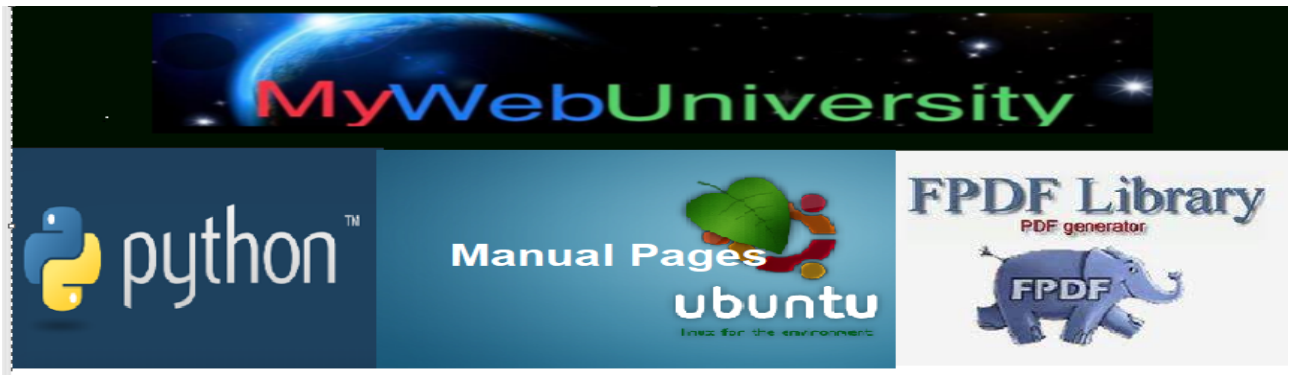

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'perf-c2c.1'

$ man perf-c2c.1

PERF-C2C(1) perf Manual PERF-C2C(1)

NAME

perf-c2c - Shared Data C2C/HITM Analyzer.

SYNOPSIS

perf c2c record [<options>] <command>

perf c2c record [<options>] -- [<record command options>] <command>

perf c2c report [<options>]

DESCRIPTION

C2C stands for Cache To Cache.

The perf c2c tool provides means for Shared Data C2C/HITM analysis. It allows you to track down the cacheline contentions.

On Intel, the tool is based on load latency and precise store facility events provided by Intel CPUs. On PowerPC, the tool uses random instruction sampling with thresholding feature. On AMD, the tool uses IBS op pmu (due to hardware limitations, perf c2c is not supported on Zen3 cpus). On Arm64 it uses SPE to sample load and store operations, therefore hardware and kernel support is required. See perf-arm-spe(1) for a setup guide. Due to the statistical nature of Arm SPE sampling, not every memory operation will be sampled.

These events provide: - memory address of the access - type of the access (load and store details) - latency (in cycles) of the load access

The c2c tool provide means to record this data and report back access details for cachelines with highest contention - highest number of HITM accesses.

The basic workflow with this tool follows the standard record/report phase. User uses the record command to record events data and report command to display it.

RECORD OPTIONS

-e, --event=

Select the PMU event. Use perf c2c record -e list to list available events.

-v, --verbose

Be more verbose (show counter open errors, etc).

-l, --ldlat

Configure mem-loads latency. Supported on Intel, Arm64 and some AMD processors. Ignored on other archs.

On supported AMD processors:

- /sys/bus/event_source/devices/ibs_op/caps/ldlat file contains '1'.
- Supported latency values are 128 to 2048 (both inclusive).
- Latency value which is a multiple of 128 incurs a little less profiling overhead compared to other values.
- Load latency filtering is disabled by default.

-k, --all-kernel

Configure all used events to run in kernel space.

-u, --all-user

Configure all used events to run in user space.

REPORT OPTIONS

-k, --vmlinux=<file>

vmlinux pathname

-v, --verbose

Be more verbose (show counter open errors, etc).

-i, --input

Specify the input file to process.

-N, --node-info

Show extra node info in report (see NODE INFO section)

-c, --coalesce

Specify sorting fields for single cacheline display. Following fields are available: tid,pid,iaddr,dso (see COALESCE)

-g, --call-graph

Setup callchains parameters. Please refer to perf-report man page for details.

--stdio

Force the stdio output (see STDIO OUTPUT)

--stats

Display only statistic tables and force stdio mode.

--full-symbols

Display full length of symbols.

--no-source

Do not display Source:Line column.

--show-all

Show all captured HITM lines, with no regard to HITM % 0.0005 limit.

-f, --force

Don?t do ownership validation.

-d, --display

Switch to HITM type (rmt, lcl) or peer snooping type (peer) to display and sort on.

Total HITMs (tot) as default, except Arm64 uses peer mode as default.

--stitch-lbr

Show callgraph with stitched LBRs, which may have more complete callgraph. The perf.data file must have been obtained using perf c2c record --call-graph lbr. Disabled by default. In common cases with call stack overflows, it can recreate better call stacks than the default lbr call stack output. But this approach is not foolproof. There can be cases where it creates incorrect call stacks from incorrect matches. The known limitations include exception handing such as setjmp/longjmp will have calls/returns not match.

--double-cl

Group the detection of shared cacheline events into double cacheline granularity. Some architectures have an Adjacent Cacheline Prefetch feature, which causes cacheline sharing to behave like the cacheline size is doubled.

-M, --disassembler-style=

Set disassembler style for objdump.

--objdump=<path>

Path to objdump binary.

C2C RECORD

The perf c2c record command setup options related to HITM cacheline analysis and calls standard perf record command.

Following perf record options are configured by default: (check perf record man page for details)

-W,-d,--phys-data,--sample-cpu

The following table lists the events monitored on different architectures. Unless specified otherwise with the -e option, the tool will select the default events.

Arch	Configuration	Options	Events
Intel	Default	-e ldlat-loads	cpu/mem-loads,ldlat=30/P
		-e ldlat-stores	cpu/mem-stores/P
	Load only	-e ldlat-loads	cpu/mem-loads,ldlat=30/P
	Store only	-e ldlat-stores	cpu/mem-stores/P
Intel with AUX	Default	-e ldlat-loads	{cpu/mem-loads-aux/,cpu/mem-loads,ldlat=30/}:P
		-e ldlat-stores	cpu/mem-stores/P
	Load only	-e ldlat-loads	{cpu/mem-loads-aux/,cpu/mem-loads,ldlat=30/}:P
	Store only	-e ldlat-stores	cpu/mem-stores/P
AMD	Default	-e mem-ldst	ibs_op// (without latency support)
			ibs_op/ldlat=30/ (with latency support)
PowerPC	Default	-e ldlat-loads	cpu/mem-loads/
		-e ldlat-stores	cpu/mem-stores/
	Load only	-e ldlat-loads	cpu/mem-loads/
	Store only	-e ldlat-stores	cpu/mem-stores/
Arm	Default	-e spe-ldst	

arm_spe_0/ts_enable=1,pa_enable=1,load_filter=1,store_filter=1,min_latency=30/ |

SPE			
	Load only	-e spe-load	arm_spe_0/ts_enable=1,pa_enable=1,load_filter=1,min_latency=30/
	Store only	-e spe-store	arm_spe_0/ts_enable=1,pa_enable=1,store_filter=1/

User can pass any perf record option behind -- mark, like (to enable callchains and system wide monitoring):

$ perf c2c record -- -g -a

Please check RECORD OPTIONS section for specific c2c record options.

C2C REPORT

The perf c2c report command displays shared data analysis. It comes in two display modes: stdio and tui (default).

The report command workflow is following: - sort all the data based on the cacheline address - store access details for each cacheline - sort all cachelines based on user settings - display data

In general perf report output consist of 2 basic views: 1) most expensive cachelines list 2) offsets details for each cacheline

For each cacheline in the 1) list we display following data: (Both stdio and TUI modes follow the same fields output)

Index

- zero based index to identify the cacheline

Cacheline

- cacheline address (hex number)

Rmt/Lcl Hitm (Display with HITM types)

- cacheline percentage of all Remote/Local HITM accesses

Peer Snoop (Display with peer type)

- cacheline percentage of all peer accesses

LLC Load Hitm - Total, LclHitm, RmtHitm (For display with HITM types)

- count of Total/Local/Remote load HITMs

Load Peer - Total, Local, Remote (For display with peer type)

- count of Total/Local/Remote load from peer cache or DRAM

Total records

- sum of all cachelines accesses

Total loads

- sum of all load accesses

Total stores

- sum of all store accesses

Store Reference - L1Hit, L1Miss, N/A

L1Hit - store accesses that hit L1

L1Miss - store accesses that missed L1

N/A - store accesses with memory level is not available

Core Load Hit - FB, L1, L2

- count of load hits in FB (Fill Buffer), L1 and L2 cache

LLC Load Hit - LlcHit, LclHitm

- count of LLC load accesses, includes LLC hits and LLC HITMs

RMT Load Hit - RmtHit, RmtHitm

- count of remote load accesses, includes remote hits and remote HITMs;

on Arm neoverse cores, RmtHit is used to account remote accesses,

includes remote DRAM or any upward cache level in remote node

Load Dram - Lcl, Rmt

- count of local and remote DRAM accesses

For each offset in the 2) list we display following data:

HITM - Rmt, Lcl (Display with HITM types)

- % of Remote/Local HITM accesses for given offset within cacheline

Peer Snoop - Rmt, Lcl (Display with peer type)

- % of Remote/Local peer accesses for given offset within cacheline

Store Refs - L1 Hit, L1 Miss, N/A

- % of store accesses that hit L1, missed L1 and N/A (no available) memory

level for given offset within cacheline

Data address - Offset

- offset address

Pid

- pid of the process responsible for the accesses

Tid

- tid of the process responsible for the accesses

Code address

- code address responsible for the accesses

cycles - rmt hitm, lcl hitm, load (Display with HITM types)

- sum of cycles for given accesses - Remote/Local HITM and generic load

cycles - rmt peer, lcl peer, load (Display with peer type)

- sum of cycles for given accesses - Remote/Local peer load and generic load

cpu cnt

- number of cpus that participated on the access

Symbol

- code symbol related to the 'Code address' value

Shared Object

- shared object name related to the 'Code address' value

Source:Line

- source information related to the 'Code address' value

Node

- nodes participating on the access (see NODE INFO section)

NODE INFO

The Node field displays nodes that accesses given cacheline offset. Its output comes in 3 flavors: - node IDs separated by , - node IDs with stats for each ID, in following format: Node{cpus %hitms %stores} (Display with HITM types) Node{cpus %peers %stores} (Display with peer type) - node IDs with list of affected CPUs in following format: Node{cpu list}

User can switch between above flavors with -N option or use n key to interactively switch in TUI mode.

COALESCE

User can specify how to sort offsets for cacheline.

Following fields are available and governs the final output fields set for cacheline offsets output:

tid - coalesced by process TIDs

pid - coalesced by process PIDs

iaddr - coalesced by code address, following fields are displayed:

Code address, Code symbol, Shared Object, Source line

dso - coalesced by shared object

By default the coalescing is setup with pid,iaddr.

STDIO OUTPUT

The stdio output displays data on standard output.

Following tables are displayed: Trace Event Information - overall statistics of memory accesses

Global Shared Cache Line Event Information

- overall statistics on shared cachelines

Shared Data Cache Line Table

- list of most expensive cachelines

Shared Cache Line Distribution Pareto

- list of all accessed offsets for each cacheline

TUI OUTPUT

The TUI output provides interactive interface to navigate through cachelines list and to display offset details.

For details please refer to the help window by pressing ? key.

CREDITS

Although Don Zickus, Dick Fowles and Joe Mario worked together to get this implemented, we got lots of early help from Arnaldo Carvalho de Melo, Stephane Eranian, Jiri Olsa and Andi Kleen.

C2C BLOG

Check Joe?s blog on c2c tool for detailed use case explanation:

https://joemario.github.io/blog/2016/09/01/c2c-blog/

SEE ALSO

perf-record(1), perf-mem(1), perf-arm-spe(1)