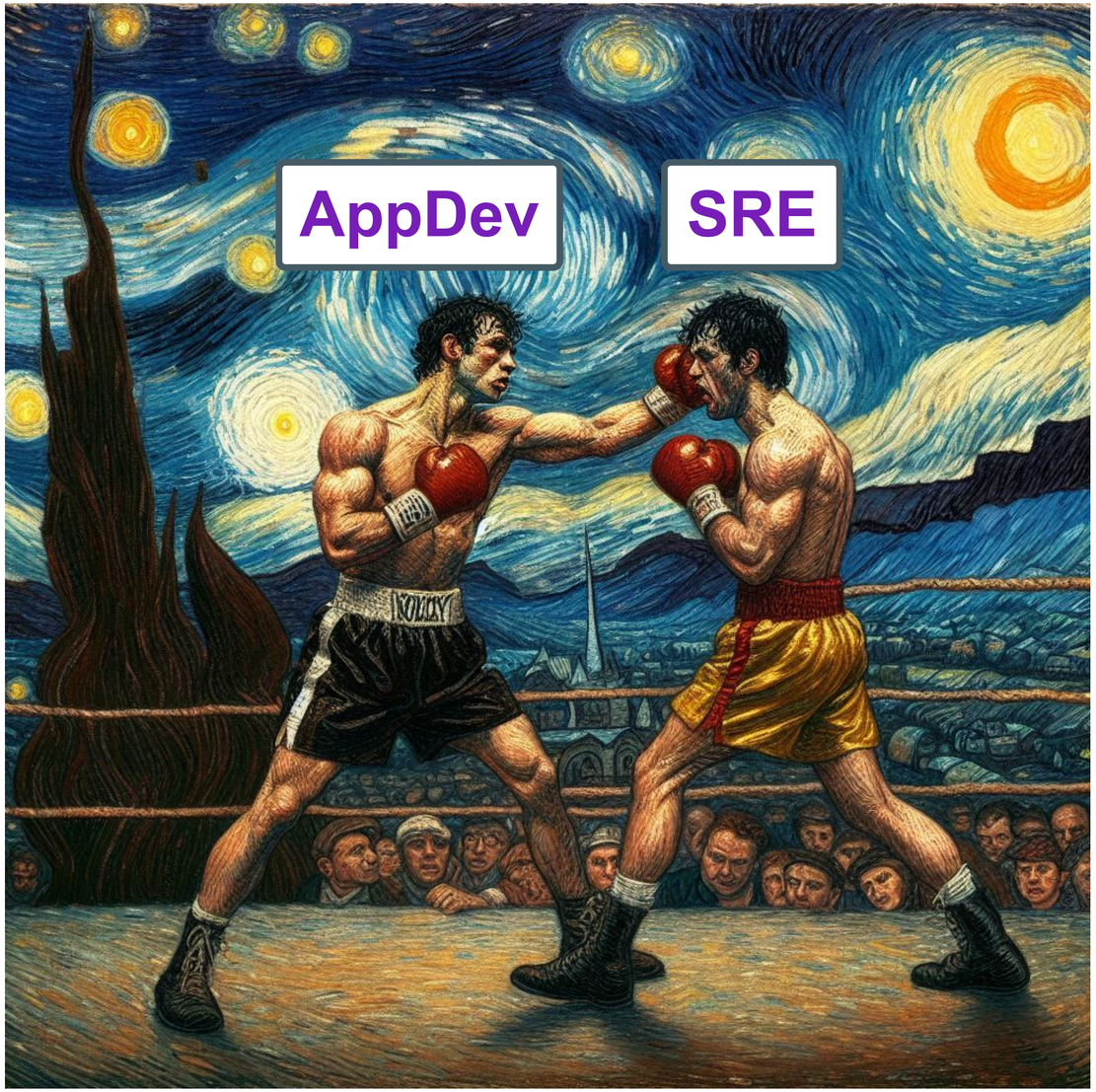
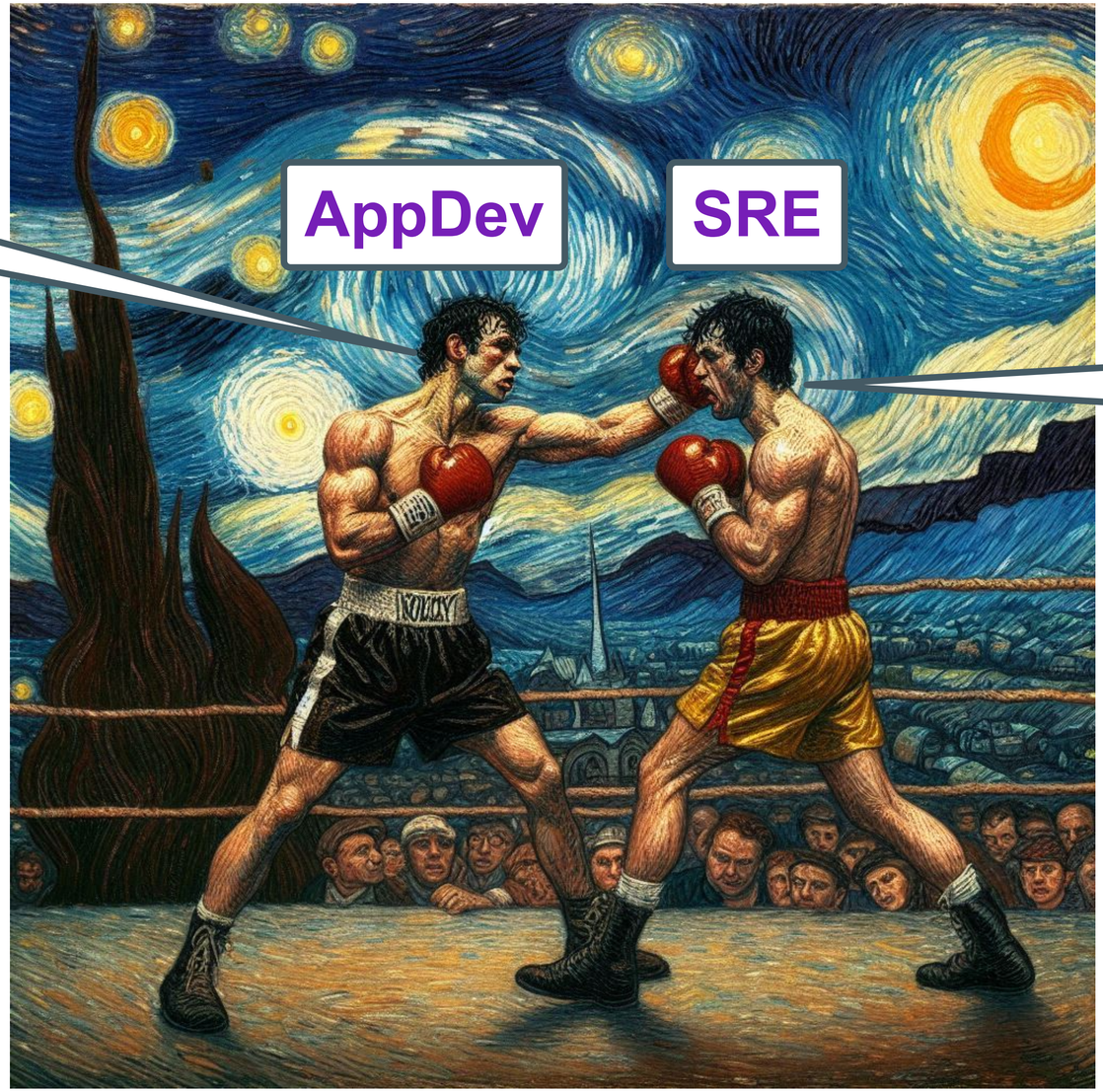
Rock around the (synchronization)

Improve performance with high precision time!

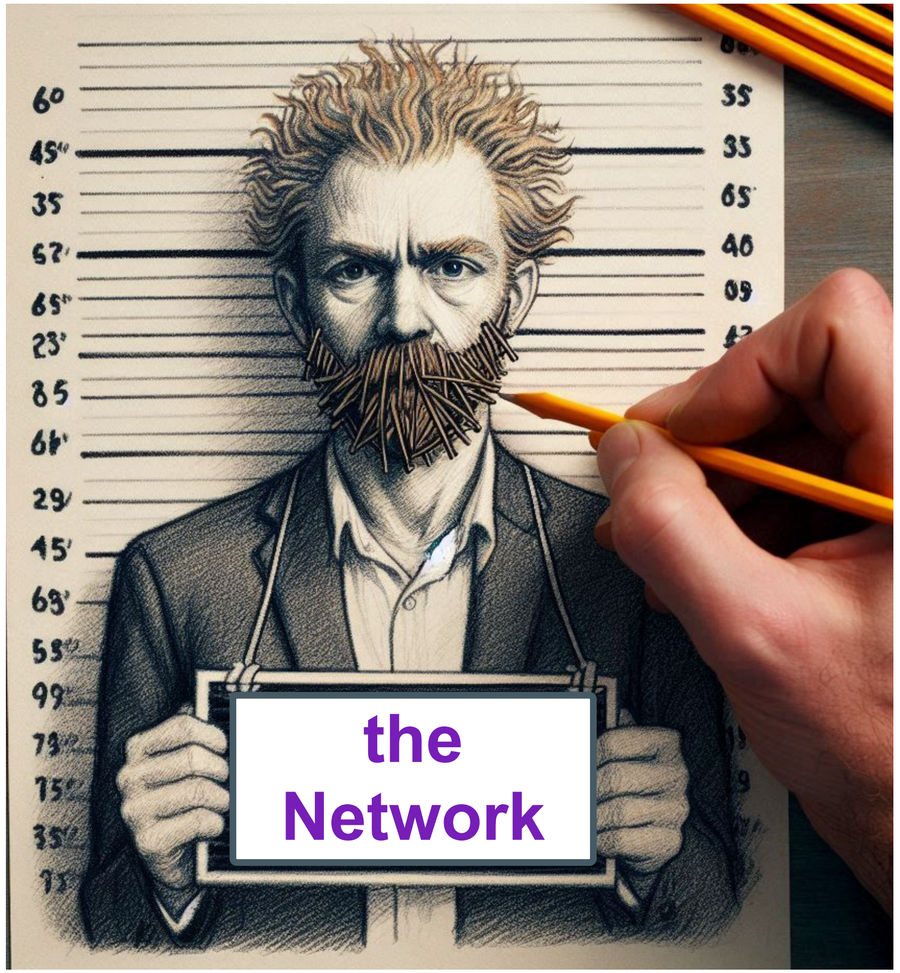
Lerna Ekmekcioglu
Sr Solutions Engineer, Clockwork Systems


AppDev
SRE

My app is slow!
It’s the network!👉



Is it the app or 👉



?

Demo: ↗️  latencies

Ecommerce site

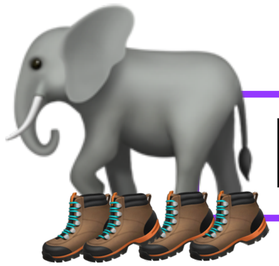
Demo: ↗️  latencies

Network pipe

Ecommerce site

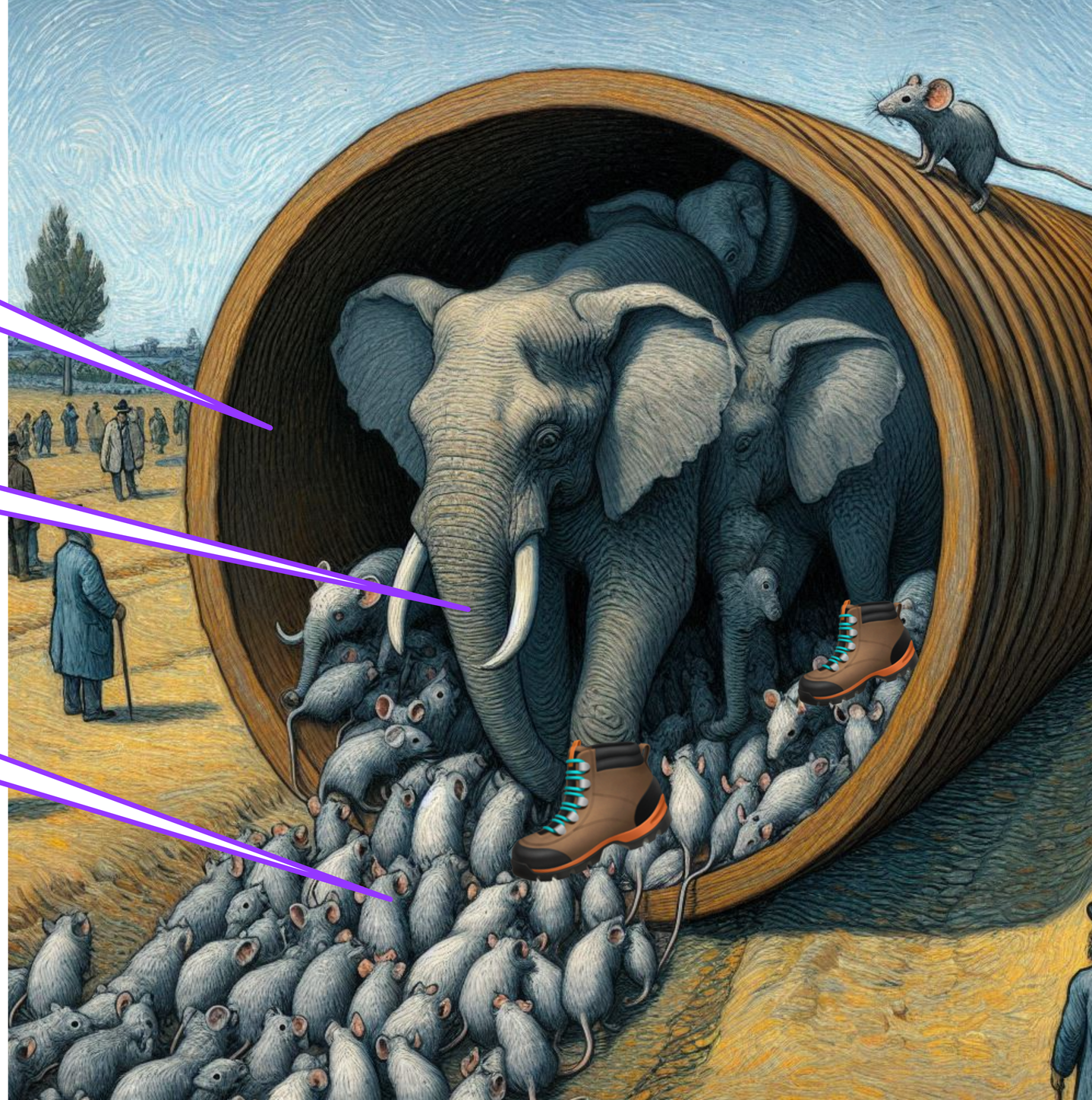
Demo: ↗️ 🐿️ latencies

Network pipe

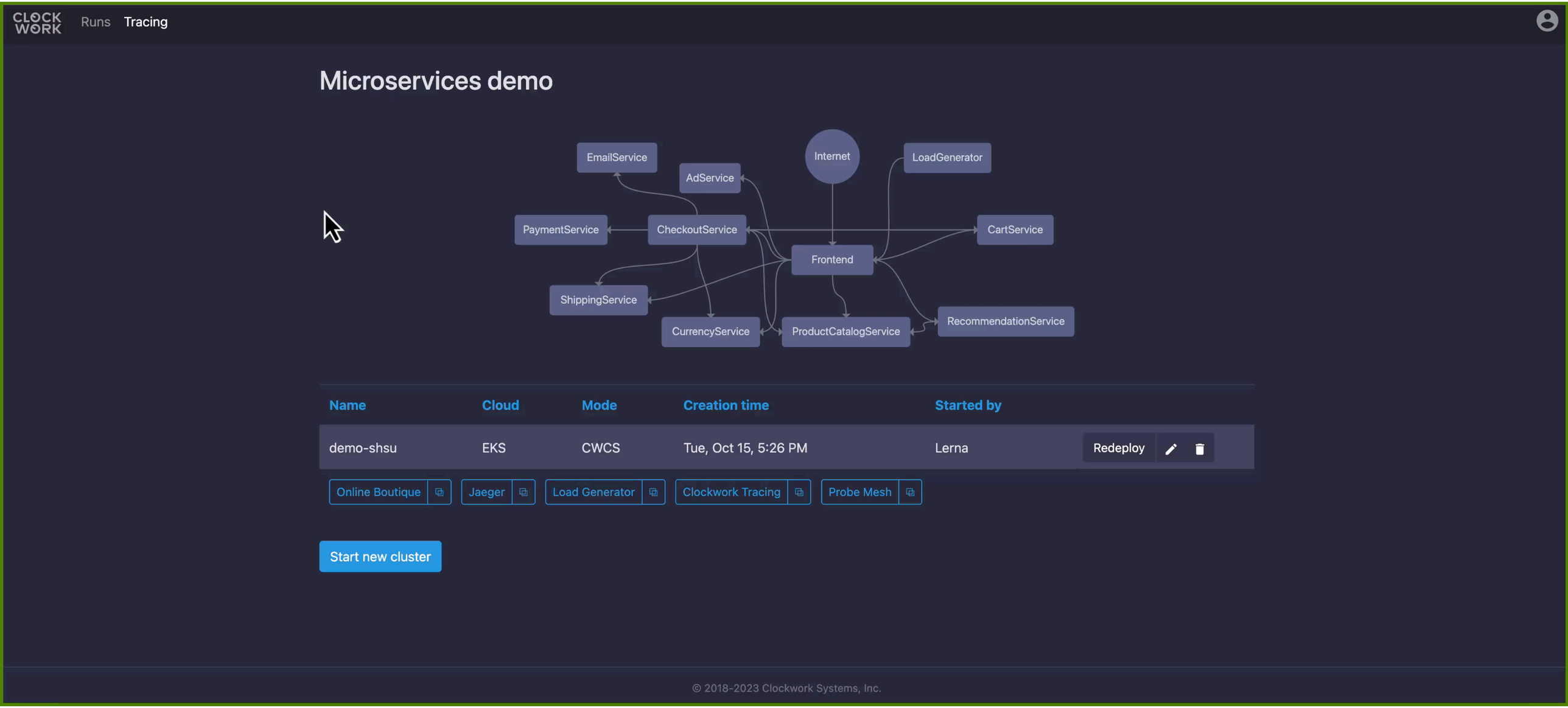


Data workload

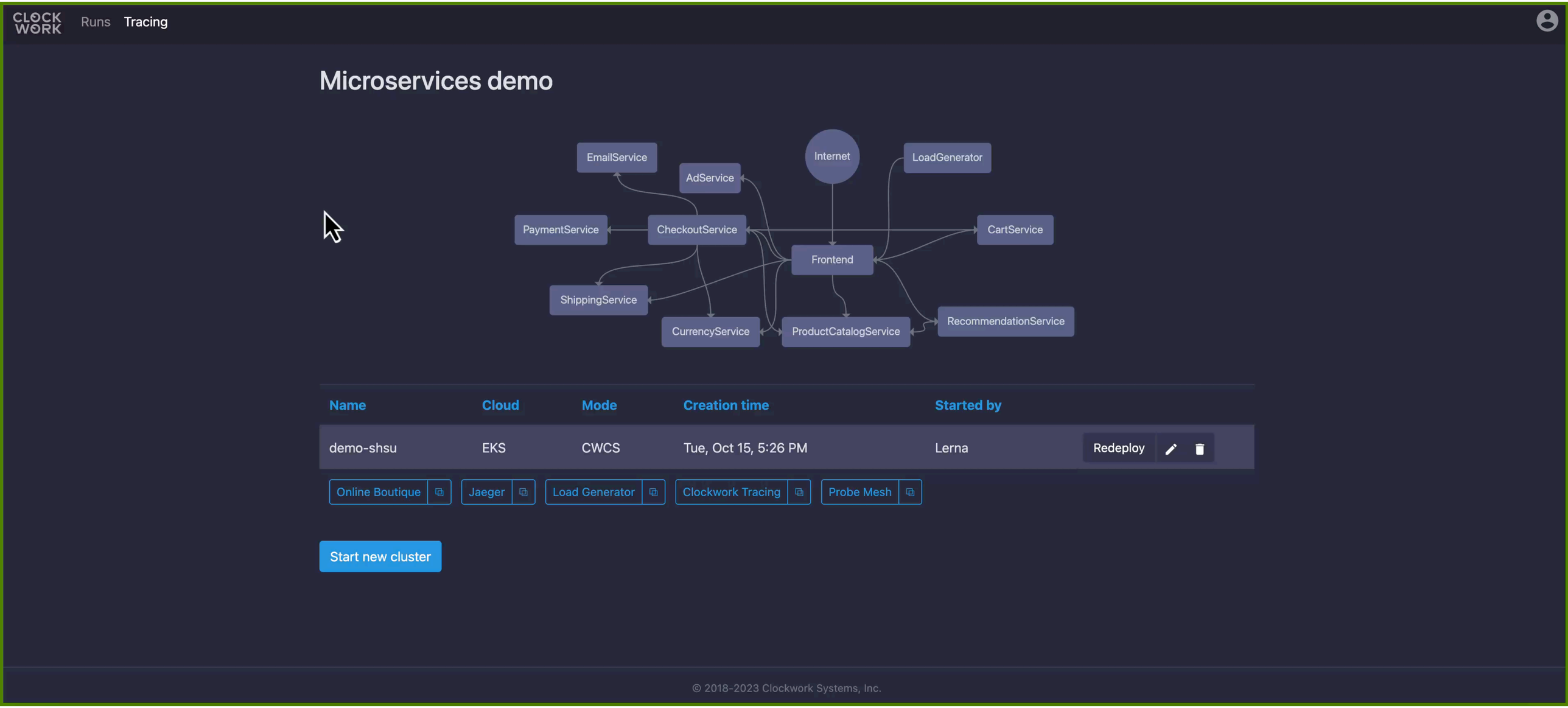
Ecommerce site



Demo: ↗️ tail latencies for 🐭



Demo: ↗️ tail latencies for 🐭



Why do we need accurate time?

Minimize tail latencies and control congestion

Performance

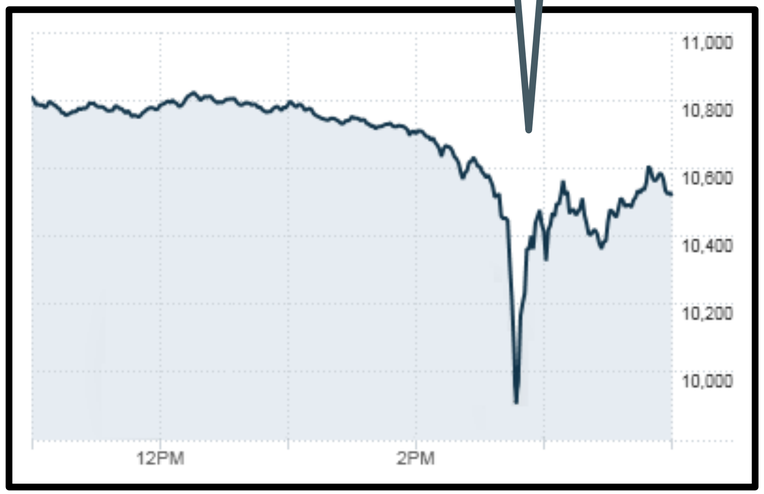
Why do we need accurate time?

Minimize tail latencies and control congestion

👩‍💻

Performance

Prevent market abuse



👩🏽‍📢

Max divergence from UTC 100µs

Compliance

Flash crash, CC0

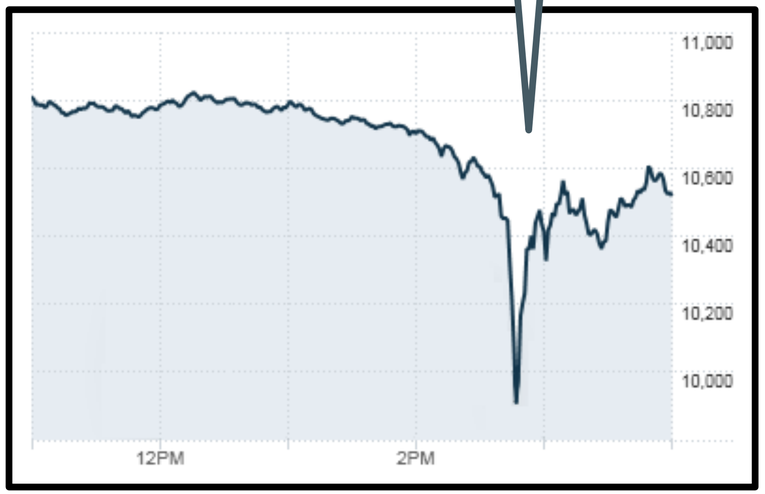
Why do we need accurate time?

Minimize tail latencies and control congestion

👩🏽‍💻

Performance

Prevent market abuse



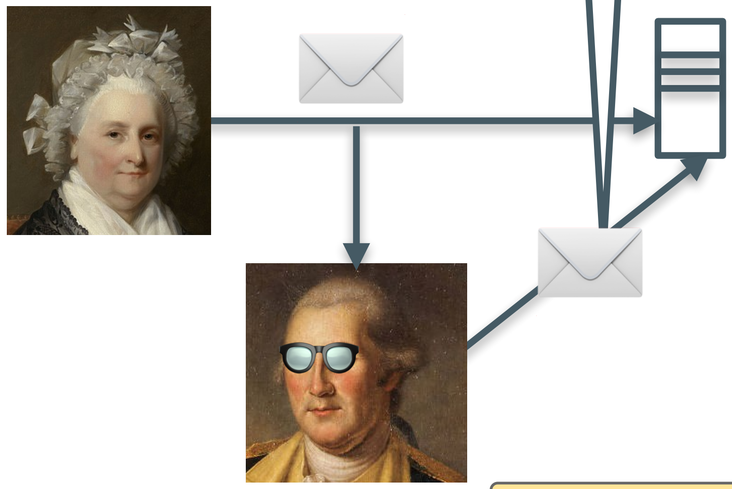
👩🏽‍📣

Max divergence from UTC 100µs

Compliance

Flash crash, CC0

Prevent replay attacks



👨🏽

Accurate timestamps on requests

Security

George Washington, CC0

Martha Washington, CC0

🕙 It’s 10:00!

🕯️ 8 to 12?

💦 9:00?

Candle clock
CC BY-SA 3.0 DEED

Ancient Persian water clock
CC BY-SA 3.0 DEED

Pendulum Clock, CC0

🕙 It’s 10:00!

🕯️ 8 to 12?

[Candle clock](#)
[CC BY-SA 3.0 DEED](#)

💦 9:00?

[Ancient Persian water clock](#)
[CC BY-SA 3.0 DEED](#)

It’s **10:01!**

[Pendulum Clock](#), [CC0](#)

Better precision 🎯

Precise frequency

Quartz crystal

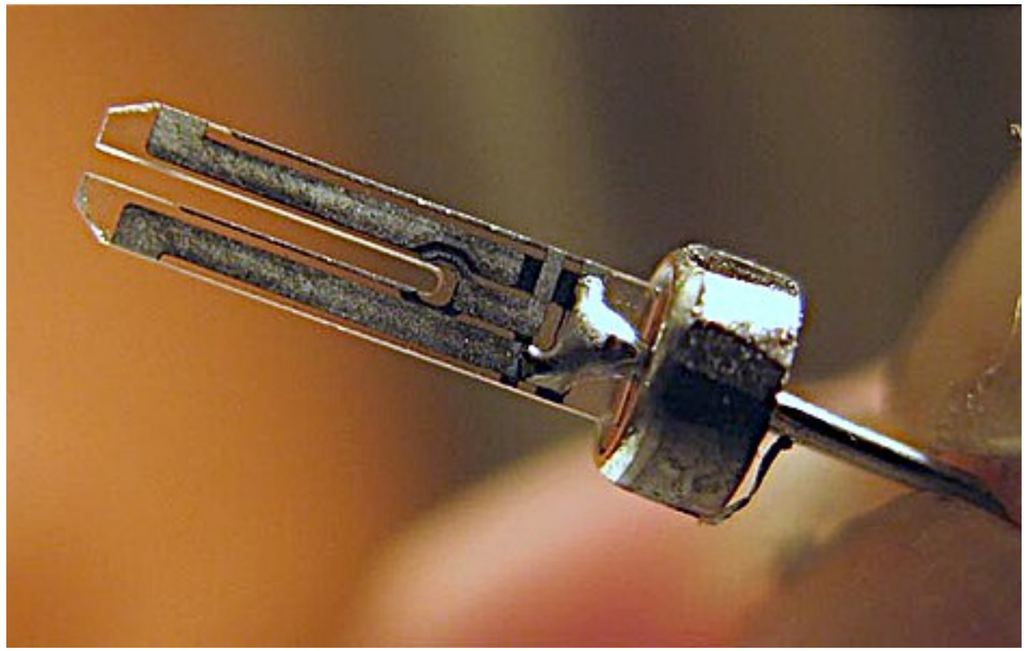
Precise frequency

It’s (Piezo)electric!

⚡electric field ↔️ mechanical force 💪

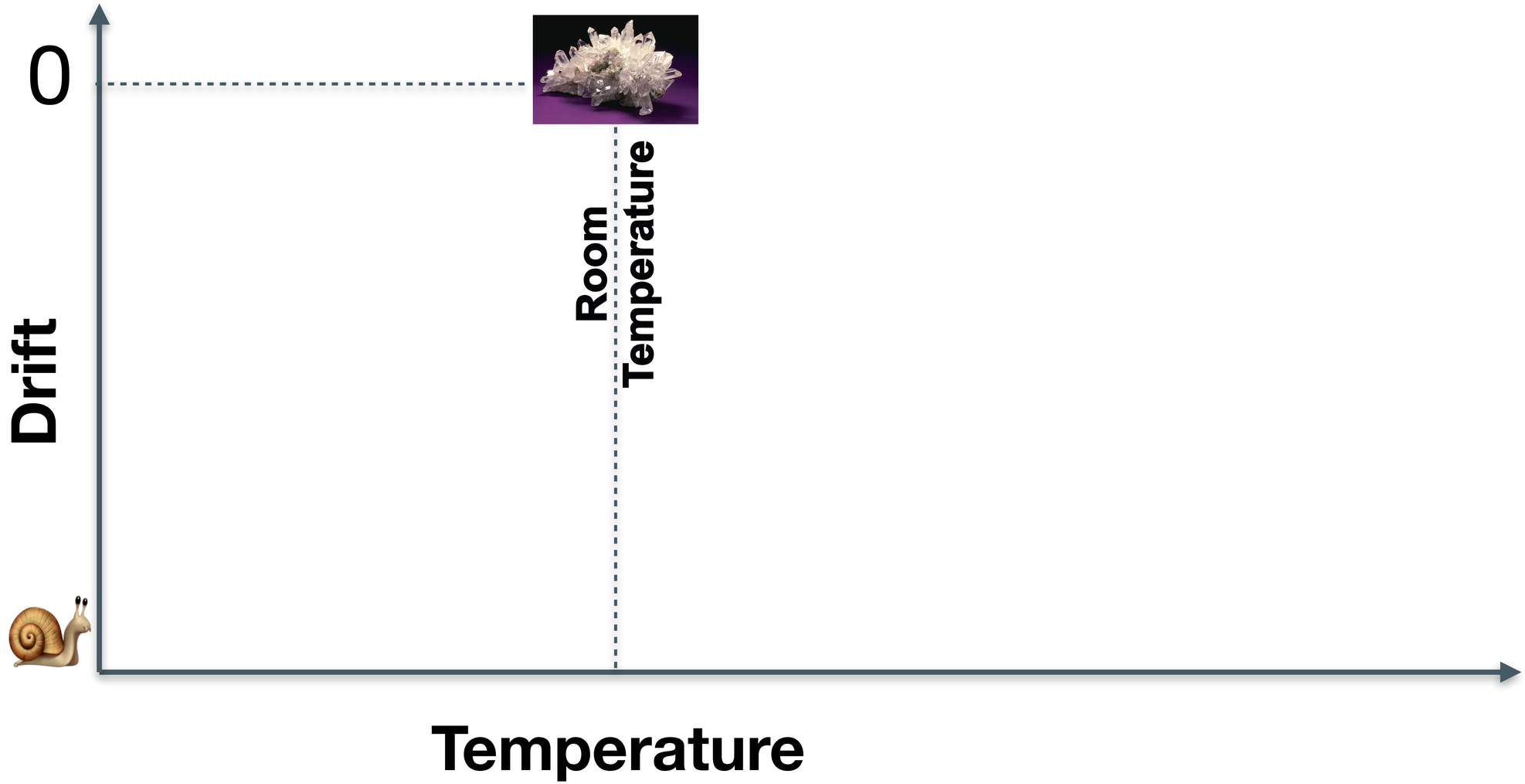
[Quartz crystal](#)

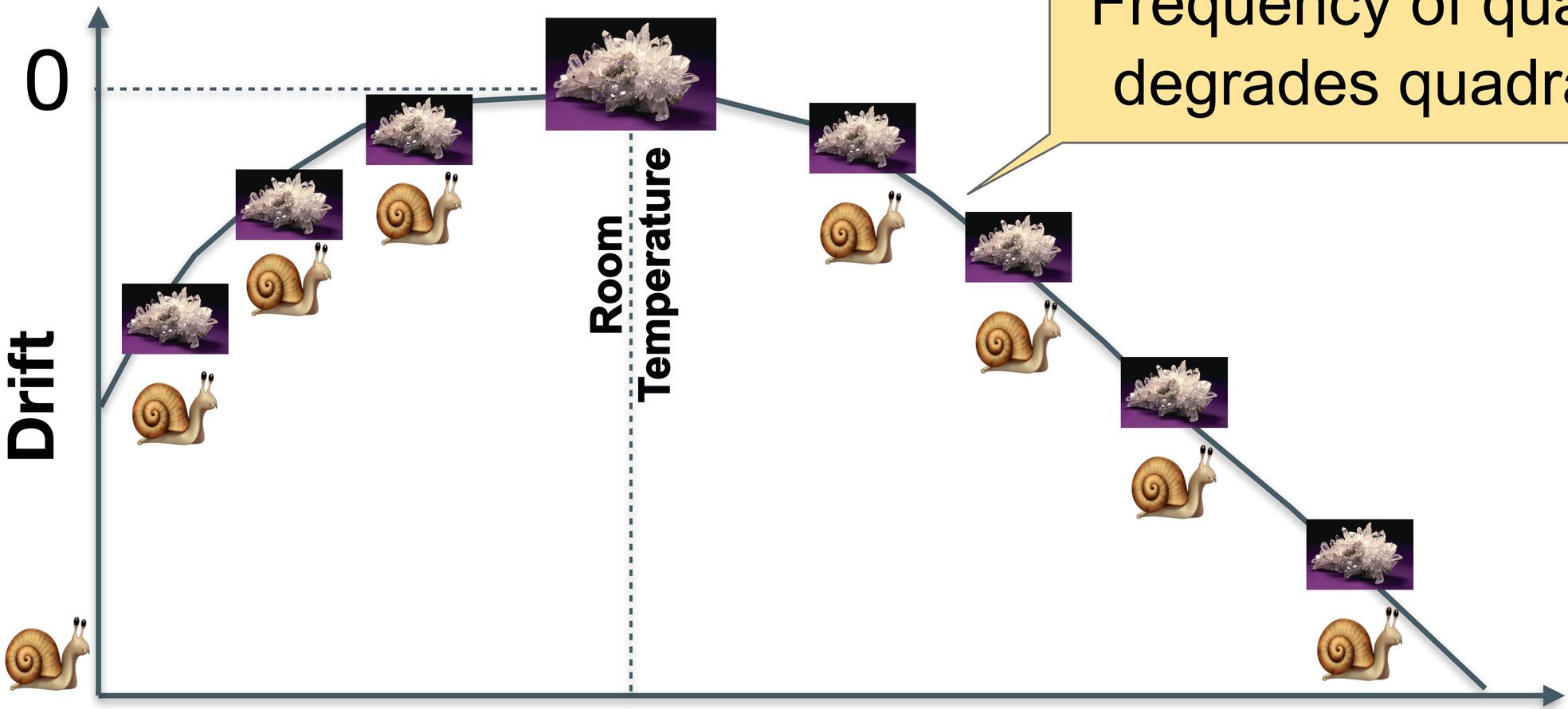
CC0



[Quartz crystal tuning fork](#)

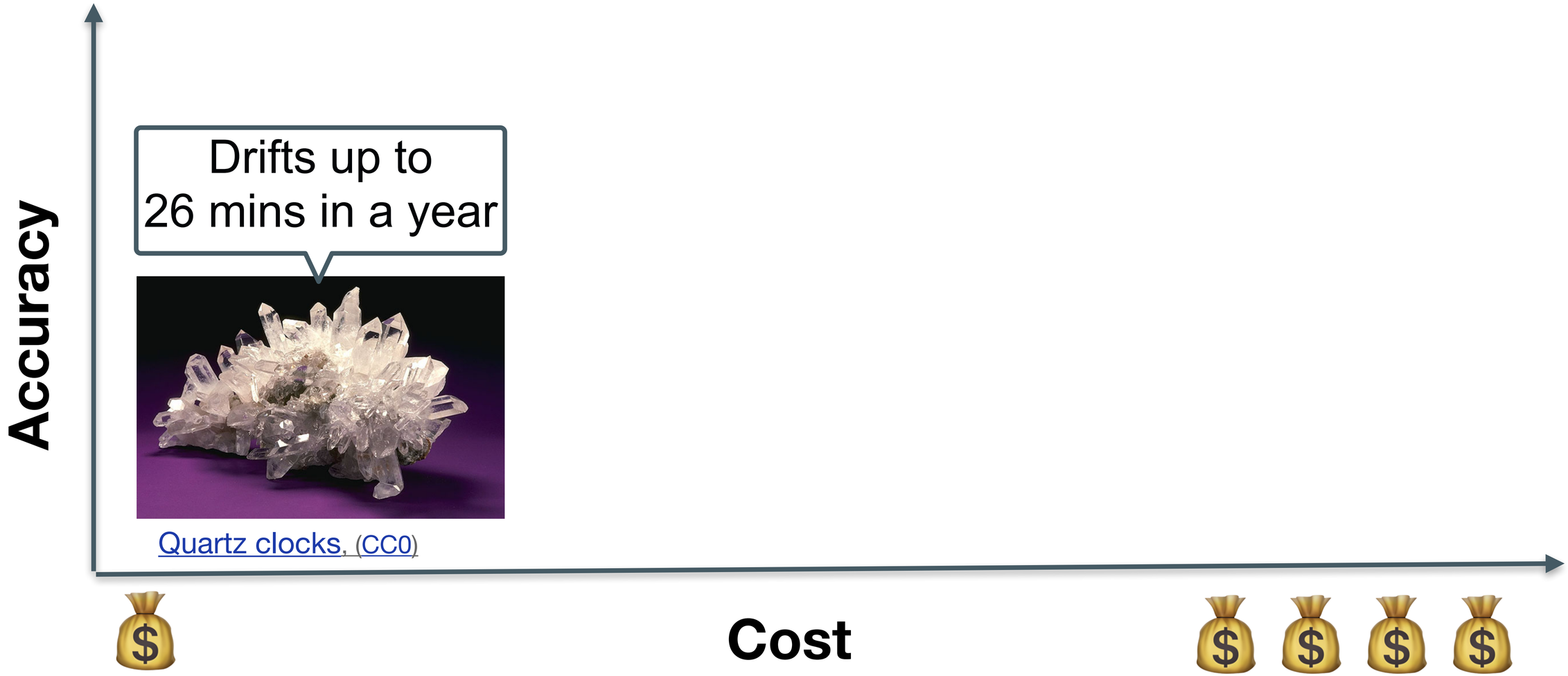
CC0


0
Drift
Room Temperature
Temperature



Frequency of quartz crystal degrades quadratically 🔥

Time sources



Time sources

Drifts ±1second
in millions of years

Atomic clocks

(CC BY-SA 3.0 DEED)

Drifts up to
26 mins in a year

Quartz clocks, (CC0)

Accuracy

Cost

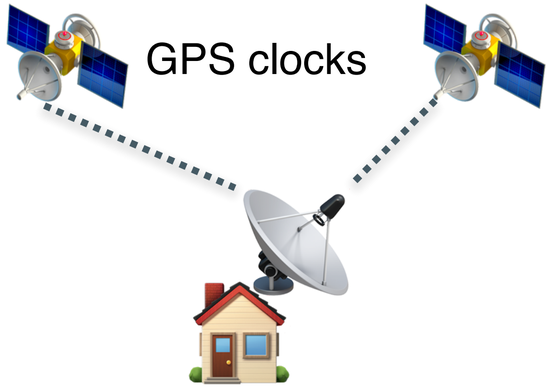
💰

💰 💰 💰 💰

Time sources

Drifts ±1second
in millions of years

GPS clocks



Drifts up to
26 mins in a year

Quartz clocks, (CC0)

Atomic clocks

(CC BY-SA 3.0 DEED)

Accuracy

Cost

💰

💰 💰 💰 💰

Bonnie and Clyde

Bonnie and Clyde, Library of Congress collection

Bonnie and Clyde

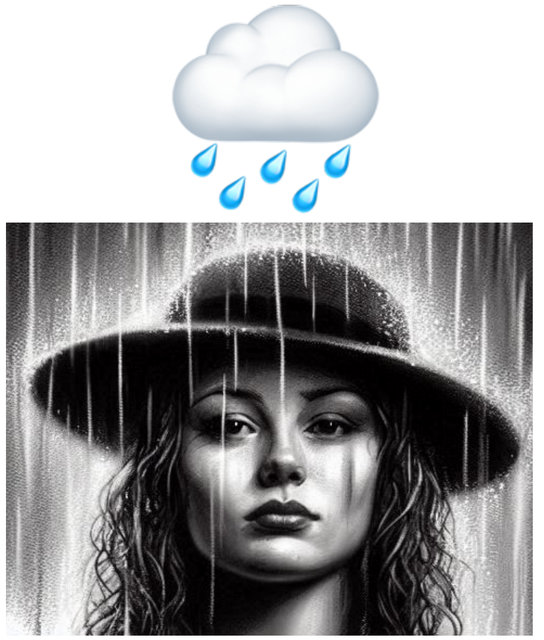
Bonnie and Clyde, Library of Congress collection

(CC0)

Bonnie and Clyde

(CC0)

Bonnie and Clyde



Bonnie and Clyde

Synchronize clocks

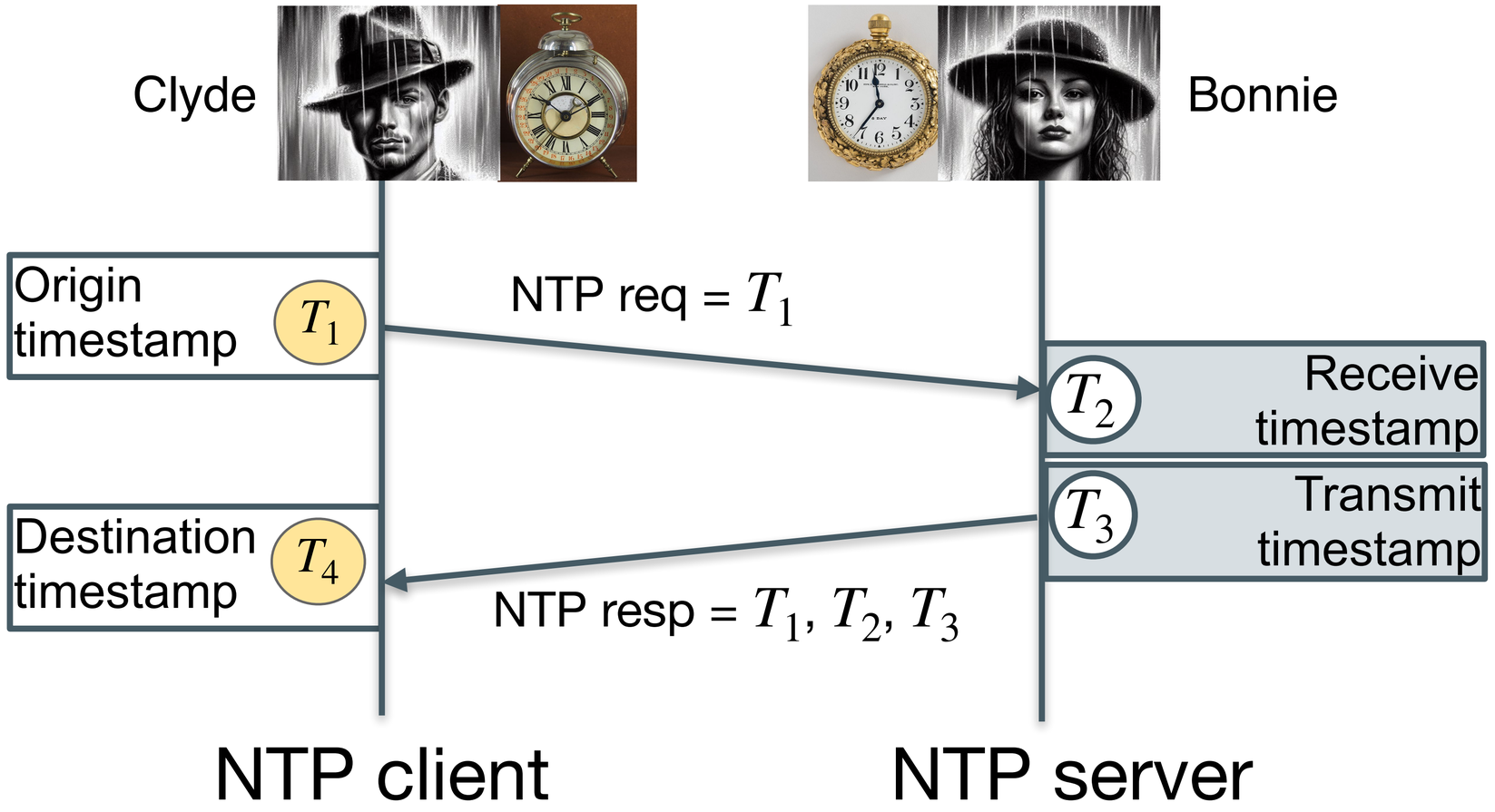
NTP client

NTP server

So Bonnie and Clyde travel in time to use NTP

DeLorean Time Machine by JMortonPhoto.com & OtoGodfrey.com (CC BY-SA 4.0)

NTP flow

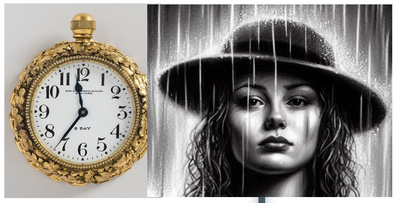


1900 desk clock (CC0)

1900 clock (CC0)

NTP flow

Clyde



Bonnie

Origin timestamp T_1

NTP req = T_1

T_2 Receive timestamp

T_3 Transmit timestamp

Destination timestamp T_4

NTP resp = T_1, T_2, T_3

Clyde has T_1, T_2, T_3, T_4

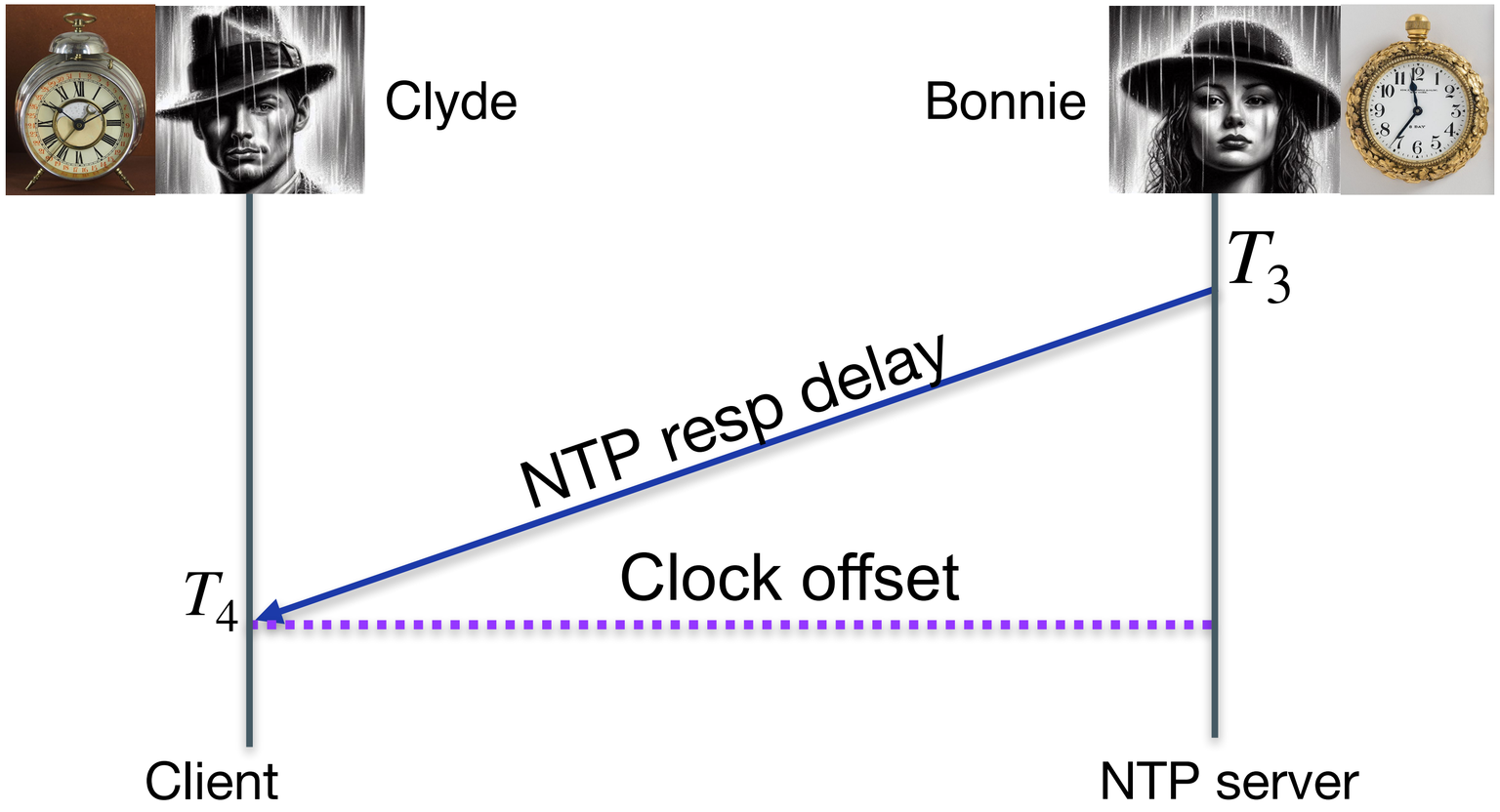
NTP client

NTP server

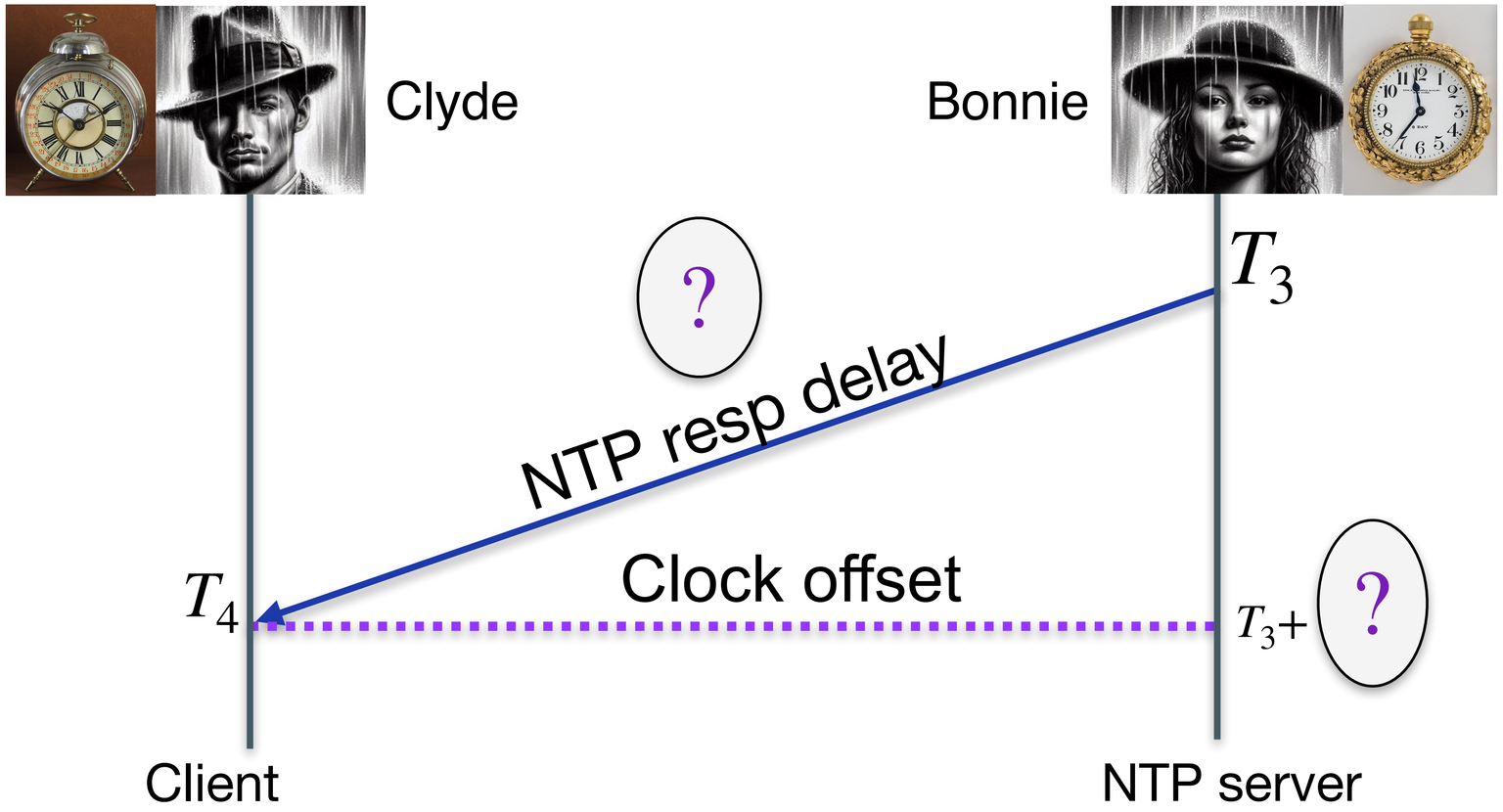
[1900 desk clock](#) ([CC0](#))

[1900 clock](#) ([CC0](#))

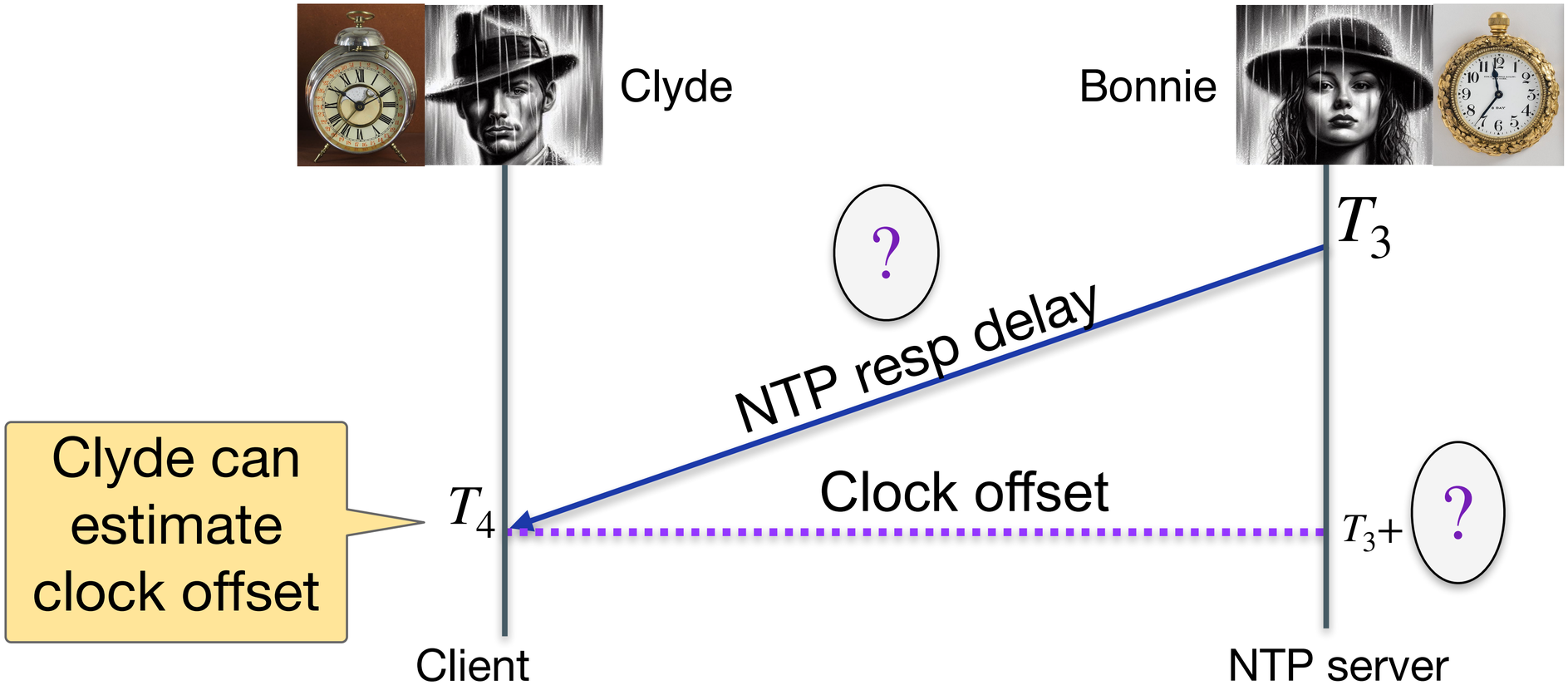
NTP: estimating clock offset



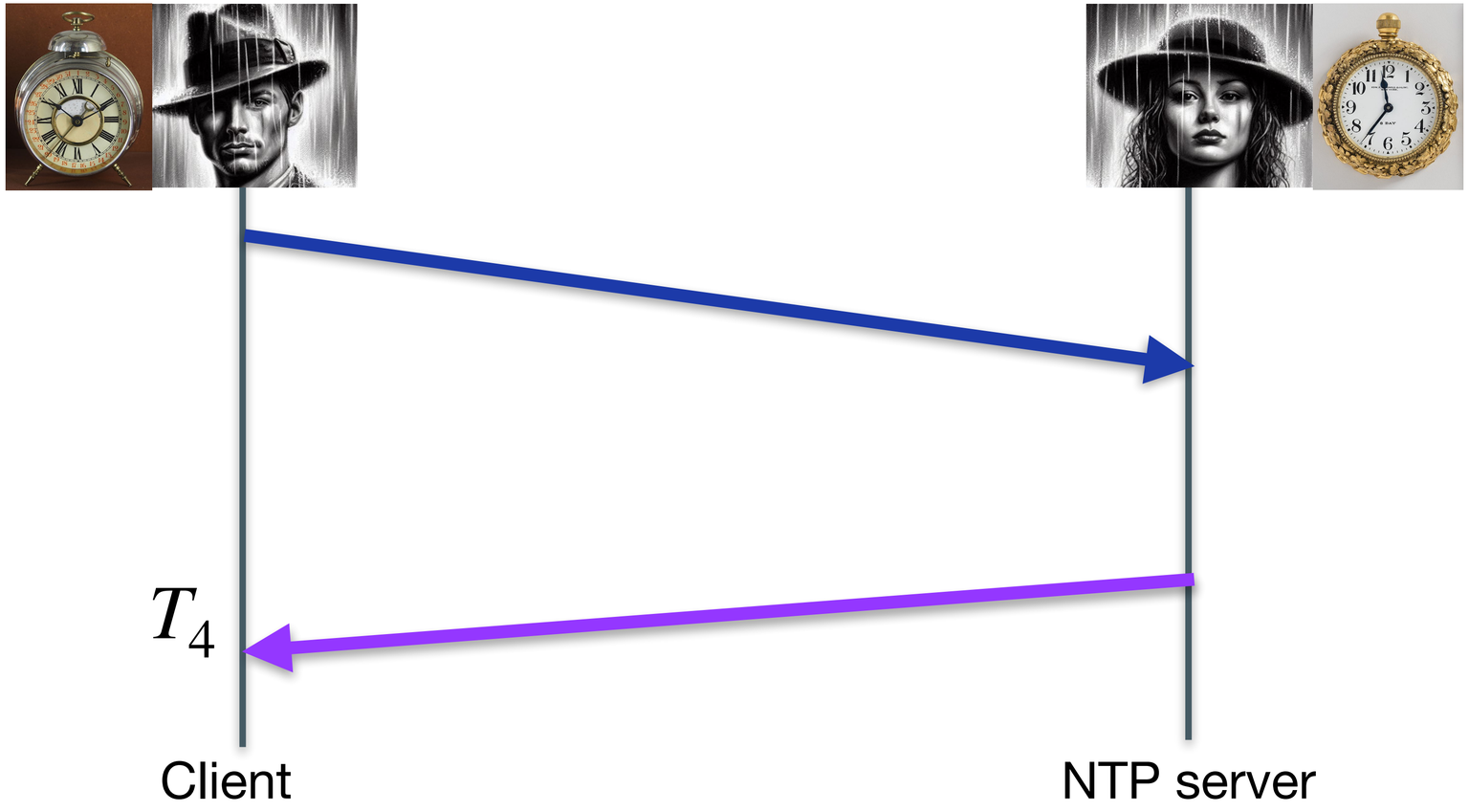
NTP: estimating clock offset



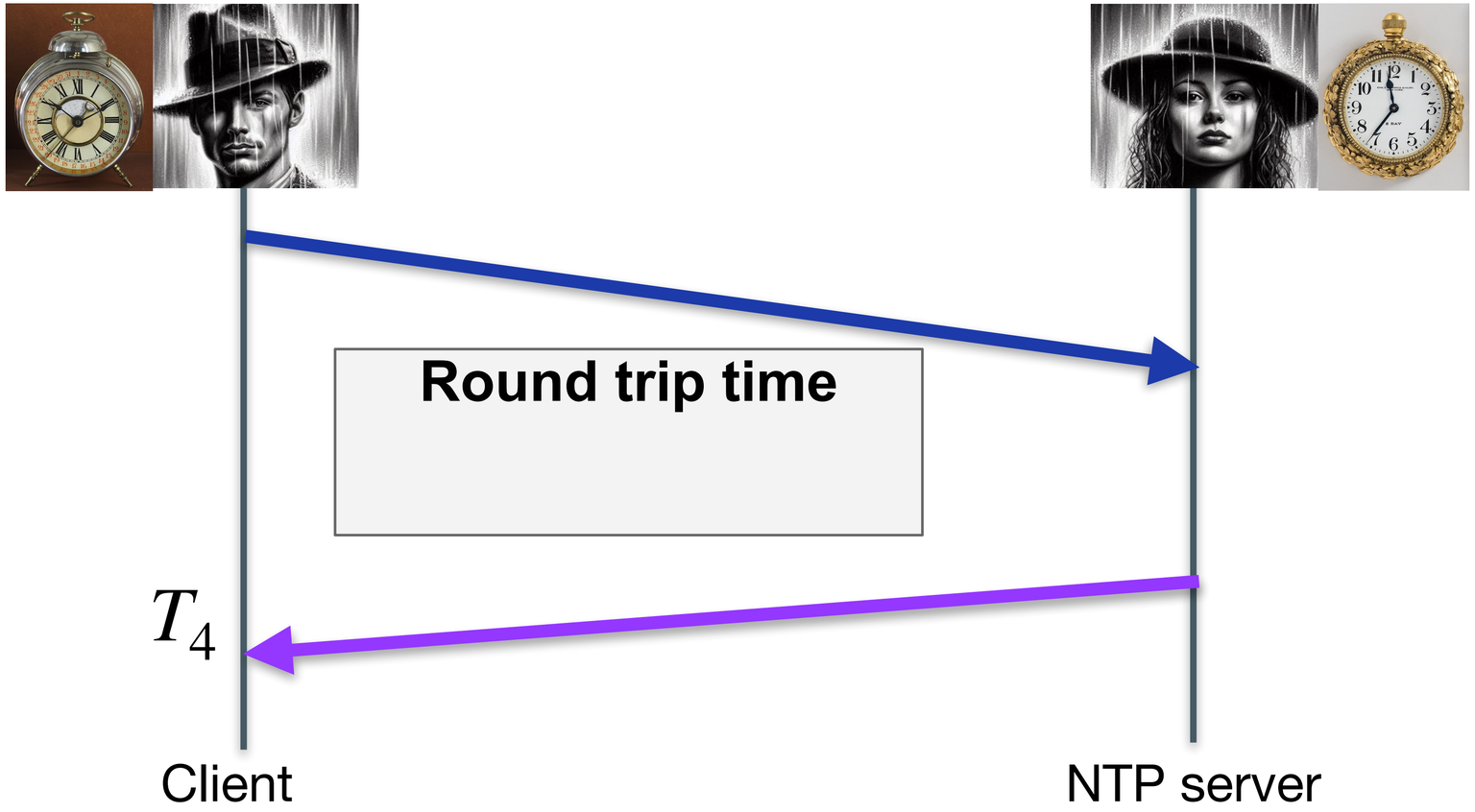
NTP: estimating clock offset



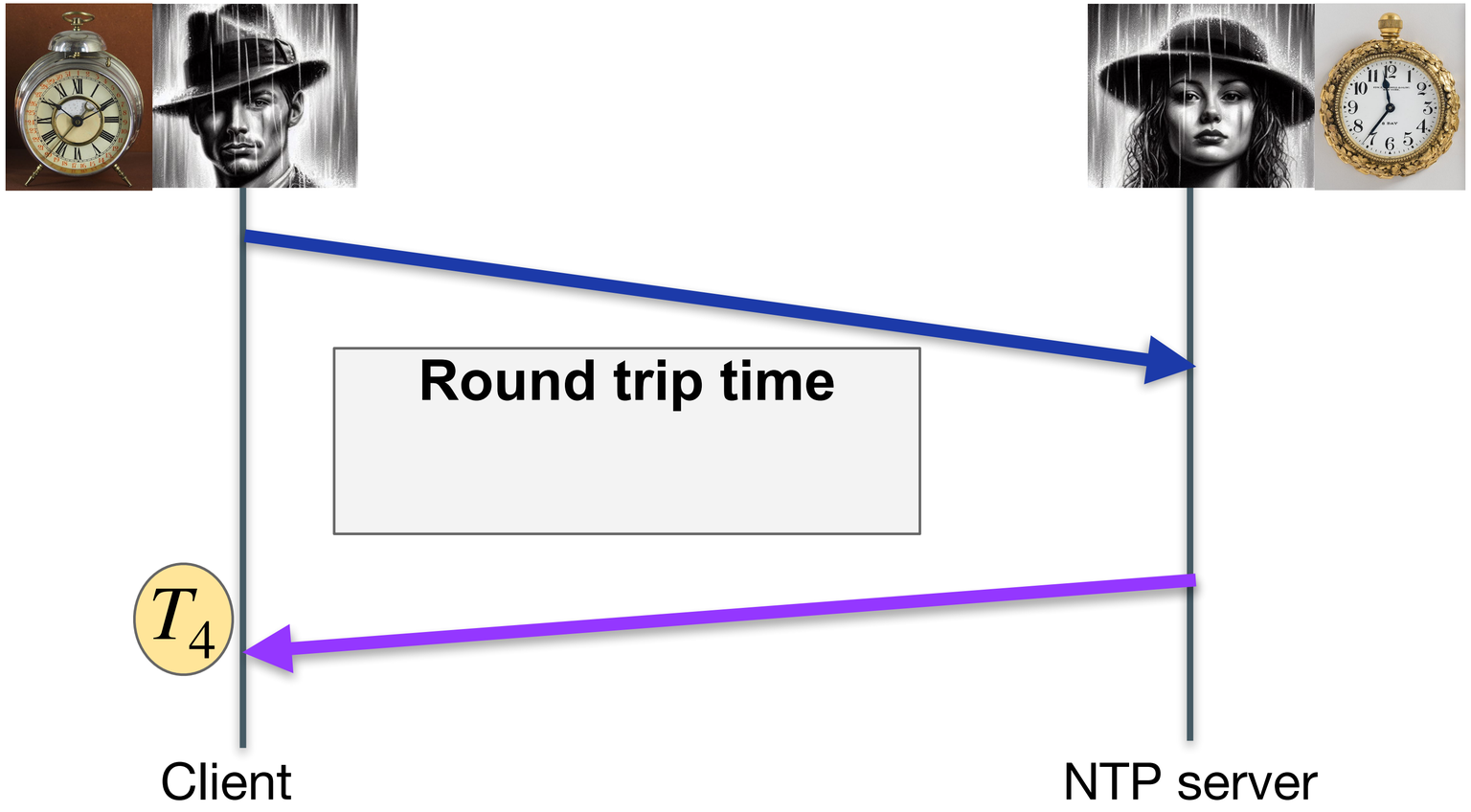
NTP: estimating clock offset



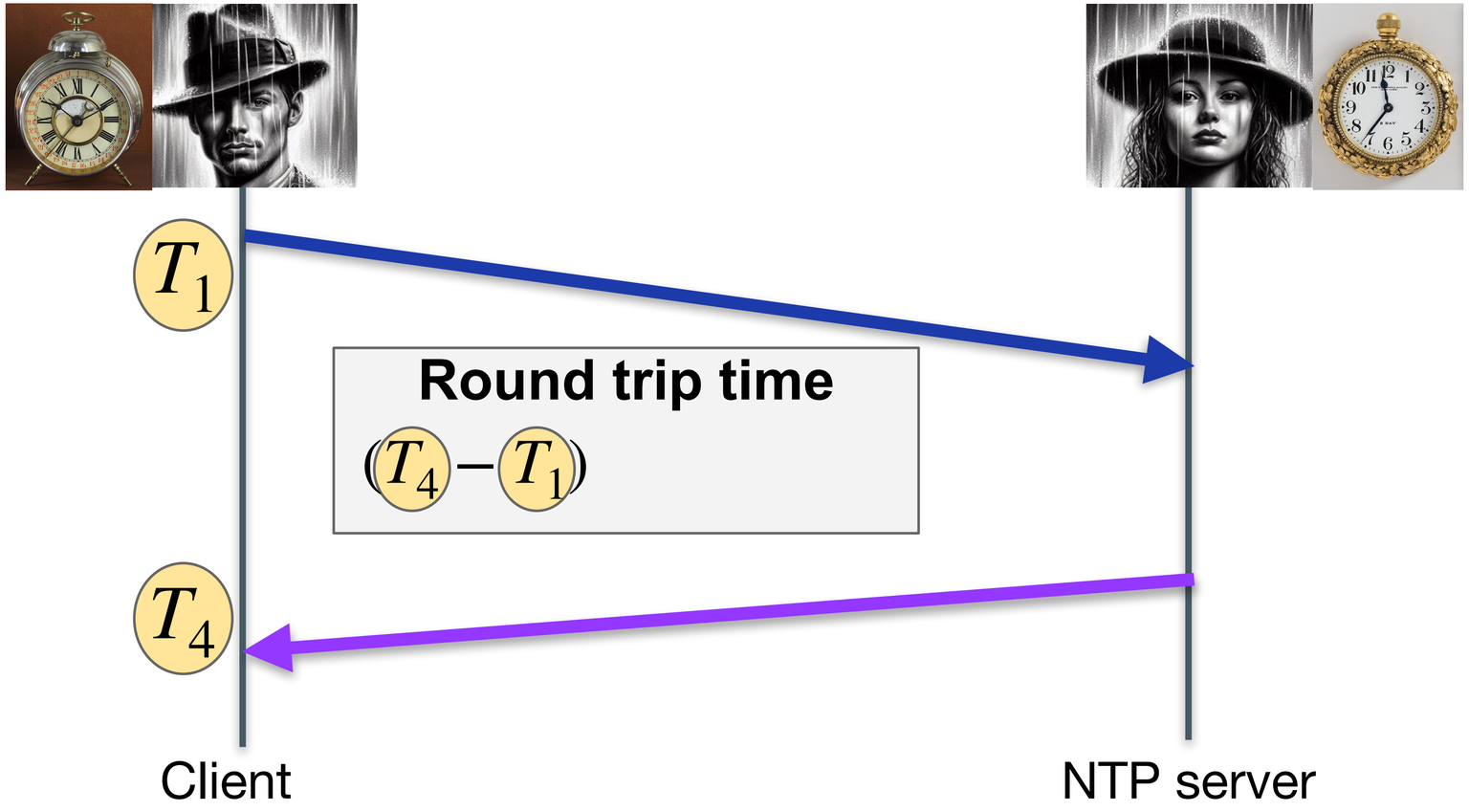
NTP: estimating clock offset



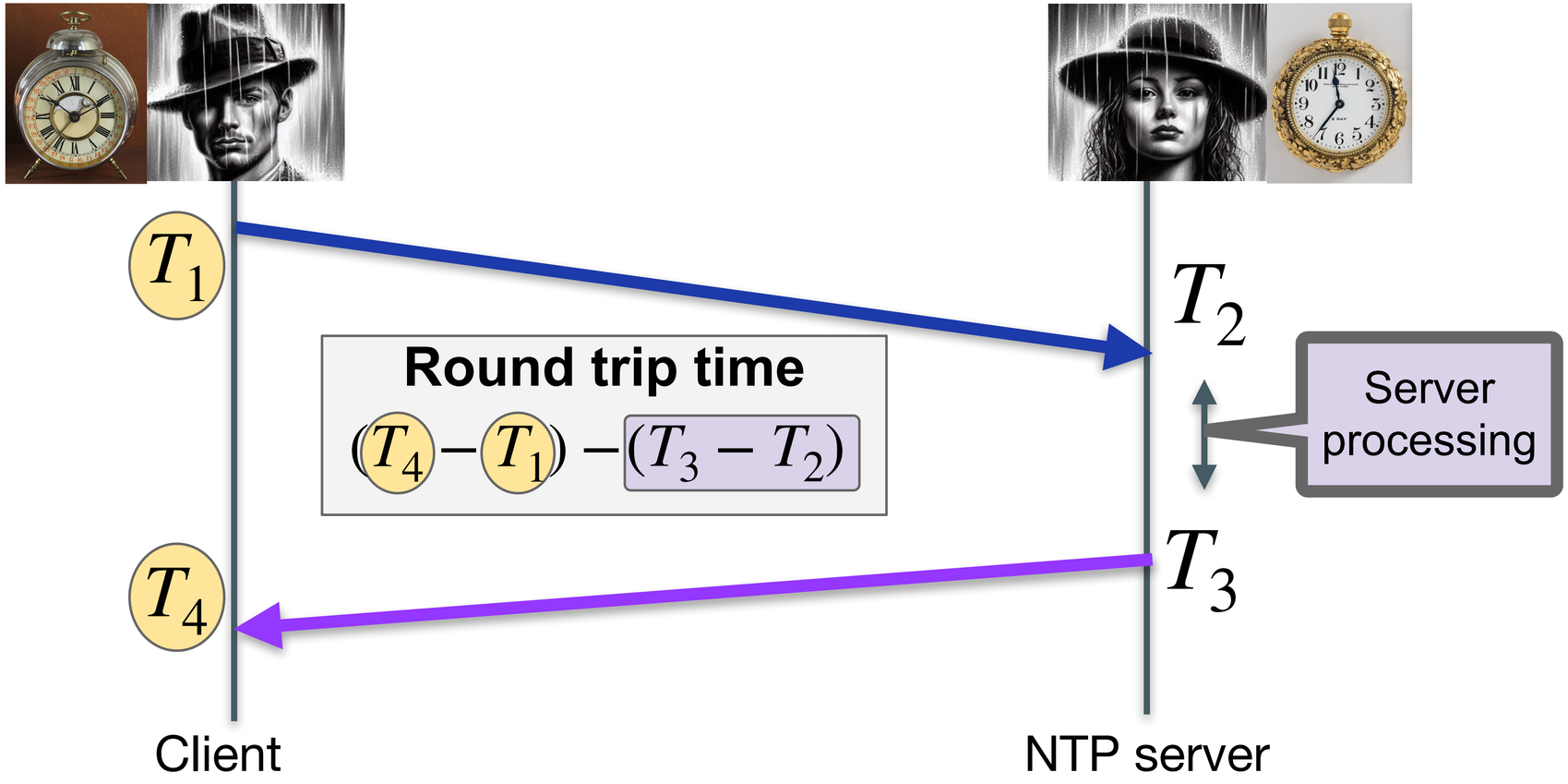
NTP: estimating clock offset



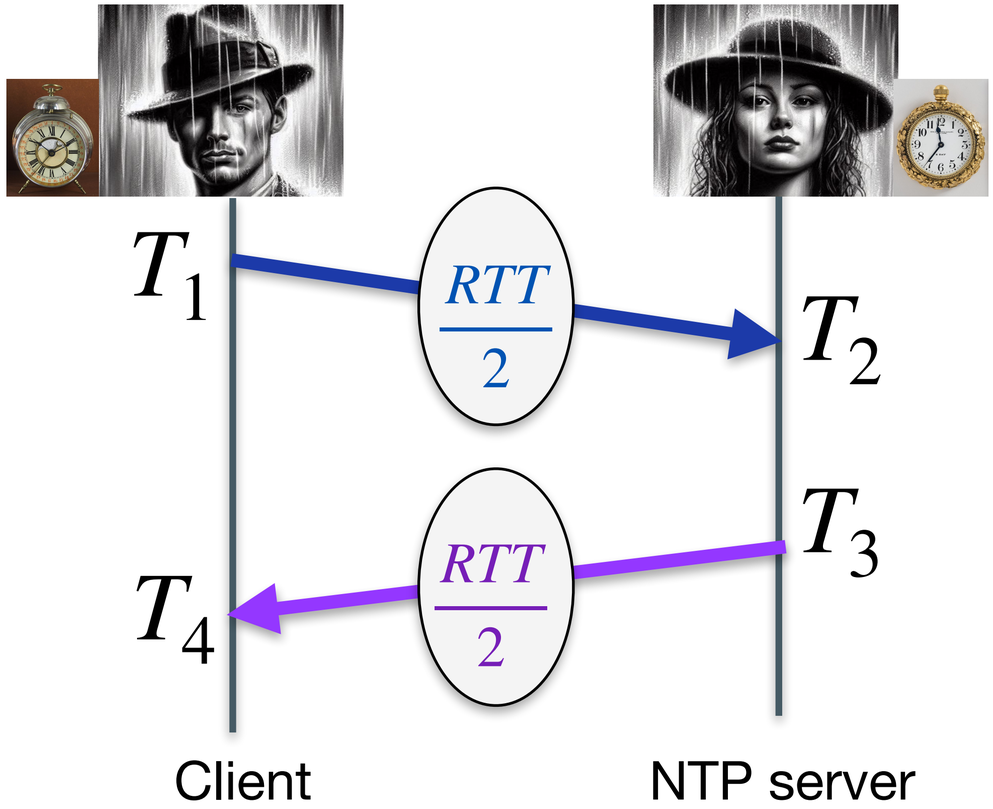
NTP: estimating clock offset



NTP: estimating clock offset



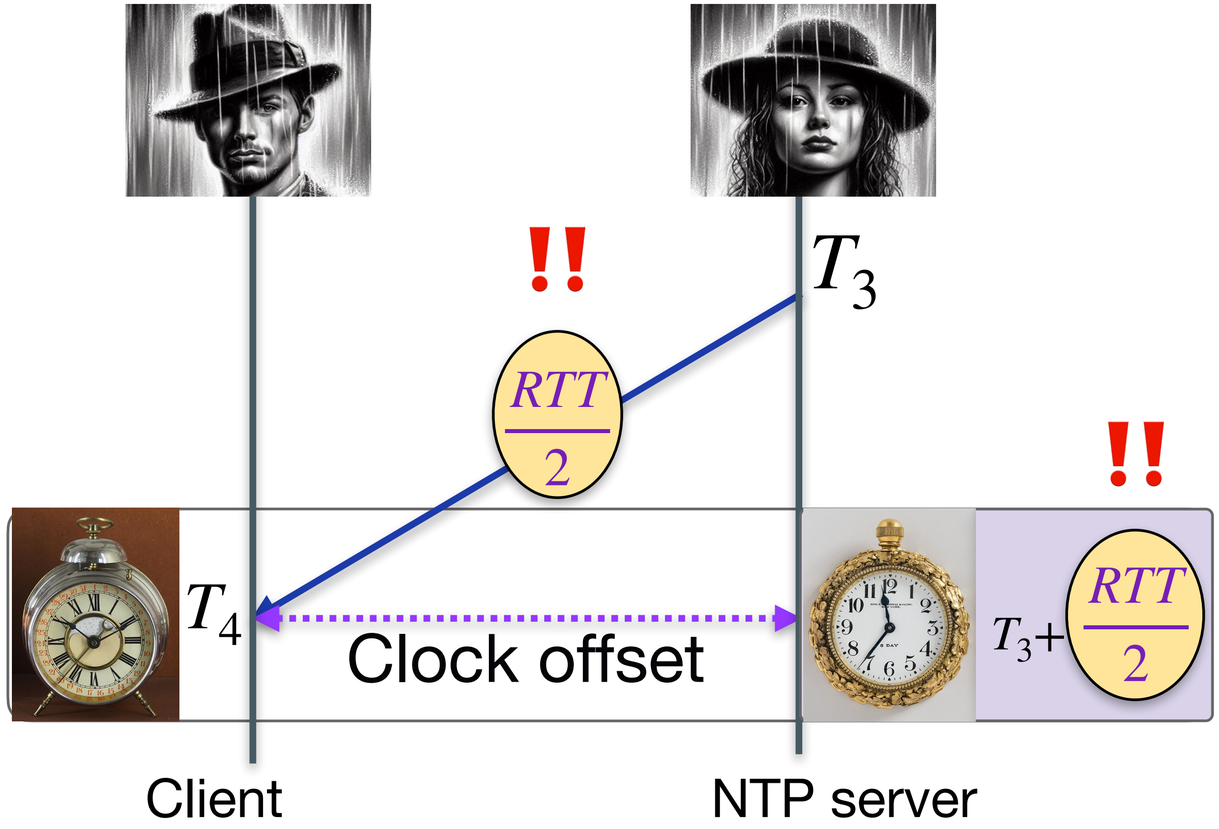
NTP: estimating clock offset



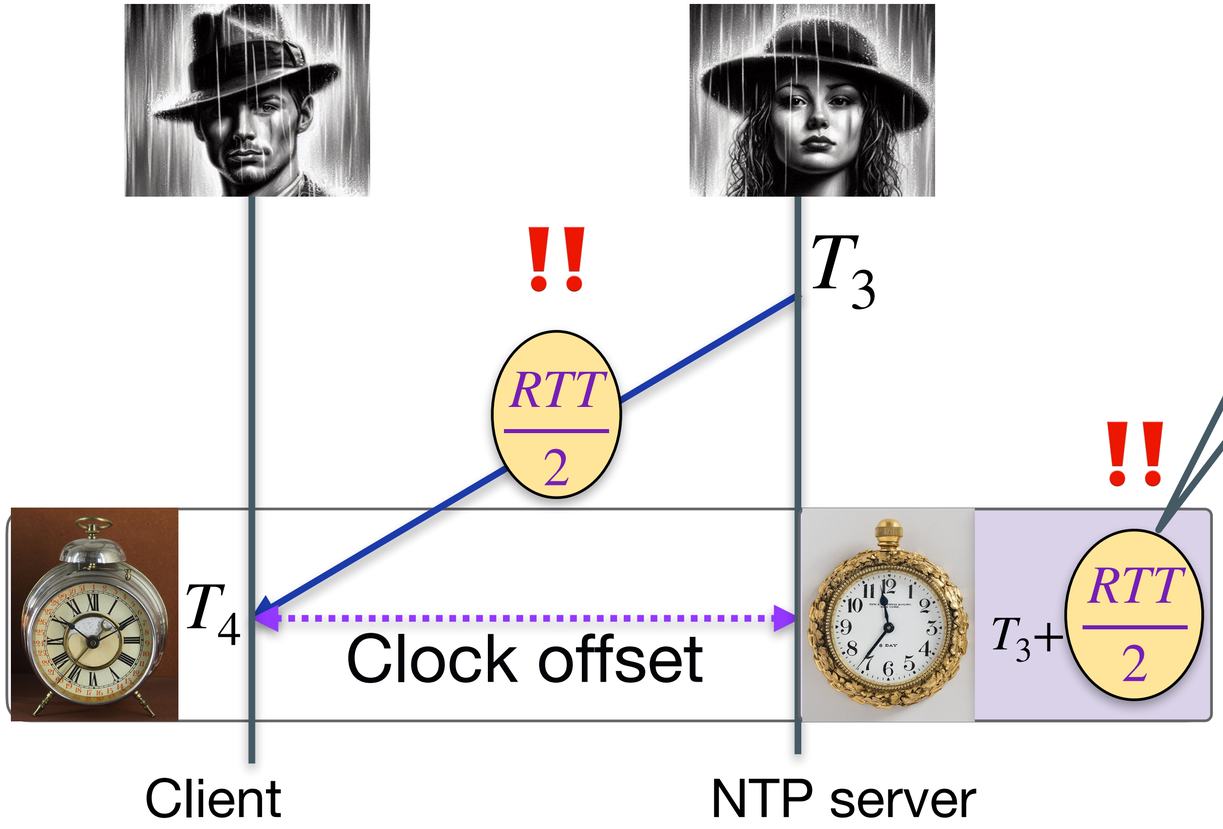
!!

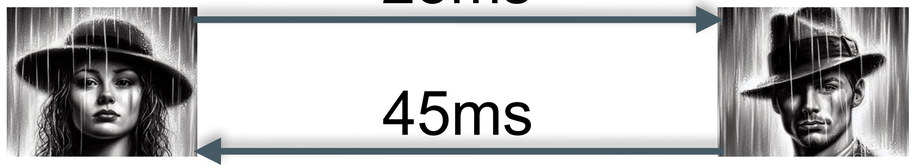
NTP assumes **request and response delays are equal**

NTP: estimating clock offset



NTP: estimating clock offset





Path **asymmetries** contribute to NTP clock sync **errors!**

NTP: Disciplining the clock to minimize clock offset θ

🐳 Best practice

Slew

🐎 or 🐢

$\theta < 125ms$

NTP: Disciplining the clock to minimize clock offset θ

🐋 Best practice

Slew

🐎 or 🐢

$\theta < 125ms$

Step

⏮ or ⏭

$125ms < \theta < 1000s$

NTP: Disciplining the clock to minimize clock offset θ

🐋 Best practice

Slew

🐎 or 🐢

$\theta < 125ms$

Step

⏮ or ⏭

$125ms < \theta < 1000s$

Alert

$\theta >= 1000s$

Times are hard during the Great Depression

The Shamrock Shakedown

August 29, 2024

Outlaws Turn to Algorithms!

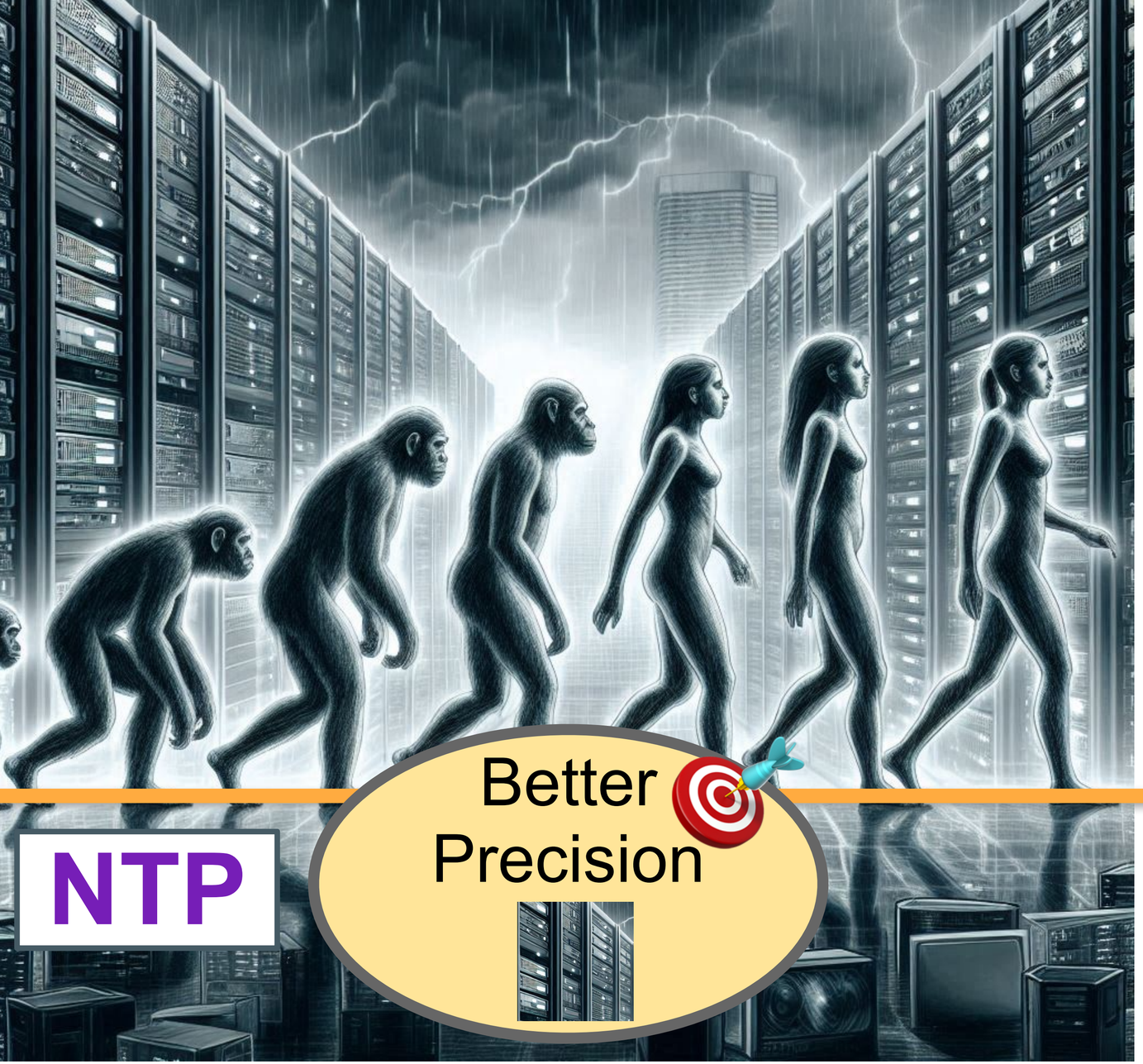
Times are hard during the Great Depression

The Shamrock Shakedown

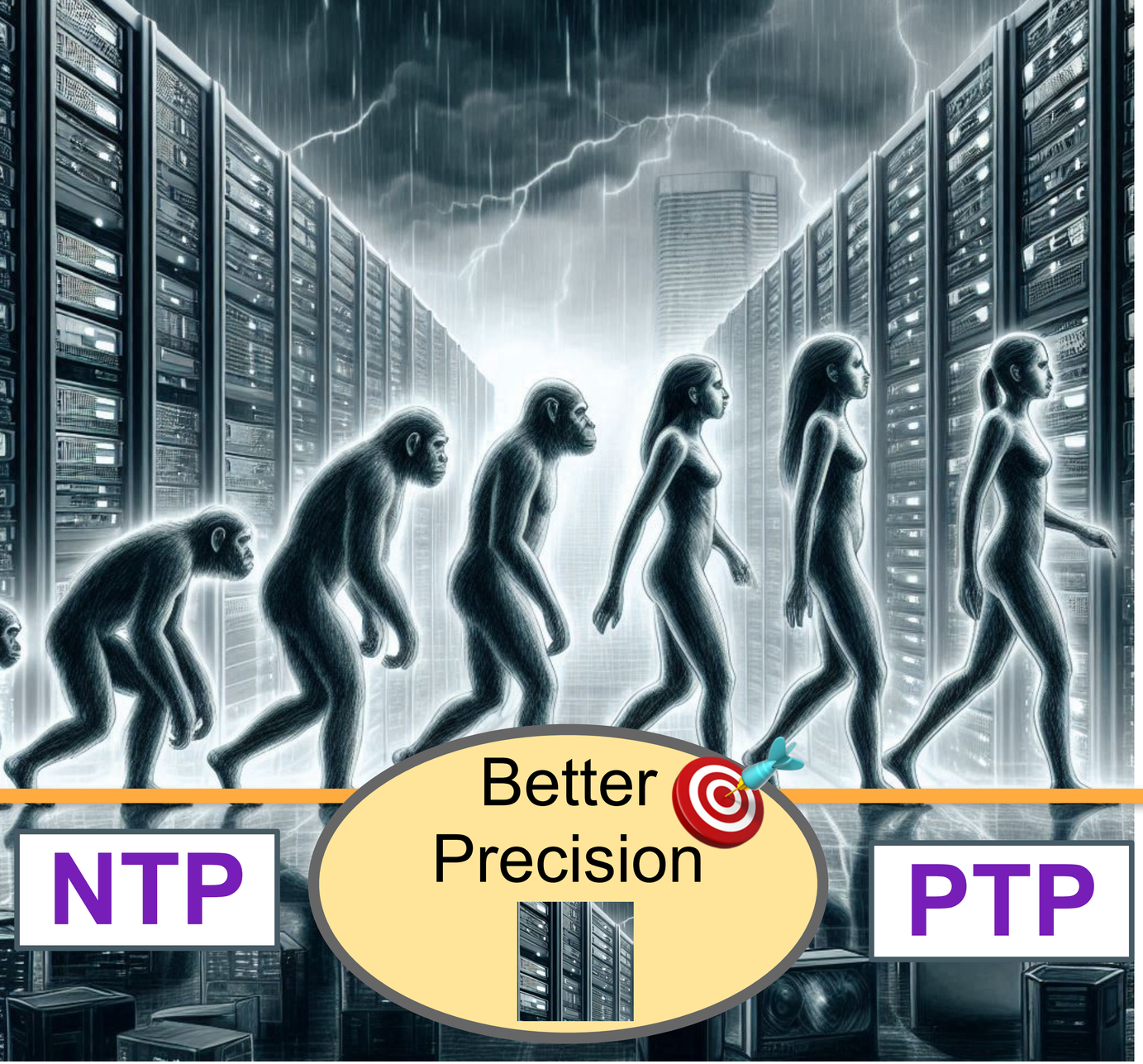
August 29, 2024

Outlaws Turn to Algorithms!

Bonnie and Clyde Launch High-Stakes Hedge Fund



PTP: IEEE 1588 ⚡



PTP: IEEE 1588 ⚡

Relies on electrical engineering concepts 👷🏾‍♀️

PTP ⚡

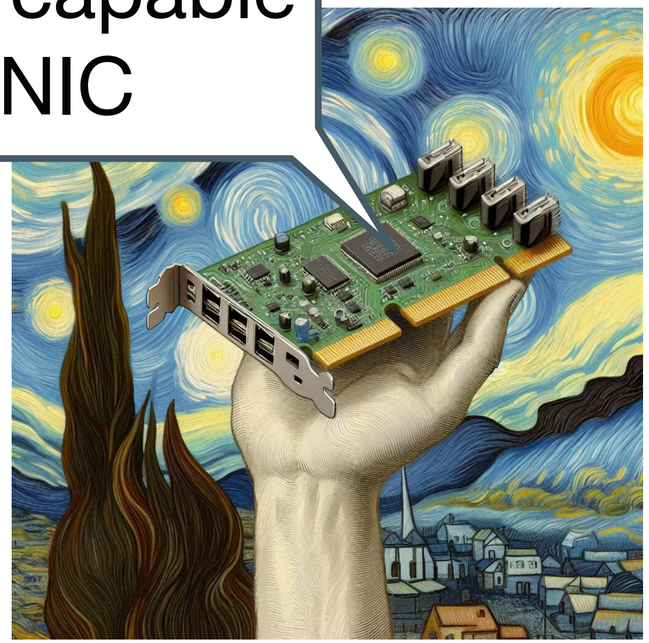
PTP ⚡

For fine grained accuracy 

PTP ⚡

For fine grained accuracy 🕸️💸

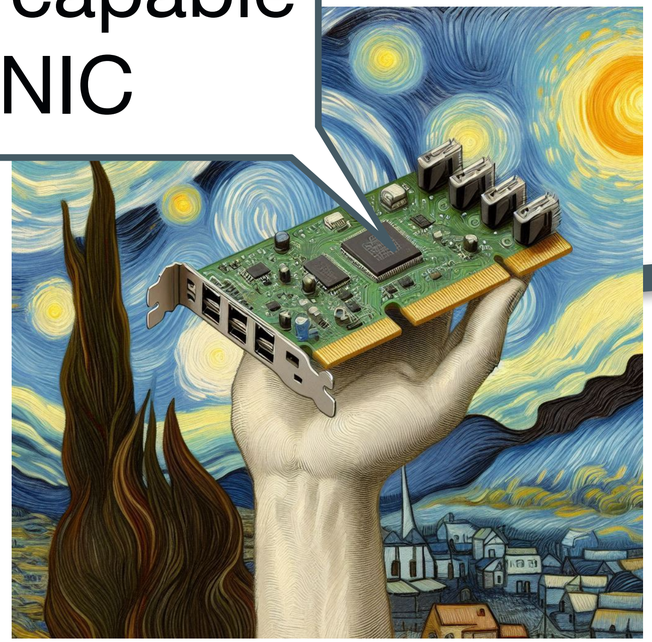
PTP capable NIC

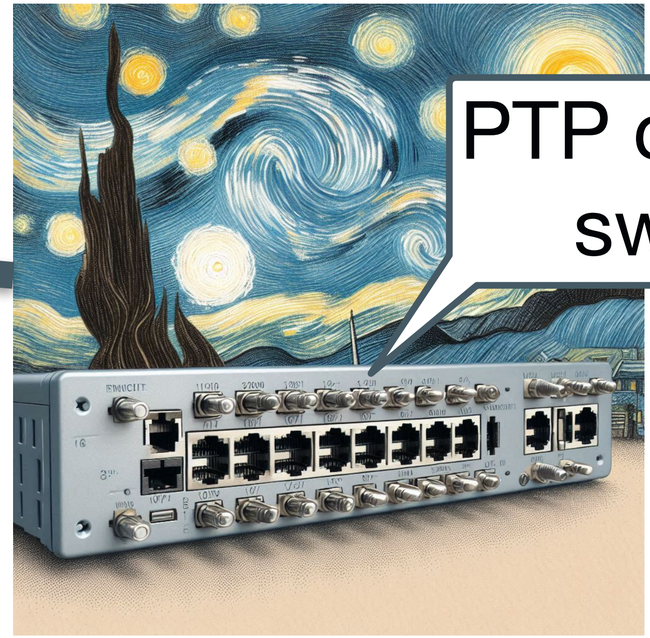


PTP ⚡

For fine grained accuracy 🕸️💸

PTP capable NIC





PTP capable switch

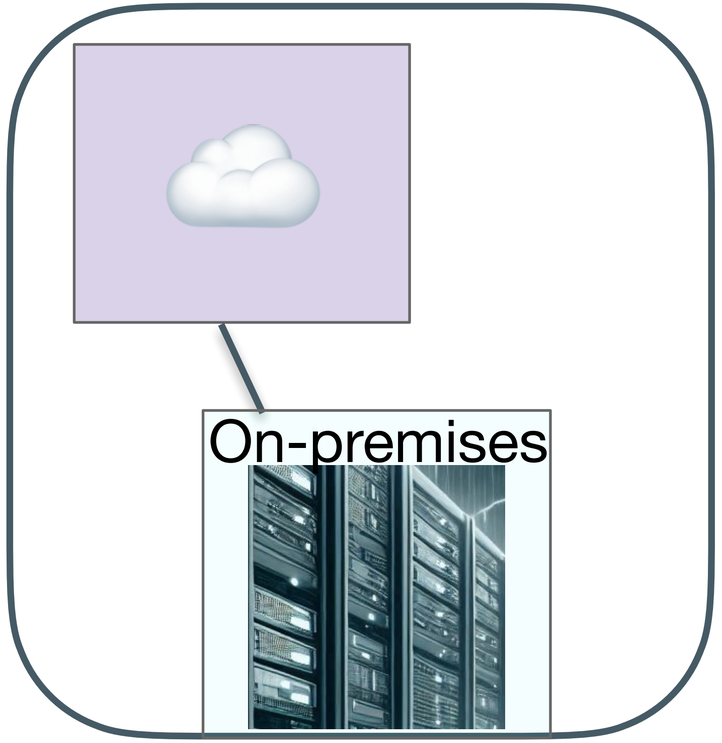
October 28, 2024

The Shamrock Shakedown

October 28, 2024

B&C Ventures' NextGen Heist expands to the cloud!

Huygens



Better precision

Huygens clock synchronization

- Software based
- High precision 🎯

https://www.usenix.org/conference/nsdi18/presentation/geng

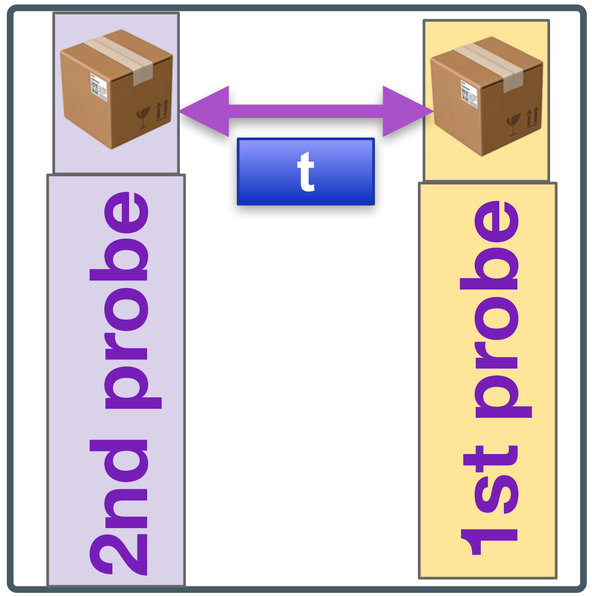
Huygens clock synchronization

- Software based
- High precision 🎯

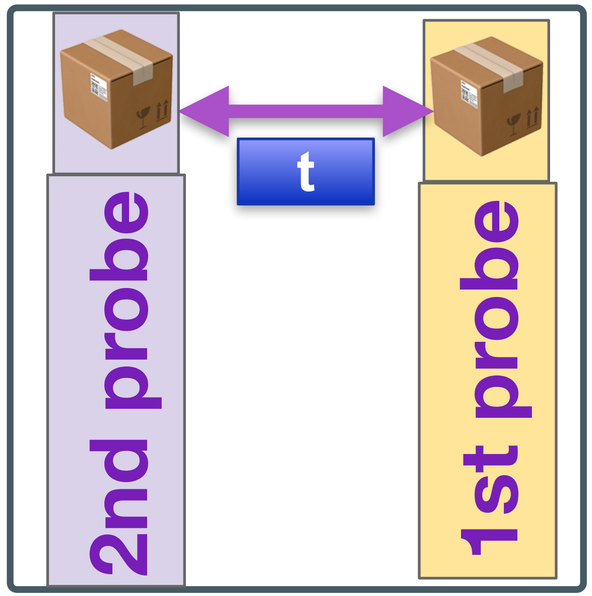
Measures
one way delays

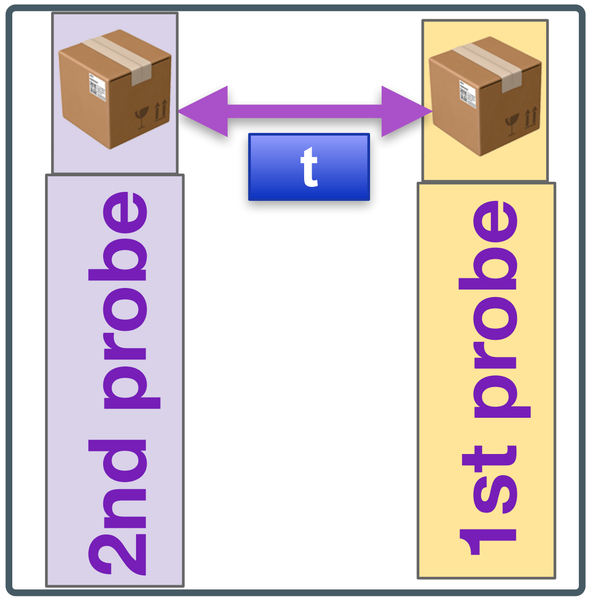
https://www.usenix.org/conference/nsdi18/presentation/geng

Huygens: Coded probes

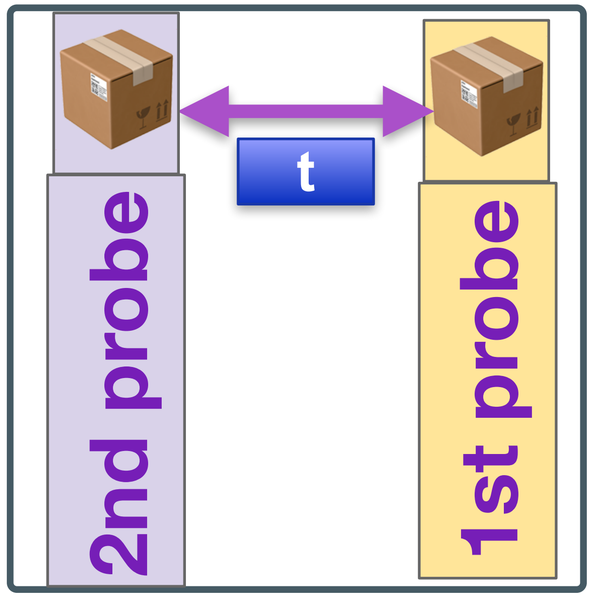


Huygens: Coded probes



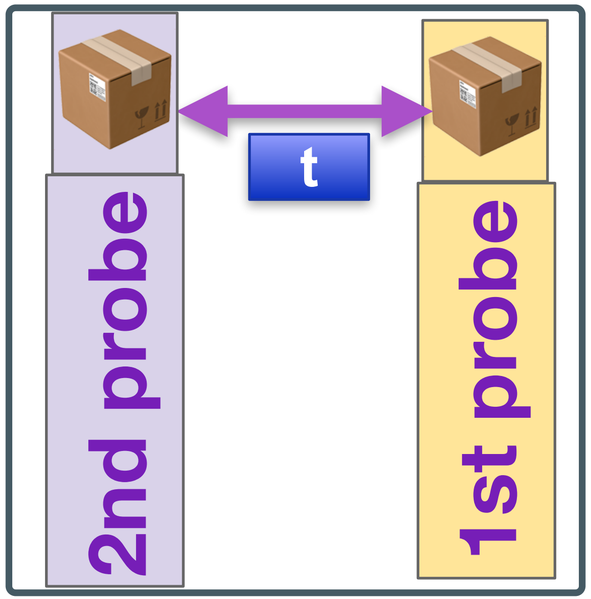


Huygens: Coded probes

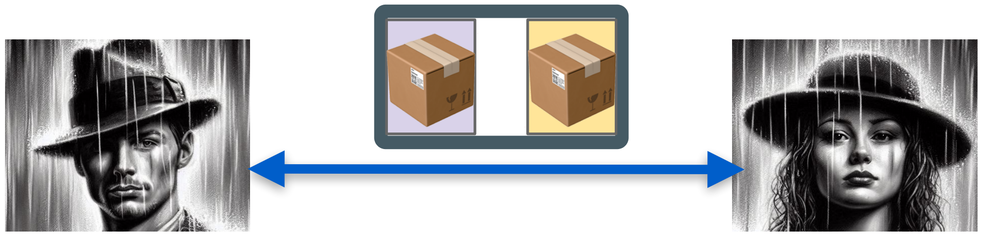


Pure coded probes:

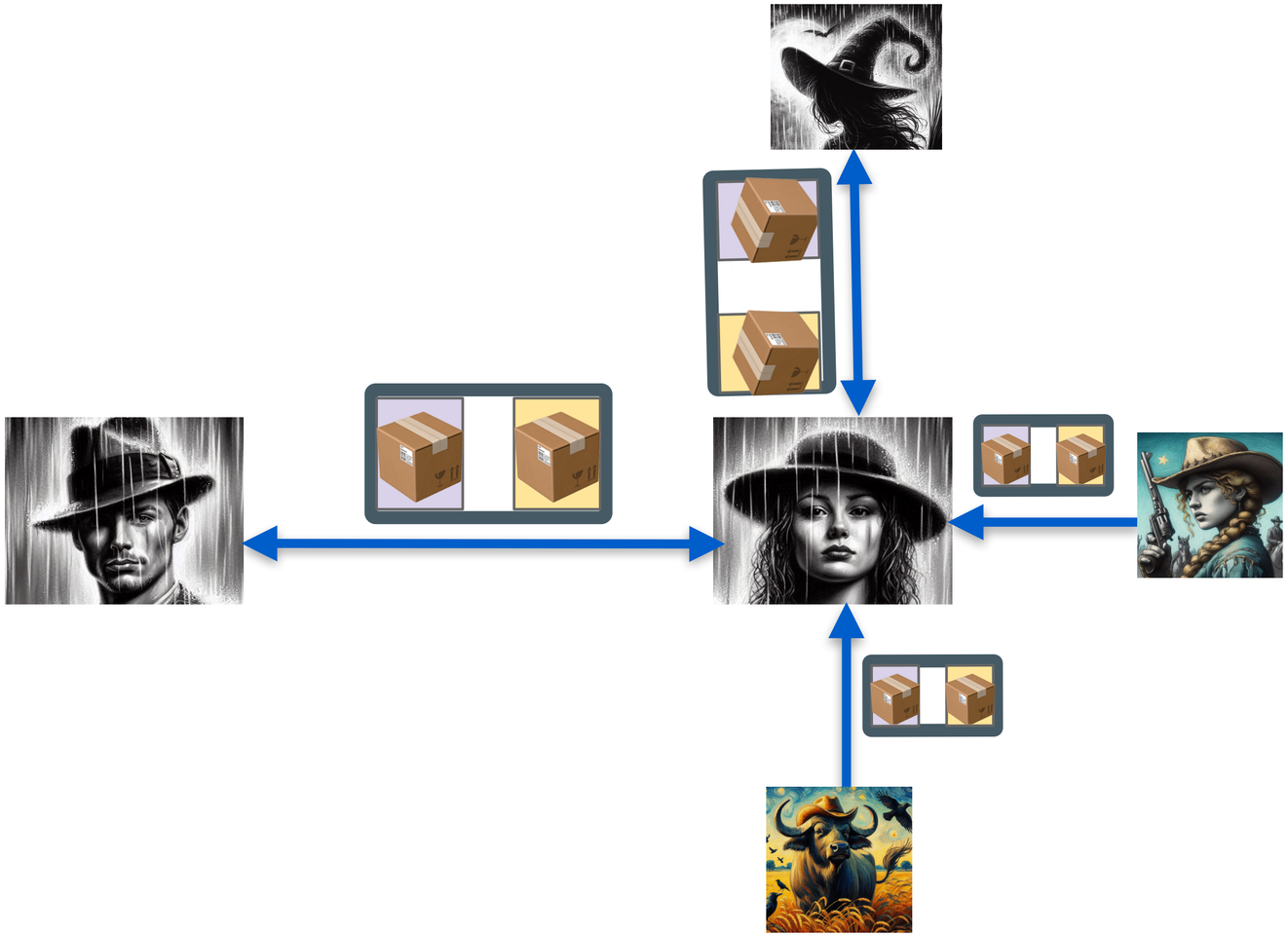
If **t** remains the same then the probe pair didn't suffer from delays



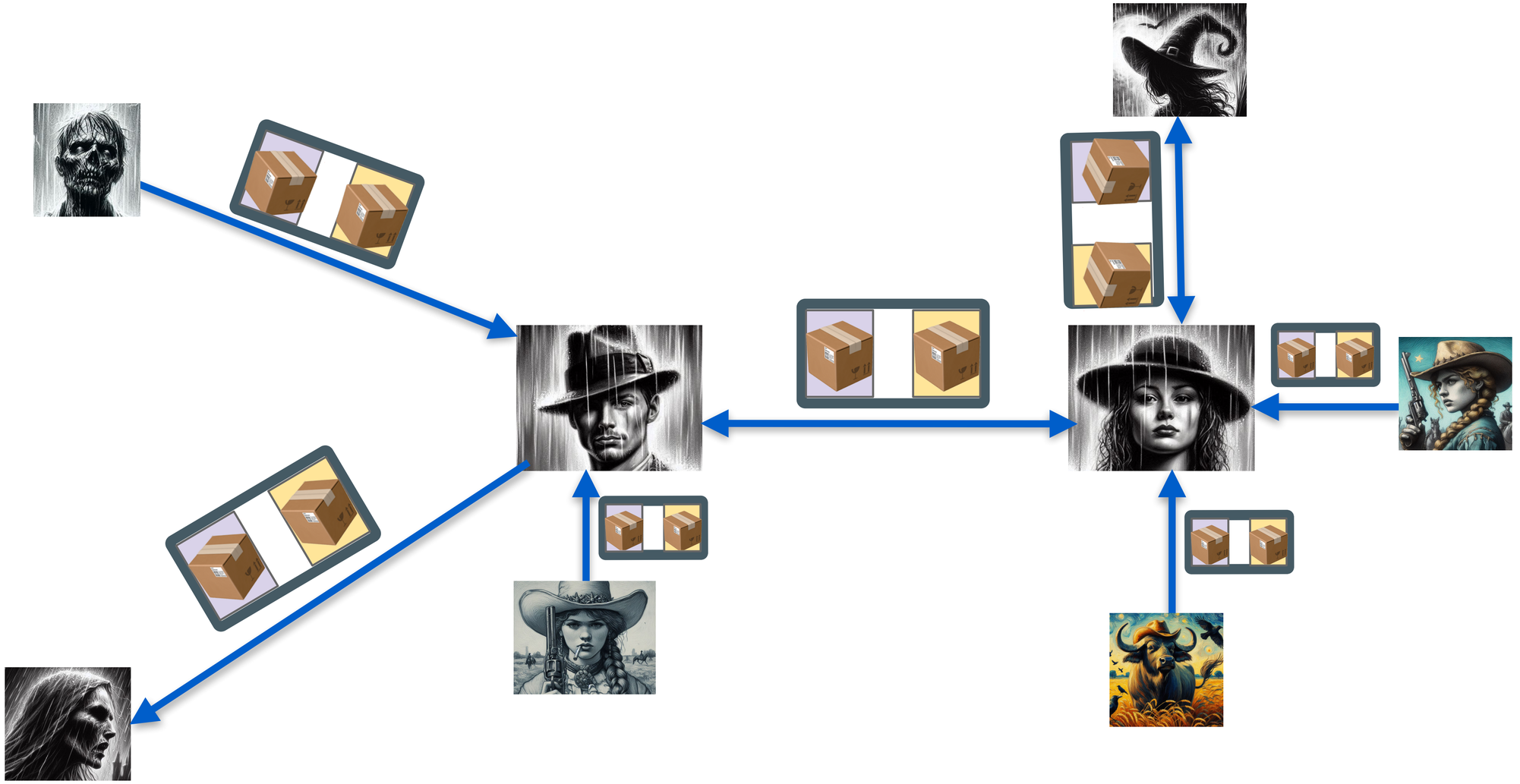
Huygens: probe mesh



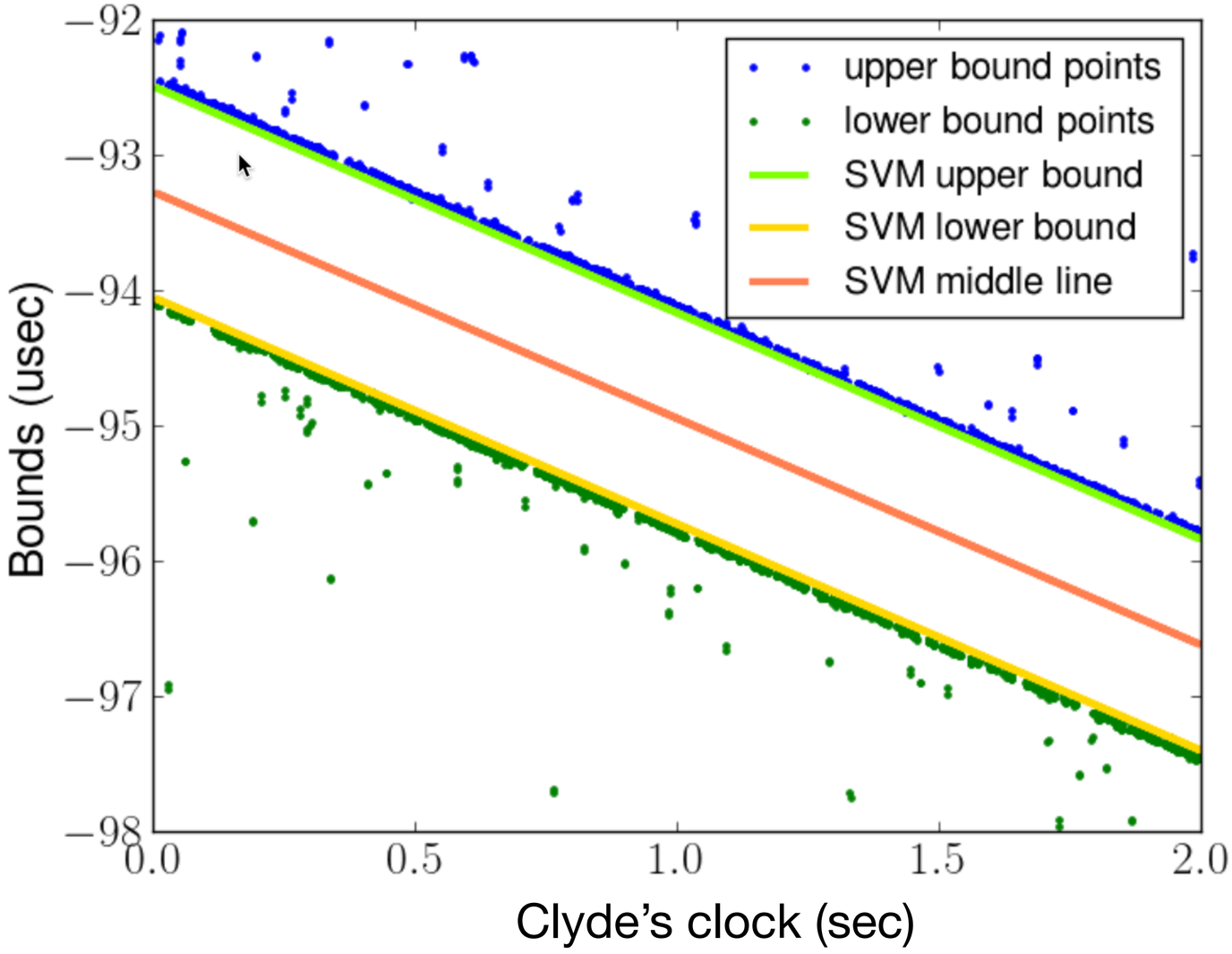
Huygens: probe mesh



Huygens: probe mesh



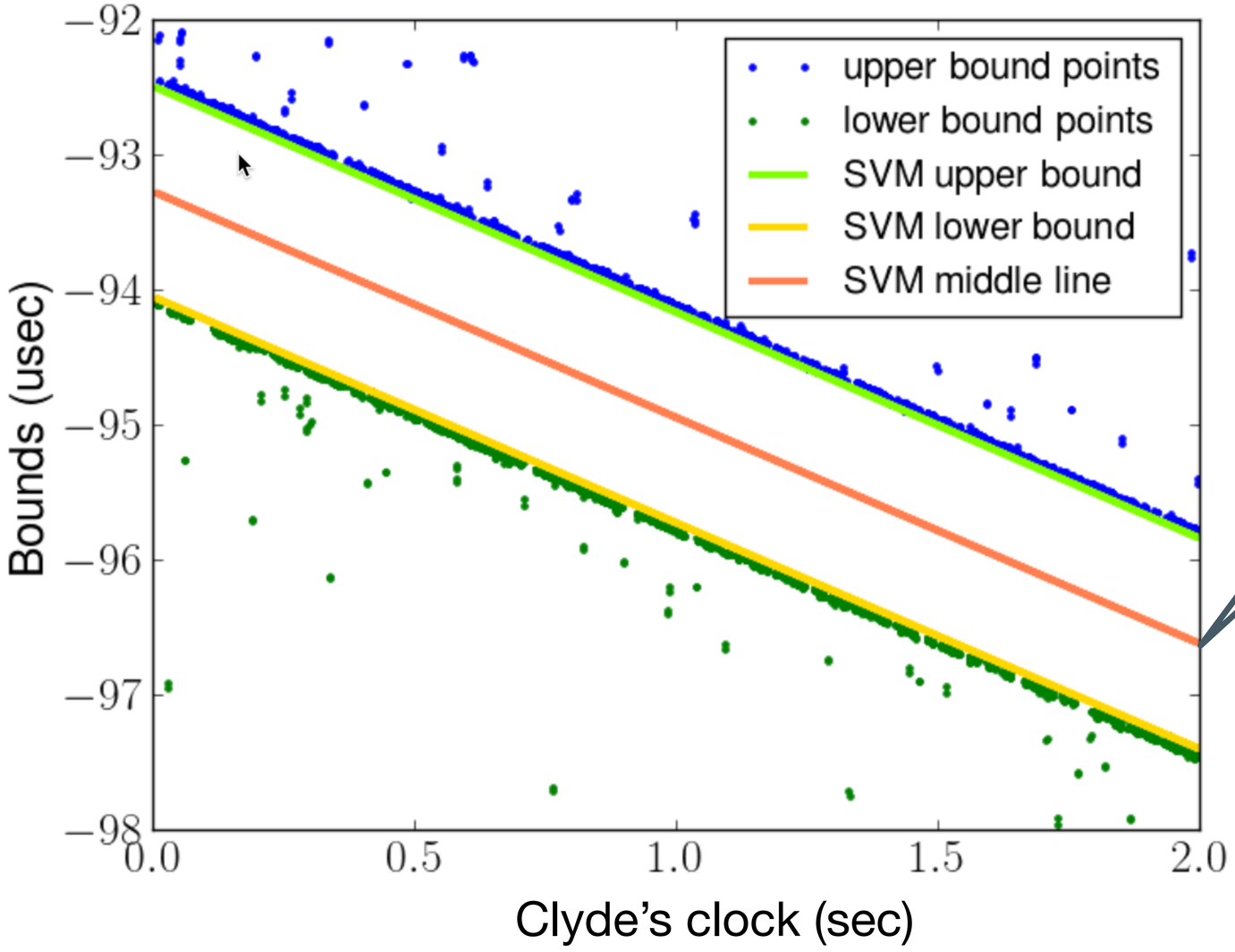
Huygens: estimating offset and drift (step 1)



Over 2 second intervals
as temperature 🌡️ is constant

https://www.usenix.org/conference/nsdi18/presentation/geng

Huygens: estimating offset and drift (step 1)

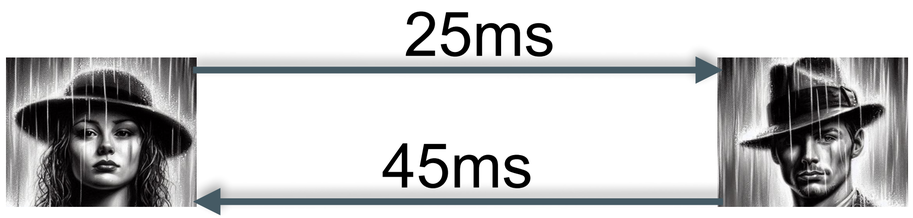


Apply ML classifier
for high precision estimation
of offset and drift

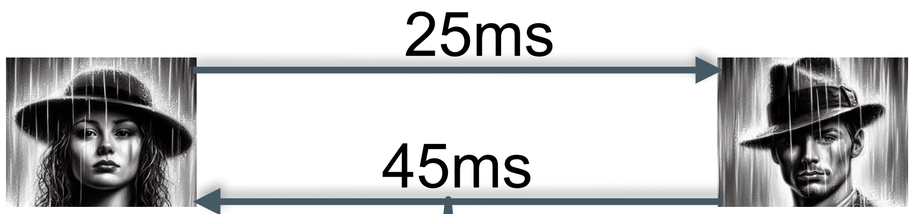
Over 2 second intervals
as temperature 🌡️ is constant

https://www.usenix.org/conference/nsdi18/presentation/geng

NTP: pairwise synchronization



NTP: pairwise synchronization

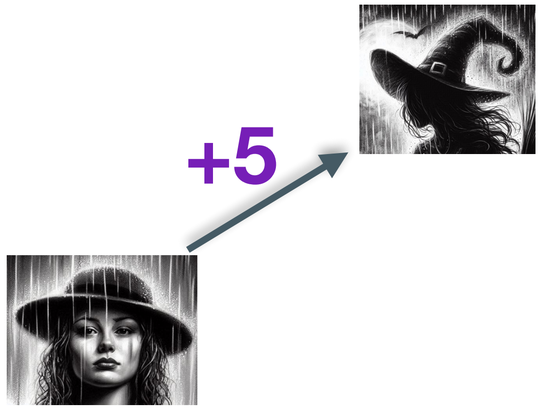


Prone to errors
due to
path asymmetry

Huygens

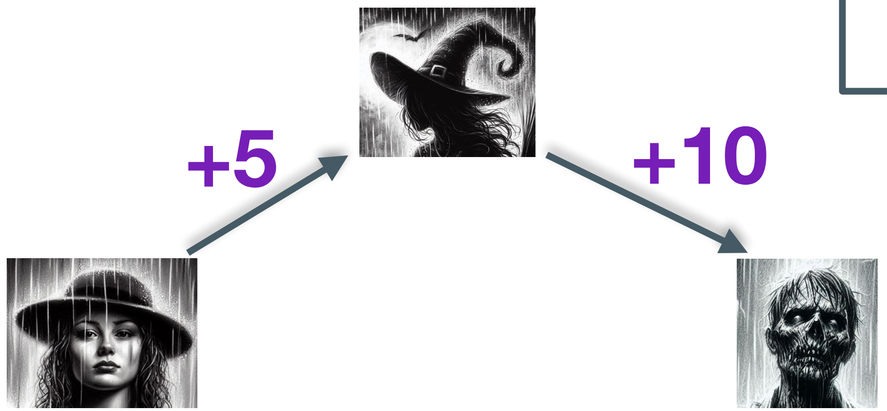
Clock sync is transitive!

Huygens



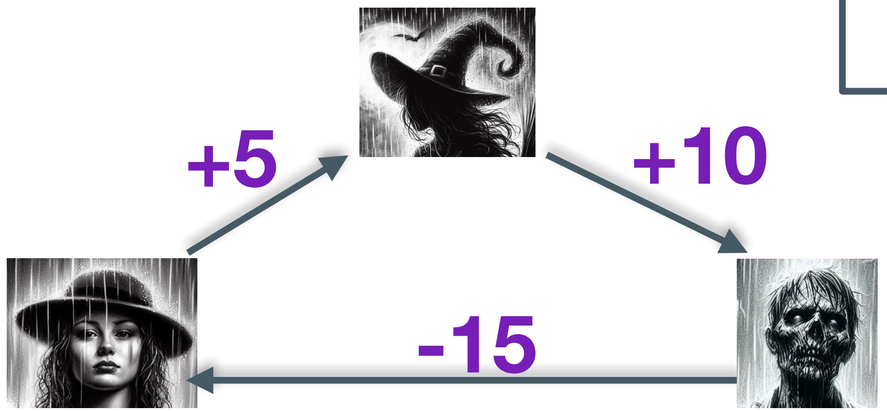
Clock sync is transitive!

Huygens



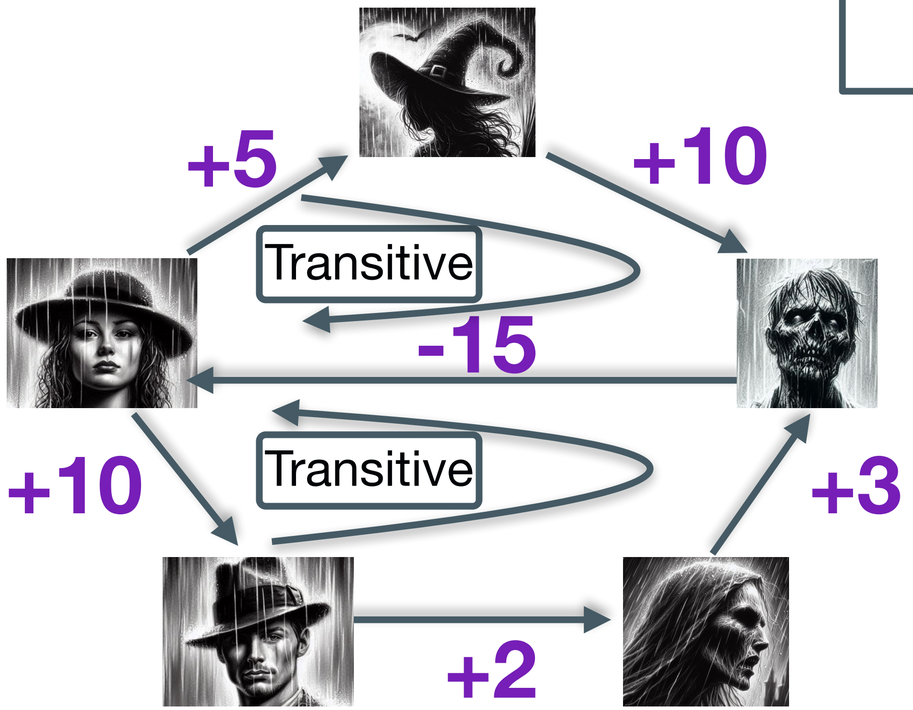
Clock sync is transitive!

Huygens



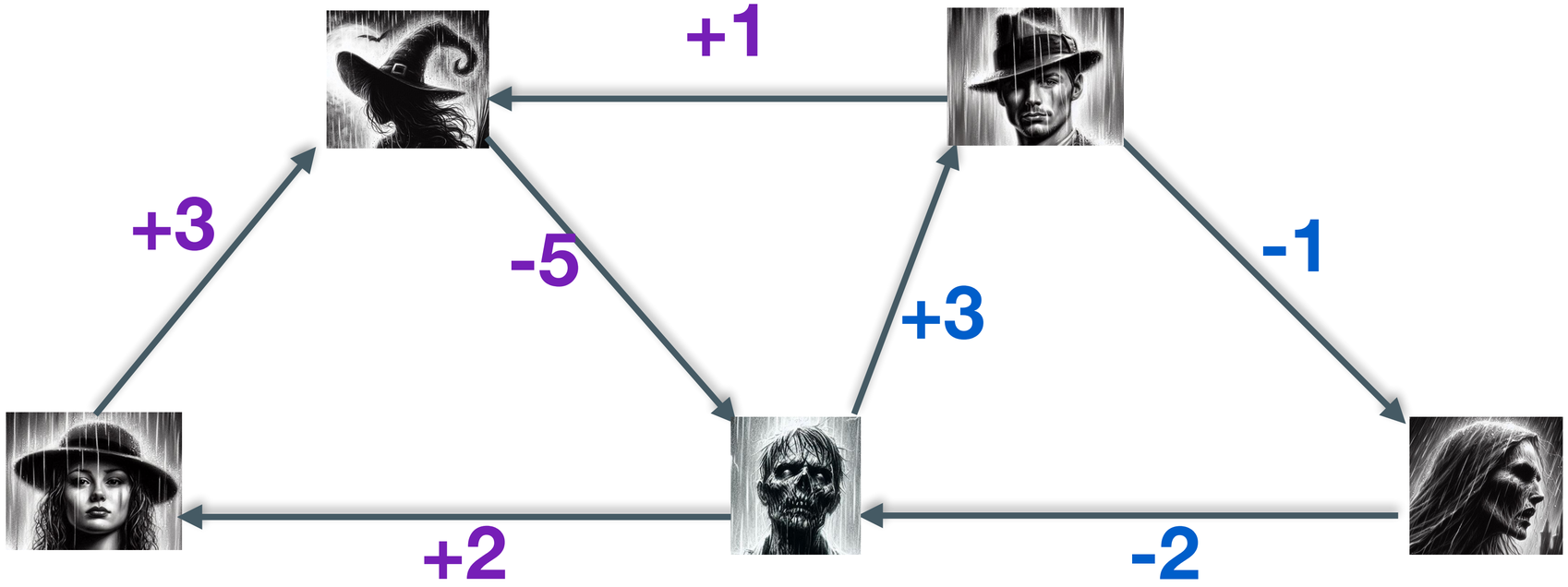
Clock sync is transitive!

Huygens

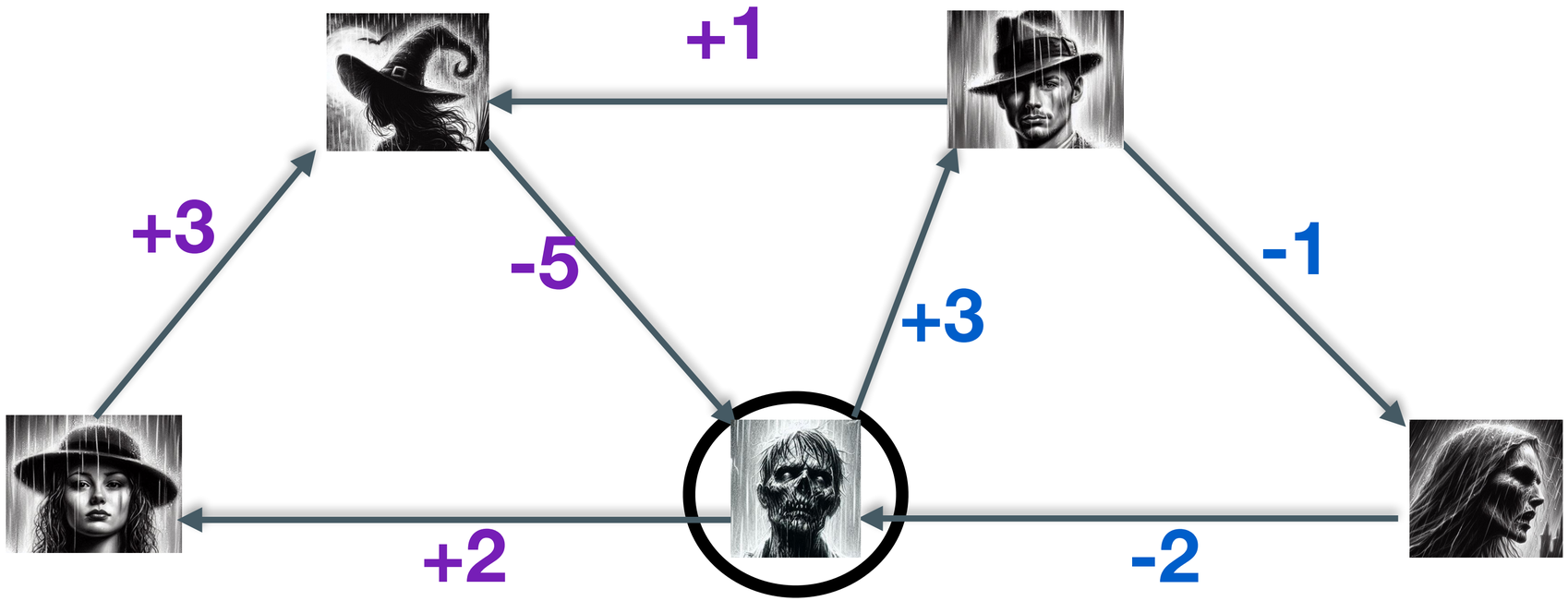


Clock sync is transitive!

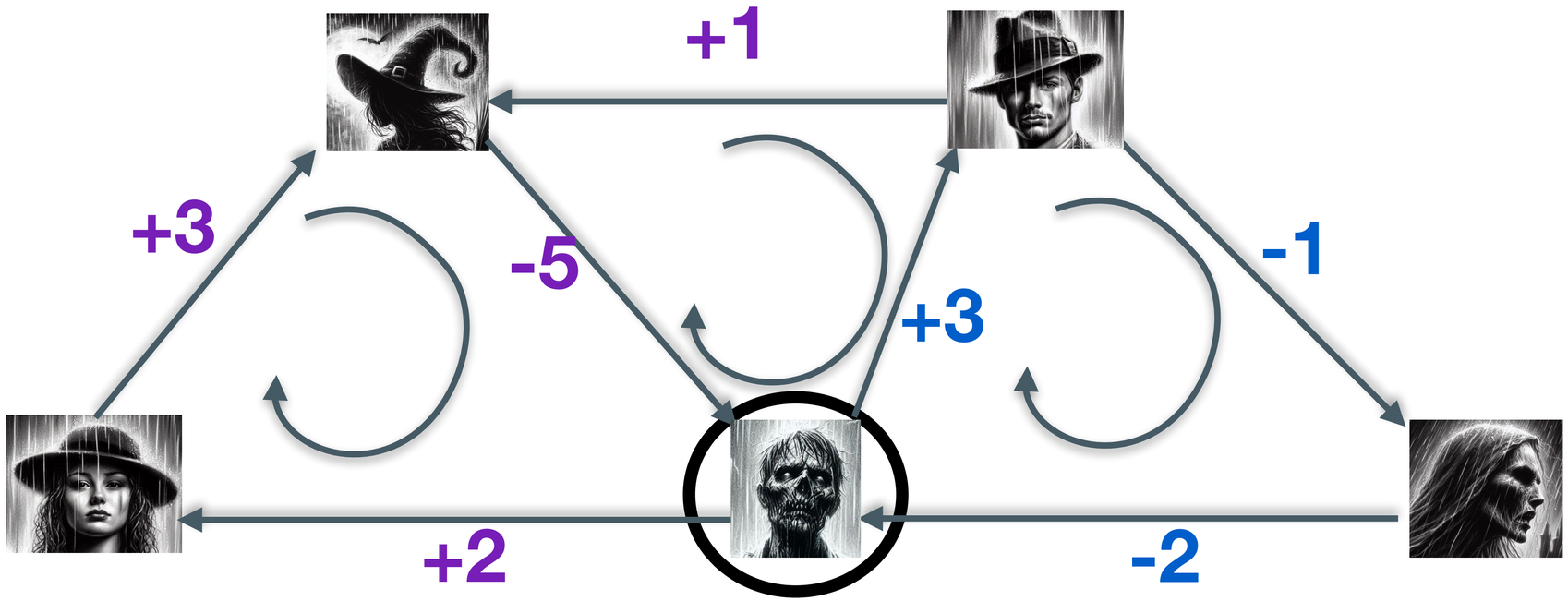
Huygens: "*triangulating*" to minimize clock sync errors



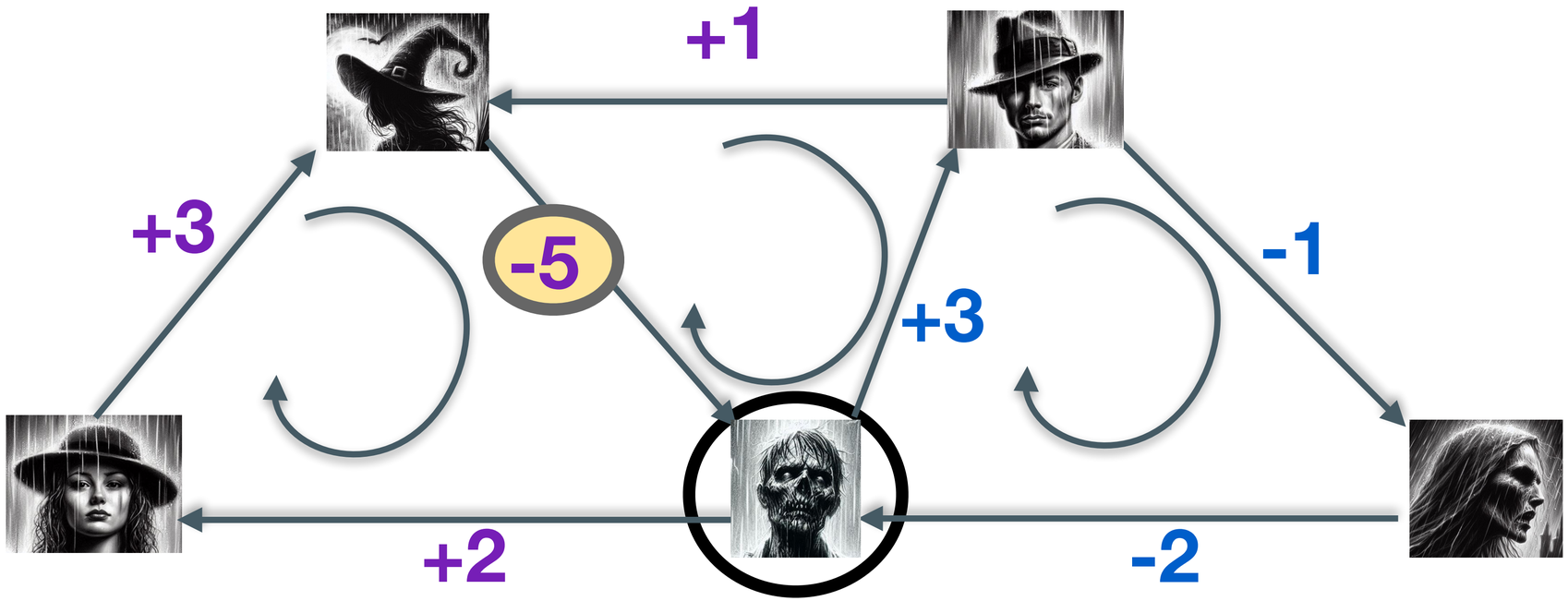
Huygens: “*triangulating*” to minimize clock sync errors



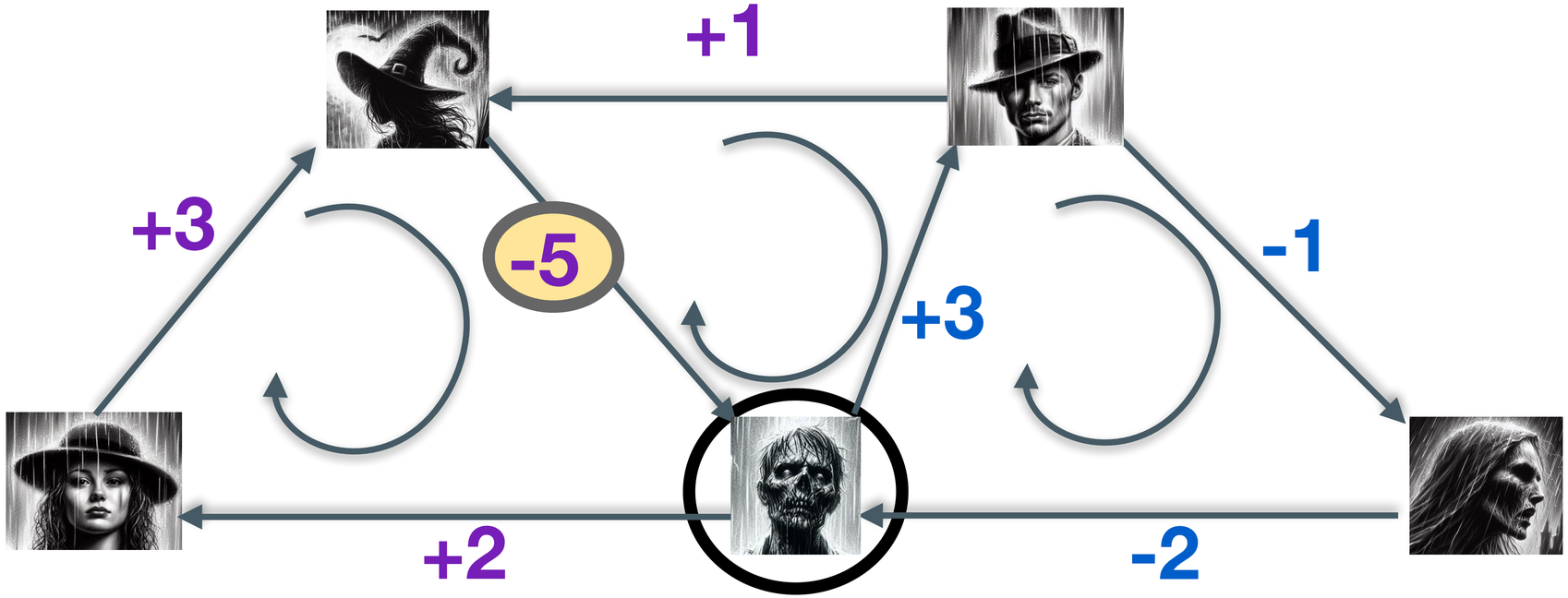
Huygens: "*triangulating*" to minimize clock sync errors



Huygens: "*triangulating*" to minimize clock sync errors

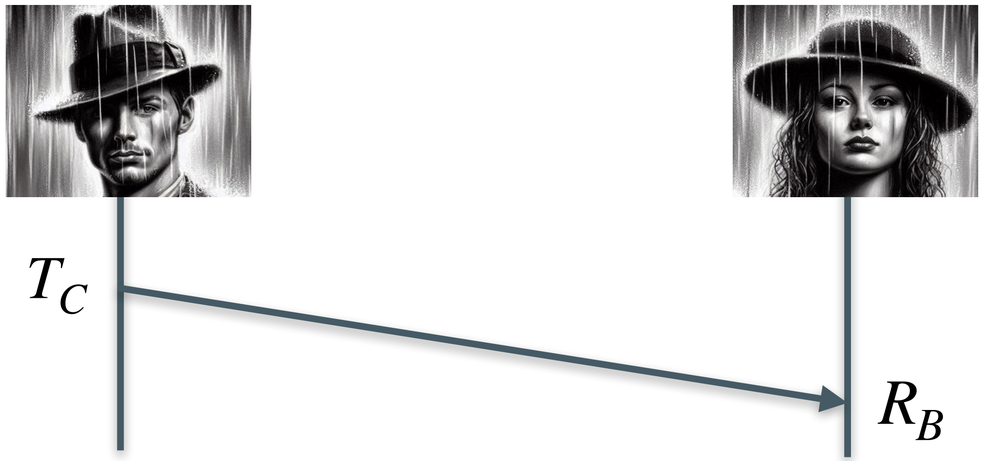


Huygens: "*triangulating*" to minimize clock sync errors

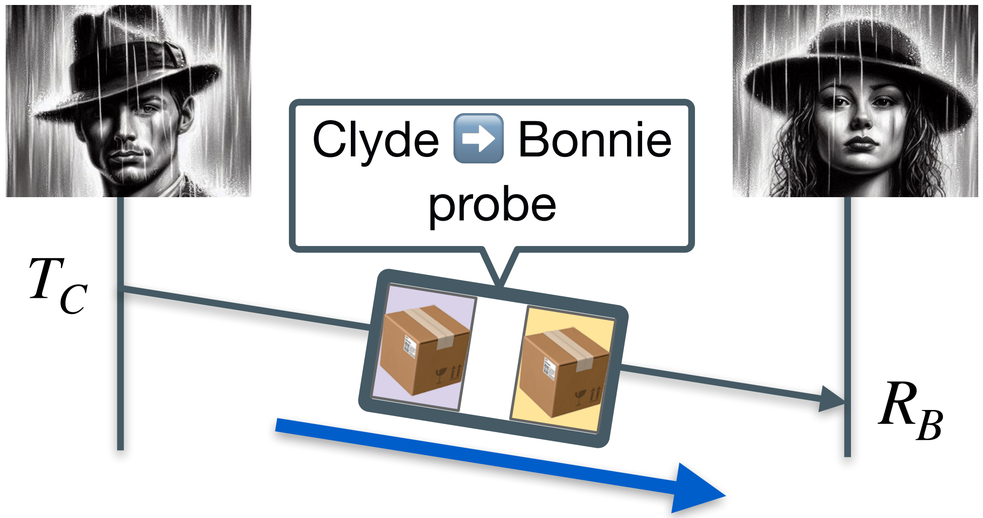


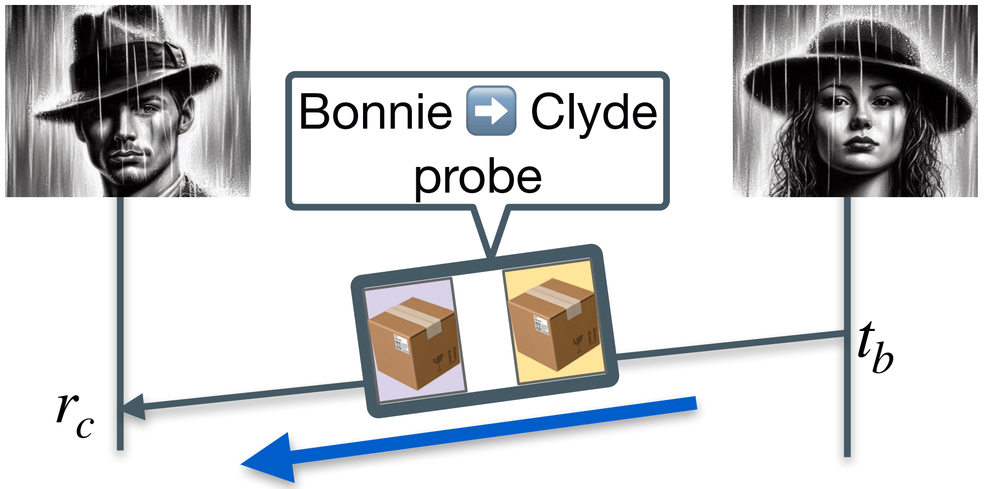
Reduces estimation errors caused by **path asymmetry**

Huygens: measuring one way delays

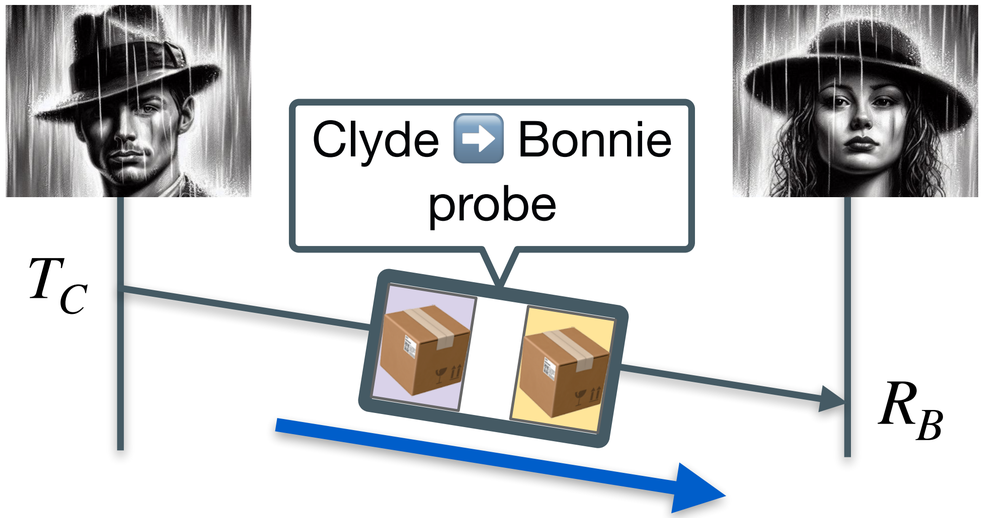


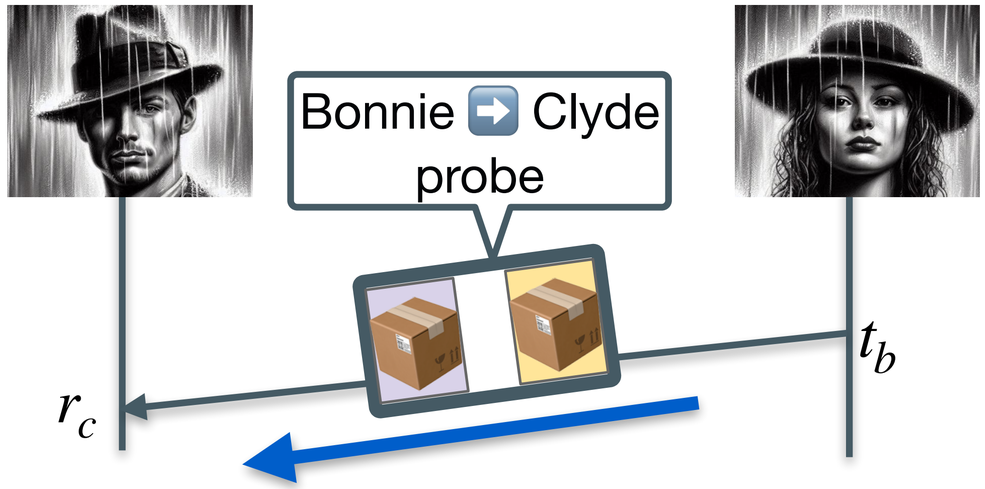
Huygens: measuring one way delays





Huygens: measuring one way delays



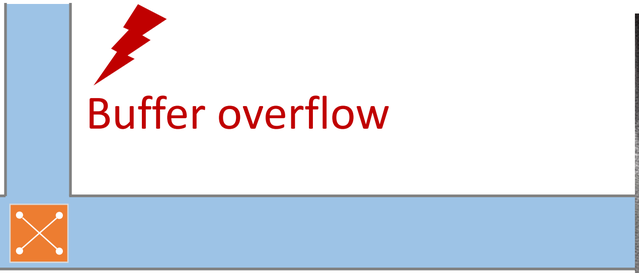


Measures forward and backward one way delays at a high precision

Sender

Receiver

Sender



Buffer overflow

Receiver

Increase tail latencies

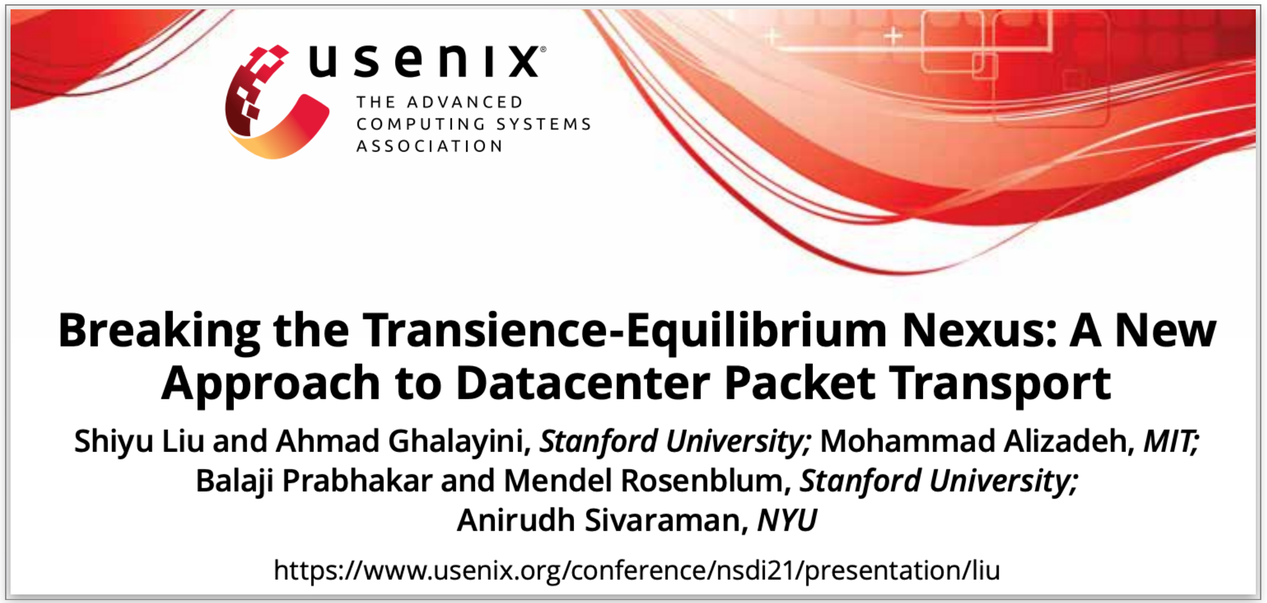
Retransmits,
Timeouts

Sender

Buffer overflow

Receiver

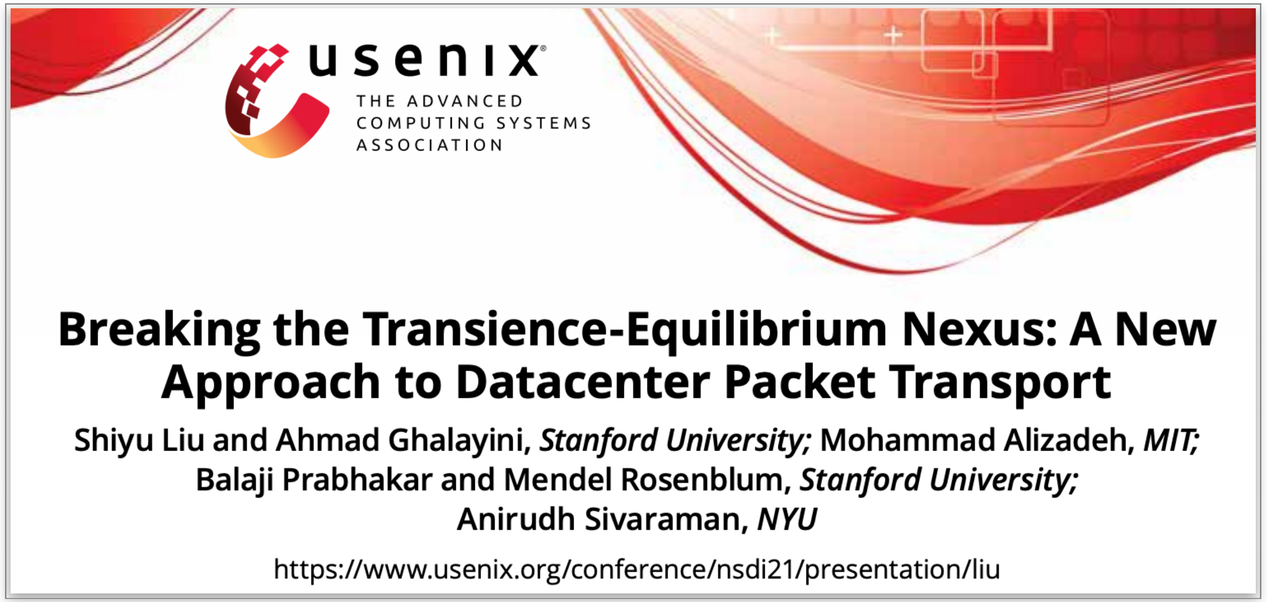
On-Ramp congestion control



https://www.usenix.org/conference/nsdi21/presentation/liu

On-Ramp congestion control

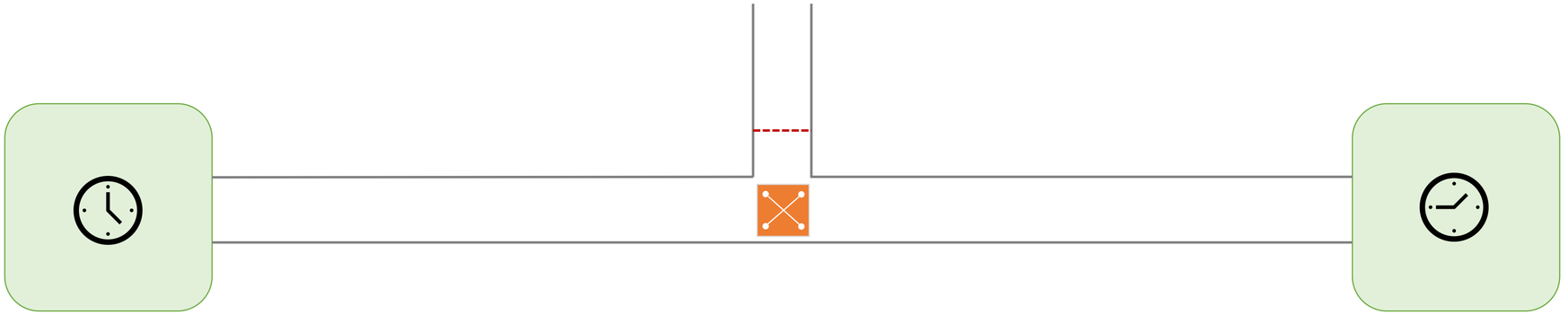
Detects congestion early and controls it



https://www.usenix.org/conference/nsdi21/presentation/liu

On-Ramp: buffer headroom

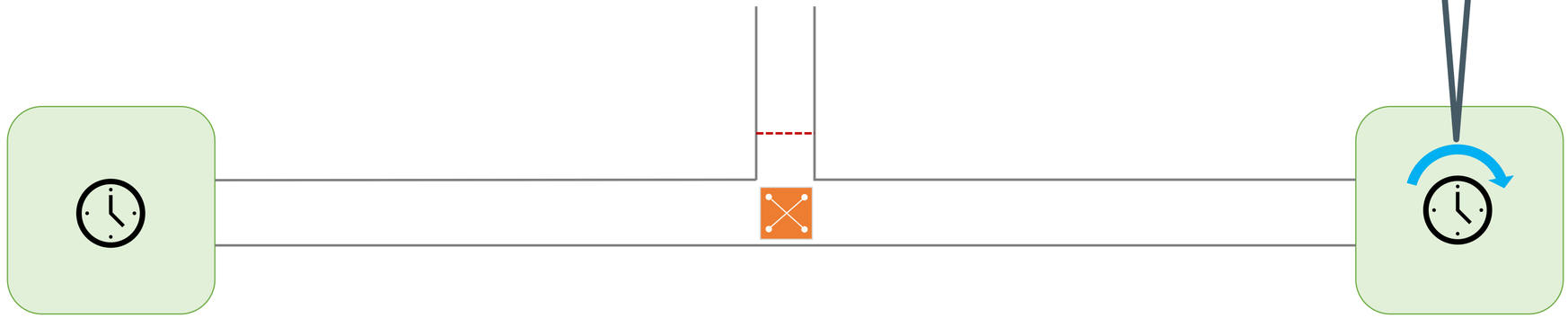
On-Ramp: buffer headroom



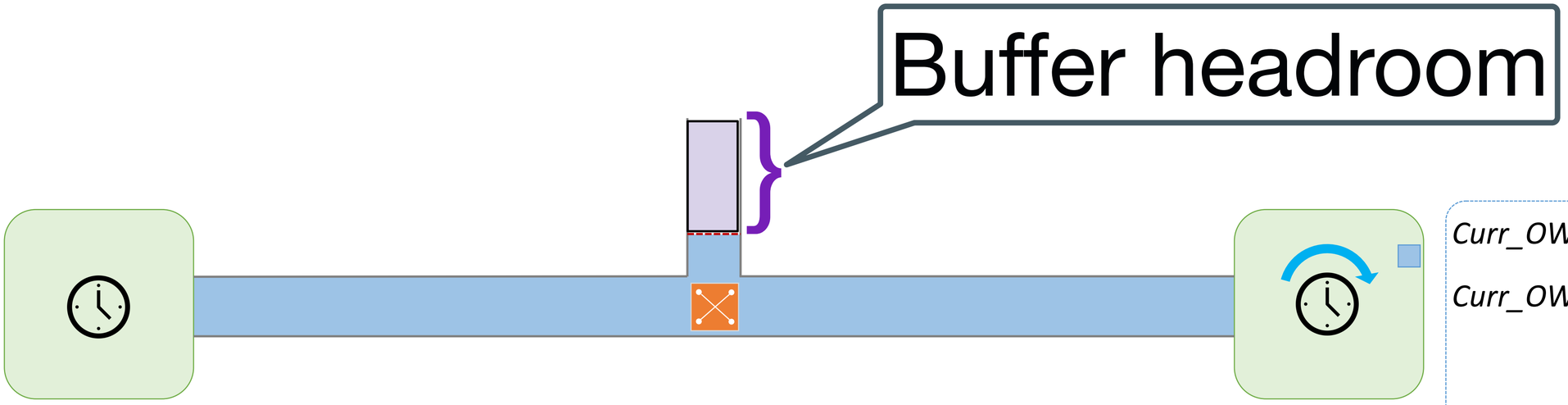
Sender

Receiver

On-Ramp: buffer headroom

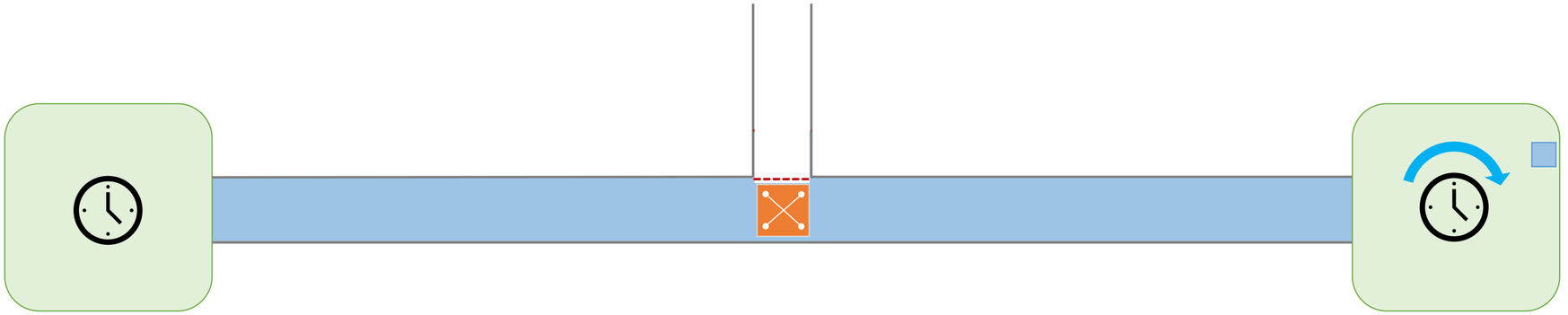


On-Ramp: buffer headroom



Let’s pause for a few microseconds

On-Ramp: buffer headroom



Sender

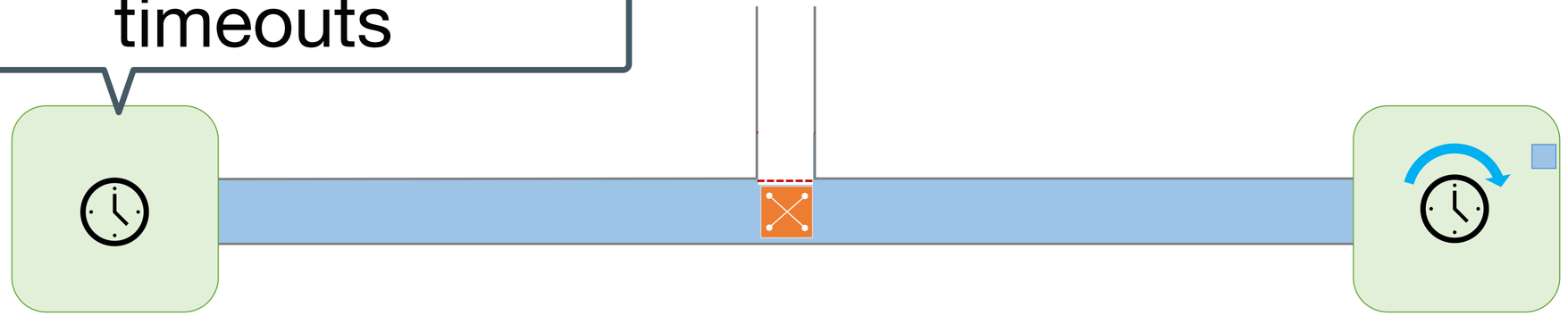
Receiver

$Curr\_OWD = T_{Rx} - T_{Tx}$

$Curr\_OWD > Thresh$?

On-Ramp: buffer headroom

Eliminates retransmits, timeouts



Sender

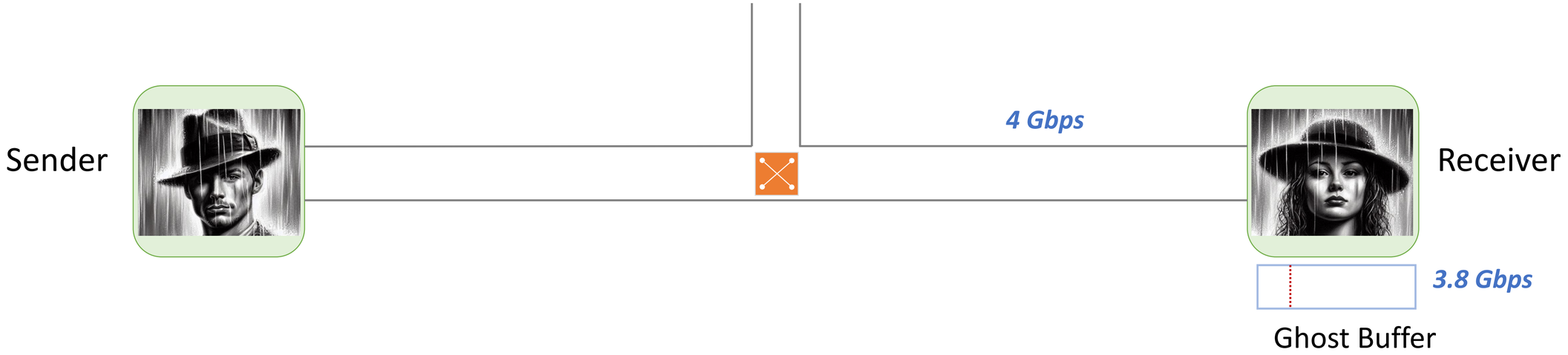
Receiver

Curr_OWD = $T_{Rx} - T_{Tx}$

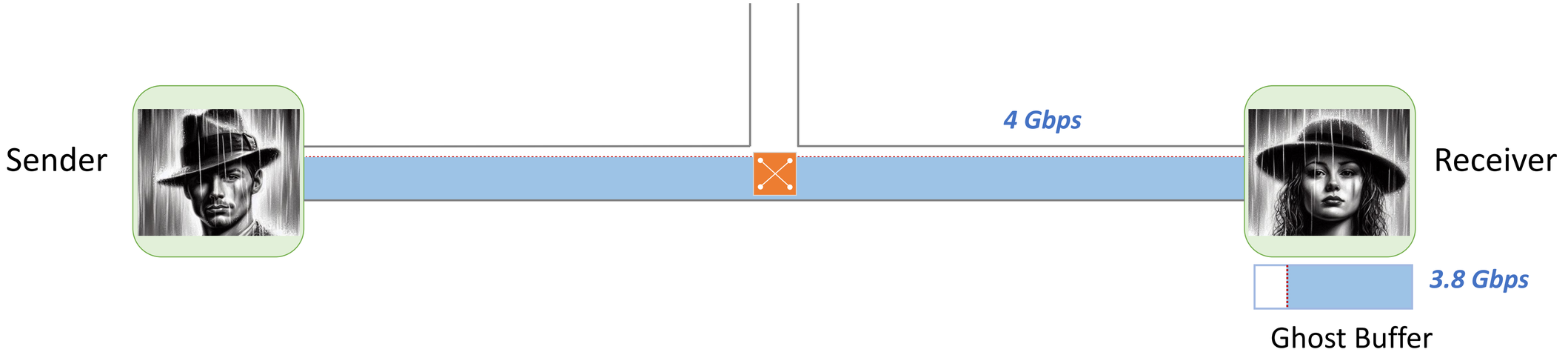
Curr_OWD > *Thresh* ?

Bring it on!

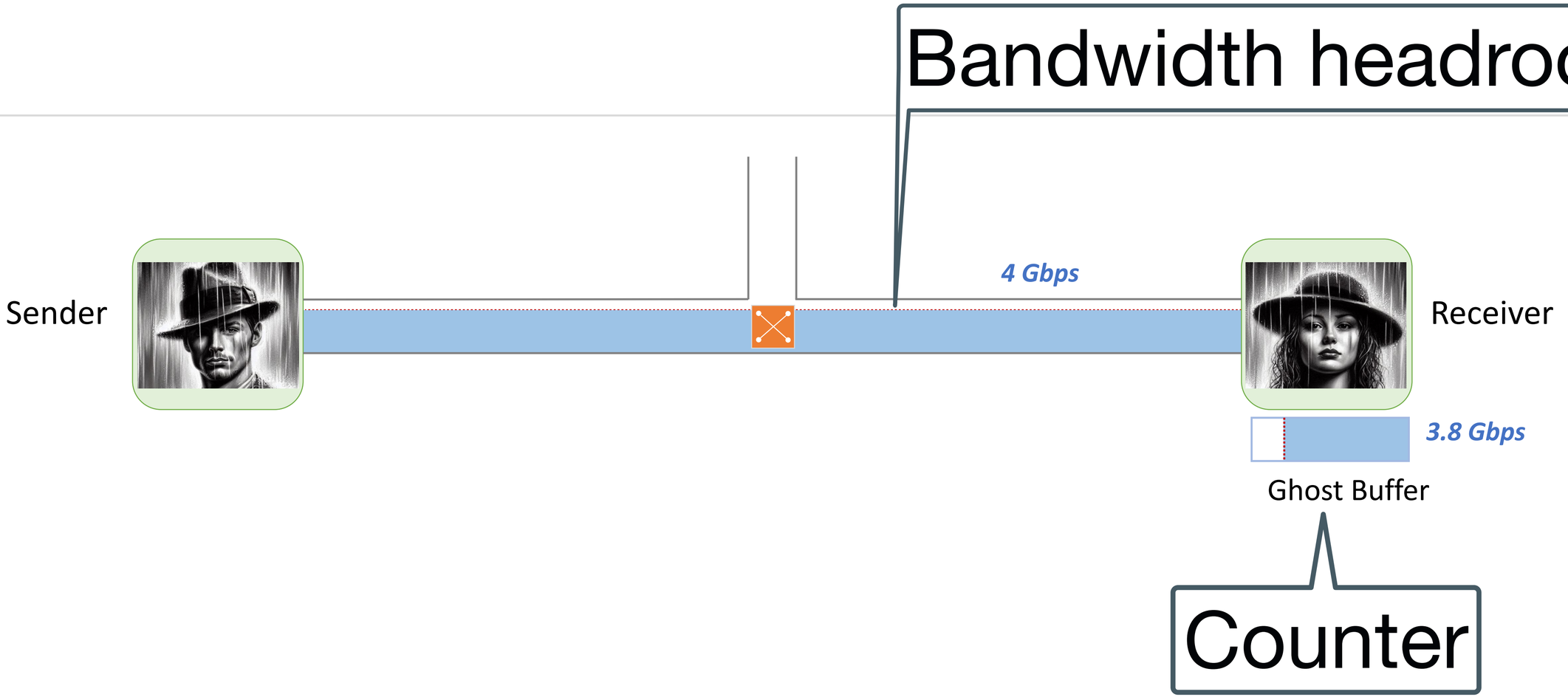
On-Ramp: optimize goodput with bandwidth headroom



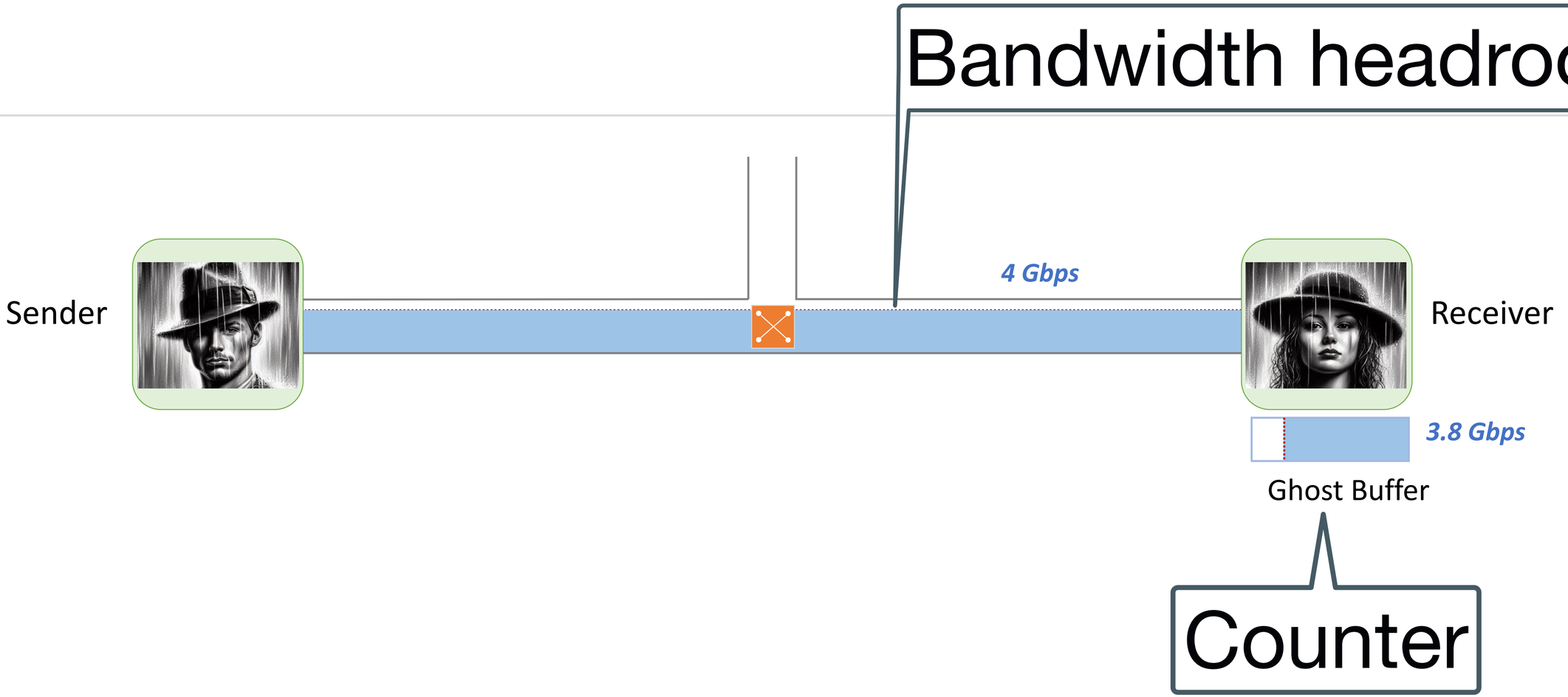
On-Ramp: optimize goodput with bandwidth headroom



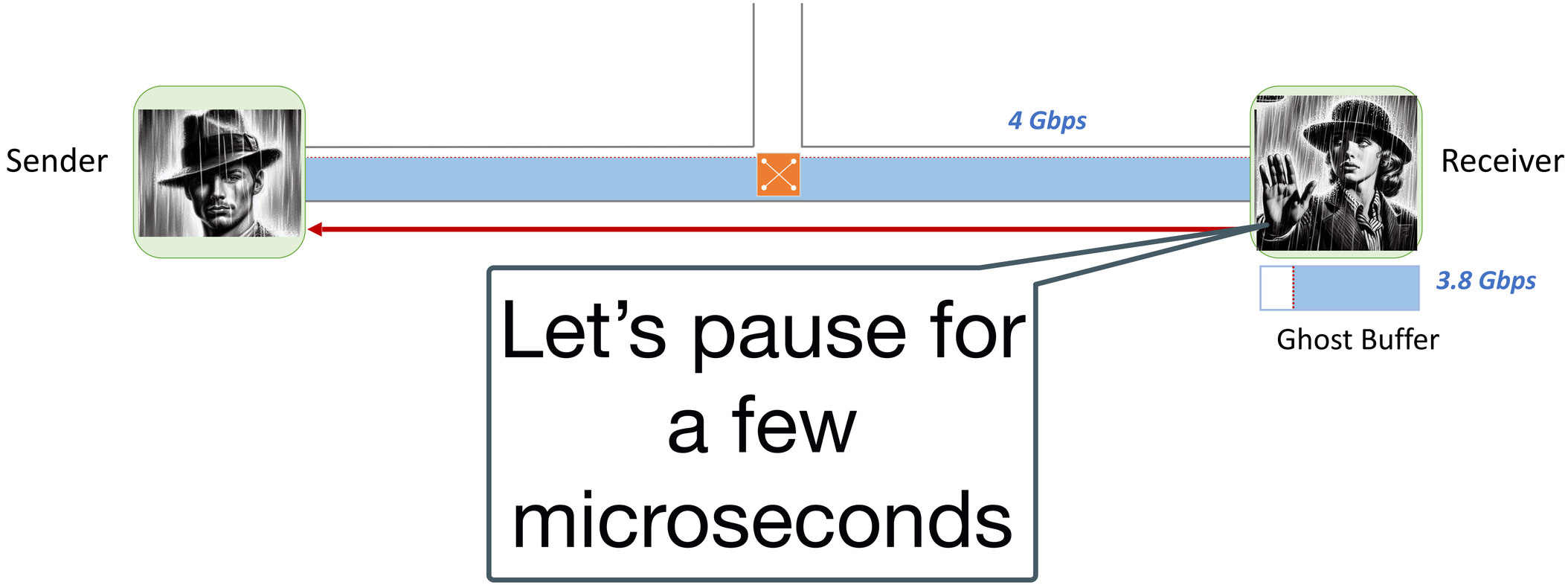
On-Ramp: optimize goodput with bandwidth headroom



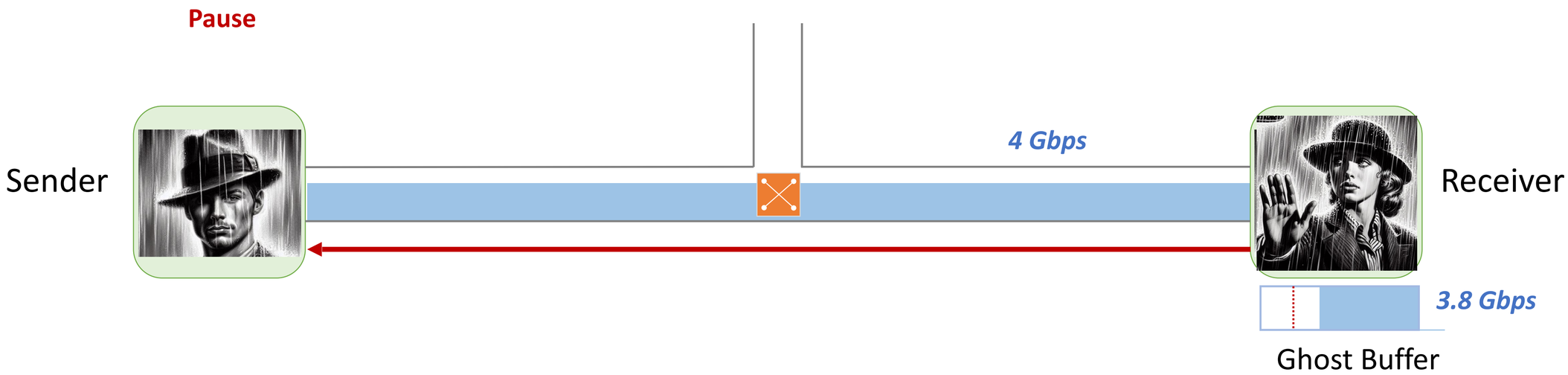
On-Ramp: optimize goodput with bandwidth headroom



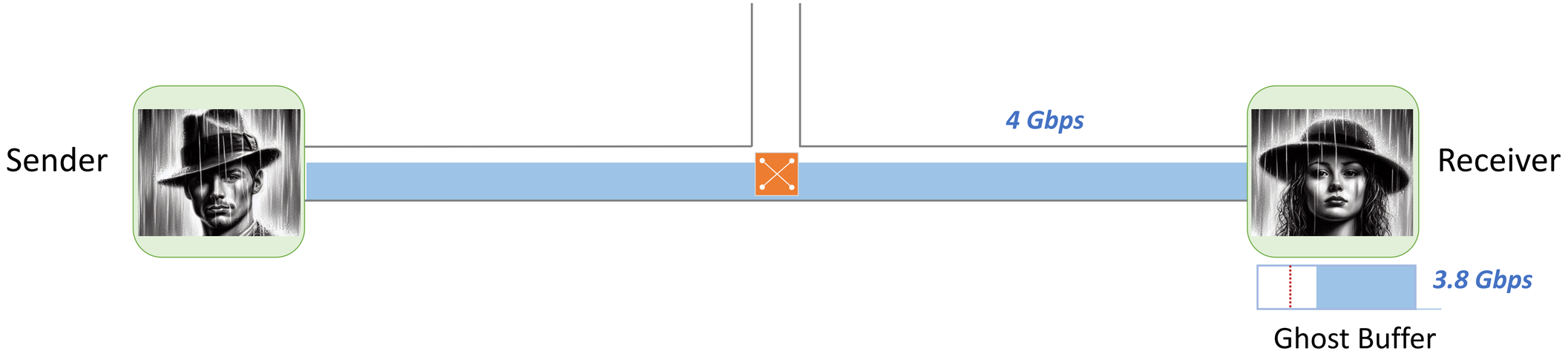
On-Ramp: optimize goodput with bandwidth headroom



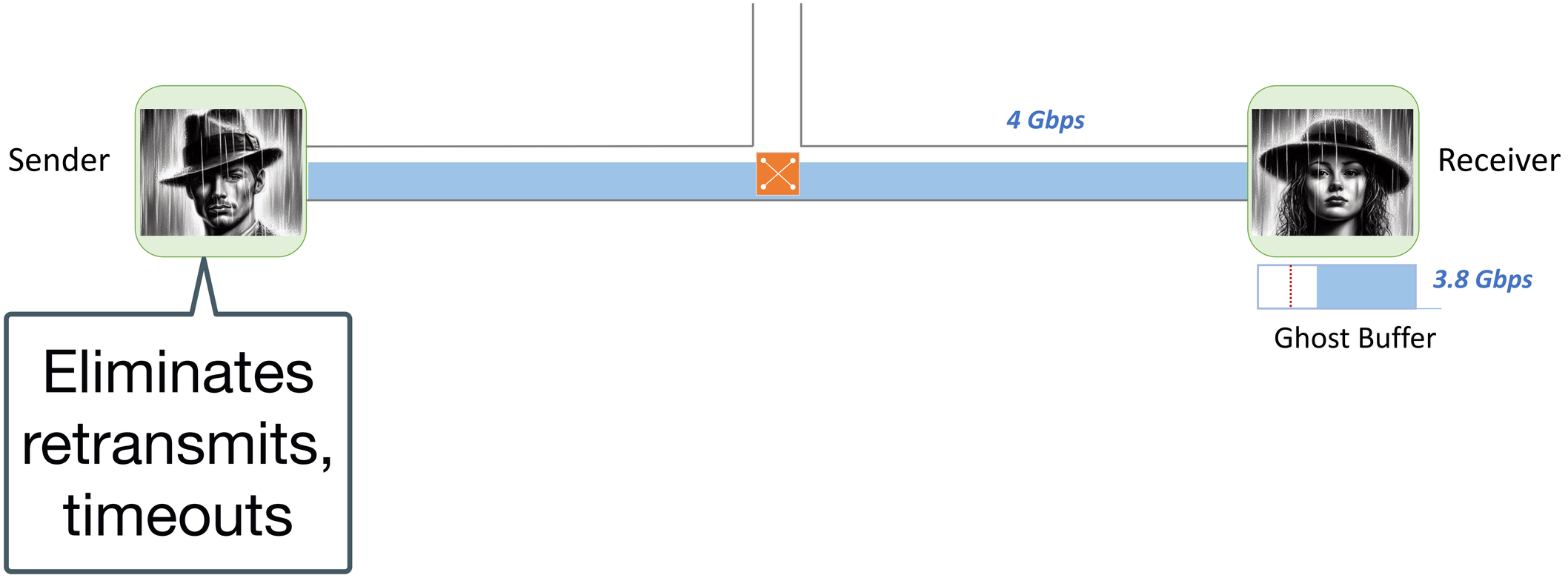
On-Ramp: optimize goodput with bandwidth headroom



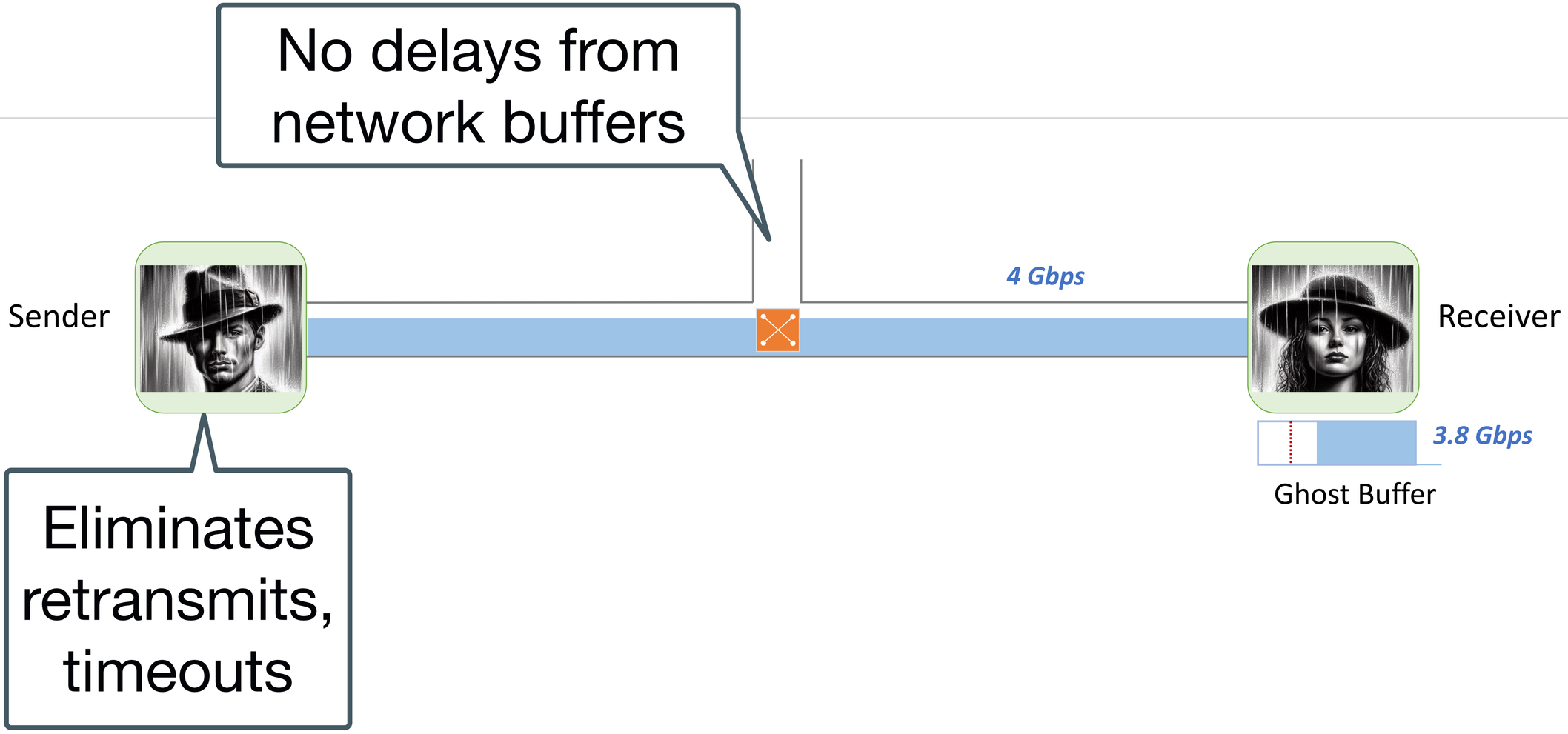
On-Ramp: optimize goodput with bandwidth headroom



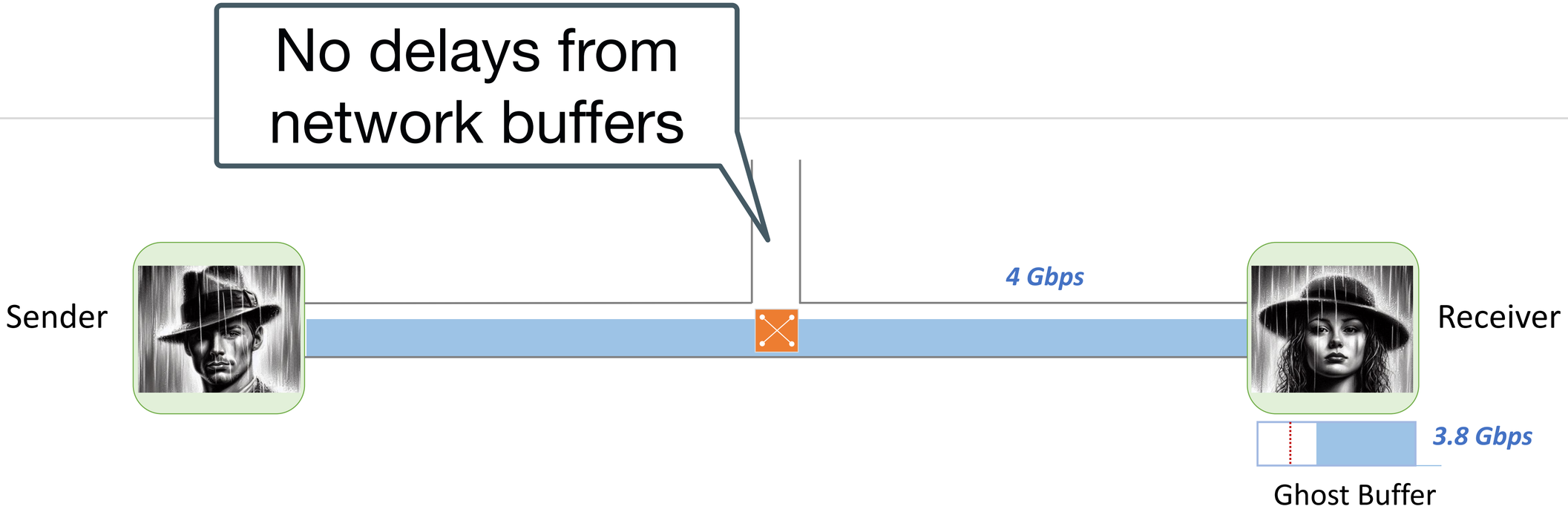
On-Ramp: optimize goodput with bandwidth headroom



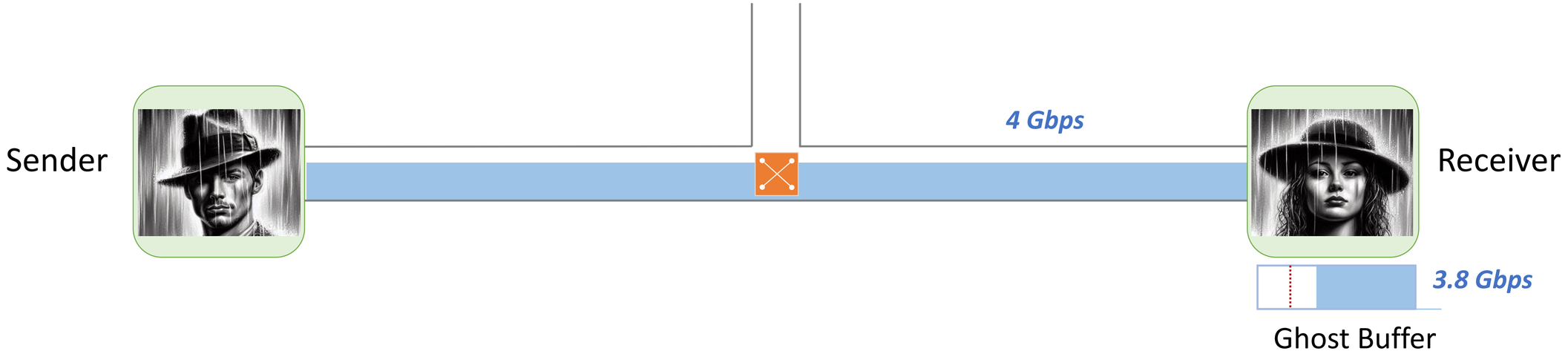
On-Ramp: optimize goodput with bandwidth headroom



On-Ramp: optimize goodput with bandwidth headroom



On-Ramp: optimize goodput with bandwidth headroom



Bandwidth slicing from receiver's end

Senders

4 Gbps

Receiver

S1

S3

S2

Bandwidth slicing from receiver's end

Senders

4 Gbps

Receiver

S1

S3

S2

2.3 Gbps

Bandwidth slicing from receiver's end

Senders

4 Gbps

Receiver

S1

S3

S2

2.3 Gbps

1.0 Gbps

Bandwidth slicing from receiver's end

Senders

4 Gbps

Receiver

S1

S3

S2

0.5 Gbps

2.3 Gbps

1.0 Gbps

Bandwidth slicing from receiver's end

Receiver slices 🔪 bandwidth according to its desired **priority**

Senders

Receiver

4 Gbps

S1

0.5 Gbps

S3

2.3 Gbps

🔪 GB3

S2

1.0 Gbps

Bandwidth slicing from receiver's end

Receiver slices 🔪 bandwidth according to its desired **priority**

Senders

Receiver

4 Gbps


S1

3 *0.5 Gbps*


S3

1 *2.3 Gbps* 🔪 GB3


S2

2 *1.0 Gbps* 🔪 GB2

Bandwidth slicing from receiver's end

Receiver slices 🔪 bandwidth according to its desired **priority**

Senders

Receiver

4 Gbps


S1

3 *0.5 Gbps* 🔪 GB1


S3

1 *2.3 Gbps* 🔪 GB3


S2

2 *1.0 Gbps* 🔪 GB2

If it is

I did it!

If it is

Huygens

With **clock offsets and drifts** at a high precision, we can measure 

delays.

On-Ramp

If we know one way delays, we can **detect** and **control** upcoming **congestion at a high precision**.

Thank You

Huygens

Huygens

On-Ramp

On-Ramp

Lerna Ekmekcioglu

🐳lerna@clockwork.io