★perf cheat sheet★

Julia Evans
@b0rk

sourced from brendangregg.com/perf.html, which has many more great examples

important command line arguments

- `-a` : entire system
- `-g` : record stack traces
- `-e` : choose an event to record
- `-p` : specify a PID
- `-F` : pick sample frequency

perf top: get updates live!

```
# Sample CPUs at 49 Hertz, show top symbols:
perf top -F 49

# Sample CPUs, show top process names and segments:
perf top -ns comm,dso

# Count system calls by process, refreshing every 1 second:
perf top -e raw_syscalls:sys_enter -ns comm -d 1

# Count sent network packets by process, rolling output:
stdbuf -oL perf top -e net:net_dev_xmit -ns comm | strings
```

sampling

tracing

perf stat: count events! CPU counters!

```
# CPU counter statistics for COMMAND:
perf stat COMMAND

# *Detailed* CPU counter statistics for COMMAND:
perf stat -ddd command

# Various basic CPU statistics, system wide:
perf stat -e cycles,instructions,cache-misses -a

# Count system calls for PID, until Ctrl-C:
perf stat -e 'syscalls:sys_enter_*' -p PID

# Count block device I/O events for the entire system, for 10 seconds:
perf stat -e 'block:*' -a sleep 10
```

Reporting

```
# Show perf.data in an ncurses browser:
perf report

# Show perf.data as a text report:
perf report --stdio

# List all events from perf.data:
perf script

# Annotate assembly instructions from perf.data
# with percentages
perf annotate [--stdio]
```

perf trace: trace system calls & other events

```
# Trace syscalls system-wide
perf trace

# Trace syscalls for PID
perf trace -p PID
```

perf record: record profiling data

records into perf.data file

```
# Sample CPU functions for COMMAND, at 99 Hertz:
perf record -F 99 COMMAND

# Sample CPU functions for PID, until Ctrl-C:
perf record -p PID

# Sample CPU functions for PID, for 10 seconds:
perf record -p PID sleep 10

# Sample CPU stack traces for PID, for 10 seconds:
perf record -p PID -g -- sleep 10

# Sample CPU stack traces for PID, using DWARF to unwind stack:
perf record -p PID --call-graph dwarf
```

perf record: record tracing data

records into perf.data file

```
# Trace new processes, until Ctrl-C:
perf record -e sched:sched_process_exec -a

# Trace all context-switches, until Ctrl-C:
perf record -e context-switches -a

# Trace all context-switches with stack traces, for 10 seconds:
perf record -e context-switches -ag -- sleep 10

# Trace all page faults with stack traces, until Ctrl-C:
perf record -e page-faults -ag
```

adding new trace events

```
# Add a tracepoint for kernel function tcp_sendmsg():
perf probe 'tcp_sendmsg'

# Trace previously created probe:
perf record -e -a probe:tcp_sendmsg

# Add a tracepoint for myfunc() return, and include the retval as a string:
perf probe 'myfunc%return +0($retval):string'

# Trace previous probe when size > 0, and state is not TCP_ESTABLISHED(1):
perf record -e -a probe:tcp_sendmsg --filter 'size > 0 && skc_state != 1' -a

# Add a tracepoint for do_sys_open() with the filename as a string:
perf probe 'do_sys_open filename:string'
```

these need kernel debuginfo