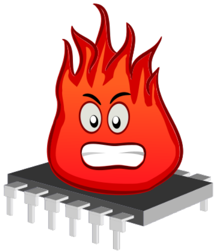
stress-ng



Improved system stressing with stress-ng

26th June 2024

New stress-ng features and the future roadmap for stress-ng

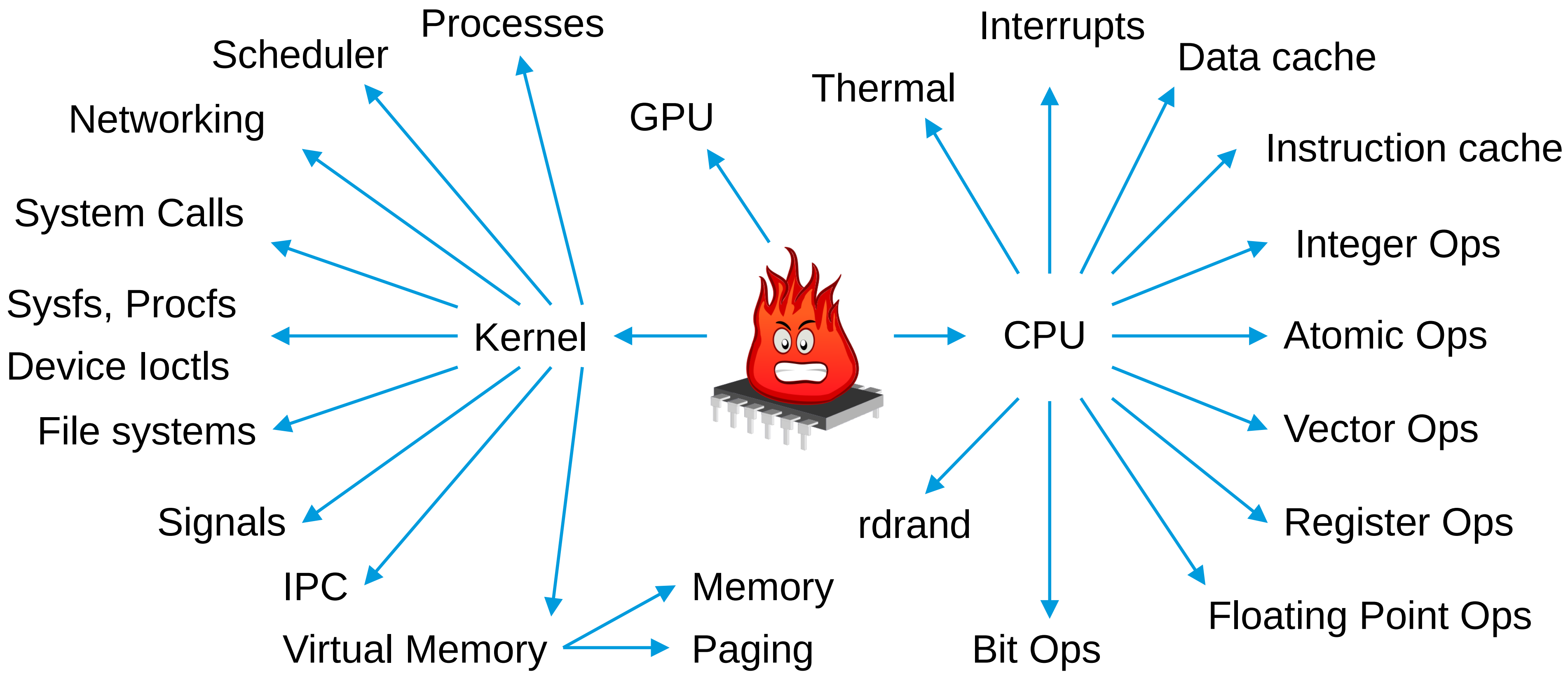
Why do stress testing?

- Find breakage points (kernel panics, races, lock-ups...)
- Check for correct system behaviour under stress
- Test modes of failure (e.g. what happens on low memory?)
- Test for stable behaviour outside of expected usage
- Exercise scaling/load (CPUs, memory, I/O) – does it scale well?
- Burn-in testing (e.g. detecting CPU / disk / memory errors)

Why use Stress-ng?

- Already found 80+ kernel bugs (Linux + *BSD)
- Kernel 0-day performance testing
  - e.g. 24 kernel performance improvements (Linux)
- Used by silicon vendors (new silicon + kernel bring-up)
- Used for kernel regression testing (e.g. Ubuntu kernel)
- LKP-tests (Linux kernel performance test tool)
- Referenced in 100+ research papers - synthetic stress testing

Stress-ng in 2024, ~340 stressors



Kernel

Processes

Scheduler

Networking

System Calls

Sysfs, Procfs

Device Ioctls

File systems

Signals

IPC

Virtual Memory

Memory

Paging

GPU

CPU

Thermal

Interrupts

Data cache

Instruction cache

Integer Ops

Atomic Ops

Vector Ops

Register Ops

Floating Point Ops

Bit Ops

rdrand

What’s new since June 2022?

- 50+ new stressors
- More stressor options for finer control and configuration
- New arch support: loong64
- Performance optimizations (using Intel vtune and perf)
- Improved SMP scaling (many CPUs, NUMA, etc)
- Improved libc + libm coverage
- Improved kernel system call coverage
- Improved portability (compilers, OS, libc, architectures, kernels)

New control options

- --oom-avoid - try to avoid Out of Memory process killing
- --oom-avoid-bytes N - specify memory threshold before OOM avoidance is activated, default N=2.5%
- --status N - show stressor run/exit/reap status every N seconds
- --permute and --with options, e.g.

```
stress-ng --with cpu,matrix,vecmath,fp --permute 5 -t 10s
stress-ng --with vm,mmap,brk,mremap --all 8 -t 10m
```

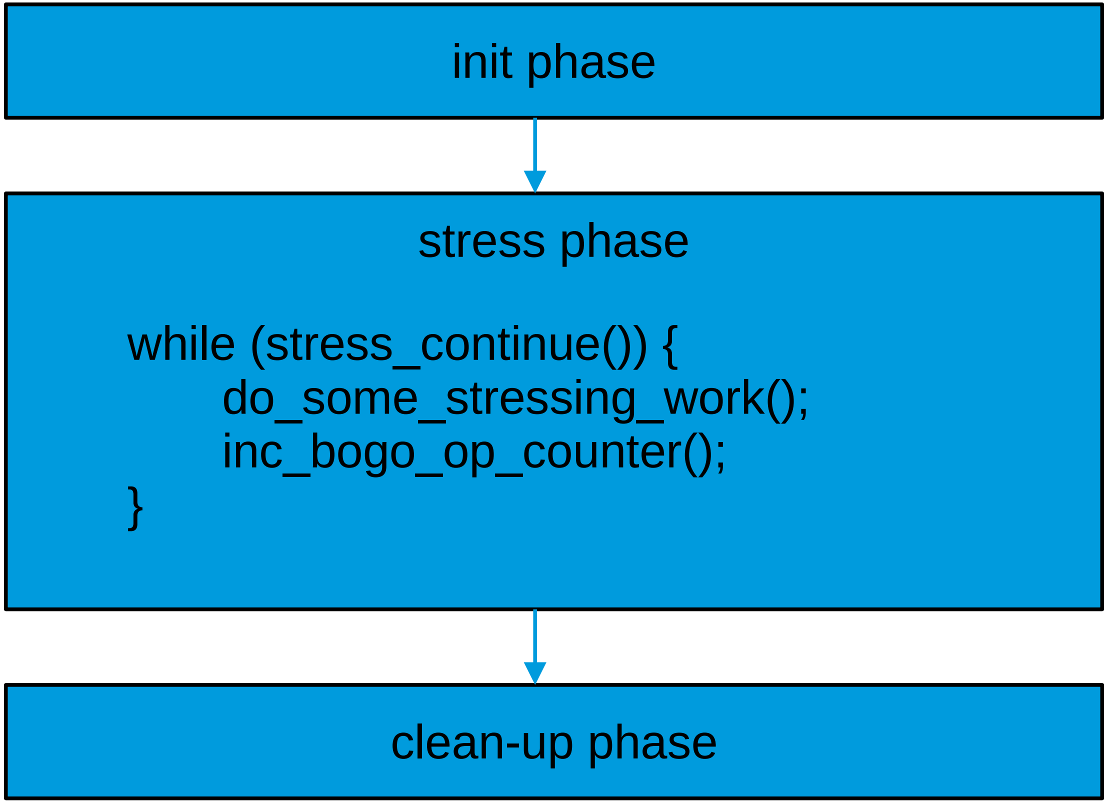
- --progress - show stressor progress and estimated completion time, useful with --seq options

What is a Stressor?

Normally a single process forked from stress-ng

Stressor may be one or more child process or one or more pthreads in more complex stress cases.

Stressor terminates on SIGALRM or reached maximum bogo-op count



init phase

stress phase

```
while (stress_continue()) {
        do_some_stressing_work();
        inc_bogo_op_counter();
}
```

clean-up phase

New Stressors: CPU compute

- factor – factorization of huge integers using GNU Multi-Precision Library (GMP)
- fma – fused multiply-add instructions
- fp – various sized floating point format stressor
- fractal – SMP scalable fractal generator
- mprf – reliable floating point exercising (GNU MPFR lib)
- prime – large integer prime number search (GMP)
- rotate – exercise 8/16/32/64/128 bit left/right rotate ops

New Stressors: libm + Eigen lib

- besselmath – libm bessel math functions
- eigen – 2D matrix stressors using Eigen C++ library
- expmath – libm exponential functions
- logmath – libm logarithmic functions
- monte-carlo – monte-carlo computations (pi, e, etc)
- powmath – libm power functions
- trig – libm trigonometric functions (sin, cos, tan etc)

New Stressors: Vector and Neural Network ops

- vecfp – vector floating point math operations
- vecshuf – vector instructions, data shuffling operations
- vnni - vector neural network instructions (x86 vnni)

New Stressors: Memory

- bitonicsort – bitonic integer sorting
- insertionsort – standard O(N^2) insertion sort
- mmapfiles – attempt to memory map 500,000+ files
- pagemove – exercise page moving using remap()
- vma – random address space memory mapping operations

New Stressors: Data and Instruction Cache

- cacheline – multi-process shared memory cacheline validation
- far-branch – calls to thousands of randomly allocated functions
- flushcache – hammer i-cache and d-cache flushing
- llc-affinity – exercise lower level cache (e.g. L3) while changing CPU affinity

New Stressors: Scheduling

- min-nanosleep – measure minimal nanosleep() duration for different linux schedulers
- mtx – iso C mutex stressor
- prio-inv – exercise thread priority inversion (RT kernels)
- race-sched – racy scheduling with CPU affinity
- ring-pipe – copy data around a ring of processes using pipes
- time-warp – check for clock time warping
- workload – random run time work loads

New Stressors: signal handling

- sigbus – SIGBUS signal exerciser (BUS errors)
- sigxcpu – SIGXCPU (ulimited cpu run time signal)
- sigxfsiz – SIFXFSIZ (ulimited file size signal)
- signest – now supports 25+ nested signals

New Stressors: CPU opcodes

- priv-instr – test trap handling of privileged instructions
    - ARM, Alpha, HPPA, Loong64, M68000, MIPS32/64, PPC64, S390x, SH4, SPARC64, x86.
- regs – exercise CPU register copying
    - All arches as above
- waitcpu – CPU wait/pause delay (exercise low power states)
    - ARM (yield), x86 (pause, tpause, umwait), PPC64 (yield, mdoio, mdoom), RISC-V (pause), loong64 (dbar)
- x86cpuid – exercise x86 CPUID instruction mixes

New Stressors: System calls / kernel interfaces

- cgroup – exercise cgroup v2 (mount/read/write/umount)
- fd-fork – file descriptor copying using fork()
- fsize – exercise 32 bit/64 bit file size limits
- metamix – mixed concurrent file meta data race exerciser
- mseal – Linux 6.10 memory sealing
- syscall – exercise as many system calls as possible
- umount – exercise racy file system unmounts
- unlink – racy file unlink (removal) stressor

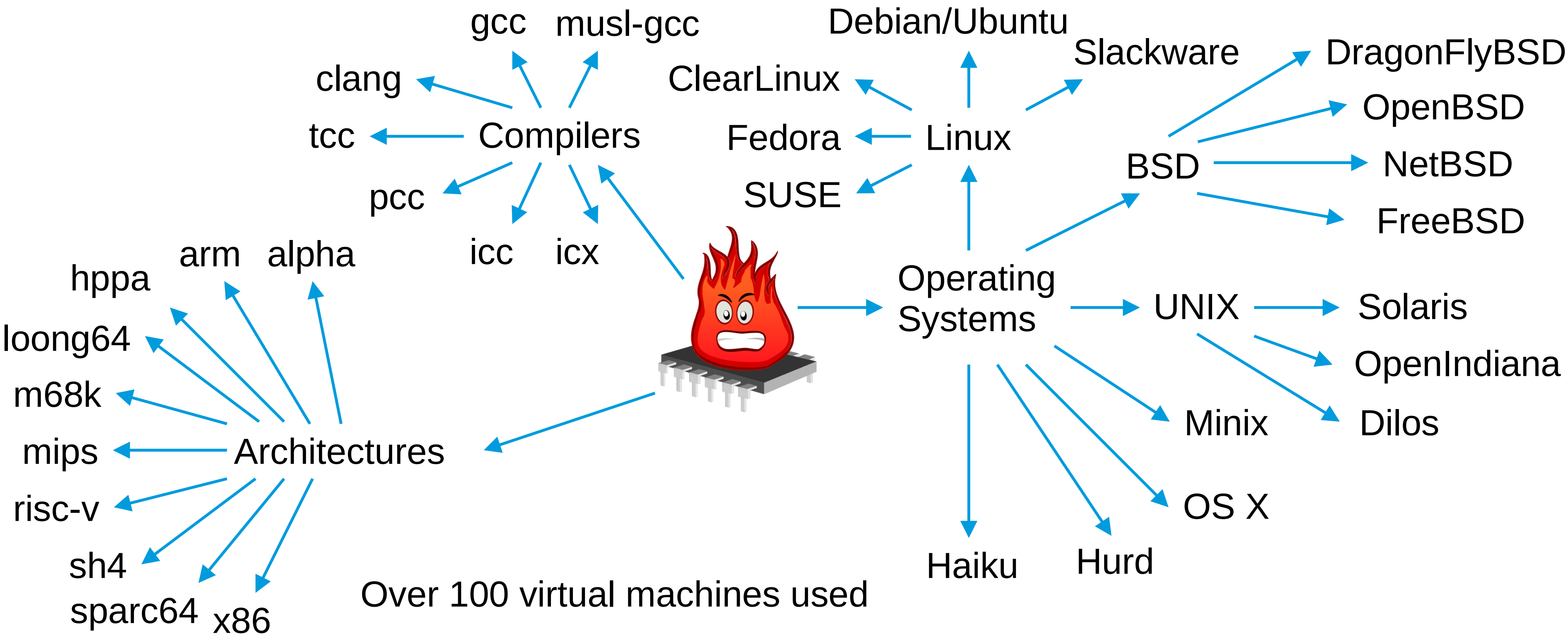
What drives stress-ng development?

- New kernel features (system calls, ioctls, sysfs/procfs, devices)
- Kernel gcov coverage holes (checked on each new kernel)
  - Directed coverage testing, another never ending task!
- New processor features (e.g. vector, AI, etc)
- New architectures (e.g. loong64)
- Kernel bugs (implement some reproducers)
- User requests and user provided stressors
  - Contributions always welcome!

Roadmap

- Synchronized stressor start, --sync-start for V0.18.01+
  - Create all stressor instances; ready wait; start
- Improved libc + libm coverage
  - e.g. OpenBSD 7.5 sincos() SIGSEGV issue
- Power measurements (x86 RAPL power) for V0.18.02+
- Monthly Release Cadence (normally 1st week of the Month)
- Focus on portability

Portability – Release Testing



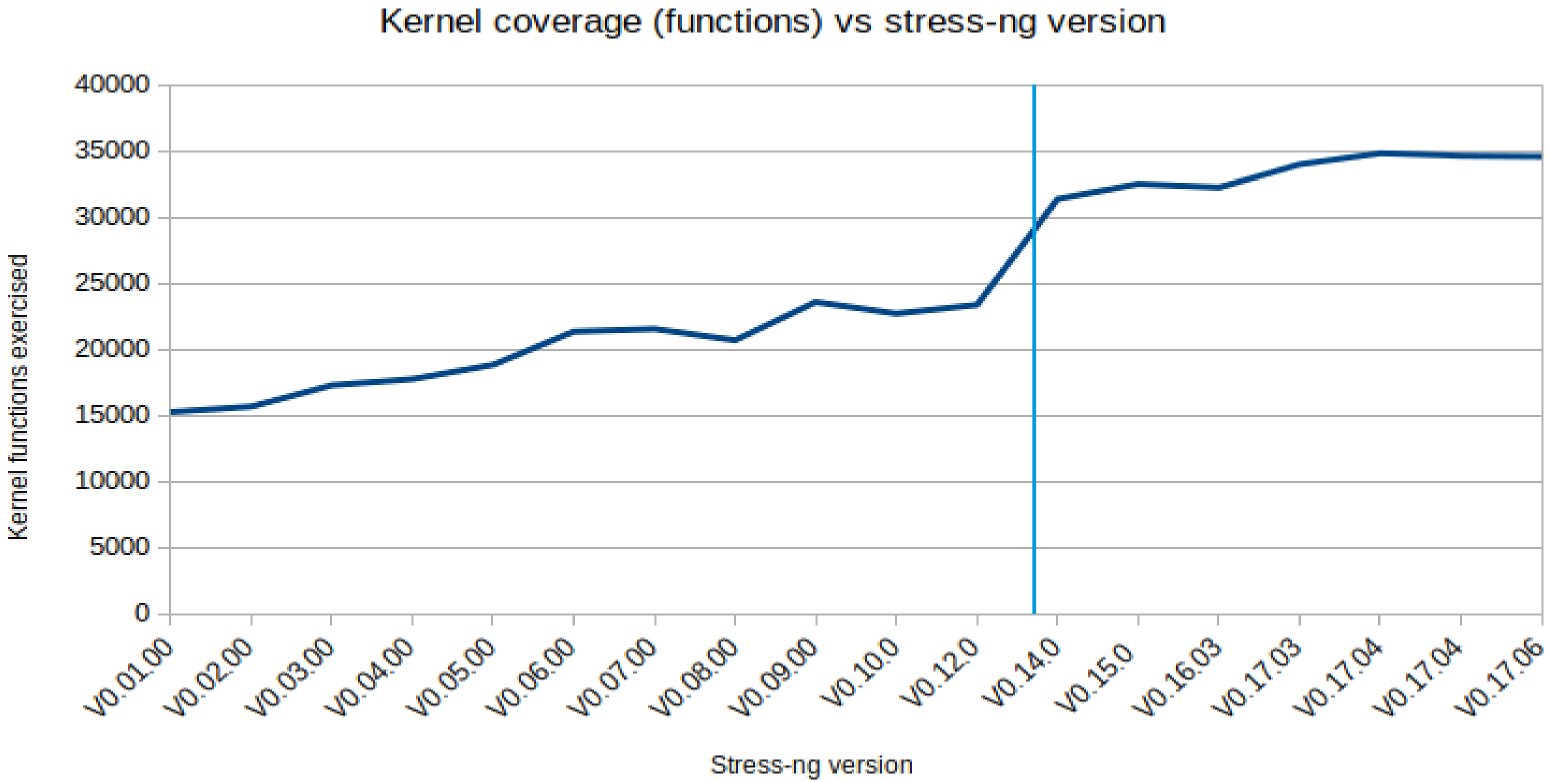
Compilers
gcc
musl-gcc
clang
tcc
pcc
icc
icx

Architectures
arm
alpha
hppa
loong64
m68k
mips
risc-v
sh4
sparc64
x86

Operating Systems
Linux
Debian/Ubuntu
ClearLinux
Fedora
SUSE
Slackware
BSD
DragonFlyBSD
OpenBSD
NetBSD
FreeBSD
UNIX
Solaris
OpenIndiana
Dilos
Minix
OS X
Hurd
Haiku

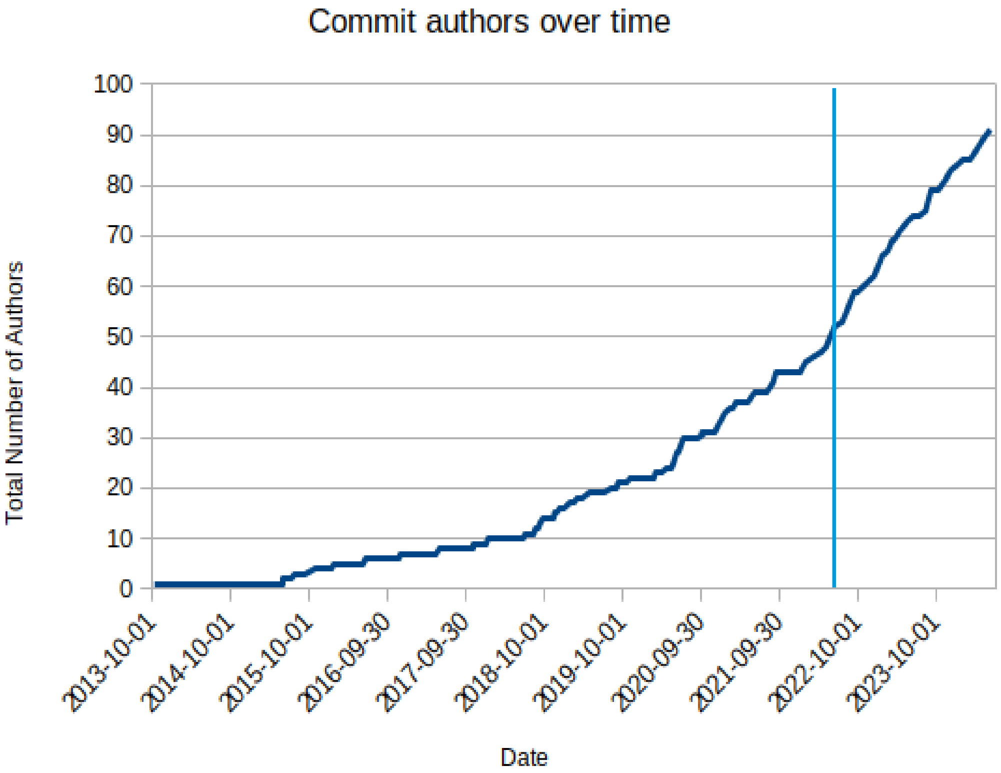
Over 100 virtual machines used

Kernel Test Coverage



Some development stats for last 2 years

- 4500+ commits
- 45+ new commit authors
  - Growing developer community
- 50+ new stressors
- 55,000+ new lines of code
- 32 tagged releases



Find out more

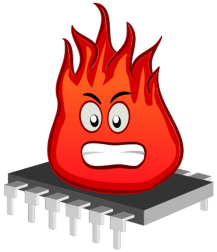
Read the manual (man page), ‘make pdf’ to make PDF version

- Plenty of per-stressor information
- About 100 pages – a lot of options!
- Future work: write a quick start man page

Quick start Reference Guide:

https://wiki.ubuntu.com/Kernel/Reference/stress-ng

Project Information + Questions



github.com/ColinIanKing/stress-ng

email: colin.i.king@gmail.com

Any Questions?