# **eBFP指令集**

   eBPF 定义了一套特有的RISC 指令集，包含数据存取(ST/LD), 算术/逻辑运算ALU(add,sub,mul,mod...), 跳转(JMP,RET)等，

在学习eBPF指令集时，可以把eBPF当做与x86/arm汇编平级的一门语言。

## **1 eBPF寄存器**

eBPF有11个 64 bit 寄存器(r0-r11)和一个PC寄存器，其中

r0: 保存返回值

R1-R5 : 用于传递参数

R6-R9: 通用寄存器，在call调用时进行压栈保护

R10: 栈帧寄存器

eBPF寄存器可以与X86寄存器一一对应

    R0 - rax   R1 - rdi    R2 - rsi     R3 - rdi  
    R4 - rcx   R5 - r8     R6 - rbx  R7 - r13  
    R8 - r14   R9 - r15   R10 - rbp

## **2. eBPF指令编码**

eBPF 指令编码由五部分组成，一共占用8Byte  
IMG_259

imm:32  32位立即数  
off:16    16位偏移  
src\_reg  源寄存器  
dst\_reg  目的寄存器  
op 8位操作码，又继续细分为

### **2.1 ALU和JMP指令**

op 8bit可以分为三部分:

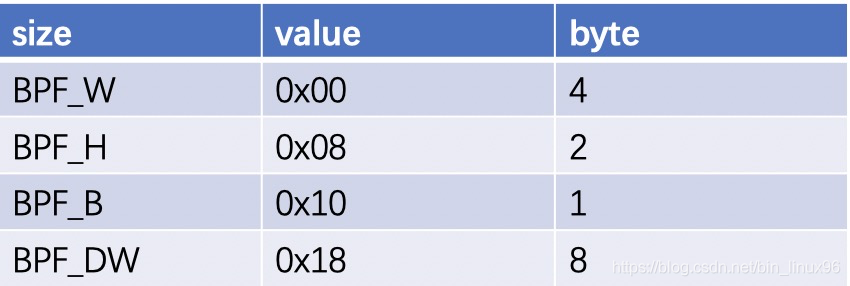
IMG_260  
instruction class(3bit): 指令所属的类别  
source(1bit): 表示源操作数是用寄存器还是立即数  
BPF\_X:使用src\_reg作为源操作数  
BPF\_K:使用imm32作为源操作数  
operation code(4bit):操作码(2.3节)

### **2.2 load和store指令**

op 8bit也可以分为三部分

IMG_261

size: 表示load/store的大小



mode: 操作模式  
BPF\_IMM：load立即数到寄存器  
BPF\_ABS/BPF\_IND:用于操作packet  
BPF\_MEM:store到内存  
BPF\_XADD:独占add

### **2.3 指令类**

### **IMG_263**

LD/ST表示源操作数为立即数  
LDX/STX表示源操作数为src\_reg  
MOV: 寄存器之间赋值  
NEG:取反