All About the Perf Fuzzer

Vince Weaver

vincent.weaver@maine.edu

Linux Plumbers 2016 — 3 November 2016

perf_fuzzer

- http://web.eece.maine.edu/~vweaver/projects/perf_events/fuzzer/
- Fuzzes the perf_event_open() syscall
- A *very* complicated system call
- Core PEO code based on Trinity, shares trinity's perf_event_open()

Motivation

- Varied
- Anyone not using `perf` (e.g. PAPI) is essentially untested
- Want HPC sysadmins to feel confident they can set perf_event_paranoid to 0 and not face crashes
- Once I started finding crashes, felt obligated to try to get things fixed

SW Challenges of `perf_event_open()`

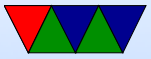
- Is not just `perf_event_open()`
  Trinity et al. exercise `perf_event.h` well
- The *other* parts of the manpage:
  `mmap()`, `poll()`, `prctl()`, `read()`, `ioctl()`, `signa`
- Files under `/proc` and `/sys`
- General perf state (`fork()`, `exec()`)
- BPF, ftrace, cgroups, breakpoints

HW Challenges of `perf_event_open()`

- Arch specific registers
- Doesn't virtualize well or at all
- Can cause unusually high-NMI loads
- Hard to make deterministic, partly because underlying events are nondeterministic

Current Status

- 4.9-rc0 just before rc1 (rc1+ breaks my system)

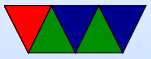


Paranoid 2 (user-access only)

machine	warnings	time to crash	kernel
p4	1	7m49s	4.9-rc0
core2	1	n/a (7days+)	4.9-rc0
haswell	1	3d9h26m	4.9-rc0
skylake	1	7d8h37m	4.9-rc0
a10	1	2d	4.9-rc0
sparc	0	30s	3.2
pi2	?	n/a	4.8?

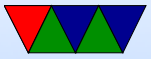
Paranoid 1 (user and kernel)

machine	warnings	time to crash	kernel
core2	2	1d15h20m	4.9-rc0
haswell	0	21h25m	4.9-rc0
skylake	?	n/a (5d+)	4.9-rc0
a10	?	2h15m	4.9-rc0



Paranoid 0 (per-cpu,uncore)

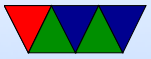
machine	warnings	time to crash	kernel
core2	3	21h19m	4.9-rc0
haswell	3	8h58m	4.9-rc0
skylake	0	4h50m	4.9-rc0
a10	1	7h55m	4.9-rc0



Paranoid -1, root (tracepoints)

machine	warnings	time to crash	kernel
core2	0	14m	4.9-rc0
haswell	?	didn't run	4.9-rc0
skylake	?	34m	4.9-rc0
a10	?	lost log	4.9-rc0

machine	warnings	time to crash	kernel
core2	?	1h13m	4.9-rc0



Open Questions

- Can we design a tool to auto-generate the equivalent of perf_fuzzer?
- Can we enhance kernel to make it possible to debug the weird perf_fuzzer generated deadlocks?
- Can we make anyone care about fuzzing?
- Can we make people contribute back?

Future Work

- cgroups
- eBPF
- Other features that require root permission

Questions?

http://web.eece.maine.edu/~vweaver/projects/perf_events/fuzzer/2016_fuzzer_update.pdf

vincent.weaver@maine.edu