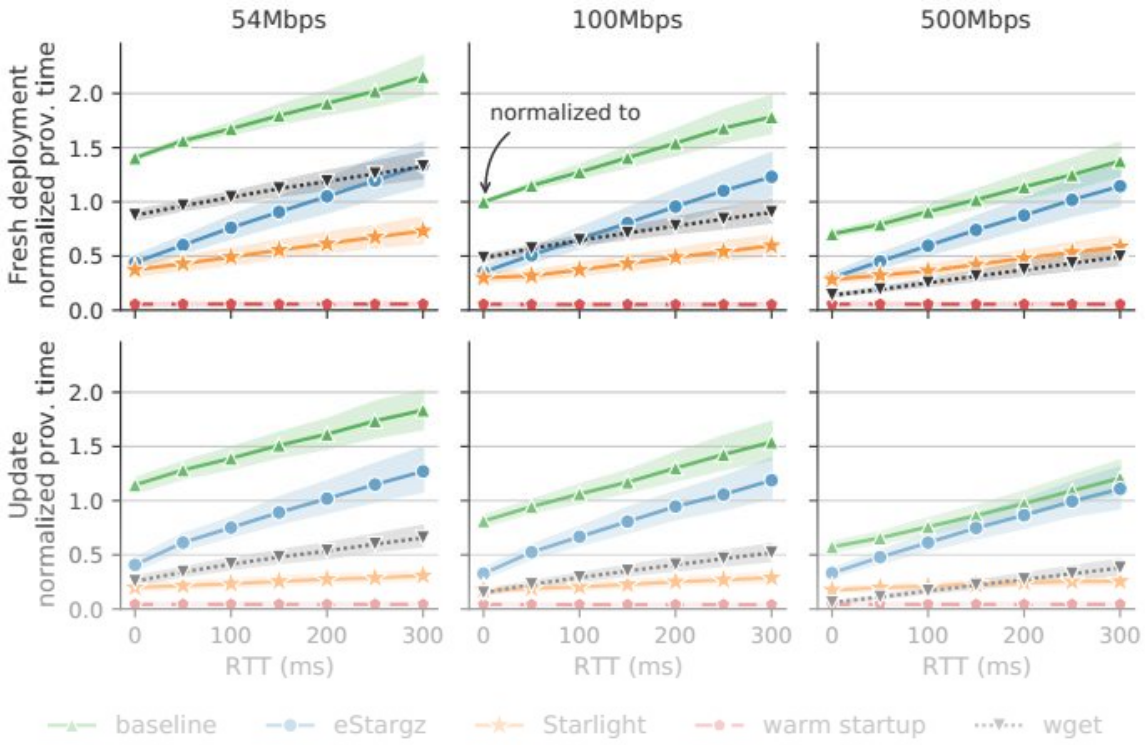
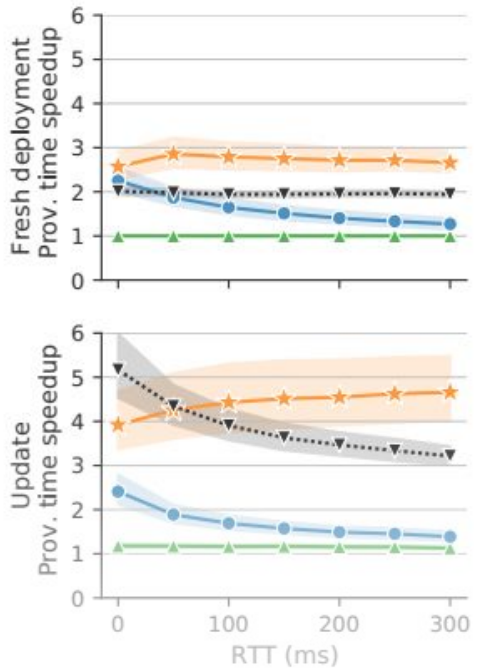
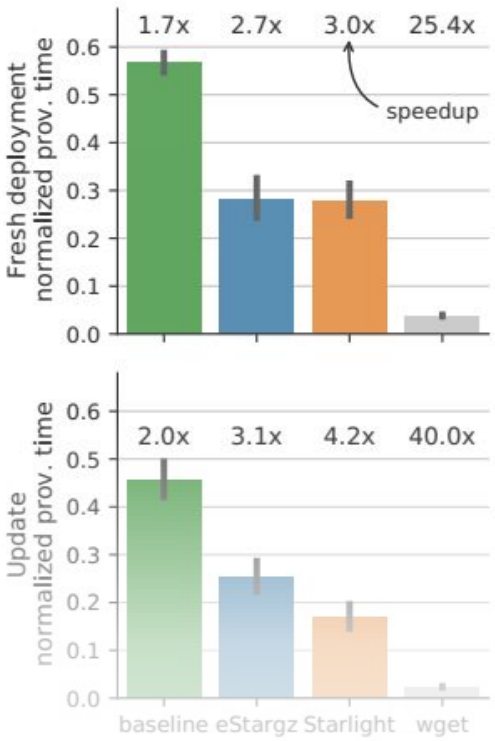






Presenting Complex Figures

Moshe Gabel

UNIVERSITY OF TORONTO

Figures Tell a Story

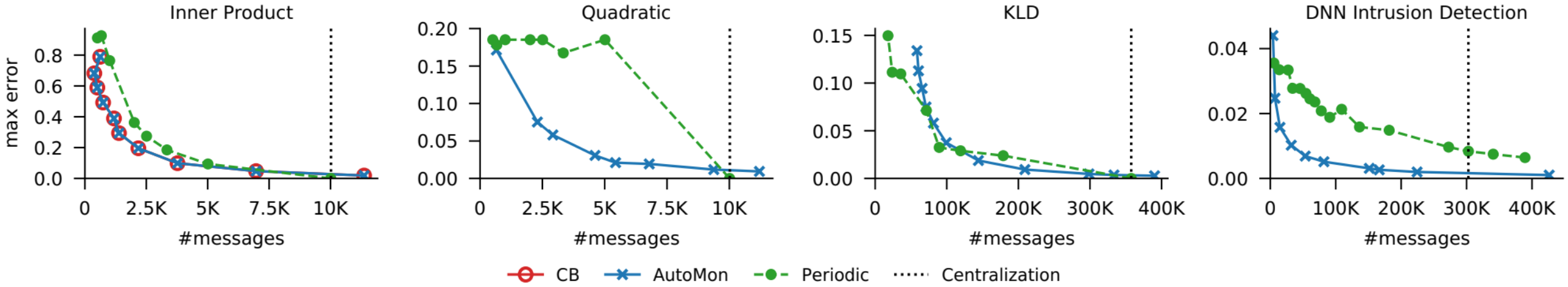
Story from our SIGMOD'22 paper.

"AutoMon can **compute** complex, **arbitrary functions** over **distributed data** streams with **reduced communication** compared to other methods"

Hadar Sivan, Moshe Gabel, Assaf Schuster. "AutoMon: Automatic Distributed Monitoring for Arbitrary Multivariate Functions". SIGMOD 2022

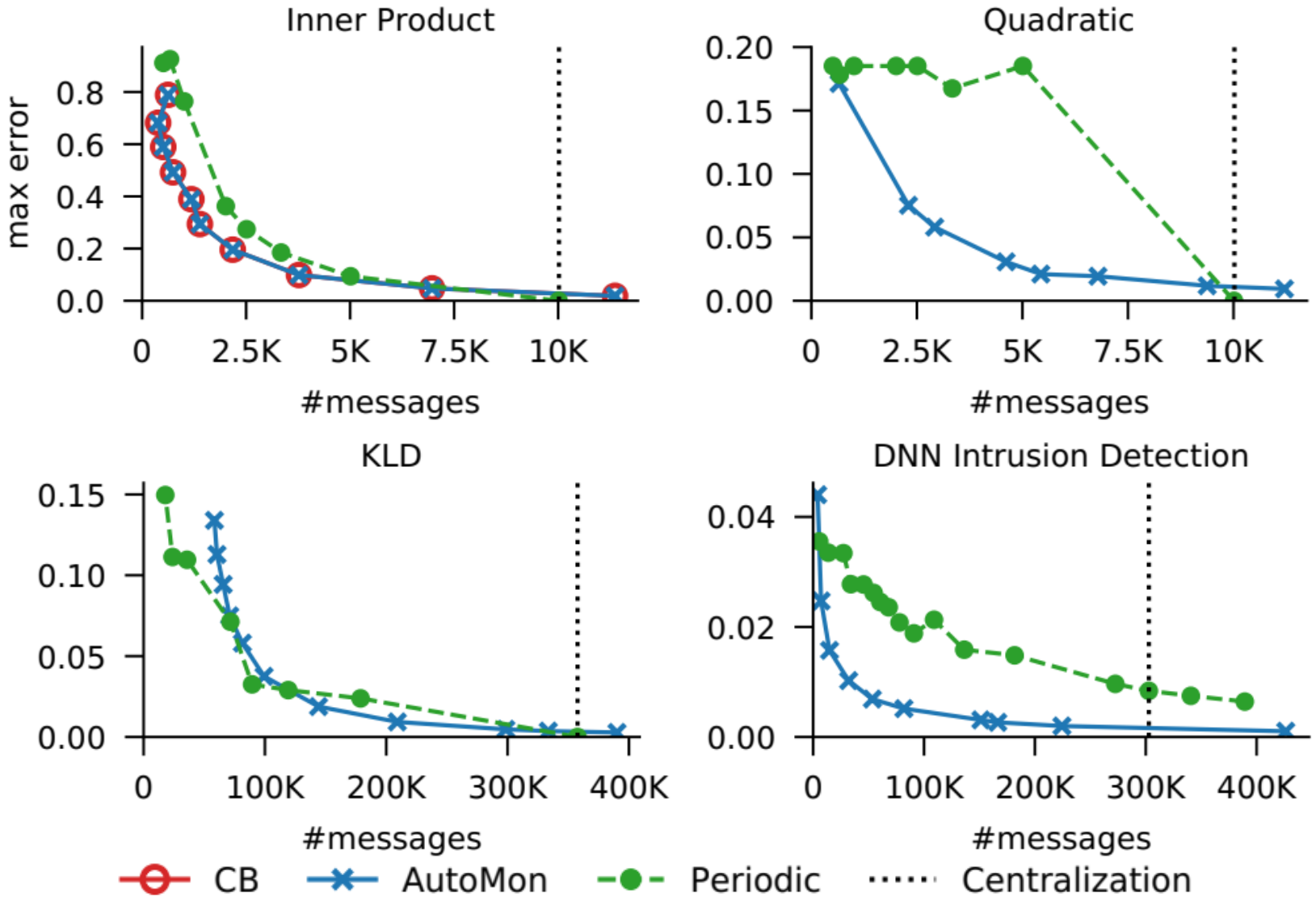
Figures Tell a Story

- But what if it is complex?



- …ouch.
- And I can’t see anything

Rearrange?



- Better.
- But still like a slap in the face.
- What to do?

Talks are Not Papers!

- Shocking!
- Thank you for coming to my TED talk.
- Actually, I am not surprised at all.
- So stop copy/pasting complex figures!

Talks are Not Papers

What works in a paper…

- Supporting text, caption
- Read the paper (… you wish)
- Time to dive into figure
- Can re-read the figure

Talks are Not Papers

What works in a paper…

- Supporting text, caption
- Read the paper (… you wish)
- Time to dive into figure
- Can re-read the figure

…does not work on a slide:

- Audience barely following
- No time to read, re-read
- What to focus on?
- Hard to read + listen

Talks are Not Papers

What works in a paper…

- ▶ Supporting text, caption
- ▶ Read the paper (… you wish)
- ▶ Time to dive into figure
- ▶ Can re-read the figure

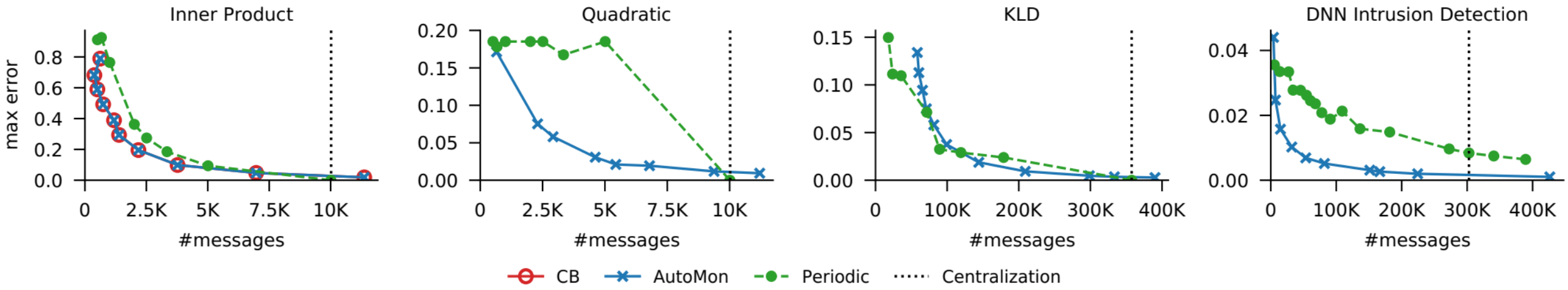
…does not work on a slide:

- ▶ Audience barely following
- ▶ No time to read, re-read
- ▶ What to focus on?
- ▶ Hard to read + listen

Guide the audience through figure.
Teach them how to read it.

Let's Try Again

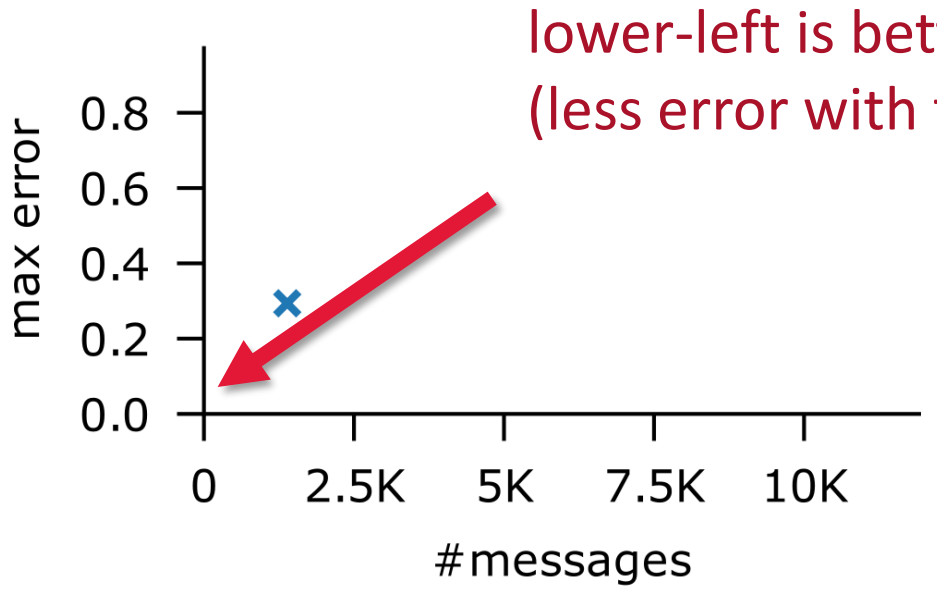
- Presenting results for AutoMon for different error levels
  - (ignore unfamiliar concepts)



Results: Error-Comm Tradeoff

from [SIGMOD'22]

- One run with specific ϵ
  - X axis – total sent messages
  - Y axis – max error across run

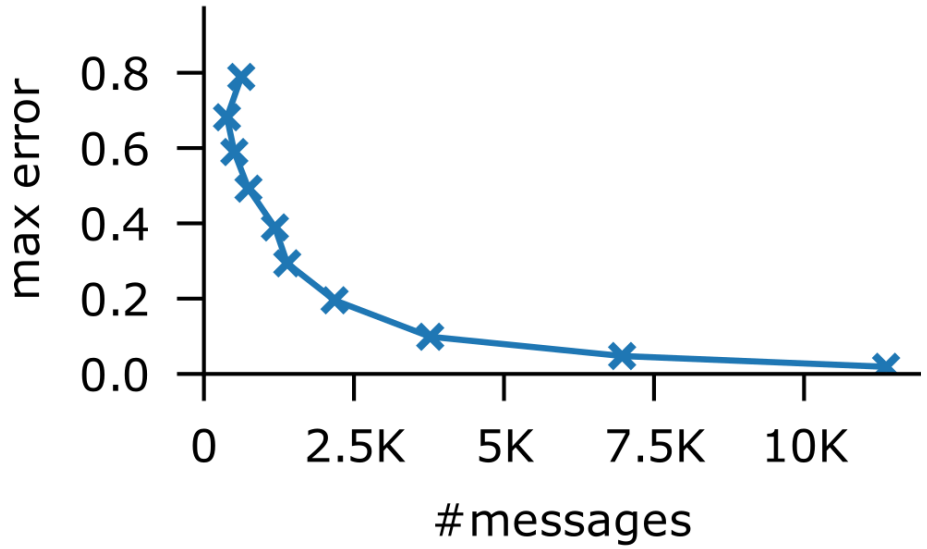


lower-left is better
(less error with fewer messages)

Results: Error-Comm Tradeoff

from [SIGMOD'22]

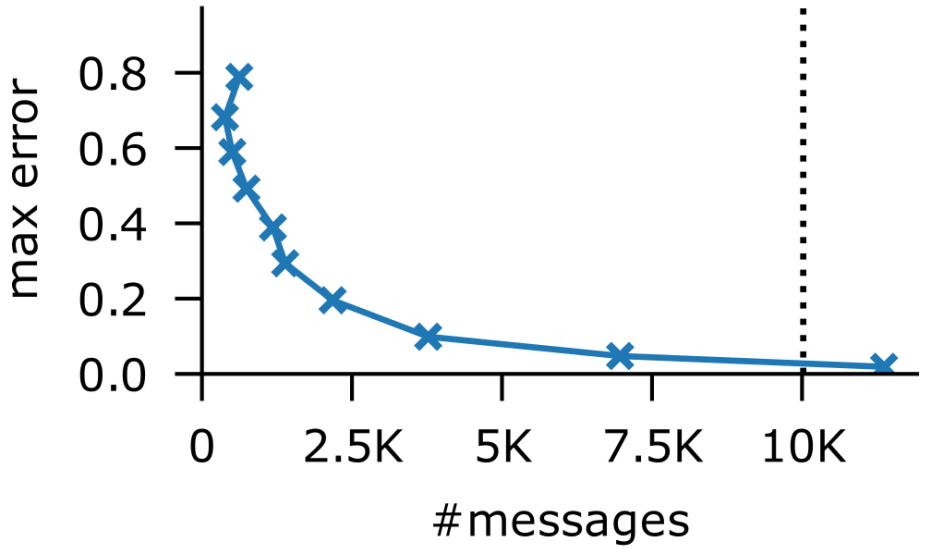
- Test on a range of ϵ
- Trade-off curve of **AutoMon** on this data and function



Results: Error-Comm Tradeoff

from [SIGMOD'22]

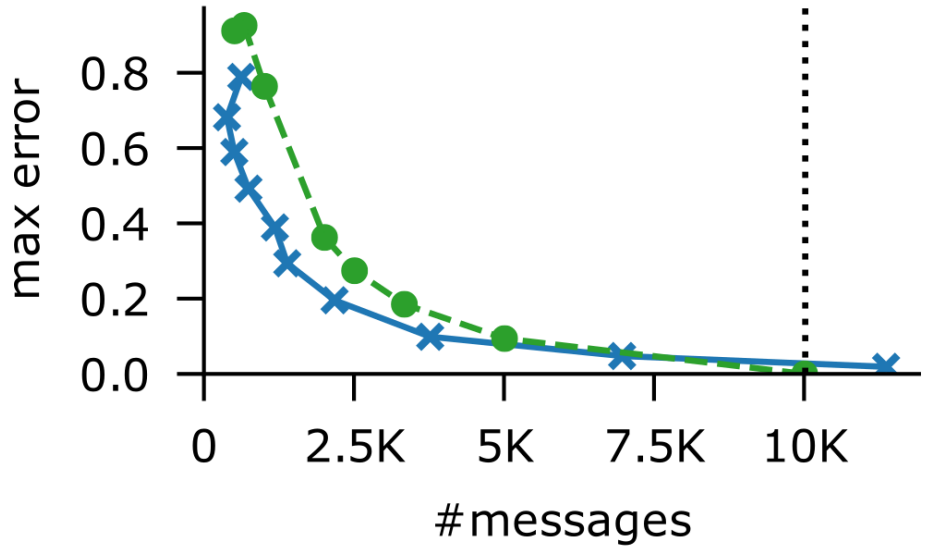
- Test on a range of ϵ
- Trade-off curve of **AutoMon** on this data and function
- **Centralization**: **just send all data updates.**
  - No error
  - State-of-the-art for sketches (they reduce message size, not number of messages)



Results: Error-Comm Tradeoff

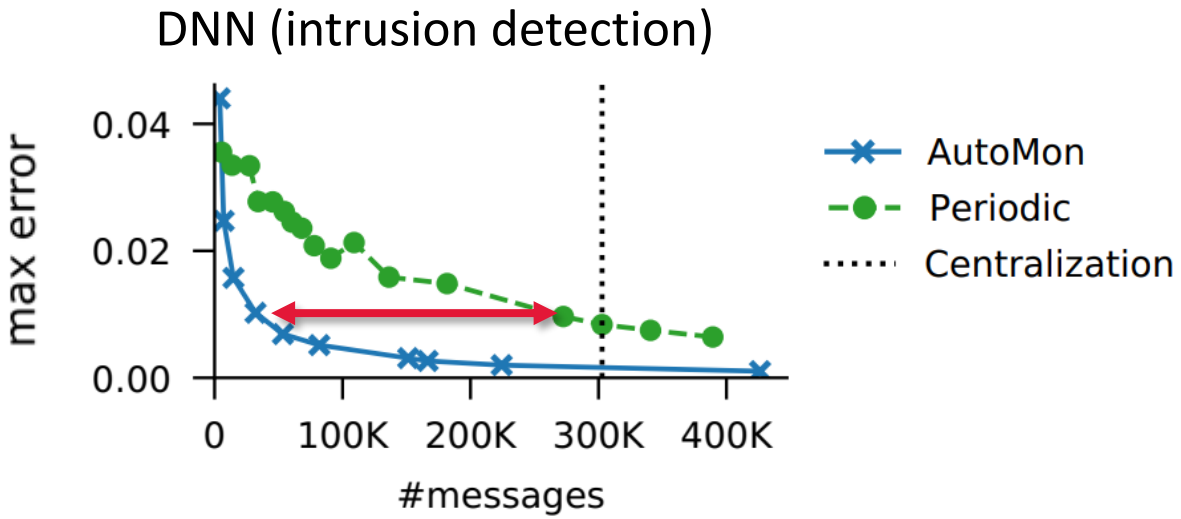
from [SIGMOD'22]

- Test on a range of ϵ
- Trade-off curve of **AutoMon** on this data and function
- **Centralization**
- **Periodic**: send every N updates
  - Non-adaptive
  - The common approach



Results: Error-Comm Tradeoff

from [SIGMOD'22]



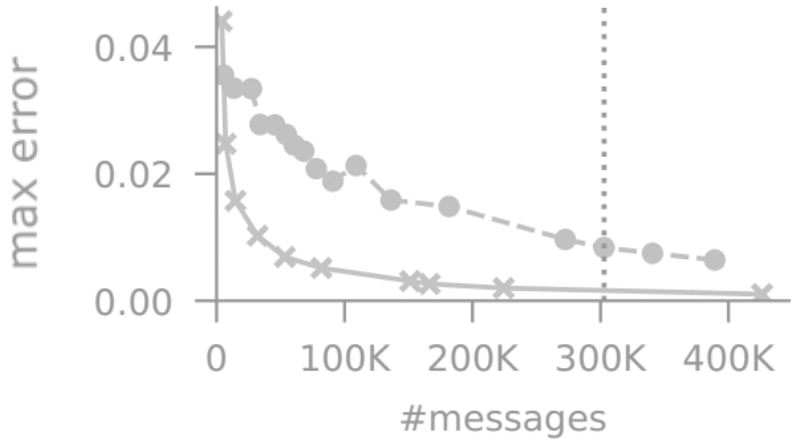
Value changes **slowly:**

- Periodic wastes messages
- AutoMon is adaptive and communication-efficient
- **2% comm** with low error

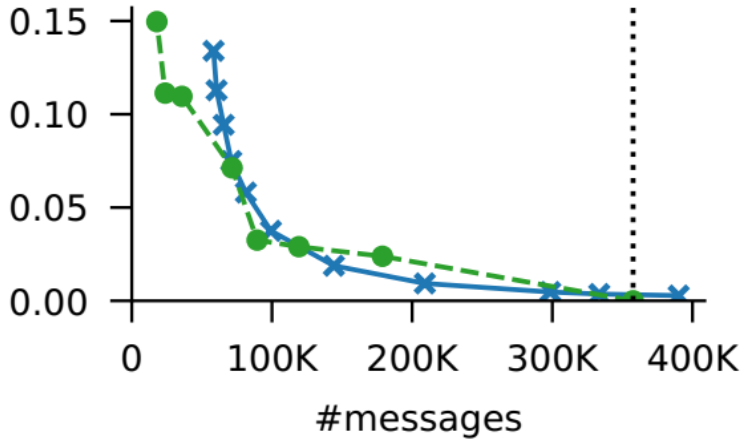
Results: Error-Comm Tradeoff

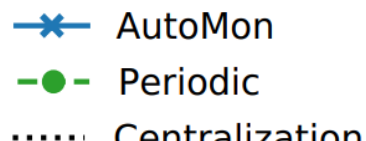
from [SIGMOD'22]

DNN (intrusion detection)



KLD (pollution monitoring)





Value changes **slowly:**

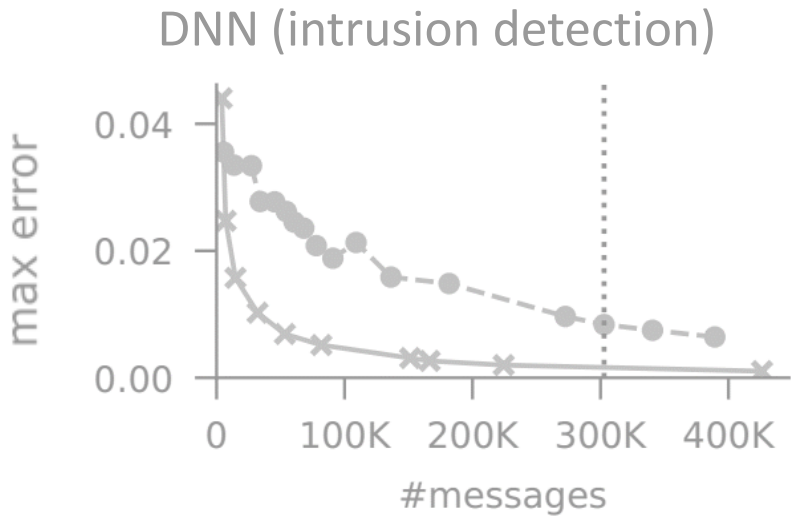
- Periodic wastes messages
- AutoMon is adaptive and communication-efficient
- **2% comm** with low error

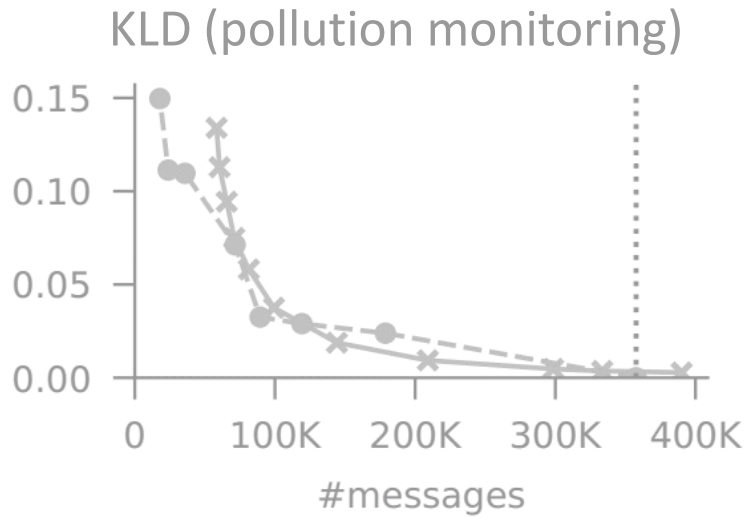
Value changes **gradually:**

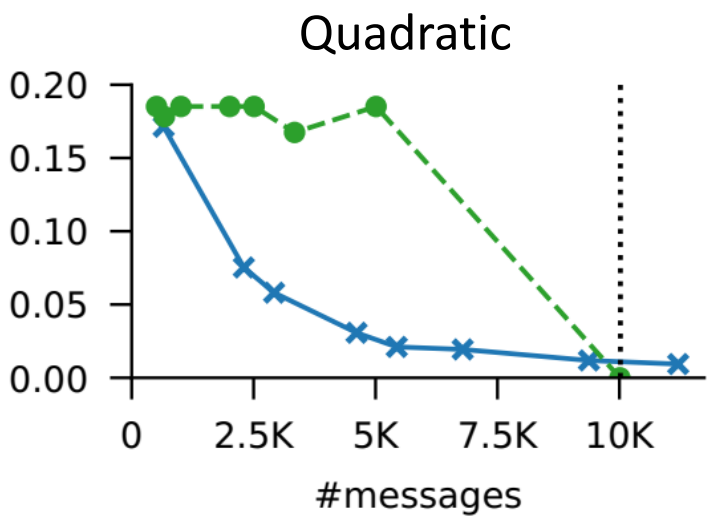
- AutoMon performance similar to Periodic
- AutoMon guarantees error, and is adaptive

Results: Error-Comm Tradeoff

from [SIGMOD'22]







Value changes **slowly:**

- Periodic wastes messages
- AutoMon is adaptive and communication-efficient
- **2% comm** with low error

Value changes **gradually:**

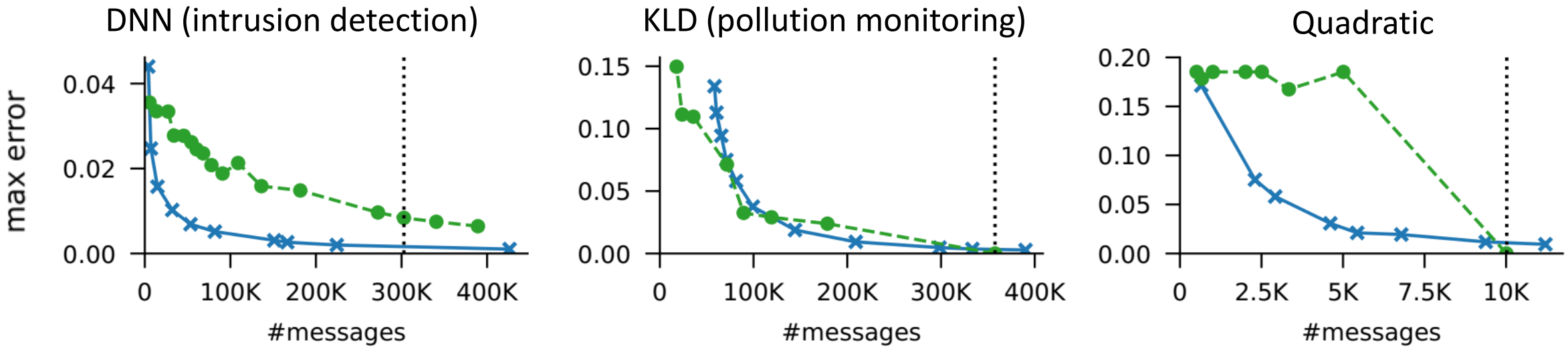
- AutoMon performance similar to Periodic
- AutoMon guarantees error, and is adaptive

Value changes **quickly:**

- Large error in Periodic
- AutoMon is adaptive: smooth, superior tradeoff

Results: Error-Comm Tradeoff

from [SIGMOD'22]



AutoMon:

- ✓ provides equivalent or **superior tradeoff** to current approaches …
- ✓ **automatically** from source code.

Summary

- Do not stick to paper figure.
- Show small part first
- ...then expand.
- Trim unneeded figures.
- Emphasize important conclusions
  - Explicitly.

Summary

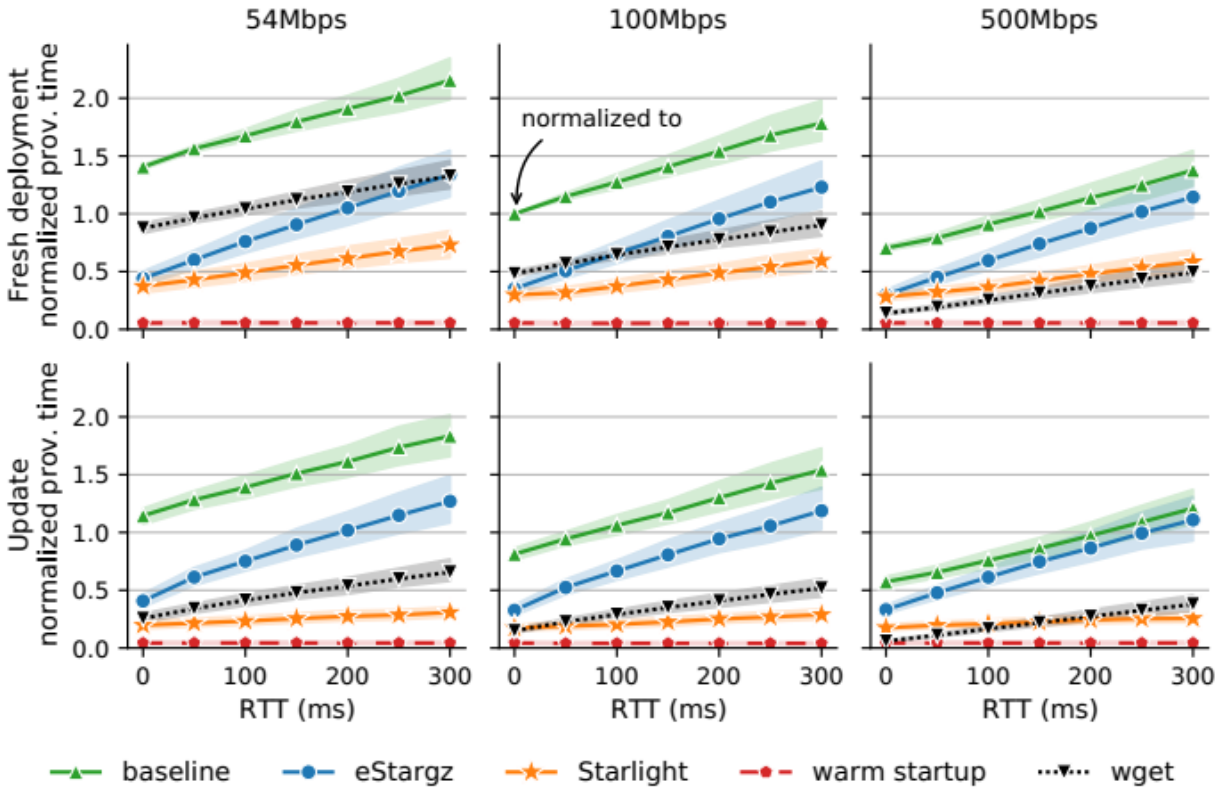
- Do not stick to paper figure.
- Show small part first
- ...then expand.
- Trim unneeded figures.
- Emphasize important conclusions
  - Explicitly.

Not enough time!

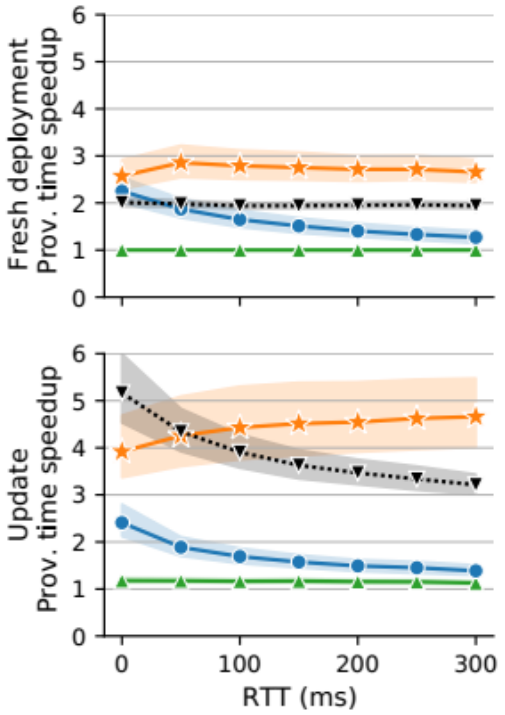
Zathras says:

- Understanding > showing
- Practice the talk
- Reduce content

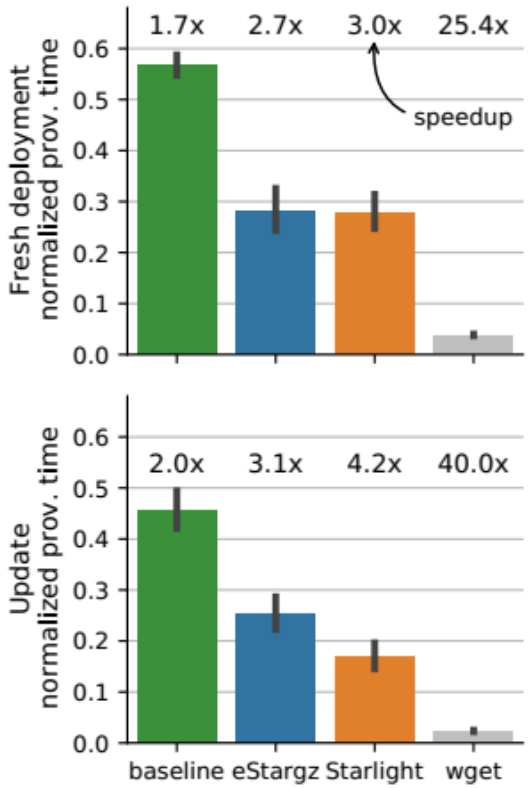
Another Example: Aggregated Results



(a) Edge and WAN.



(b) Speedup with 100Mbps.



(c) Cloud time and speedup.

From our paper: Jun Lin Chen, Daniyal Liaqat, Moshe Gabel, Eyal de Lara.
"Starlight: Fast Container Provisioning on the Edge and over the WAN". NSDI 2022

What to do With Aggregates?

- Figure aggregates results on 20 containers
  - Impossible to understand.
- Story:
  "**Starlight reduces provisioning time**, compared to SotA approaches"
- Strategy:
  - Build results for **one** container, method by method
  - Summarize conclusions.
  - Show most important aggregate.
  - Send readers to paper for others.

Evaluation

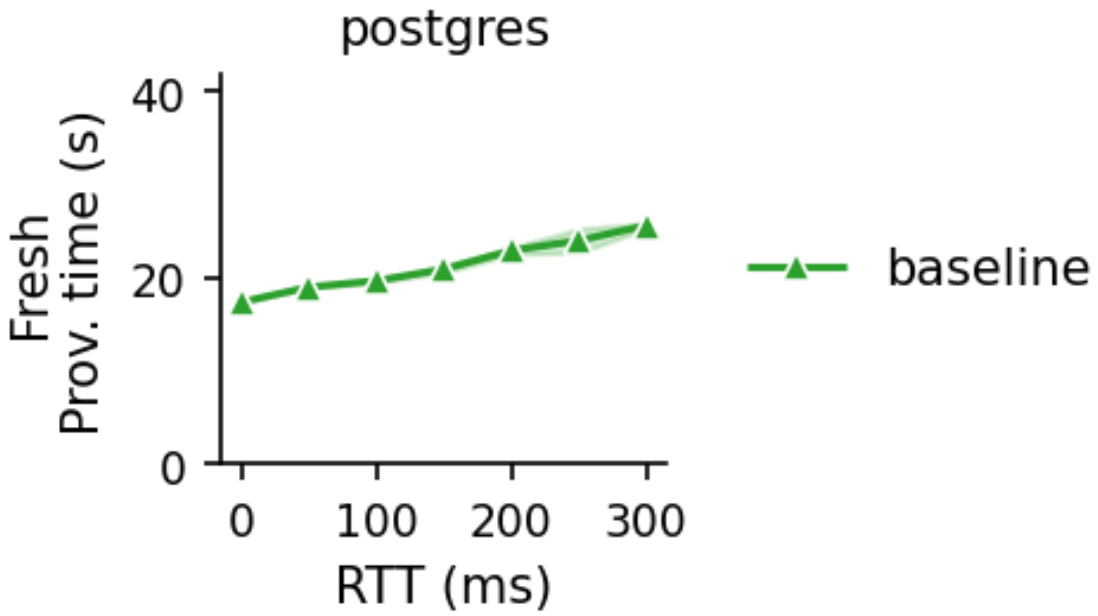
from [NSDI'22]

- 21 popular containers
  - 15B+ downloads in Docker Hub
  - Run each until ready
- Controlled deployments
  - tc controls bandwidth, RTT
- Real-world deployment
  - WAN covers 3 continents

Effect of Round-trip Time

from [NSDI'22]

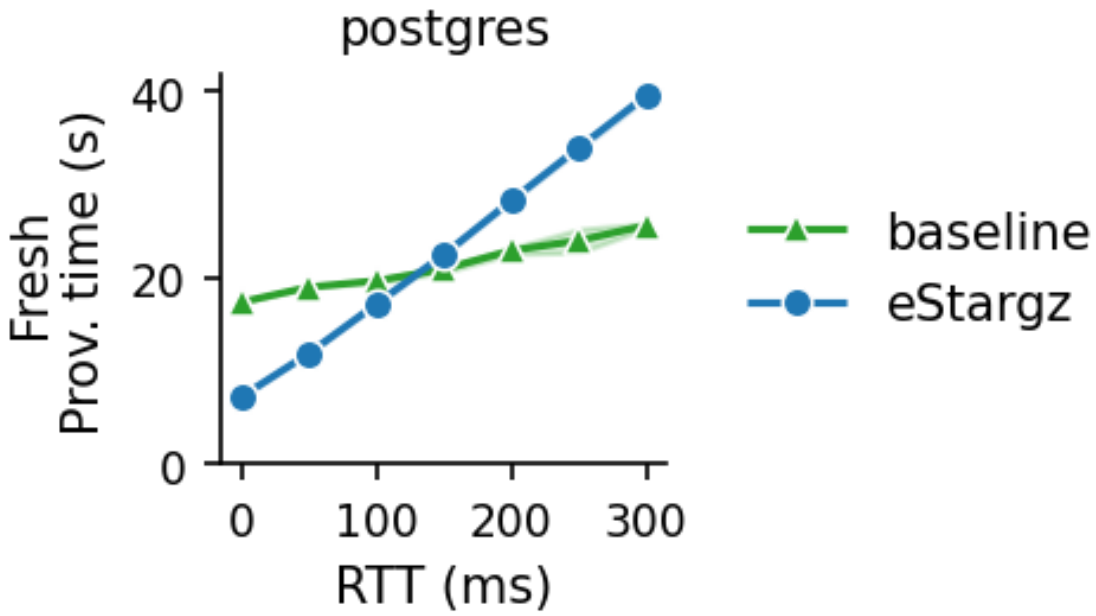
- X = RTT 0 to 300ms, 100Mbs
- Y = resulting provisioning time
  - **containerd v1.5.0**



Effect of Round-trip Time

from [NSDI'22]

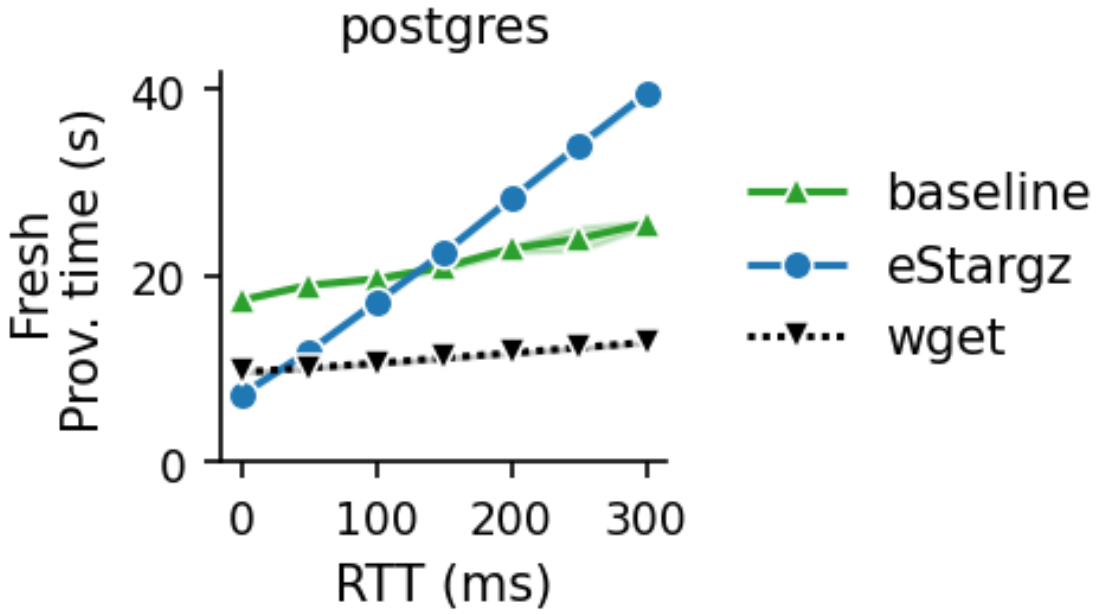
- X = RTT 0 to 300ms, 100Mbs
- Y = resulting provisioning time
  - **containerd v1.5.0**
  - **eStargz v0.6.3**: pull-based, on-demand



Effect of Round-trip Time

from [NSDI'22]

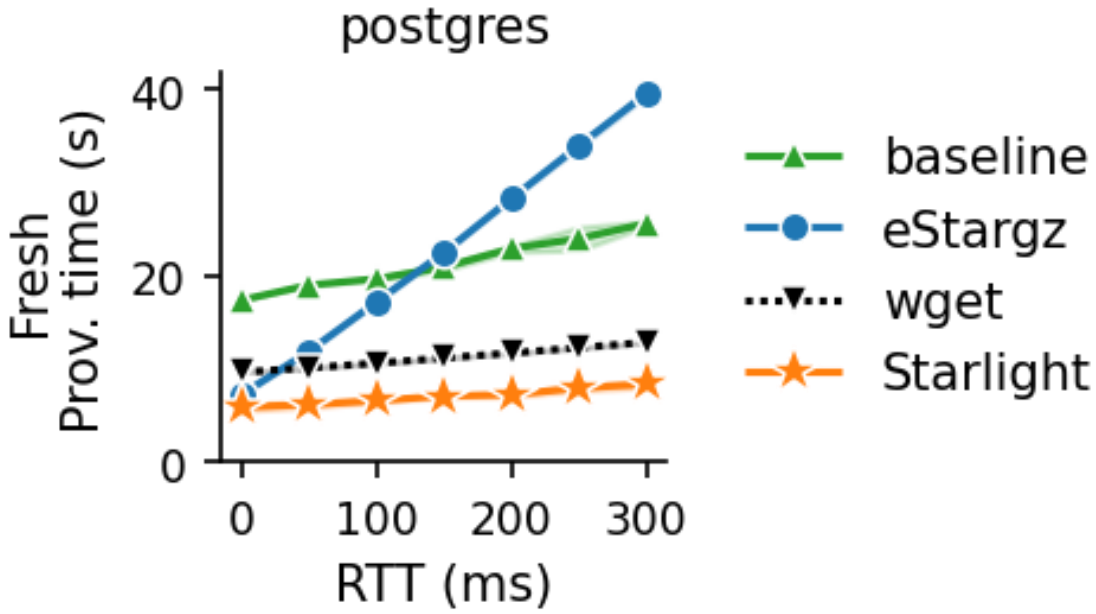
- X = RTT 0 to 300ms, 100Mbs
- Y = resulting provisioning time
  - **containerd v1.5.0**
  - **eStargz v0.6.3**: pull-based, on-demand
  - **Download** optimized delta bundle
    - Lower bound without early start



Effect of Round-trip Time

from [NSDI'22]

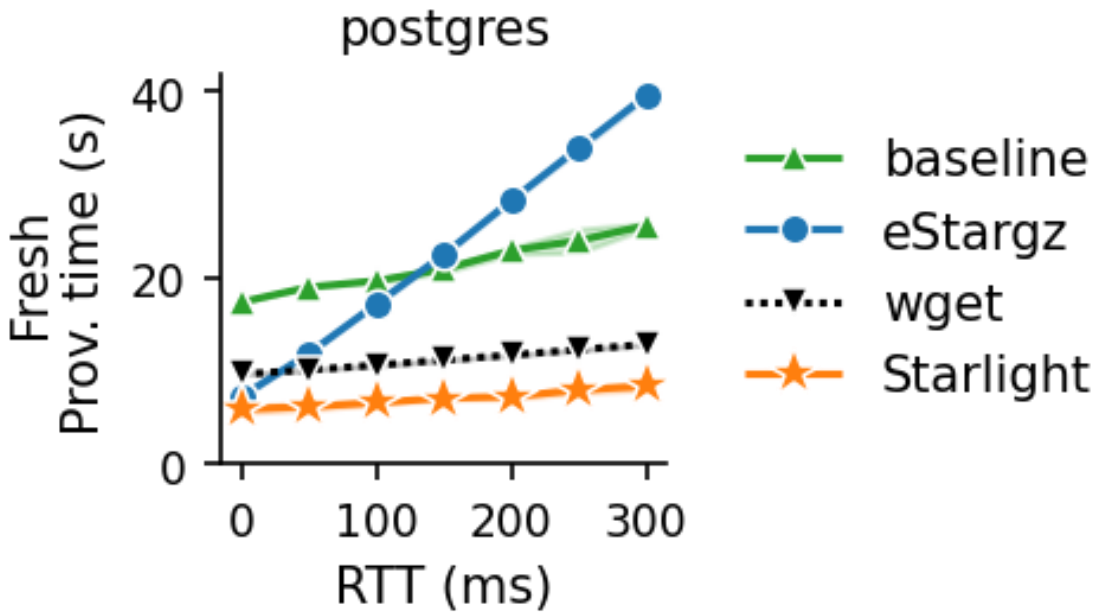
- X = RTT 0 to 300ms, 100Mbs
- Y = resulting provisioning time
  - **containerd v1.5.0**
  - **eStargz v0.6.3**: pull-based, on-demand
  - **Download** optimized delta bundle
    - Lower bound without early start
  - **Starlight**



Effect of Round-trip Time

from [NSDI'22]

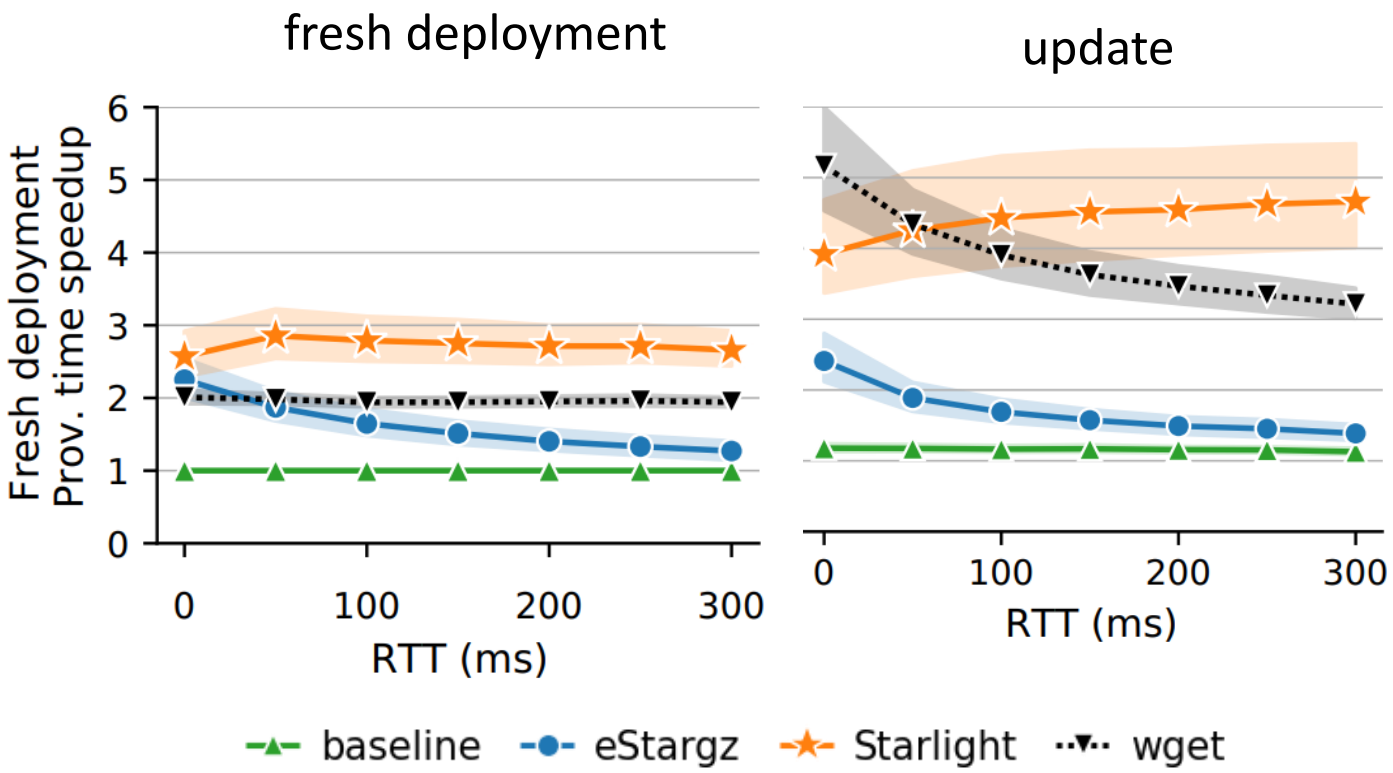
Starlight:
- ✓ Fastest (even with cloud RTT).
- ✓ Scales well with latency.
- ✓ Outperforms wget.



How Much Faster Are We?

from [NSDI'22]

speedup over containerd
(harmonic mean of 21 containers)



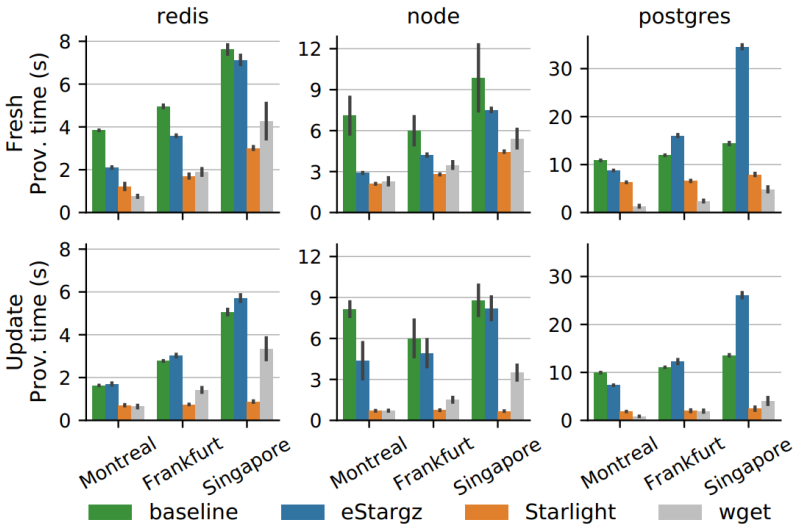
Starlight:

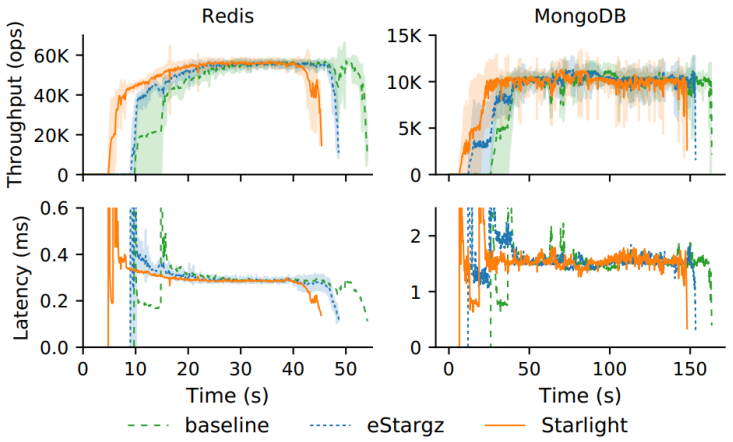
- ✓ Outperforms containerd 3x, estargz 1.9x
- ✓ Faster than wget
- ✓ Scales better with RTT
- ✓ Extremely fast updates: 4—5x compared to fresh

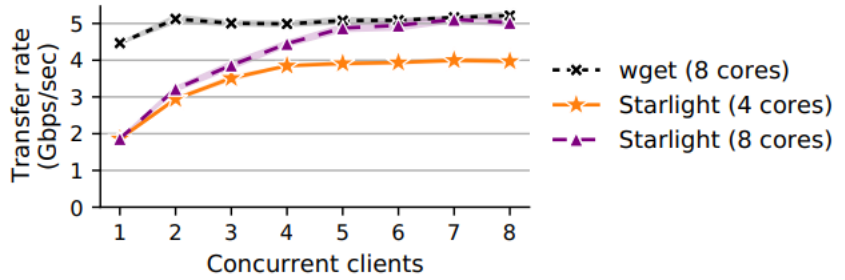
Additional Experiments

from [NSDI'22]

- On WAN → similar results.
- In cloud → Starlight fastest.
- Overhead.
- Varying bandwidth.
- Proxy scalability.



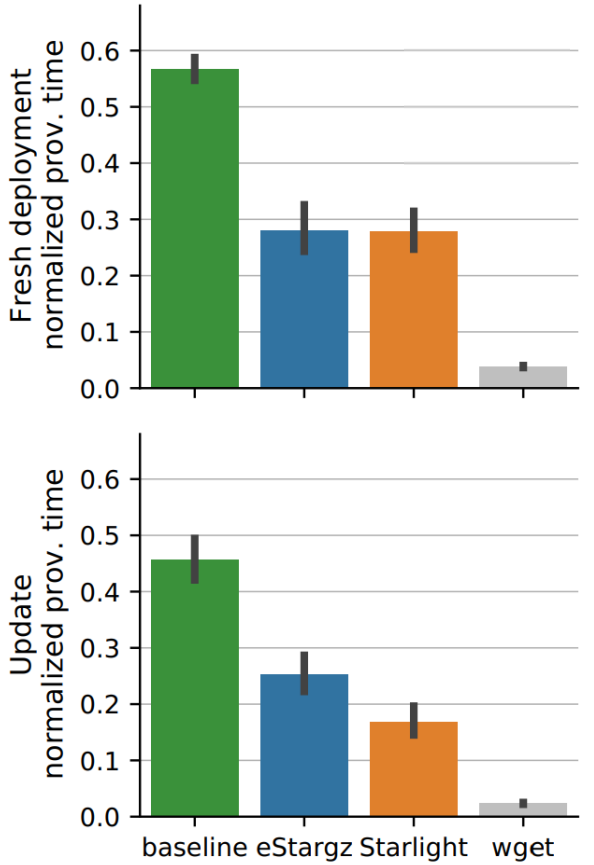




What about on Cloud?

from [NSDI'22]

- Amazon datacenter
- No limits on bandwidth, latency



Fresh

Updates

- ✓ Starlight and eStargz outperform containerd
- ✓ Starlight 35% faster than eStargz in updates
- ✓ At 5Gbps, hard to outperform wget

Example: Architecture Diagram

- From NSDI ‘22 again
- Architecture diagram with steps.
- Strategy:
  Show steps and interaction

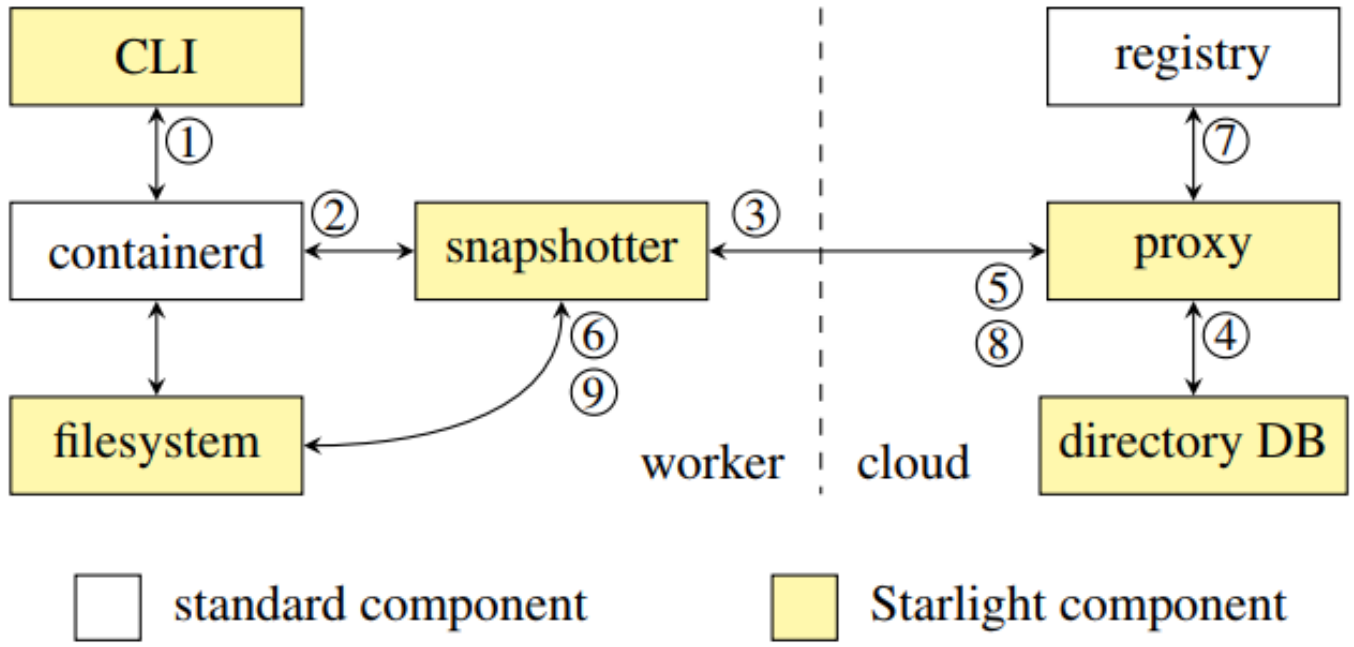
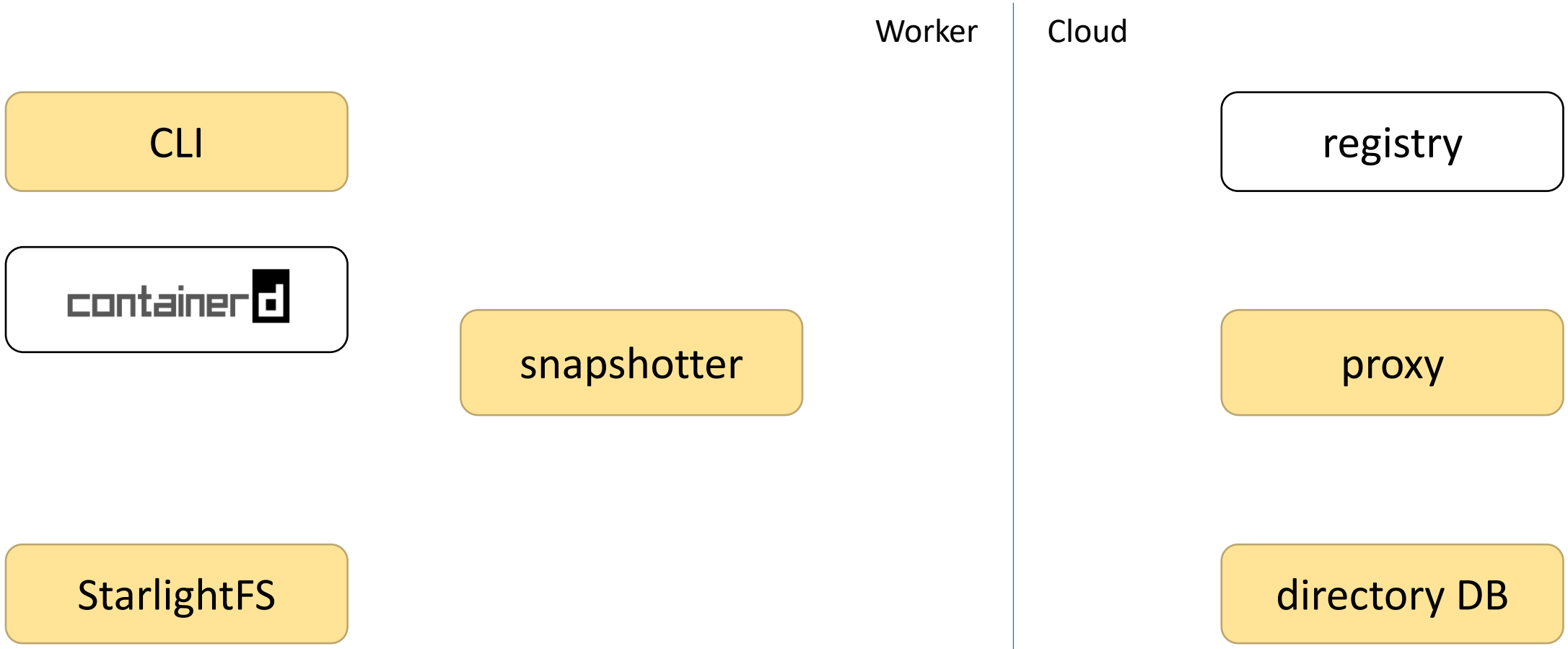


Figure 2: Starlight architecture.

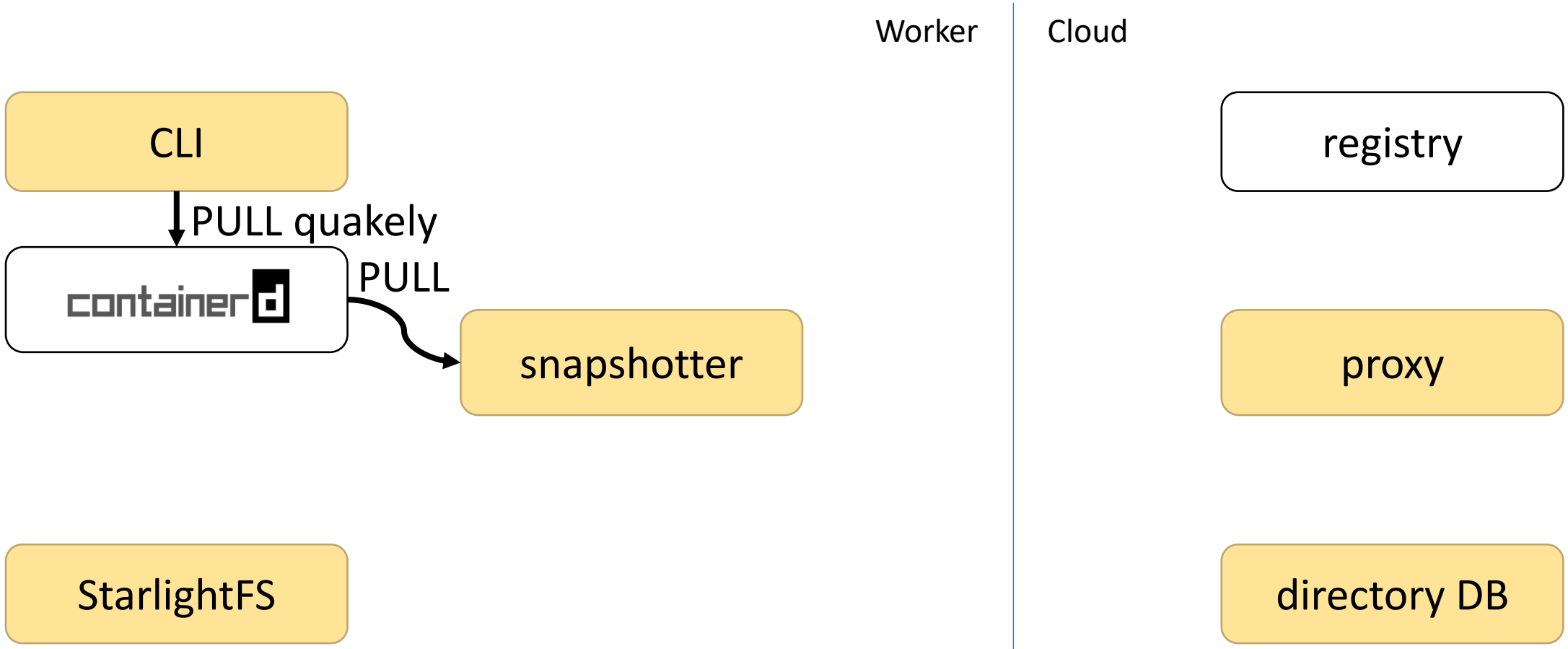
Starlight Operation

from [NSDI'22]



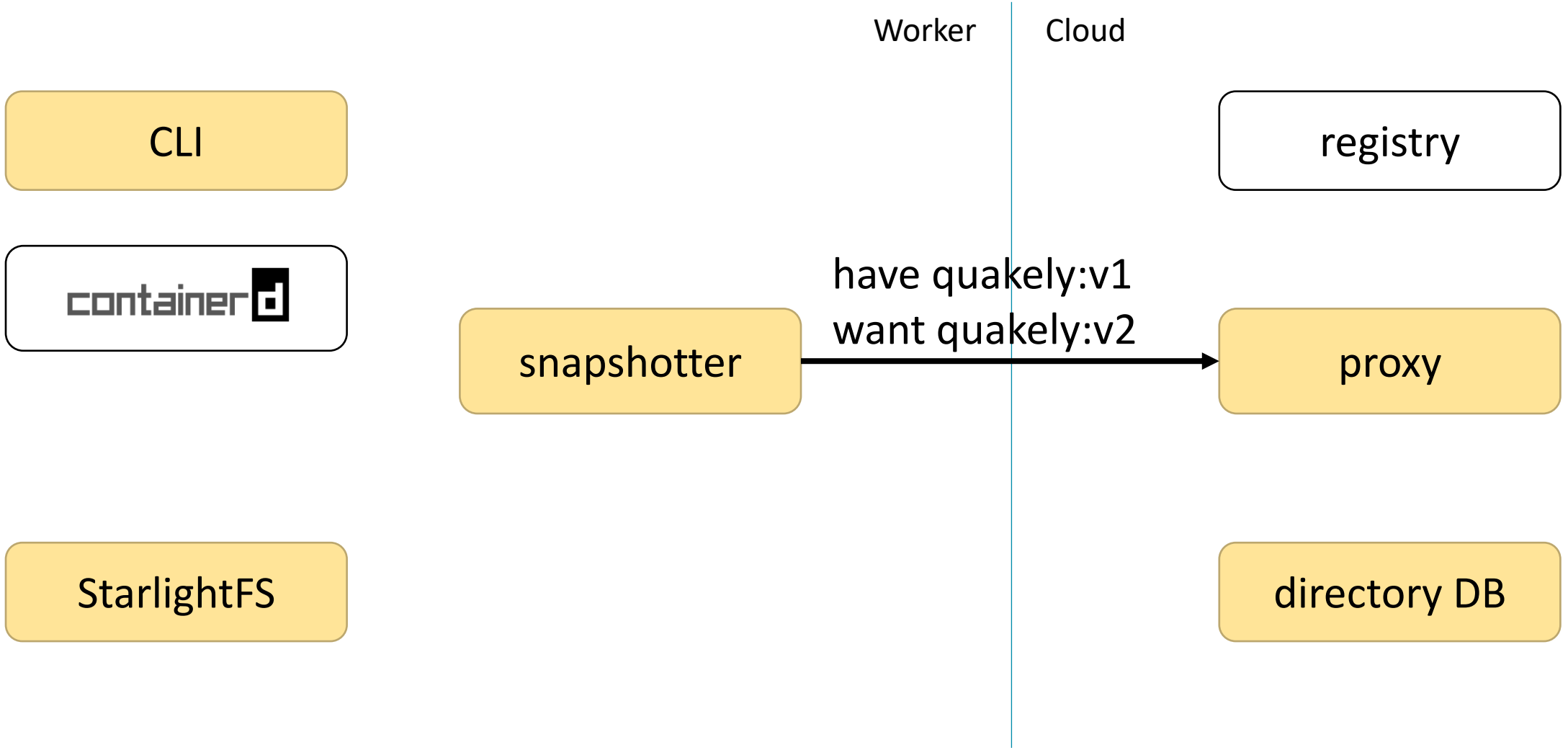
Starlight Operation

from [NSDI'22]



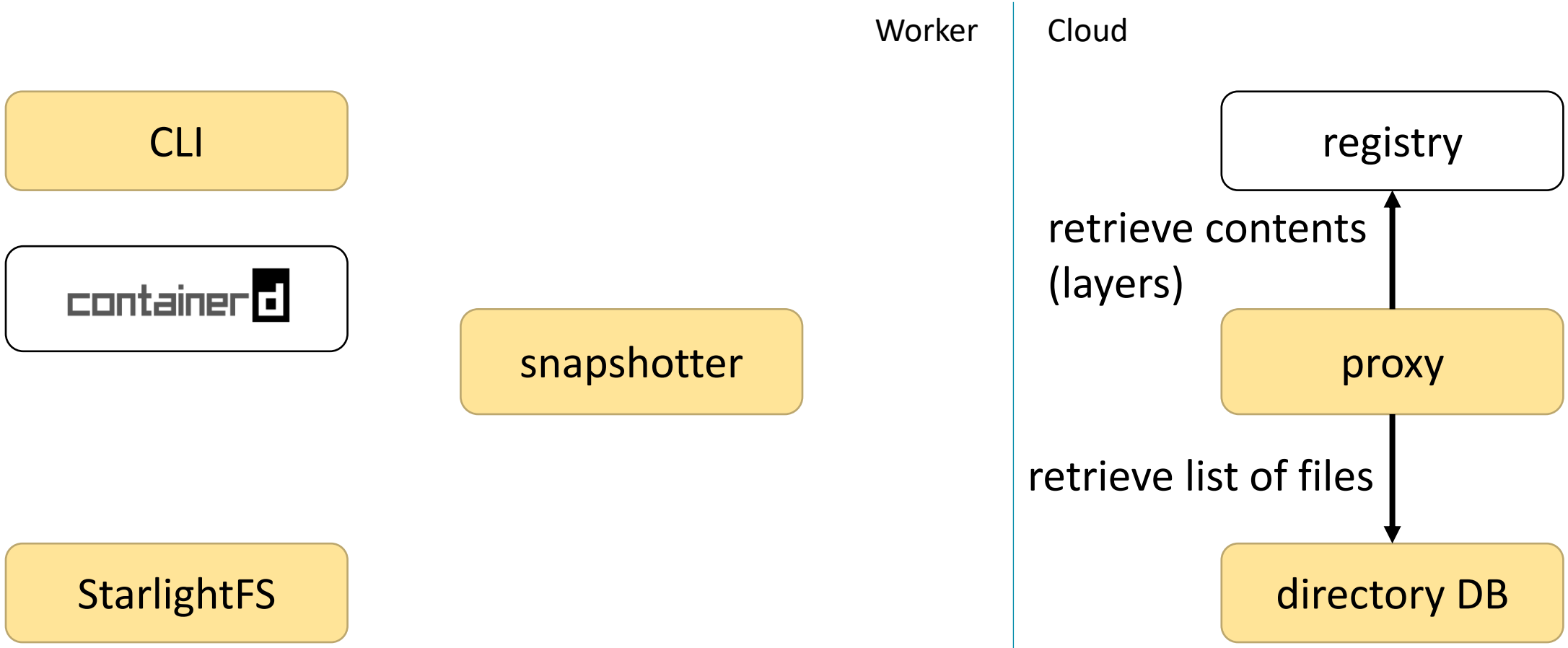
Starlight Operation

from [NSDI'22]



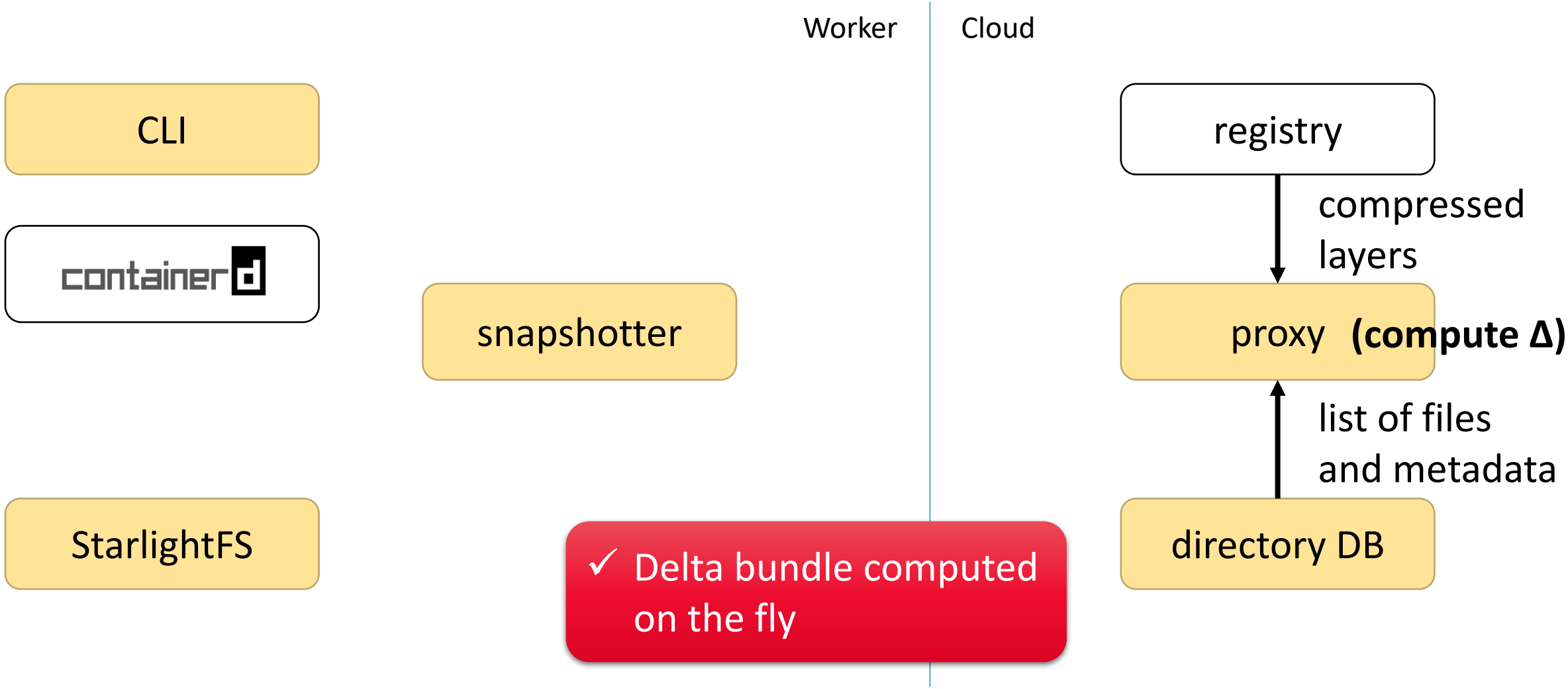
Starlight Operation

from [NSDI'22]



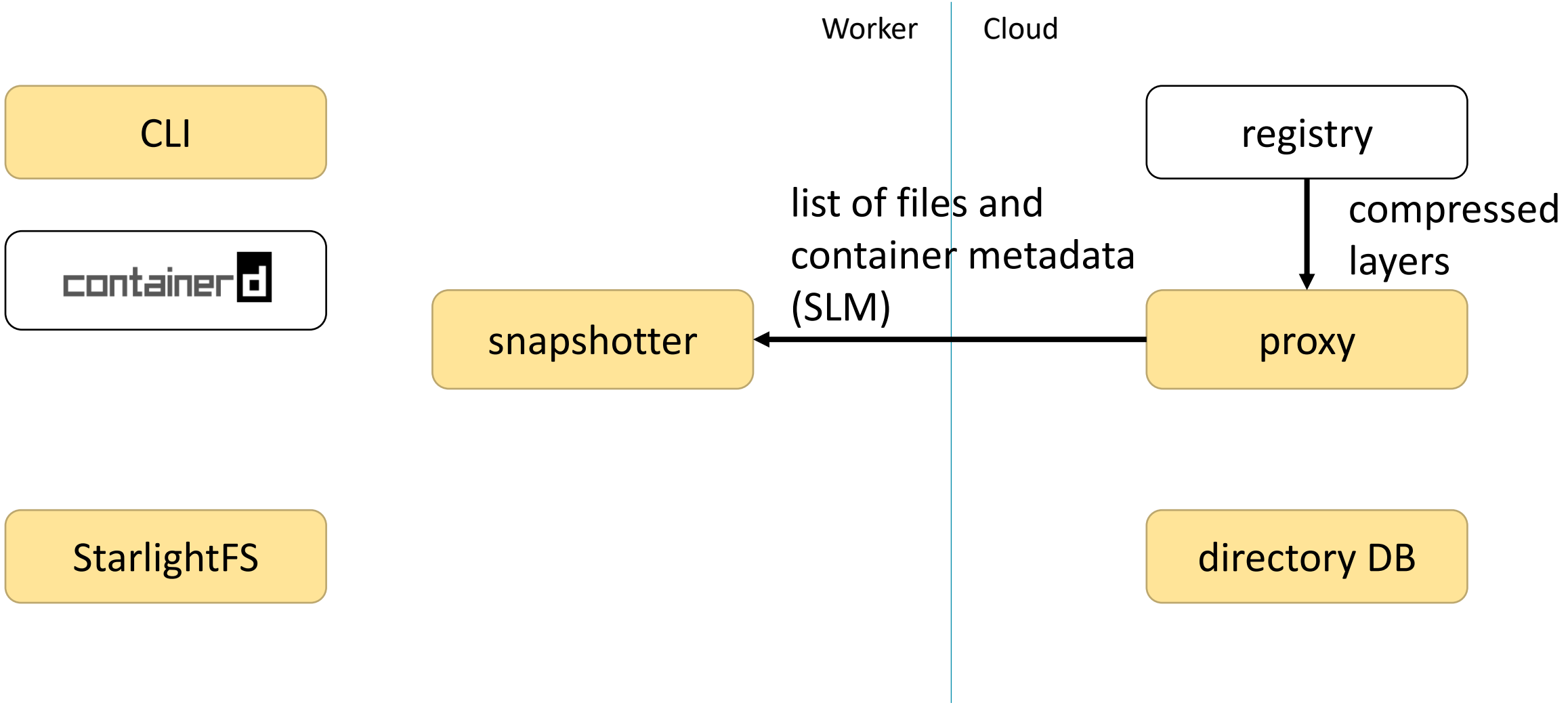
Starlight Operation

from [NSDI'22]



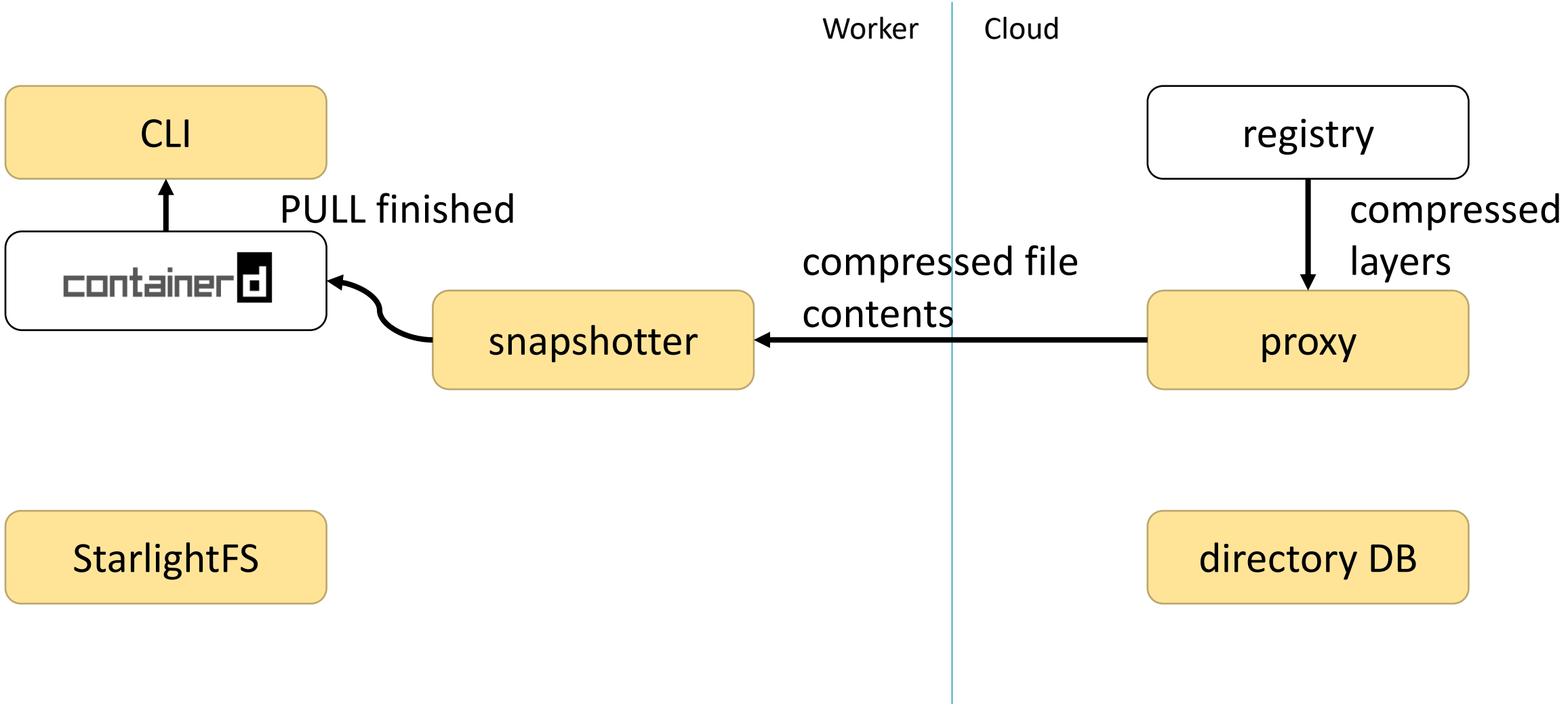
Starlight Operation

from [NSDI'22]



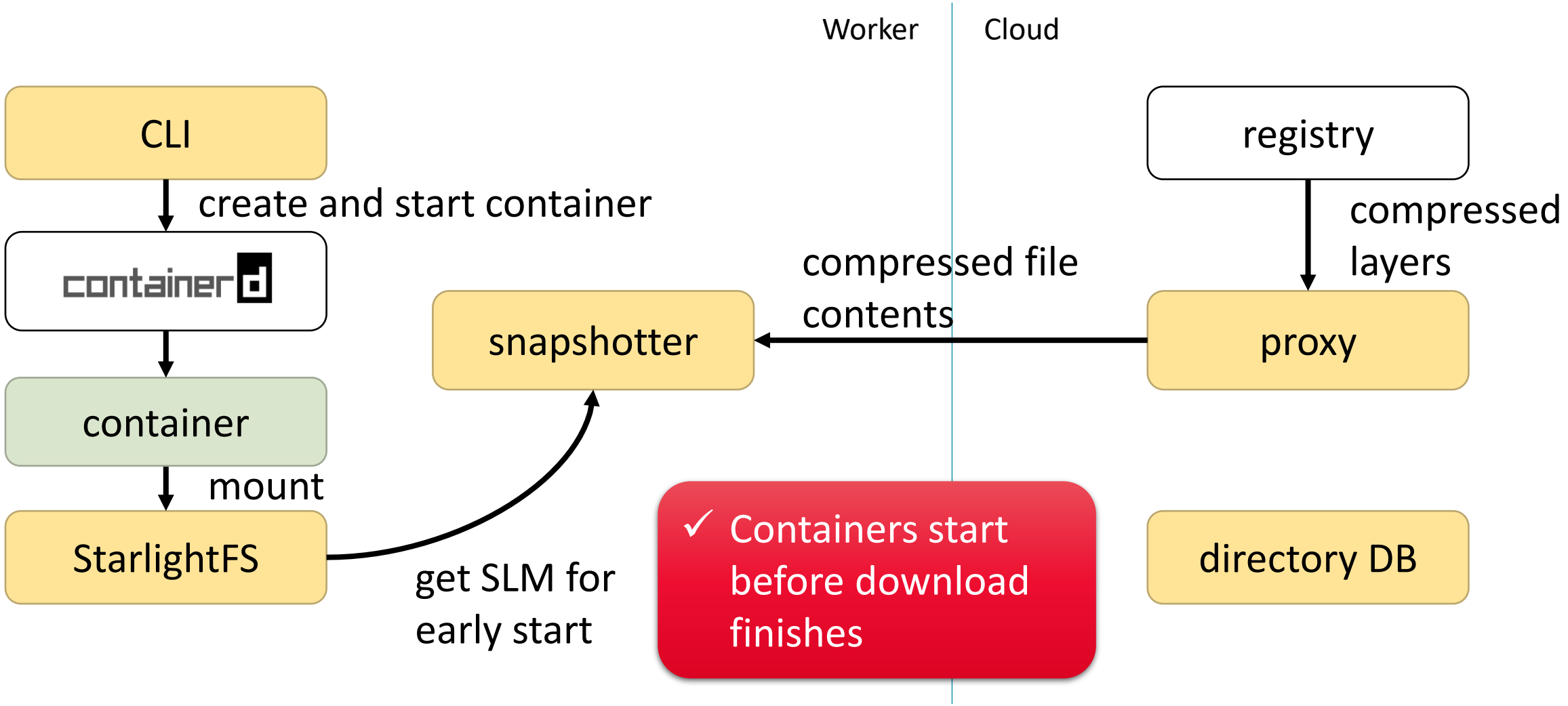
Starlight Operation

from [NSDI'22]



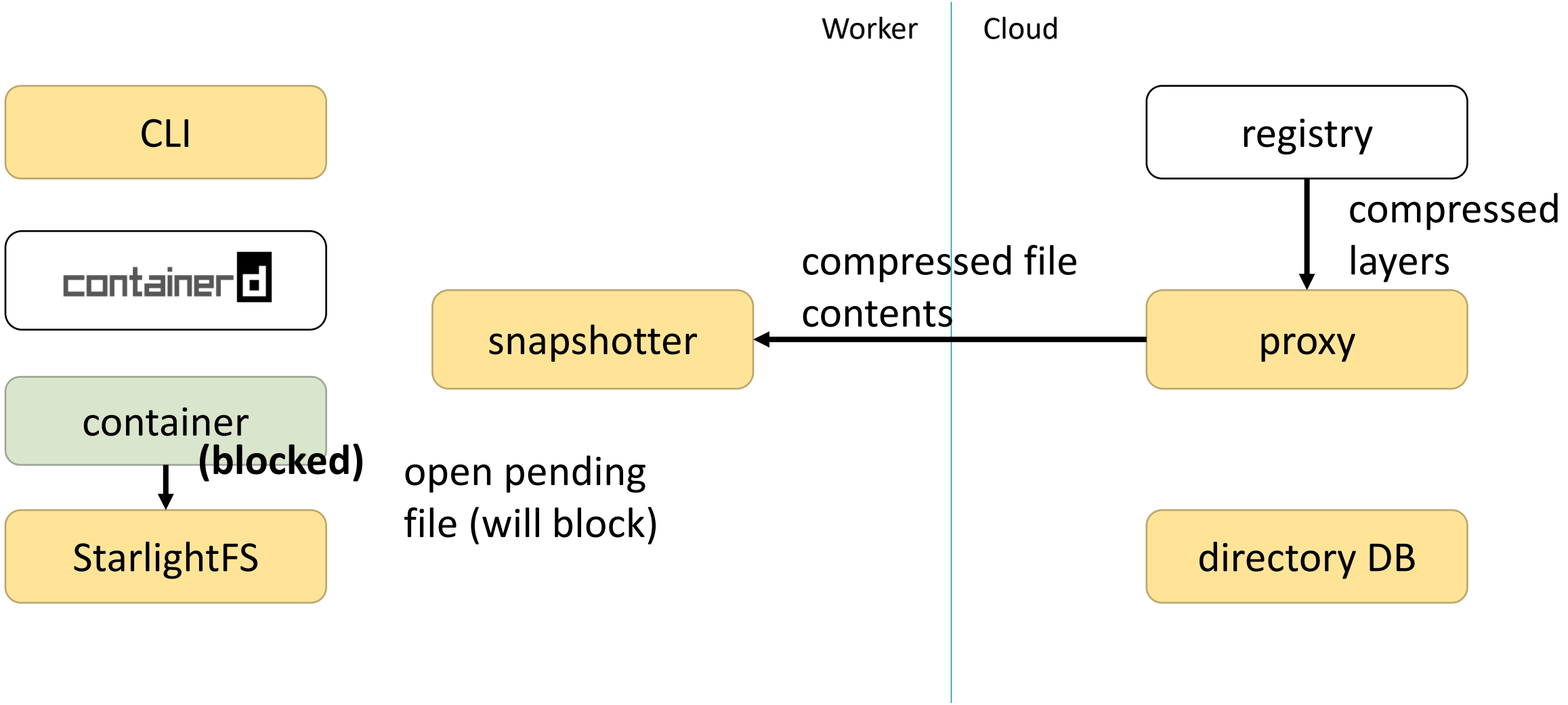
Starlight Operation

from [NSDI'22]



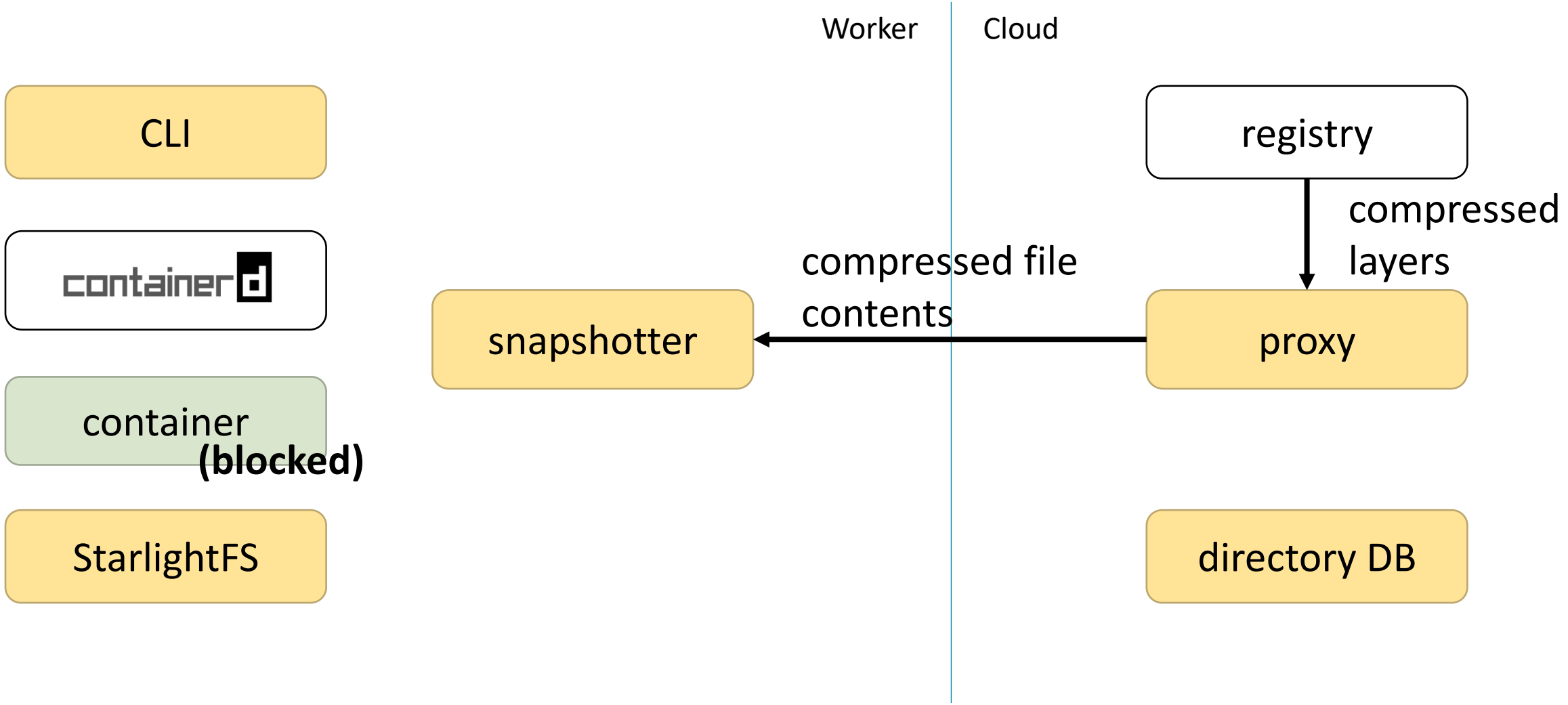
Starlight Operation

from [NSDI'22]



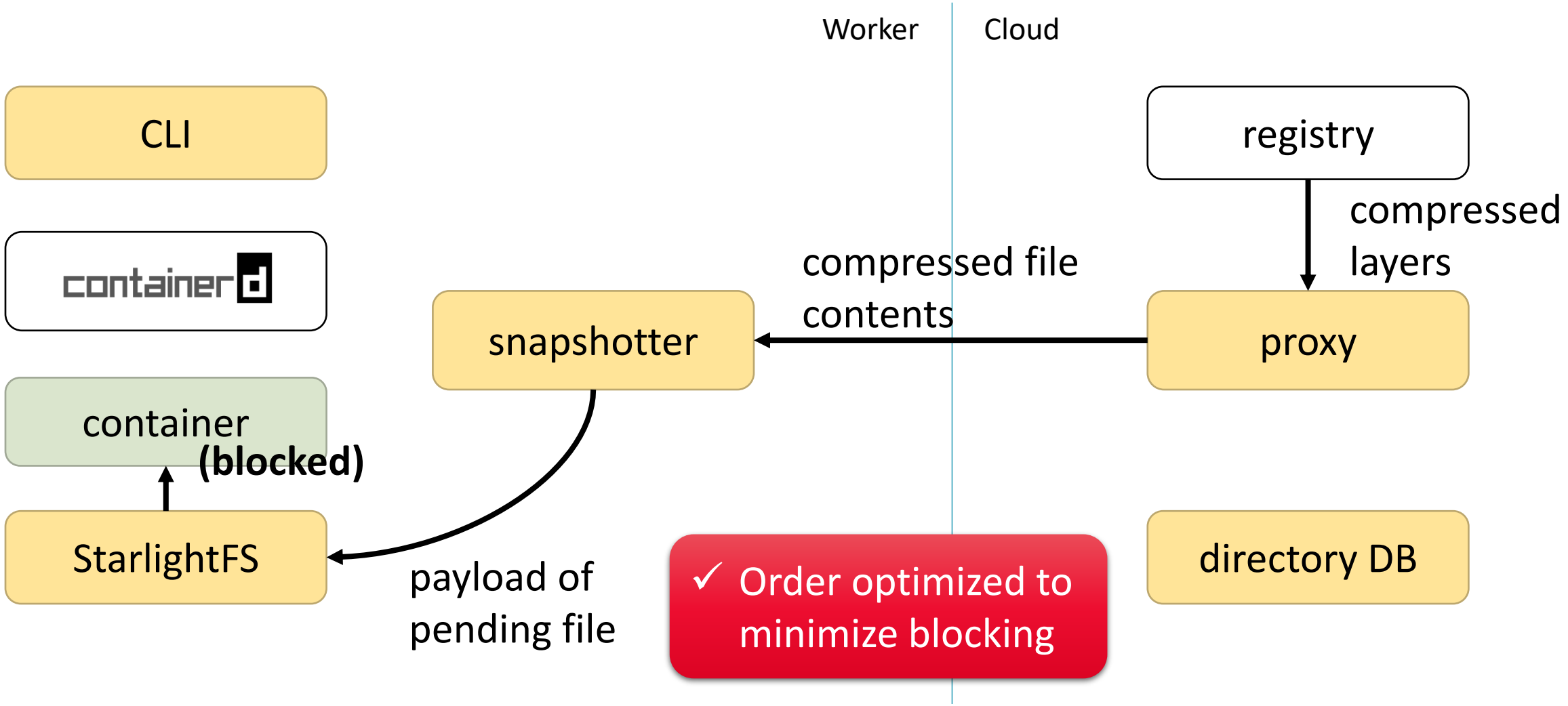
Starlight Operation

from [NSDI'22]



Starlight Operation

from [NSDI'22]



Thank you!