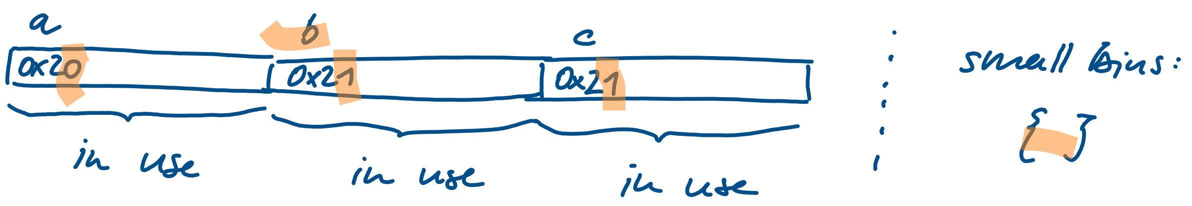
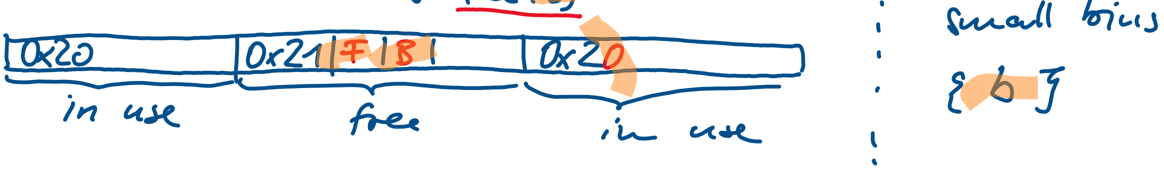
Double free

Dienstag, 30. Juni 2020 16:21

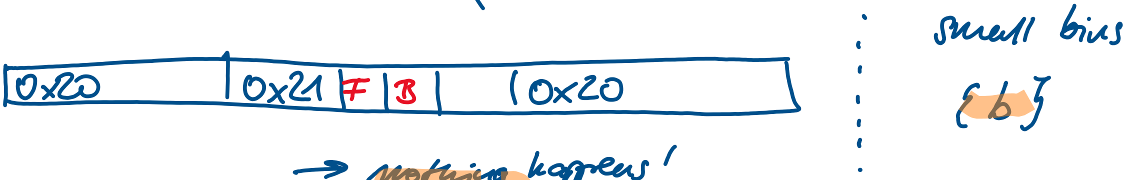
linear memory



↓ free(b)



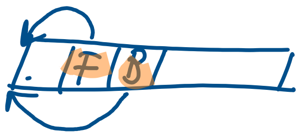
↓ free(b)



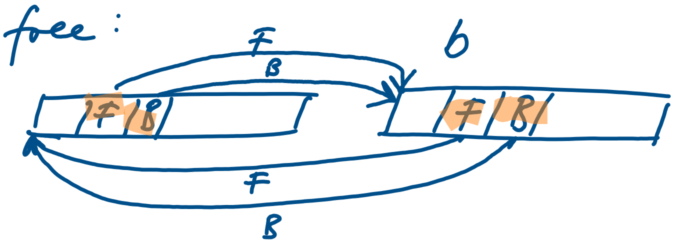
→ nothing happens!

→ next malloc returns b!

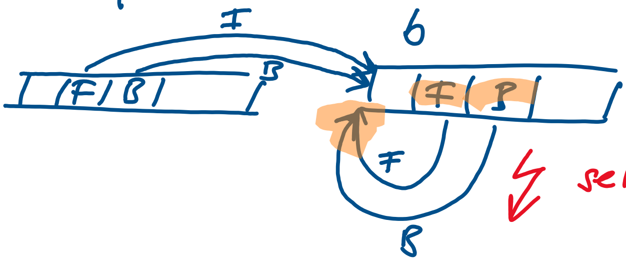
empty bin:



first free:



second free:



↯ self-reference

next malloc:

- unlink b from free list
- return b

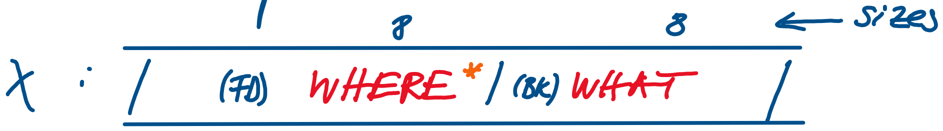
→ forever... every new malloc returns

'b again and again!

→ this also means that the FD and BK pointers are now "user data" (!!!) returned by malloc() ⇒ writeable for us!

exploit:

- free(b), free(b), malloc( )
- write FD and BK pointers to get a write what where condition
- example: X = malloc(...)



→ overwrite FD and BK pointers with strcpy, read, gets, ..

WHERE : GOT, malloc hook, free hook, ...

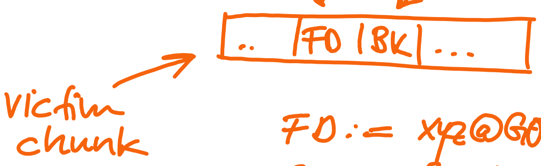
WHAT : address of injected shell code, ROP chain, ..

* slightly different

* remember the unlink macro:

FD = P → fd
BK = P → bk
FD → bk = BK
BK → fd = FD

WHERE WHAT
→ FD → bk = BK
FD + $\boxed{24}$ = BK



FD := xyz@GOT - $\boxed{24}$
BK := &shellcode

xyz can be any function that will

be called later..

full exploit sketch.

```
1) void *p = malloc(SIZE);
2) free(p);
3) free(p);
4) void *q = malloc(SIZE);
   // set first 8 bytes of q
   // set second 8bytes of q
   // e.g. overwrite puts@GOT
   // with address of shellcode
5) Void *r = malloc(SIZE) // this triggers
                          the unlink
                          overwrite
6) puts(...);  // executes the
                  Shellcode
       010110101010101110
```