# Software Security

Sai Venkata Krishnan V
IIT Madras

Institute of Smart Structures and Systems

Centre for Hardware
Security Entrepreneurship
Research & Development

# $ whoami

- PhD scholar at IIT Madras
- https://saivk.github.io/
- Area of research: programming languages security, binary analysis

# About you

# Course Logistics

- Two classes per week
  - Friday & Saturday
- 3 Assignments flexible
- Two quizzes google form / short-answer (50mins)
- All assignments and quizzes are team based
  - 4 - 5 per team
- 1 week break in between

# Course Agenda

- Week 1: Program => Process
- Week 2: Introduction to GDB and x86 Assembly
- Week 3: Buffer overflow exploit
- Week 4: Ret-2-Libc
- Week 5: ROP Exploit
- Week 6: ASLR
- Week 7 & 8: Heap exploits
- Week 9: Format string exploits
- Week 10: Current mitigations and advanced exploits

=> (Process loading into OS / Binary Format /

Process layout

=> GDB inspect process layout / stack, heap / gdb commands, assembly refreshes

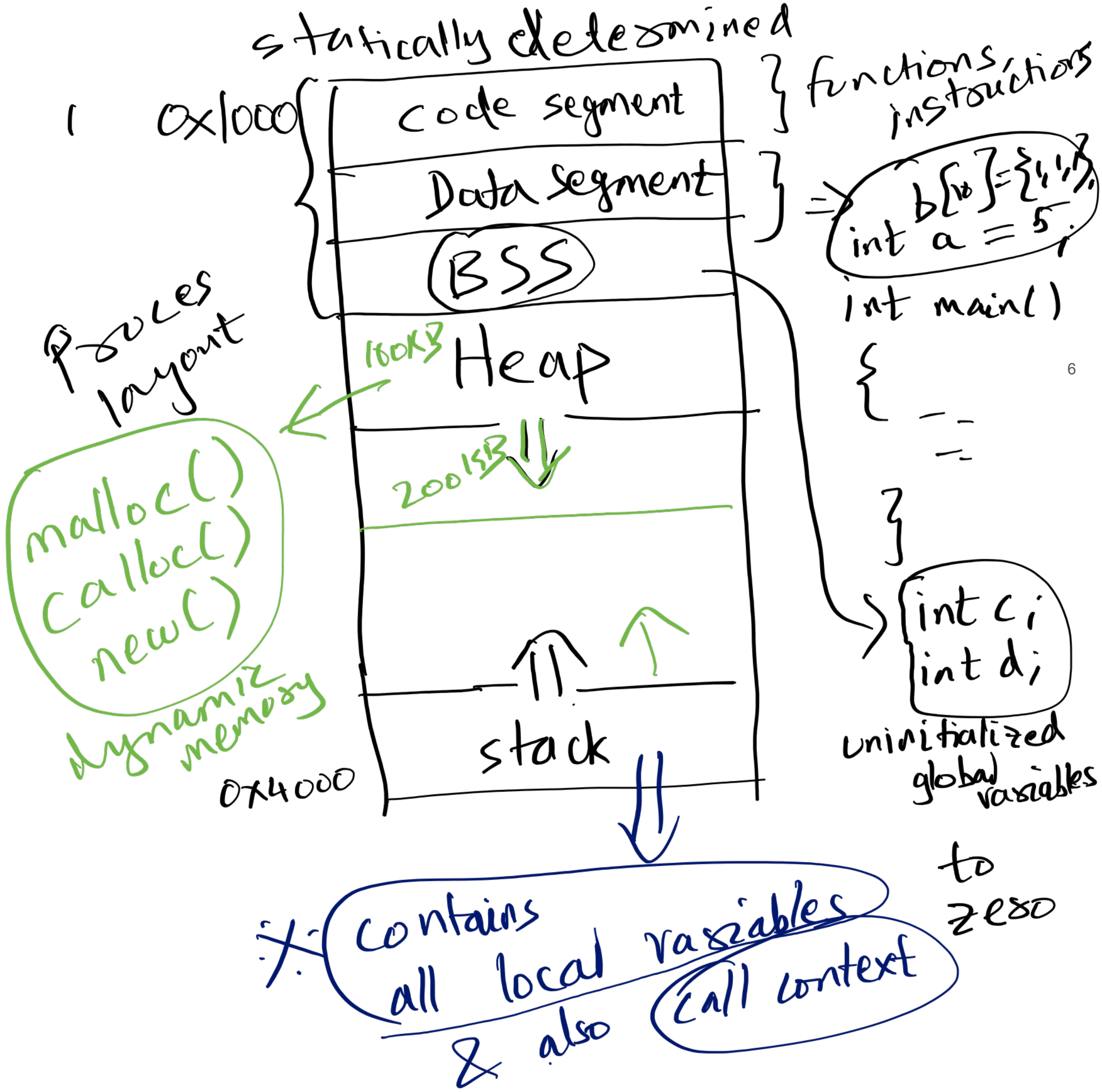
Mitigation

Binary Exploitation

CTF

capture the flag

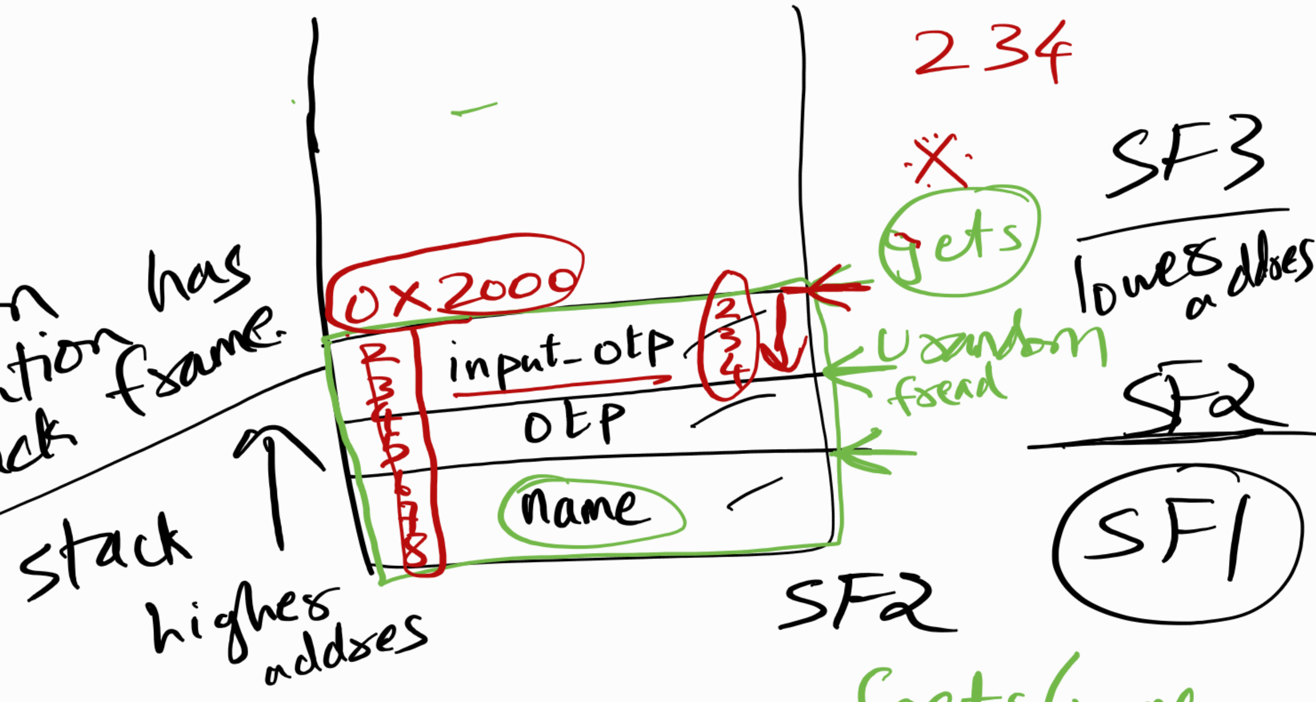
# Sneak Peek


statically determined
1 0x1000
code segment
functions, instructions
Data segment
int b[10]={1,1,};
int a = 5;
BSS
int main()
{
}
int c;
int d;
uninitialized global variables
to zero
Proces layout
100KB
Heap
200KB
malloc()
calloc()
new()
dynamic memory
stack
0x4000
contains all local variables
& also call context

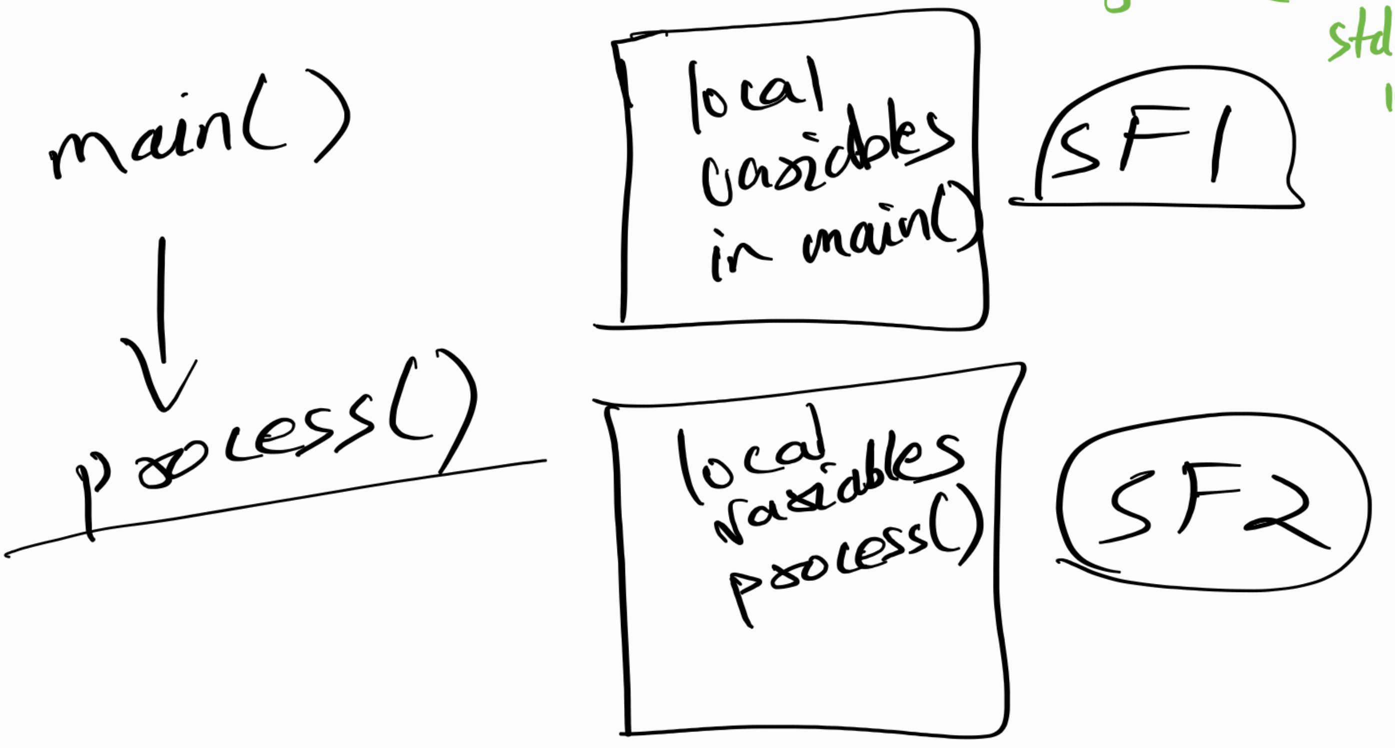
```
int process()
{   char input_otp[3];
    char otp[3];
    char name[10];
```

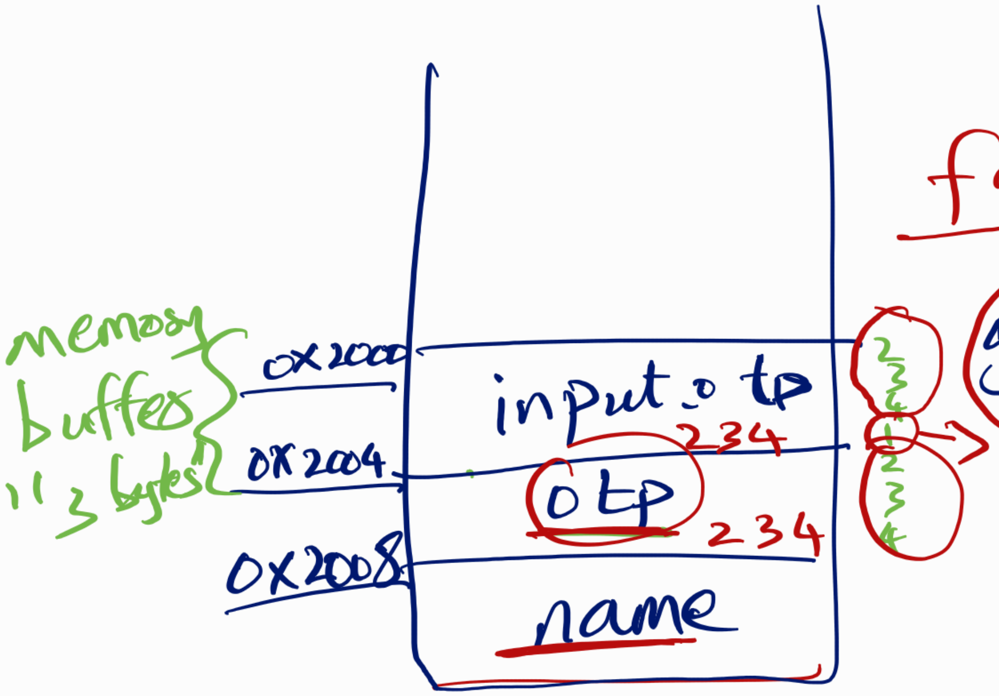
would be allocated in stack region

function has a execution stack frame.



fgets(name, stdin, 10)





fgets

gets()

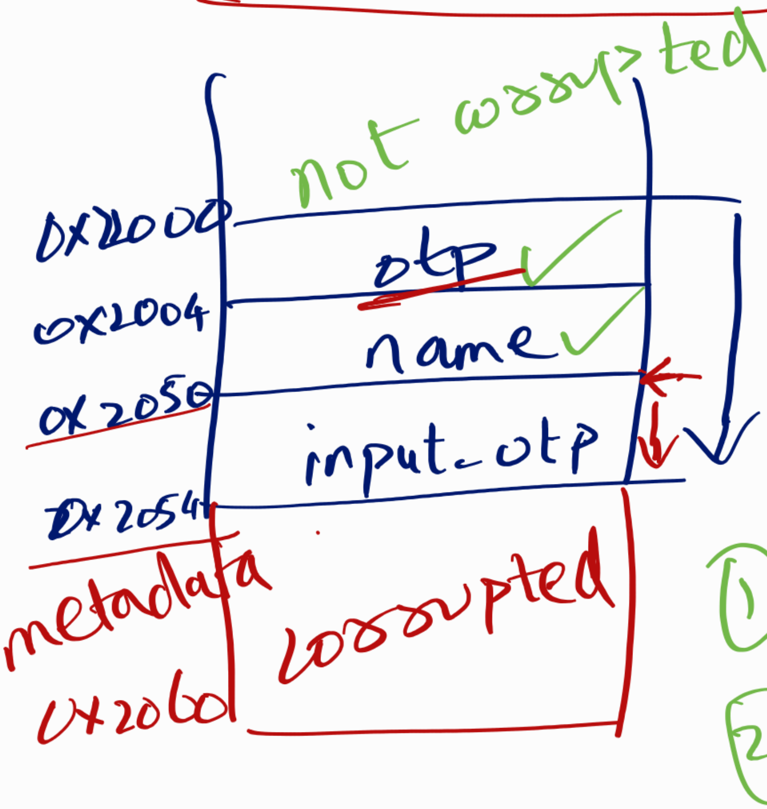
char otp[4]

char input_otp[4]

3 bytes

4 '\0' 1/2

"overflows"



gets(input_otp

① fgets

②