriver_address  
Jmp rax
```



User Mode

```
systemCall(instruction)
```



Kernel

WindowsDriver(dxgkrnl.sys)

```
systemCall(NtOpenCompositionSurfaceSectionInfo)
```

```
mov rax, ourFunctionAddress  
jmp rax
```

Our shell code

OurDriver.sys

```
ourFunction(instruction){  
  if (instruction==READ)  
    readMem()  
  else if (instruction==WRITE)  
    writeMem()  
};
```



Till now

A basic hello world driver

Communication = Hooking

What else our driver needs to do?



What makes a Kernel Cheat Driver?

1. System Call Addr.

We need the address of system module & function in order to place our hook

2. Hooking

Placing our shellcode into hook function to jump to our driver in kernel

3. Hook Handler

Handler that handles the instructions from User mode, executes and response back.

4. Clearing Traces

Clearing our loaded driver traces from PiDDBCacheTable & MmUnloadedDrivers.



Detection vectors while using hook

1. Hook Function

Almost all public system calls are sigged by AntiCheats ex:
NtOpenCompositionSurfaceSectionInfo.
Find your own system call that you can hook to.

2. Shell Code

`mov rax,xxx & jmp rax` is a classic hook shell code which is well known and sigged by AntiCheats. Create your own Shell Code that prefers mid function hooking.

```
PVOID* function = reinterpret_cast<PVOID*>(get_system_module_export("\\SystemRoot\\System32\\drivers\\dxgkrnl.sys",  
"NtOpenCompositionSurfaceSectionInfo")); // NtOpenCompositionSurfaceSectionInfo  
  
if (!function)  
    return false;  
  
BYTE orig[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 }; //0x00, 0x00, 0x00, 0x00  
  
BYTE shell_code[] = { 0x48, 0xB8 }; // mov rax, xxx // 0x48 - 0xB8  
BYTE shell_code_end[] = { 0xFF, 0xE0 }; // jmp rax // 0xFF - 0xE0
```



4. Loading Driver

Test Mode

Pay to load

Exploit

Only good for testing our driver

Get your driver signed by Microsoft by paying.

Exploit official signed driver to load our driver.

Doesn't work with Modern Anti-Cheats

Can be revoked easily if gets reported

Easy pzzz



KDMapper

<https://github.com/TheCruZ/kdmapper>

Exploits [iqvw64e.sys Intel driver](#) CVE-2015-229 to manually map non-signed drivers in kernel memory.

Automatically clears major Traces such as [MmUnloadedDrivers](#), [PiDDBCacheTable](#) & [g_KernelHashBucketList](#) but not all like [PoolBigPageTable](#).

Loading driver is easy: `kdmapper.exe driver.sys`



More Vuln Drivers?

<https://github.com/hfiref0x/KDU>

<https://github.com/eclypsium/Screwed-Drivers>

<https://guidedhacking.com/threads/vulnerable-kernel-drivers-for-exploitation.15979/>

<https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=driver>

Exploit drivers on your own



5. Creating User Mode

1. Call Hooked Function

Call the hooked function or system call with our shell code

2. Prepare Instruction

Construct the instruction according to your needs for sending to the hooked function

3. Handle the cheat logic

Aimbot, ESP, etc.

#HITB2023AMS

<https://conference.hitb.org/>



Kernel WallHack Showcase

#HITB2023AMS

<https://conference.hitb.org/>



Moving onto Ext. Hardware Cheat

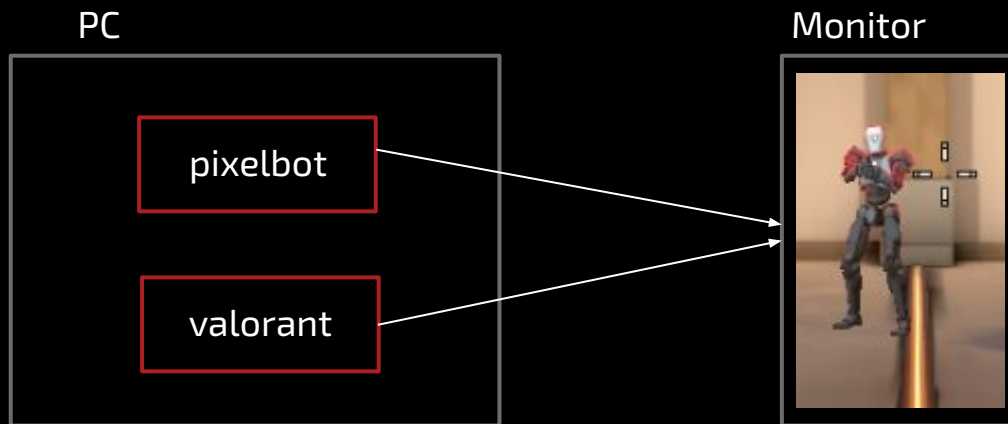
Pixelbots

Color based aimbots



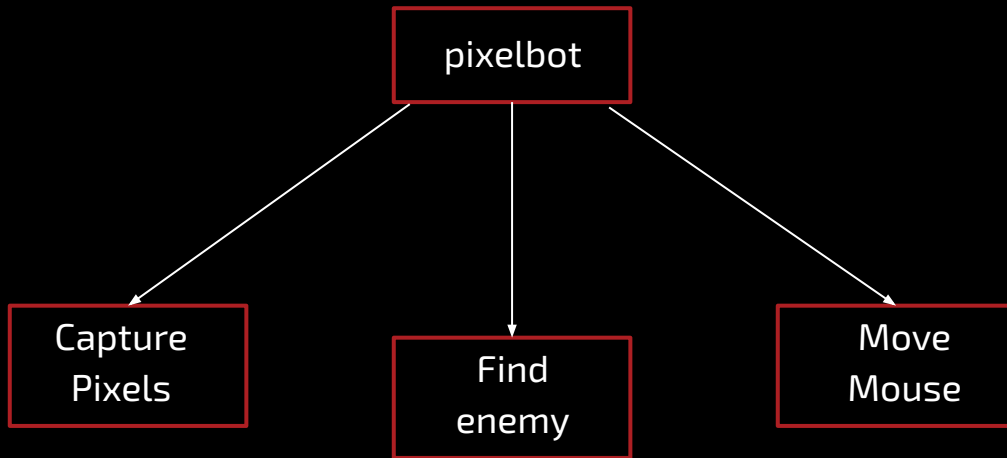


How do these work?





How do these work?





Benefits?

Very hard to detect

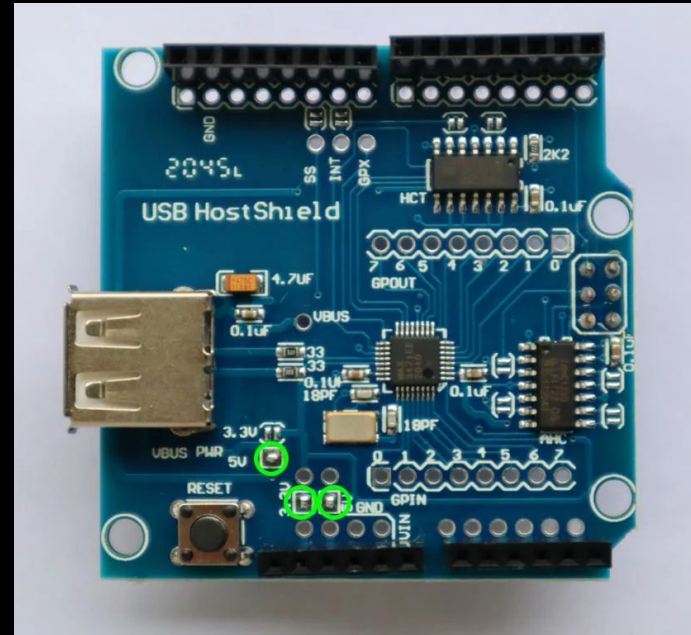
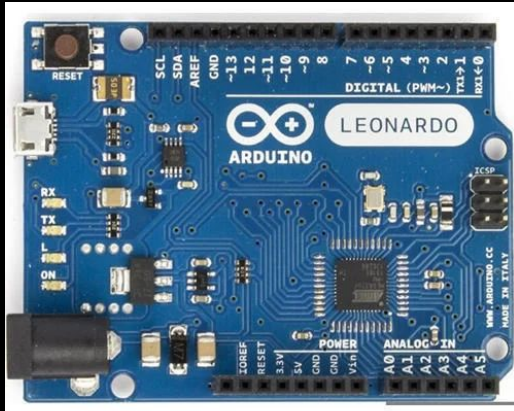
Easy & Works on almost all games

Good code = Good aimbot

Tools required

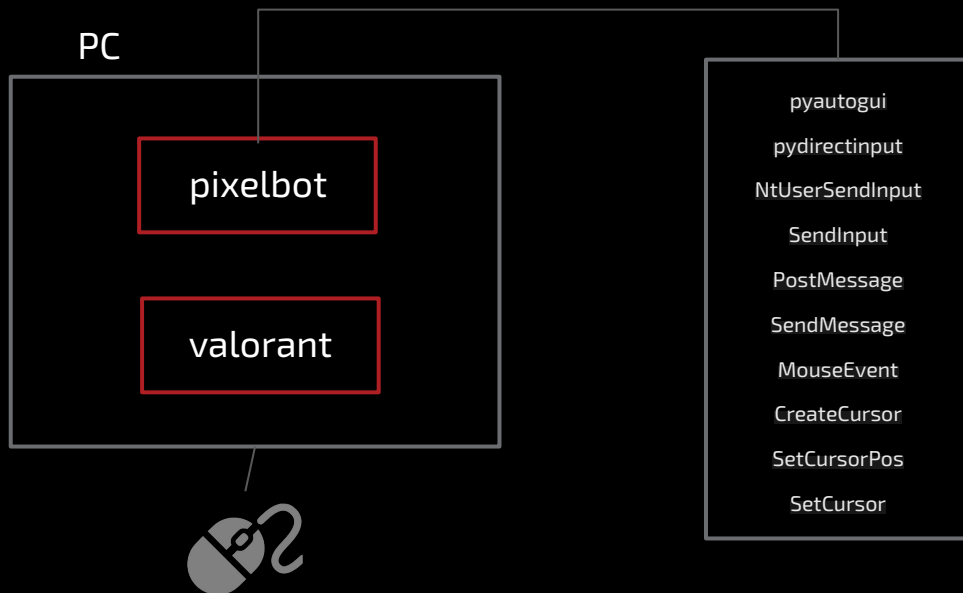
Arduino Leonardo

USB host shield soldered



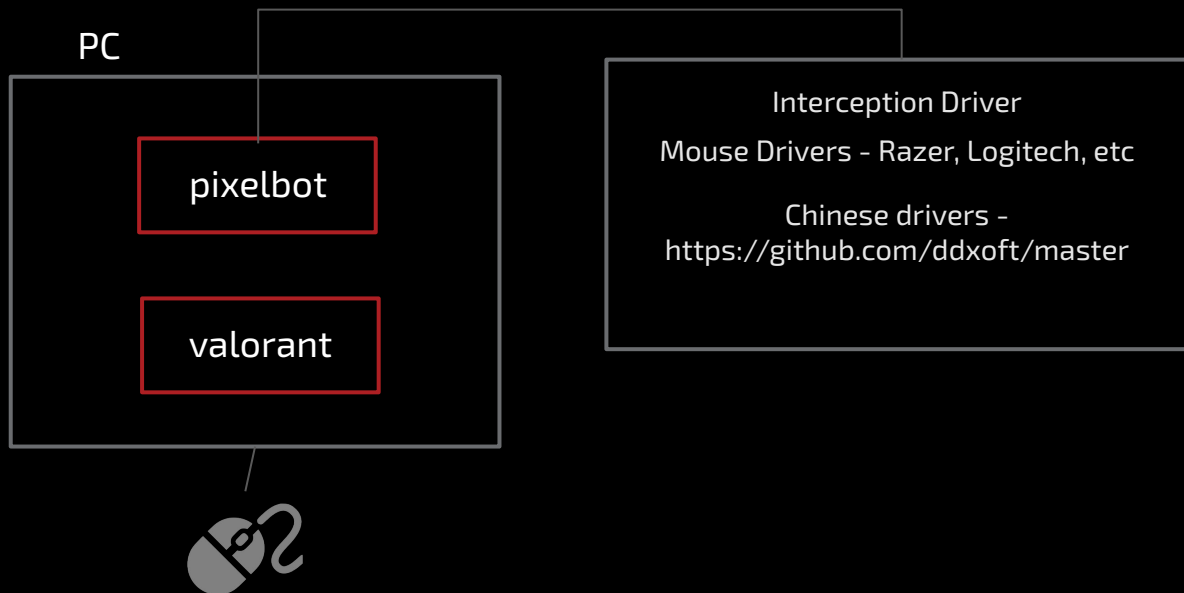


Pixelbots Gen-1 Mouse Mov.



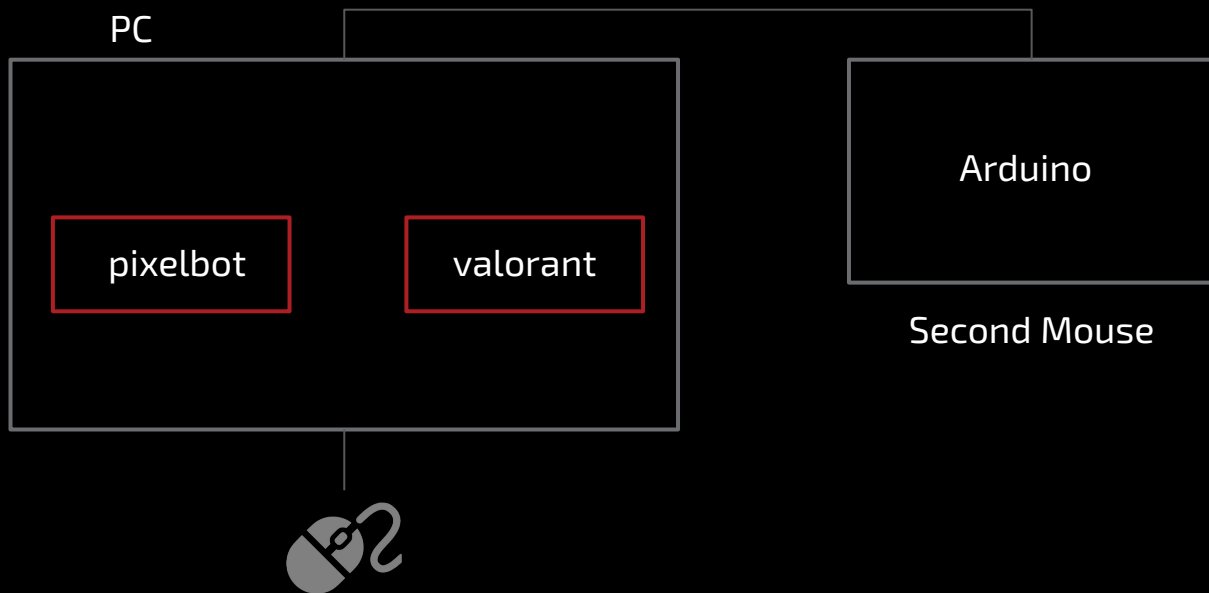


Pixelbots Gen-2 Mouse Mov.





Pixelbots Gen-3 Mouse Mov.





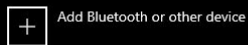
Home

Find a setting

Devices

- Bluetooth & other devices
- Printers & scanners
- Mouse
- Touchpad
- Typing
- Pen & Windows Ink

Bluetooth & other devices



Bluetooth
 Off

Mouse, keyboard, & pen

Gaming KB

Gaming Mouse

Razer Viper Mini

Devices and Printers

Control Panel > Hardware and Sound > Devices and Printers

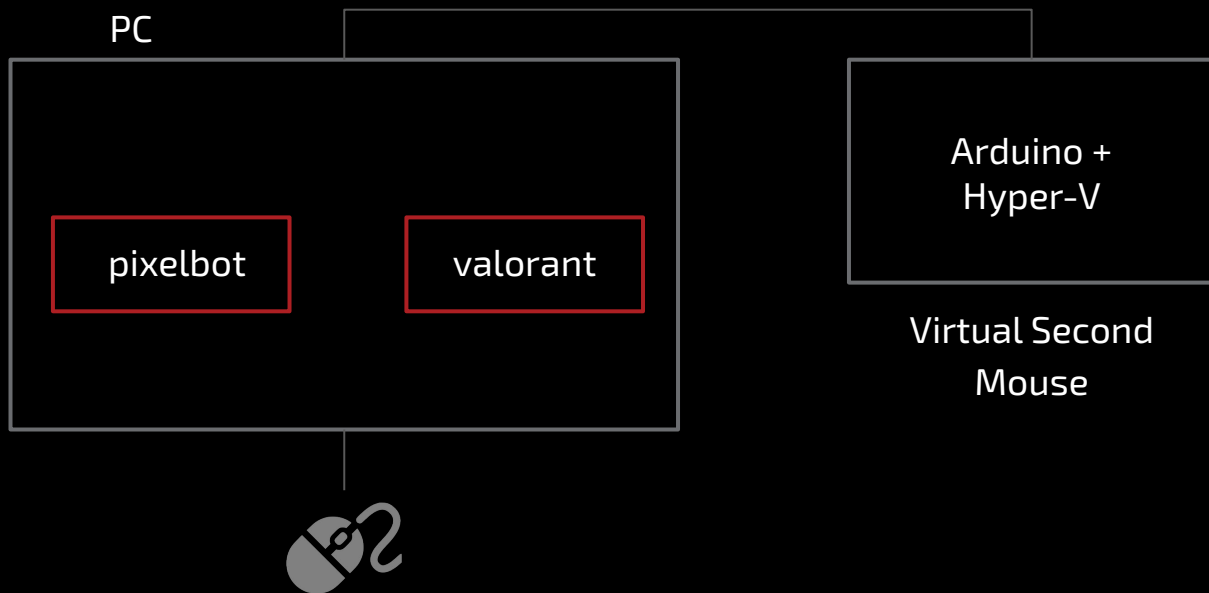
Add a device Add a printer

Devices (5)

- BenQ EX2510
- DESKTOP-~~XXXXXXXXXX~~
- Gaming KB
- Gaming Mouse
- Razer Viper Mini

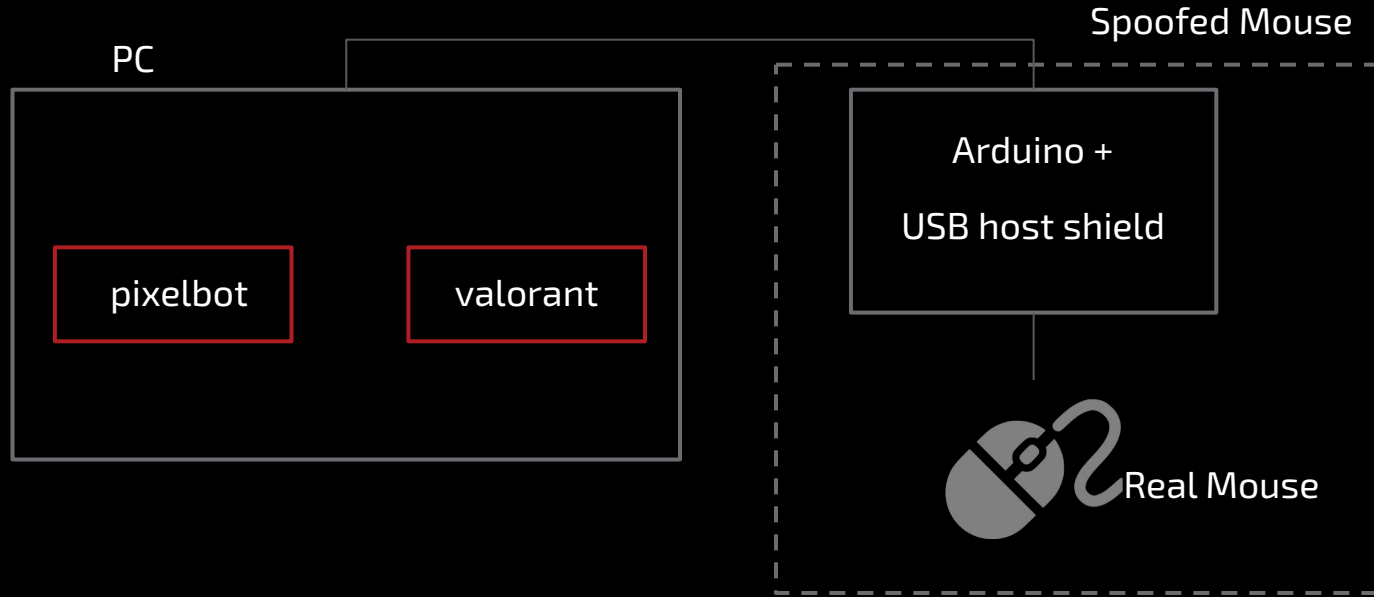


Pixelbots Gen-4 Mouse Mov.





Pixelbots Gen-5 Mouse Mov.





Settings

Home

Find a setting

Devices

- Bluetooth & other devices
- Printers & scanners
- Mouse
- Touchpad
- Typing

Bluetooth & other devices

+ Add Bluetooth or other device

Bluetooth
 Off

Mouse, keyboard, & pen

Gaming KB

Razer Viper Mini

Devices and Printers

Control Panel > Hardware and Sound > Devices and Printers

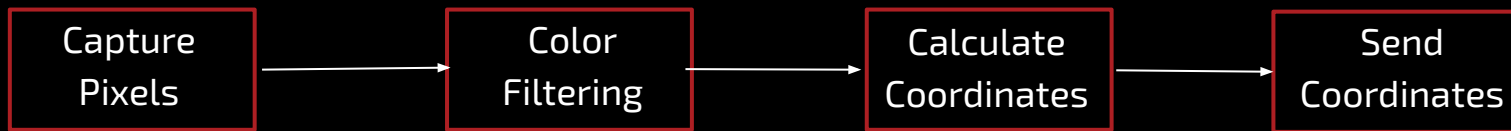
Add a device Add a printer

Devices (4)

- BenQ EX2510
- DESKTOP-~~XXXXXXXXXX~~
- Gaming KB
- Razer Viper Mini



Pixelbot Code



Challenge

Actually aim on enemy and not other things with similar color





Finding Color Range using OpenCV





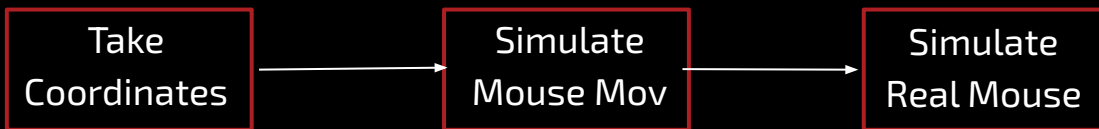
Finding Color Range using OpenCV

```
1 import cv2
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 image = cv2.imread('media/astra.jpg')
6
7 # Converting the image to hsv
8 hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
9
10 # define range of purple color in HSV
11 lower_purple = np.array([160,50,50])
12 upper_purple = np.array([180,255,255])
13
14 # Threshold the HSV image using inRange function to get only purple colors
15 mask = cv2.inRange(hsv, lower_purple, upper_purple)
16
17 plt.figure(figsize=[13,13])
18 plt.subplot(121);plt.imshow(image[:,:,:-1]);plt.title("Original Image",fontdict={'fontsize': 25});plt.axis('off');
19 plt.subplot(122);plt.imshow(mask, cmap='gray');plt.title("Mask of purple Color",fontdict={'fontsize': 25});plt.axis('off');
20 |
```

<https://cvexplained.wordpress.com/2020/04/28/color-detection-hsv/>



Arduino code





Different arduino communication methods

Serial Comm

Web server

Wireless transmitter



Serial getting monitored by vgc

Still u can hide the coordinates inside garbage

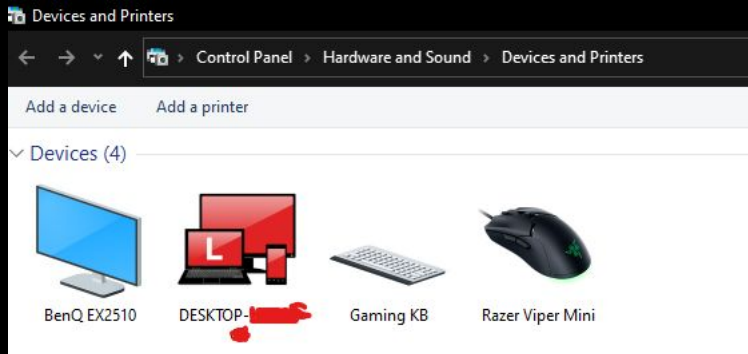
7:47:...	vgc.exe	9704	RegQueryKey	HKLM
7:47:...	vgc.exe	9704	RegOpenKey	HKLM\Software\Microsoft\COM3
7:47:...	vgc.exe	9704	RegSetInfoKey	HKLM\SOFTWARE\Microsoft\COM3
7:47:...	vgc.exe	9704	RegQueryValue	HKLM\SOFTWARE\Microsoft\COM3\Com+Enabled
7:47:...	vgc.exe	9704	RegCloseKey	HKLM\SOFTWARE\Microsoft\COM3
7:47:...	vgc.exe	9704	QueryNameInfo	C:\Windows\System32\wbem\wbemapi.dll

Spoofing arduino

C:\Program Files

(x86)\Arduino\hardware\arduino\avr\boards.txt

<https://the-sz.com/products/usbid/index.php>



```
#####
```

```
leonardo.name=Arduino Leonardo
leonardo.vid.0=0x2341
leonardo.pid.0=0x0036
leonardo.vid.1=0x2341
leonardo.pid.1=0x8036
leonardo.vid.2=0x2A03
leonardo.pid.2=0x0036
leonardo.vid.3=0x2A03
leonardo.pid.3=0x8036
```

```
leonardo.upload.tool=avrdude
leonardo.upload.protocol=avr109
leonardo.upload.maximum_size=28672
leonardo.upload.maximum_data_size=2560
leonardo.upload.speed=57600
leonardo.upload.disable_flushing=true
leonardo.upload.use_1200bps_touch=true
leonardo.upload.wait_for_upload_port=true
```

```
leonardo.bootloader.tool=avrdude
leonardo.bootloader.low_fuses=0xff
leonardo.bootloader.high_fuses=0xd8
leonardo.bootloader.extended_fuses=0xcb
leonardo.bootloader.file=caterina/Caterina-Leonardo.hex
leonardo.bootloader.unlock_bits=0x3F
leonardo.bootloader.lock_bits=0x2F
```

```
leonardo.build.mcu=atmega32u4
leonardo.build.f_cpu=1600000L
leonardo.build.vid=0x2341
leonardo.build.pid=0x8036
leonardo.build.usb_product="Arduino Leonardo"
leonardo.build.board=AVR_LEONARDO
leonardo.build.core=arduino
leonardo.build.variant=leonardo
leonardo.build.extra_flags={build.usb_flags}
```

#HITB2023AMS

<https://conference.hitb.org/>



External Hardware Aimbot Showcase



What valorant can do?

Remove outlines?

GAMEPLAY SYSTEMS UPDATES


- / Added the ability to hide Agent outlines and fresnel (the color outline on Agents)
 - / Go to Settings >> General >> Under "Other", there is an option to toggle "Hide Outlines and Fresnel".

YOLO AI

<https://www.unknowncheats.me/forum/general-programming-and-reversing/485725-guide-ai-aimbot.html>

14th January 2022, 06:54 AM

gr3gthepilot
★★★★★



Join Date: Jan 2021
Posts: 103
Reputation: 4845
Rep Power: 60
Points: 12,219, Level: 14
Level up! 2%, 1,281 Points needed
Activity: 2.4%

Last Achievements:

Guide to AI aimbot

Guide to AI aimbot

UNDER CONSTRUCTION

Guide to making your own AI aimbot

Why?
I decided to make this guide because of countless people asking me for help with making their aimbot that uses AI instead of memory reading.

What?
I will show you how to make an AI aimbot using YOLOv4 and OpenCV library (train and code the program).

How?
<https://memegenerator.net/imo/instances/66659307.jpg>

Process of making it:
The making of your aimbot I see as a two-step process:

- paste together the code that will run the AI and aim for you.
- create a dataset that you will use to train your AI to recognise whatever the fuck you want it to recognise, in most cases this will probably be the enemy player. - This is more time consuming

A few expressions:
weights / Neural network (NN) / AI / model / the stuff that actually does the magic

Making Aimbot:

So how do you make the aimbot?
I will go off of my previously posted cheats. [YOLO Aim Augmentation v2.0](#), [Valorant Cheat w Arduino and YoloV5 AI](#)

How it works?

1. we take a screenshot
2. make a blob from it
3. pass the blob through the model



Release

<https://github.com/nahoragg/KernelDriver>

<https://github.com/nahoragg/Arduino>



References

<https://guidedhacking.com/>

<https://www.unknowncheats.me/>

<https://www.youtube.com/c/NullTerminator>

Conclusion

JK



#HITB2023AMS

<https://conference.hitb.org/>



Thank you!

nahoragg@gmail.com

Twitter: nahoragg