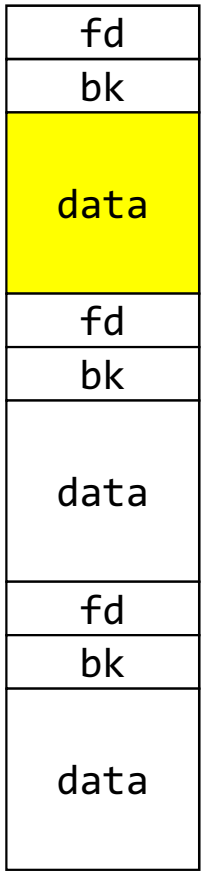
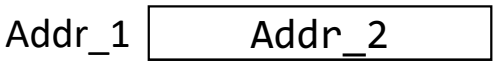
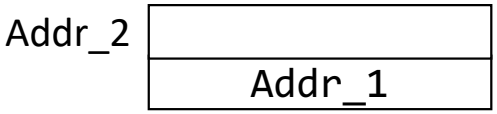



```
A = B->bk
C = B->fd
A->fd = C
C->bk = A
```

```
Addr_1 = B->bk
Addr_2 = B->fd
Addr_1->fd = Addr_2
Addr_2->bk = Addr_1
```

前提：A的data可以泄露，所以B的fd，bk都可以篡改

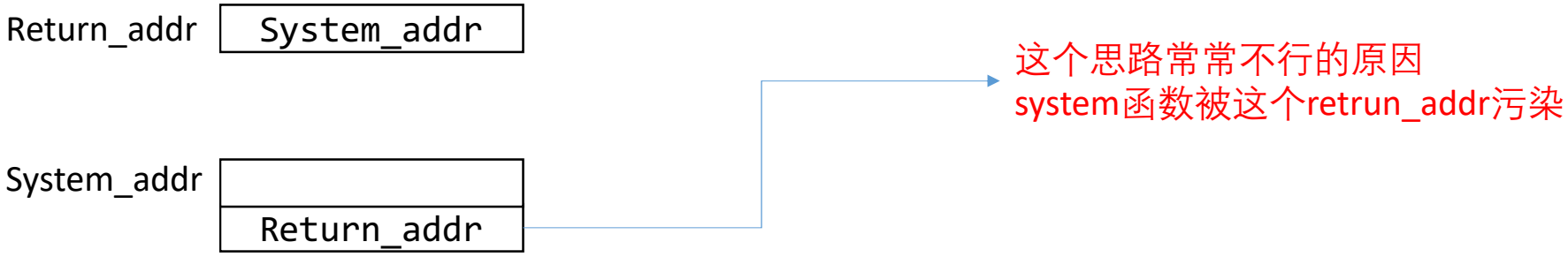




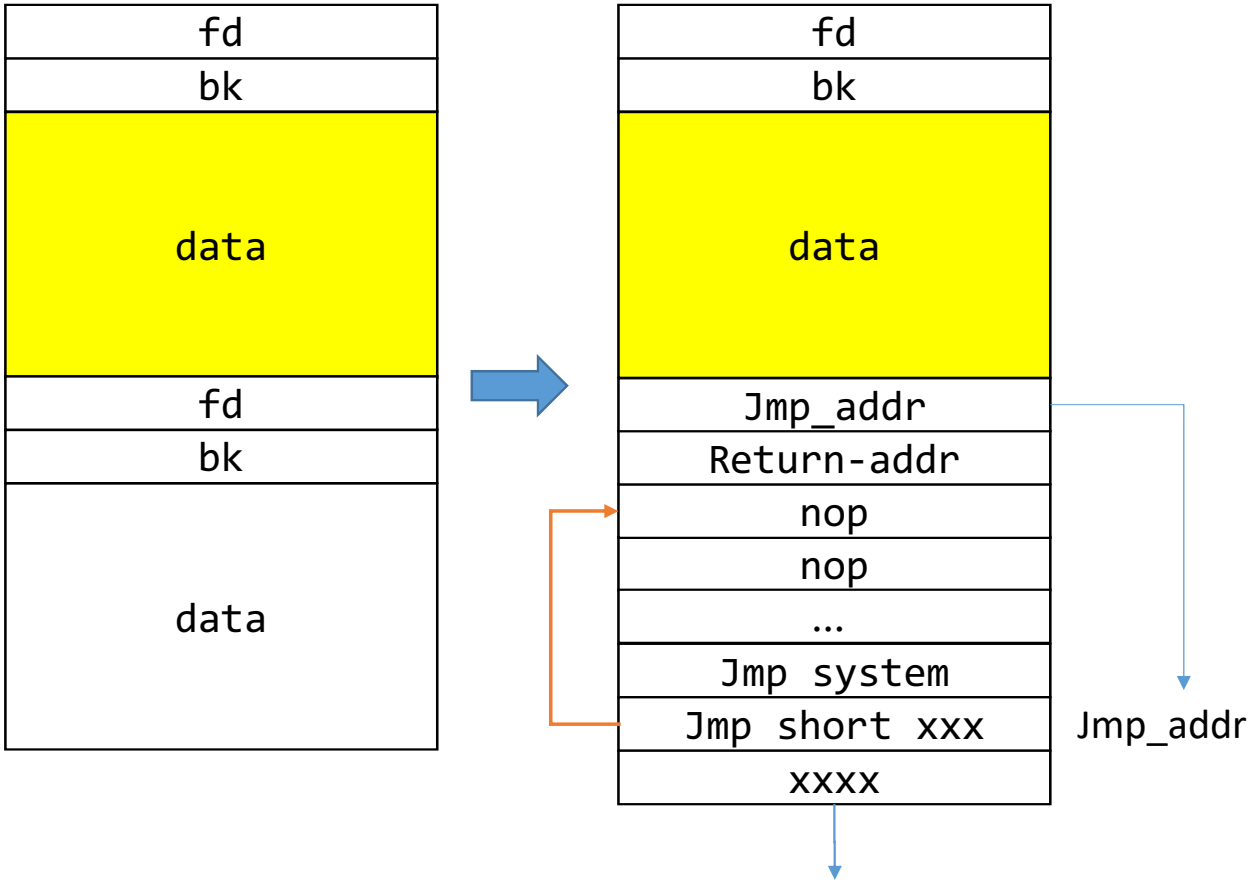
关键点不在于发现，而是如何寻找合适的Addr_1 && Addr_2

思路一

(addr_1, return_addr)--------(addr_2, system_addr)



思路二



这里最后会被改为return-addr，但无关紧要了，这也是这个思路的精髓，利用jmp直接不去执行污染后的东西

执行过程

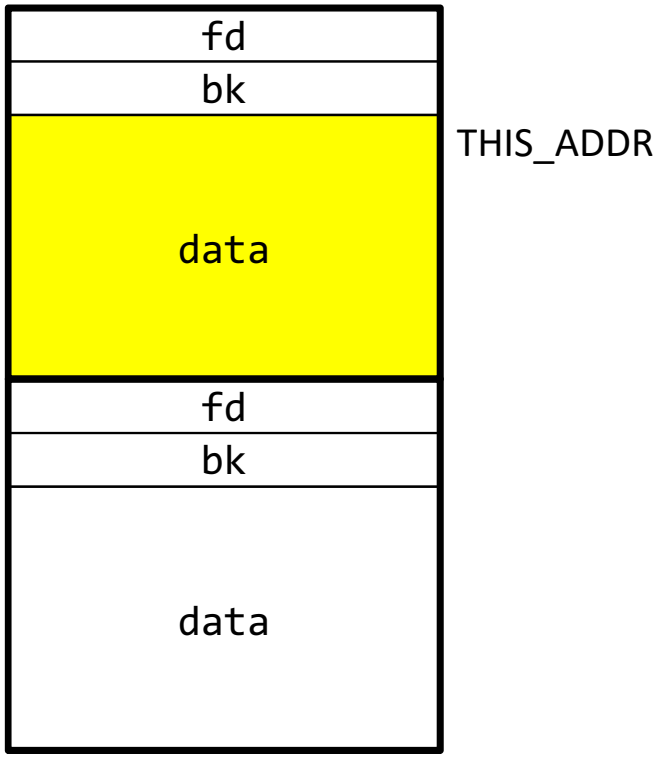
- Return_addr 后程序开始在堆上执行，开始执行位置是Jmp_addr
- 首先执行Jmp short xxx，跳转到nop
- 然后一系列nop，最后跳转到system

这个思路不行的原因
有NX，不可在堆栈上执行指令

思路三

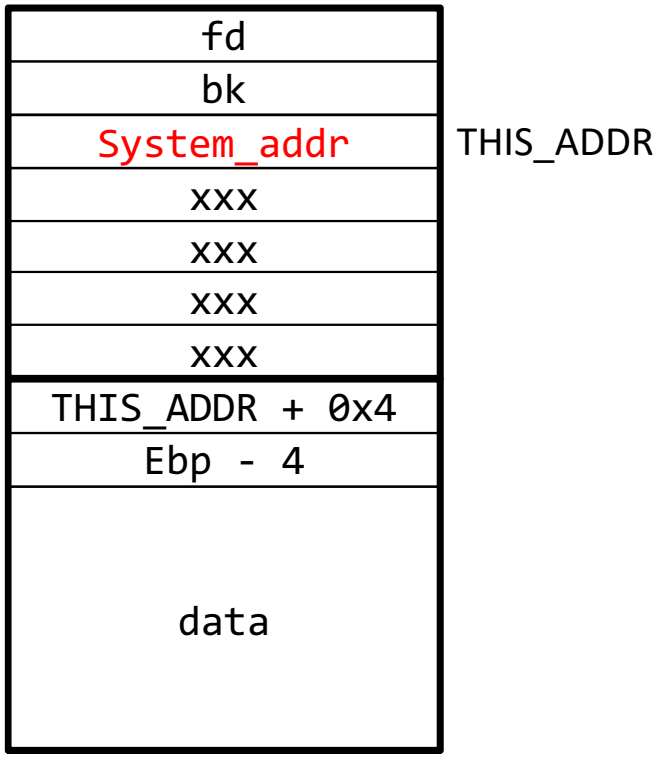
开始填充

```
<+0>   :  lea    ecx,[esp+0x4]
<+4>   :  and    esp,0xfffffff0
<+7>   :  push   DWORD PTR [ecx-0x4]
...
...
<+195> :  call   0x8048504 <unlink>
...
...
<+200> :  add    esp,0x10
<+203> :  mov    eax,0x0
<+208> :  mov    ecx,DWORD PTR [ebp-0x4]
<+211> :  leave
<+212> :  lea    esp,[ecx-0x4]
<+215> :  ret
```



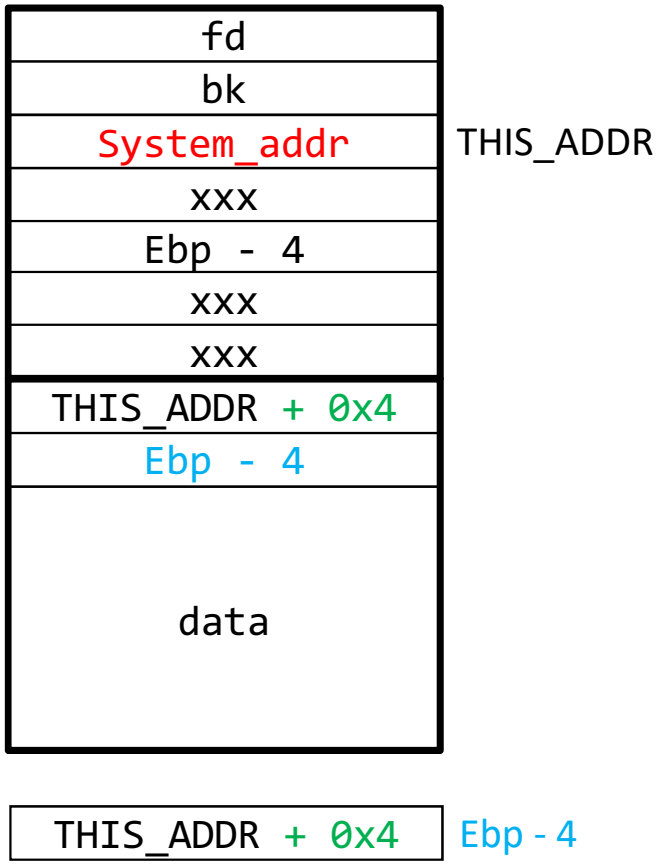
FD->bk=BK; BK->fd=FD;

```
<+0>   :  lea    ecx,[esp+0x4]
<+4>   :  and    esp,0xfffffff0
<+7>   :  push   DWORD PTR [ecx-0x4]
...
...
<+195> :  call   0x8048504 <unlink>
...
...
<+200> :  add    esp,0x10
<+203> :  mov    eax,0x0
<+208> :  mov    ecx,DWORD PTR [ebp-0x4]
<+211> :  leave
<+212> :  lea    esp,[ecx-0x4]
<+215> :  ret
```

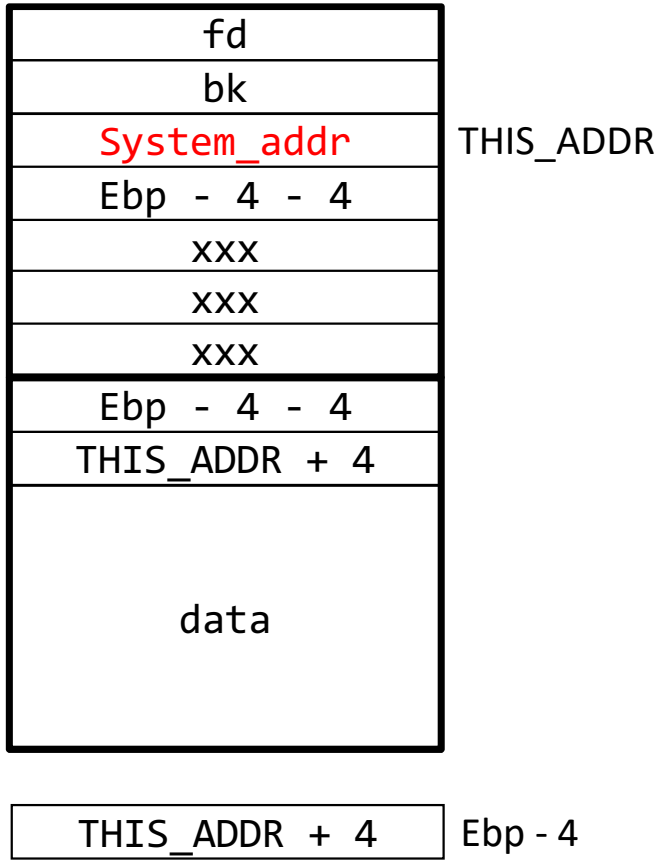


FD->bk=BK; BK->fd=FD;

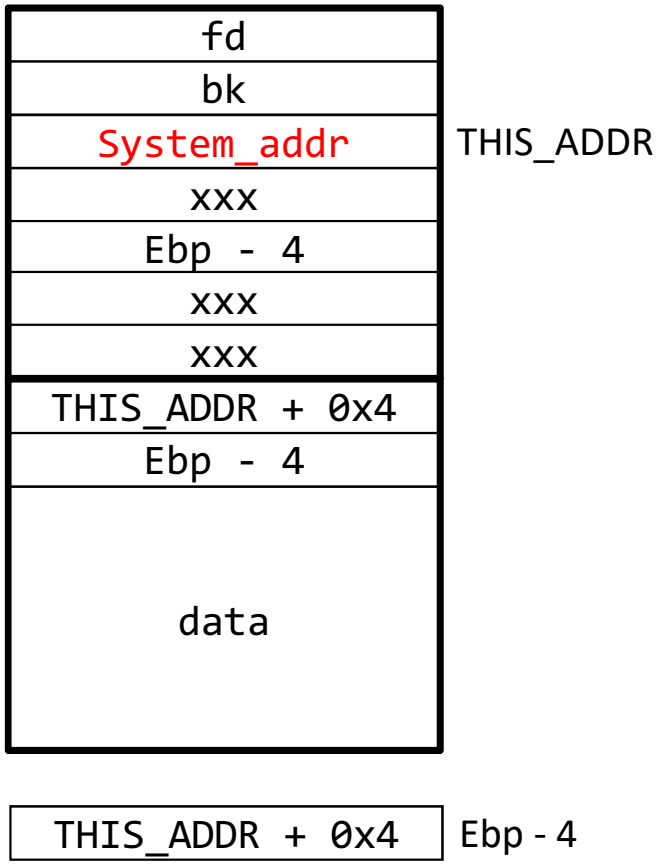
```
<+0>   :  lea    ecx,[esp+0x4]
<+4>   :  and    esp,0xfffffff0
<+7>   :  push   DWORD PTR [ecx-0x4]
...
...
<+195> :  call   0x8048504 <unlink>
...
...
<+200> :  add    esp,0x10
<+203> :  mov    eax,0x0
<+208> :  mov    ecx,DWORD PTR [ebp-0x4]
<+211> :  leave
<+212> :  lea    esp,[ecx-0x4]
<+215> :  ret
```



FD->bk=BK; BK->fd=FD;



两种选择



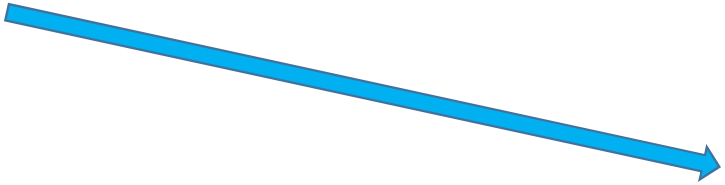
FD->bk=BK; BK->fd=FD;

```
<+0>   :  lea    ecx,[esp+0x4]
<+4>   :  and    esp,0xfffffff0
<+7>   :  push   DWORD PTR [ecx-0x4]
...
...
<+195> :  call   0x8048504 <unlink>
...
...
<+200> :  add    esp,0x10
<+203> :  mov    eax,0x0
<+208> :  mov    ecx,DWORD PTR [ebp-0x4]
<+211> :  leave
<+212> :  lea    esp,[ecx-0x4]
<+215> :  ret
```

[ebp - 0x4] = THIS_ADDR + 0x4

ecx = THIS_ADDR + 0x4

esp = [THIS_ADDR] = system



eip = esp = system