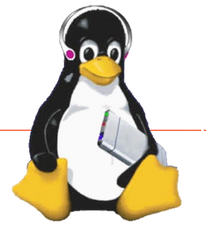

Open Source Vulnerability Matching & License Compliance

Peter Shin and David Barrett CANVASS LABS

What is demonstrated

For a vulnerability and its fix:

Package: linux
File: *kernel*bpf/verifier.c
CVE ID: CVE-2017-16995

Repository: /kernel
Function: check_alu_op
CWE ID: CWD-119

Patch Code:

			Hunk: Lines 88-94 (previously 88-91)
88	88		*/* remember the value we stored into this reg */*
89	89		regs[insn->dst_reg].type = SCALAR_VALUE;
90		-	__mark_reg_known(regs + insn->dst_reg, insn->imm);
	90	+	if (BPF_C+LASS(insn->code) == BPF_ALU64) {
	91	+	__mark_reg_known(regs + insn->dst_reg, insn->imm);
	92	+	} else {
	93	+	__mark_reg_known(regs + insn->dst_reg, (u32)insn ->imm);
91	94		}

What was improved

We use its abstract syntax:

```
TYPE check_alu_op(TYPE PARM, TYPE PARM) {
  TYPE LOCAL_VAR = FUNC_CALL(PARM);
  ...
  LOCAL_VAR[PARM] =  SCALAR_VALUE;
  FUNC_CALL(LOCAL_VAR + PARM, PARM) ;
```

To match your **altered** source code

```
int check_alu_op(struct bpf_verifier_env*  env ,
   struct bpf_insn*_env my_insn) {
   struct bpf_reg_state *regs = cur_regs(env)
   ...
   regs[my_insn->dst_reg].type =  SCALAR_VALUE;
   __mark_reg_known(
      regs + my_insn->dst_reg,  my_insn->imm);
```

And show you it is vulnerable and how to fix it

Source code or detail technical information availability

https://github.com/canvasslabs/canvass_for_security-dependency_checker