

---

## UPDATED

So, I've begun to make some updated ctf stuff. The first big part, deals with heap ctf challs, and can be found here:

<https://github.com/guyinatuxedo/Shogun>

## Nightmare

Nightmare is an intro to binary exploitation / reverse engineering course based around ctf challenges. I call it that because it's a lot of people's nightmare to get hit by weaponized 0 days, which these skills directly translate into doing that type of work (plus it's a really cool song).

### What makes Nightmare different?

It's true there are a lot of resources out there to learn binary exploitation / reverse engineering skills, so what makes this different?

- |   |   |                             |   |  |
|---|---|-----------------------------|---|--|
| 1 | * | Amount of Content           | - | There is a large amount of content in <b>this</b> course (currently over 90 challenges), laid out in a linear fashion.   |
| 2 |   |                             |   |  |
| 3 | * | Well Documented Write Ups   | - | Each challenge comes with a well documented writeup explaining how to go from being handed the binary to doing the exploit dev.  |
| 4 |   |                             |   |  |
| 5 | * | Multiple Problems per Topic | - | Most modules have multiple different challenges. This way you can use one to learn how the attack works, and then apply it to the others. Also different iterations of the problem will have knowledge needed to solve it. |
| 6 |   |                             |   |  |
| 7 | * | Using all open source tools | - | All the tools used here are free and open sourced. No IDA torrent needed.  |
| 8 |   |                             |   |  |
| 9 | * | A Place to Ask Questions    | - | So <b>if</b> you have a problem that you've been working for days and can't get anywhere (and google isn't helping).   |

I have found that resources that have many of these things to be few and far between. As a result it can make learning these skills difficult since you don't really know what to learn, or how to learn it. This is essentially my attempt to help fix some of those problems. ## Static Site

If you want, there is a static github pages site which people say looks better: <https://guyinatuxedo.github.io/>

---

## Github

A copy of all of the challenges listed, can be found on the github: <https://github.com/guyinatuxedo/nightmare>

## Special Thanks

Special thanks to these people:

- |   |             |   |   |
|---|-------------|---|---|
| 1 | noopnoop    | - | For dealing with me                               |
| 2 | digitalcold | - | For showing me how good nightmare could look with |
|   | mdbook      |   |   |
| 3 | you nerds   | - | For looking at <b>this</b>                        |

## Discord

If you get stuck on something for hours on end and google can't answer your question, try asking in the discord (or if you just feel like talking about cool security things). Here is a link to it <https://discord.gg/p5E3VZF>

Also if you notice any typos or mistakes, feel free to mention it in the Discord. With how much content is here, there is bound to be at least one.

## Index

Here is the index for all of the content in this course. Feel free to go through the whole thing, or only parts of it (don't let me tell you how to live your life). For the order that you do the challenges in a module, I would recommend starting with the first.

## Intro Departure

### 0.) Intro to the Project

### 1.) Intro to Assembly

- Intro to assembly
- Sample assembly reverse challs

---

## **2.) Intro to Tooling**

- gdb-gef
- pwntools
- ghidra

## **3.) Beginner RE**

- pico18\_strings
- helithumper\_re
- csaw18\_tourofx86pt1
- csaw19\_beleaf

## **Stack pt 0 Stack Tendencies**

### **4.) Buffer Overflow of Variables**

- Csaw18/boi
- TokyoWesterns17/just\_do\_it
- Tamu19\_pwn1

### **5.) Buffer Overflow Call Function**

- Csaw18\_getit
- Tu17\_vulnchat
- Csaw16\_warmup

#### **5.1) aslr/pie intro**

- quick aslr/pie explanation

### **6.) Buffer Overflow Call Shellcode**

- Tamu19\_pwn3
- Csaw17\_pilot

- 
- Tu18\_shelleasy

### **6.1) nx intro**

- nx explanation

### **7.) ROP Chain Statically compiled**

- dcquals19\_speedrun1
- bkp16\_simplecalc
- dcquals16\_feedme

### **7.1) stack canary intro**

- stack canary introduction

### **7.2) relro intro**

- relro introduction

### **8.) ROP Dynamically Compiled**

- csaw17\_svc
- fb19\_overflow
- hs19\_storytime
- csaw19\_babyboi
- utc19\_shellme

### **General pt 0 Stardust Challenges**

### **9.) Bad Seed**

- h3\_time
- hsctf19\_tuxtalkshow
- sunshinectf17\_prepared

---

## **10.) Format strings**

- backdoor17\_bbpwn
- twesterns16\_greeting
- pico\_echo
- watevr19\_betstar

## **11.) Index Array**

- dcquals16\_xkcd
- sawmpctf19\_dreamheaps
- sunshinectf2017\_alternativesolution

## **12.) Z3**

- tokyowesterns17\_revrevrev
- tuctf\_future
- hsctf19\_abyte

## **13.) Angr**

- securityfest\_fairlight
- plaid19\_icancount
- defcamp15\_r100

## **Stack pt 1 Return to Stack, truly a perfect game**

## **14.) Ret2system**

- asis17\_marymorton
- hxp18\_poorcanary
- tu\_guestbook

---

### **15.) Partial Overwrite**

- Tu17\_vulnchat2
- Tamu19\_pwn2
- hacklu15\_stackstuff

### **16.) SROP**

- backdoorctf\_funsignals
- inctf17\_stupiddrop
- swamp19\_syscaller
- csaw19\_smallboi

### **17.) Stack Pivot / Partial Overwrite**

- defconquals19\_speedrun4
- insomnihack18\_onewrite
- xctf16\_b0verf10w

### **18.) Ret2Csu / Ret2dl**

- ropemporium\_ret2csu
- 0ctf 2018 babystack

## **General pt 1 Armstrong challenges**

### **19.) Shellcoding pt 1**

- defconquals19\_s3
- Csaw18\_shellpointcode
- defconquals19\_s6

### **20.) Patching/Jumping**

- dcquals18\_elfcrumble

- 
- plaid19\_plaid\_part\_planning\_III

- csaw16\_gametime

## **21.) .NET Reversing**

- csaw13\_dotnet
- csaw13\_bikinibonanza
- whitehat18\_re06

## **22.) Movfuscation**

- sawmpctf19\_future
- asis18quals\_babyc
- other\_movfused

## **23.) Custom Architectures**

- h3\_challenge0
- h3\_challenge1
- h3\_challenge2
- h3\_challenge3

## **Heap Pt 0 rip Angel Beats**

### **24.) Basic Heap overflow**

- protostar\_heap1
- protostar\_heap0
- protostar\_heap2

### **25.) Intro to heap exploitation / binning**

- explanation

---

## **26.) Heap Grooming**

- explanation
- swamp19\_heapgolf
- pico\_areyouroot

## **27.) Edit Freed Chunk (pure explanation)**

- Use After Free
- Double Free
- Null Byte Heap Consolidation

## **28.) Fastbin Attack**

- explanation
- 0ctf18\_babyheap
- csaw17\_auir

## **29.) tcache**

- explanation
- dcquals19\_babyheap
- plaid19\_cpp

## **30.) unlink**

- explanation
- hitcon14\_stkof
- zctf16\_note

## **31.) Unsorted Bin Attack**

- explanation
- hitcon\_magicheap



- 
- 0ctf16\_zer0storage

### **32.) Large Bin Attack**

- largebin0\_explanation
- largebin1\_explanation

### **33.) Custom Malloc**

- csawquals17\_minesweeper
- csawquals18\_AliensVSSamurai
- csawquals19\_traveller

## **General Pt 2 Generic Isekai #367**

### **34.) Qemu / Emulated Targets**

- csaw18\_tour\_of\_x86\_pt\_2
- csaw15\_hackingtime
- csaw17\_realism

### **35.) Integer Exploitation**

- puzzle
- int\_overflow\_post
- signed\_unsigned\_int\_expl

### **36.) Obfuscated Reversing**

- csaw15\_wyvern
- csaw17\_prophecy
- bkp16\_unholy

### **37.) FS Exploitation**

- swamp19\_badfile

---

### **38.) Grab Bag**

- csaw18\_doubletrouble
- hackim19\_shop
- unit\_vars\_expl
- csaw19\_gibberish

### **Heap pt 1 heap x heap**

### **39.) House of Spirit**

- explanation
- hacklu14\_oreo

### **40.) House of Lore**

- explanation

### **41.) House of Force**

- explanation
- bkp16\_cookbook

### **42.) House of Einherjar**

- explanation

### **43.) House of Orange**

- explanation

### **44.) More tcache**

- csaw19\_poppingCaps0
- csaw19\_poppingCaps1

### **45.) Automatic Exploit Generation**

- csaw20\_rop

---

## **Ending Documentation**

- References
- What's next