Requirements for eBPF instructions:

For a range analysis, we should check if an instruction is valid, and whether

1. Program Variables:
   1. Step of bpf Checking
      1. Is\_first\_Pass
         1. DAG Checker, total instruction count is verified here
      2. Is\_Second\_Pass
         1. All paths checked, total analysis steps checked here
   2. Constants
      1. Kernel min address
      2. Kernel\_max\_address
      3. Total BPF Instructions
         1. 4k total
         2. Should be counted when is\_First\_Pass is true
      4. Total BPF Analysis Steps
         1. 64k total
         2. Should be counted when is\_Second\_Pass is true
      5. Max int size for register type based on instruction
         1. Need 8,16,32 and 64 bit separate
2. Register Limitations:
   1. R0 -> return register
   2. R1-r5 -> argument passing registers
      1. Are you calling this register in a function or an instruction
   3. R6-r9 -> callee saved registers
   4. R10 -> frame pointer READ ONLY

Add dst, src

Dst must be a register from r0-r9

Src must be a register from r0-r9

Could you take the frame pointer in r10 due to this being an add instruction?

I think I remember something about pointer addition being forbidden

These three could also change based on the type of dst and src (u32, u16)

\*dst must be a val between u64\_min and u64\_max

\*src must be a val between u64\_min and u64\_max

\*Dst + \*src must be between u64\_min + \*dst and u64\_max

Since this is an add with unsigned ints, the lower bound must be at least \*dst above min

Overflow flag for u64 ints must not be tripped

[dst] = [src] or [dst] is at least as big as [src]

Ie if src Is u32, dst must be u32 or u64

If we’re using SSA, then type matching doesn’t matter, because the type of dst will be overwritten by whatever the add ins tells it to do