Networking and Tracing

Linux Plumbers Conference 2024

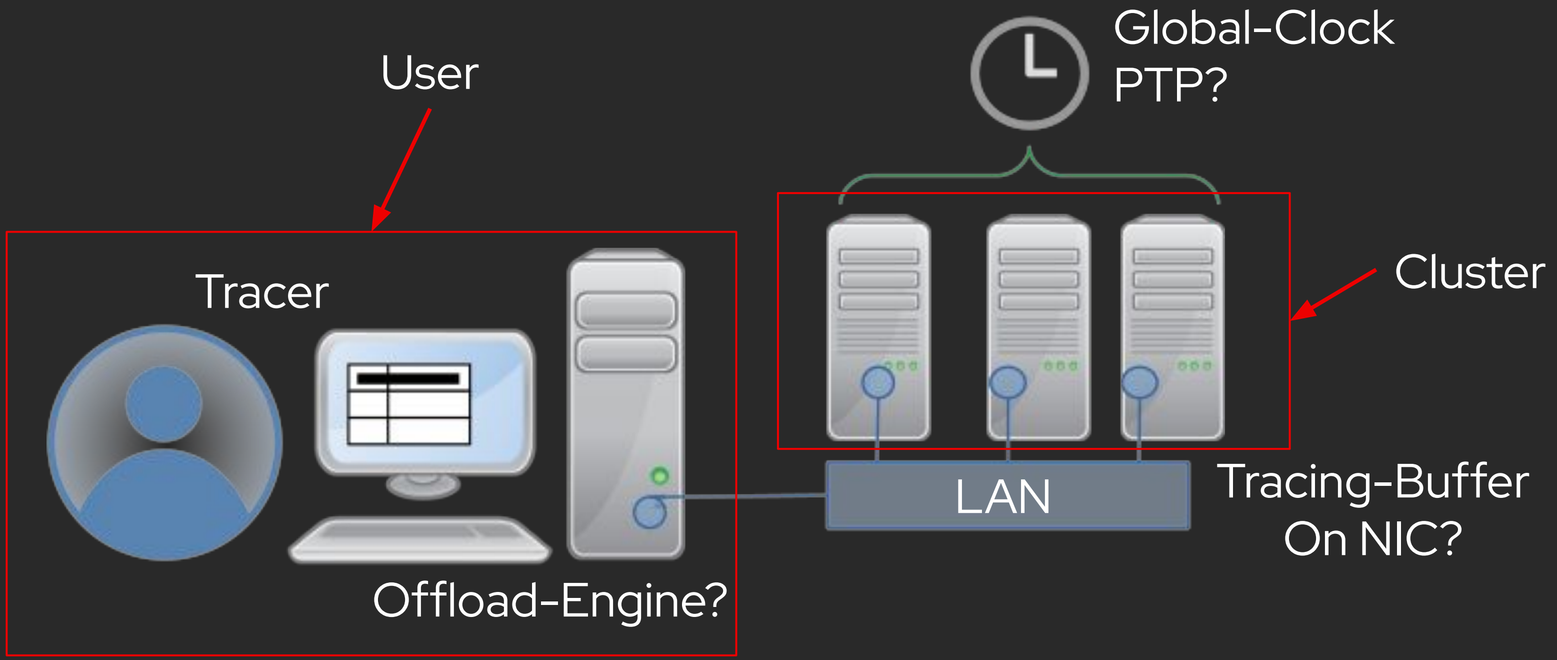
Alexander Aring

Topics to Discuss

1. PTP implementation in TRACE-CMD
2. Tracing Buffer to remote Machine
3. Tracing Buffer access over Socket API
4. Tracing Offload Engine Idea

“My” Main Use-Case (We focus on that!)

Remote
Use-Case

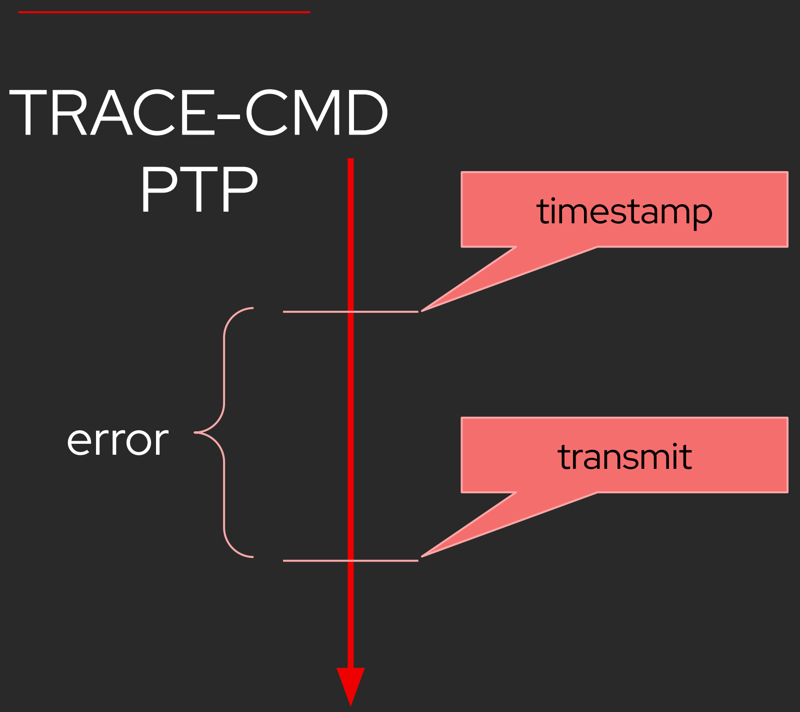


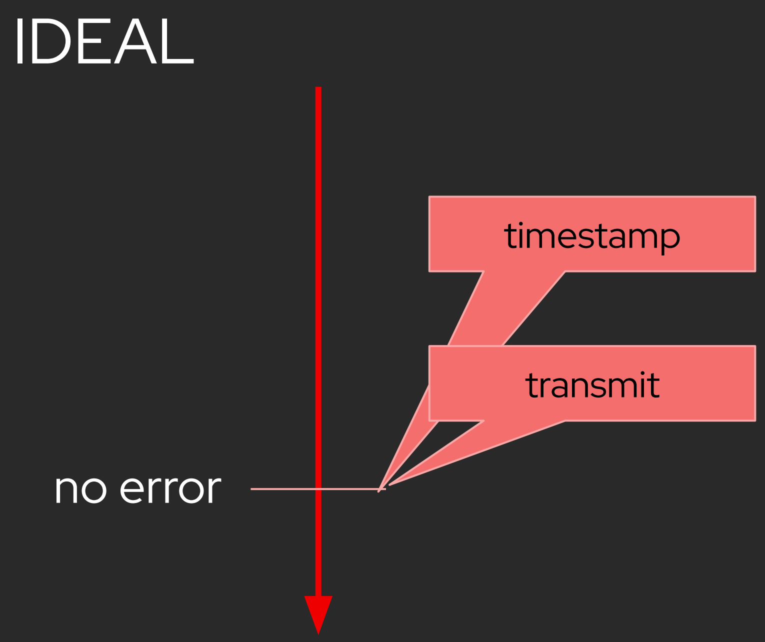
PTP Implementation in TRACE-CMD

Problem:
Timestamping done in User Space.

Only kvm tsc/vsock worked for me

TRACE-CMD Timestamping and Error





Why we care?

Causality of TraceEvents

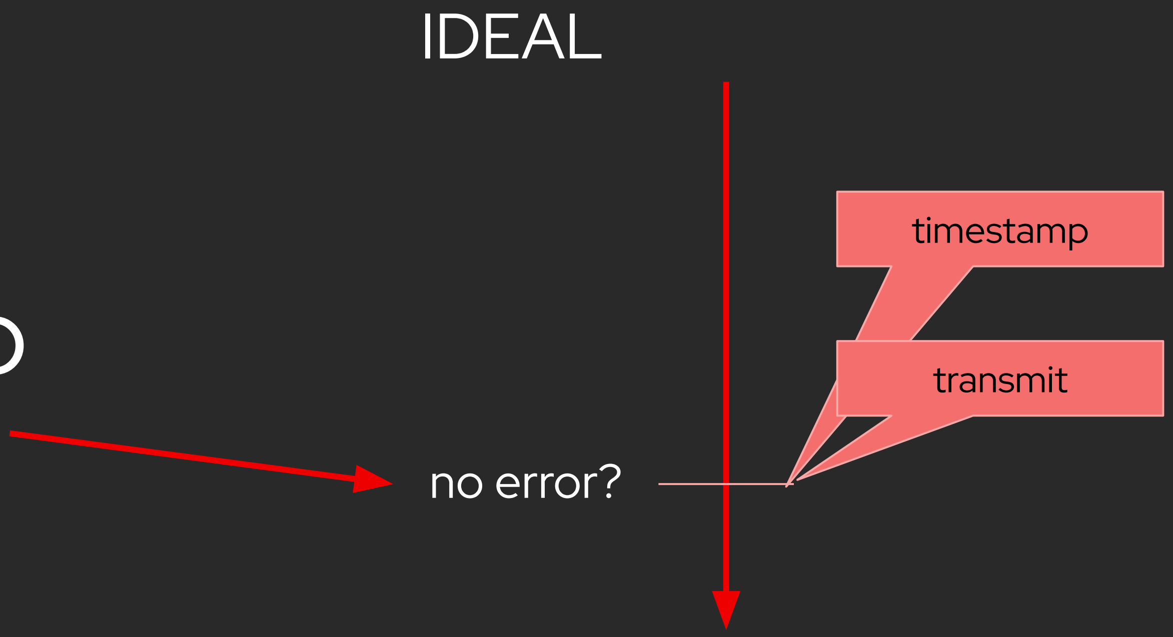
Action -> Reaction

Can we get accuracy under nanoseconds?

Using NIC HWTSTAMP?

NIC HWTSTAMP

Timestamp close to Transmit/Receive



NIC PHC - Physical Hardware Clock

Modern NICs having a PHC
NIC PHC (POSIX Clocks, adjtime(), etc.)

Using NIC PHC to sync a Tracing Clock?

Why not using linuxptp (chronyd?)?

phc2sys (sync $GLOBAL_CLOCK)?
does all the synchronization (PI~~D~~)

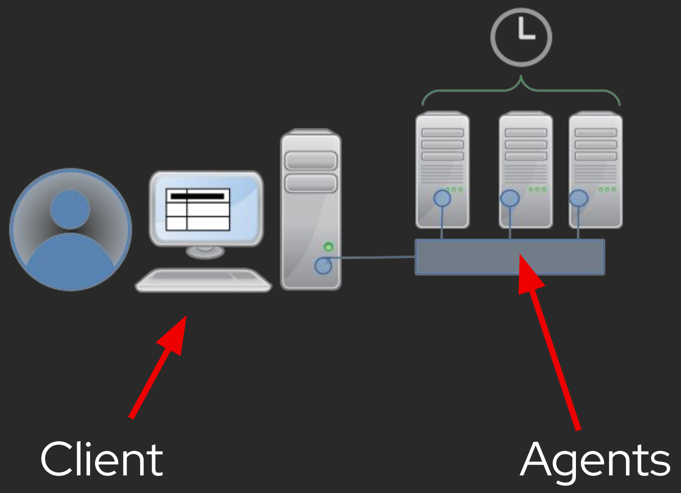
"trace-cmd -c $GLOBAL_CLOCK"?
Metadata Offset vs Synced Clock?
Better than PTP in TRACE-CMD?

Transmit Trace-Events to remote Machine

trace-cmd agents/-A

Avoid Bouncing Buffers?

- Tracing -> Socket Buffer?
- Kernel -> User (help splice()?)



NIC Ring-Buffer as Tracing-Buffer?

"net/core/page_pool.c"

DMA mapped pages on NIC Ring Buffer

Direct operate on NIC Ring Buffer?

TODO: What about metadata, tracefs, etc.?

No per-CPU Page-Pool: Cache misses?

No Cache locality!?

NIC Buffer is a shared resource among NUMA/CPUs

My opinion: A general networking problem that people want to "handle" better

No Cache Locality

My Use-Case: We don’t care?

Assume: Local Machine isn’t interested into tracing data*

Do we not have that problem anyway?

Somehow we need to send Tracing-Data over the Networking?

Wireshark as User Space Tracer?

raw-tracing-data is just accessible over AF_XDP (operates on NIC Ring-Buffer)

Runtime Dissector via Metadata (tracefs)?

NIC can generic offload “things”

Alternative to Tracing in-kernel Interpreter “kernel/trace/trace_events_filter.c”

Offload: P4*?, TC?, ?eBPF?

*Programming Protocol-Independent Packet Processors

Offloading example (u32)

Just a bunch of TLVs?
Match key-value pairs (fields)

Actions:

- Drop (Classic Filtering)
- skb->mark (“classify” Trace-Events) <- My Use-Case!
- Redirect, etc. networking stuff...

Even Local-Case Useful?

No remotes, Local Buffer, Local Tracer

No Tracing-Buffer Cache Locality

May depends on Offload to bring a benefit?

We need to see...

What to do as next: PTP?

Bring linuxptp, trace-cmd -c $GLOBAL_CLOCK together?

Look if accuracy is better than TRACE-CMD PTP Implementation?

What to do as next: NIC Ring-Buffer?

Make the User Space part as First!

~~Tracing Buffer~~ - Virtual Network Interface

Try Wireshark, try Filtering?

Then look into work on NIC Ring-Buffer?

Send it to remote machine?

Discussion, Questions?

Thanks

Red Hat