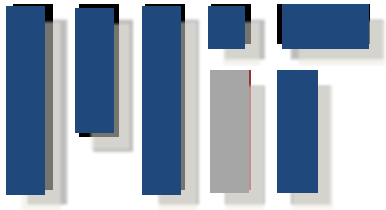
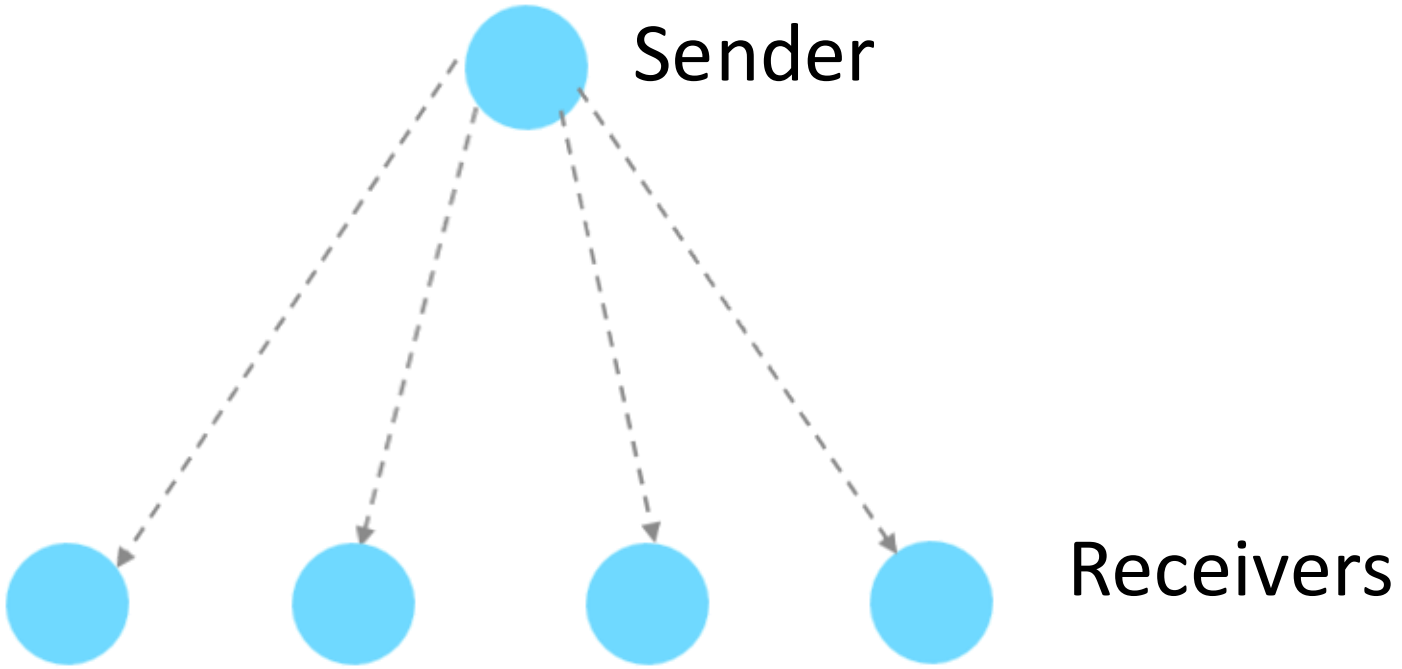
SourceSync: A Distributed Architecture for Sender Diversity

Hariharan Rahul
Haitham Hassanieh
Dina Katabi



Diversity is a fundamental property of wireless networks

Receiver Diversity



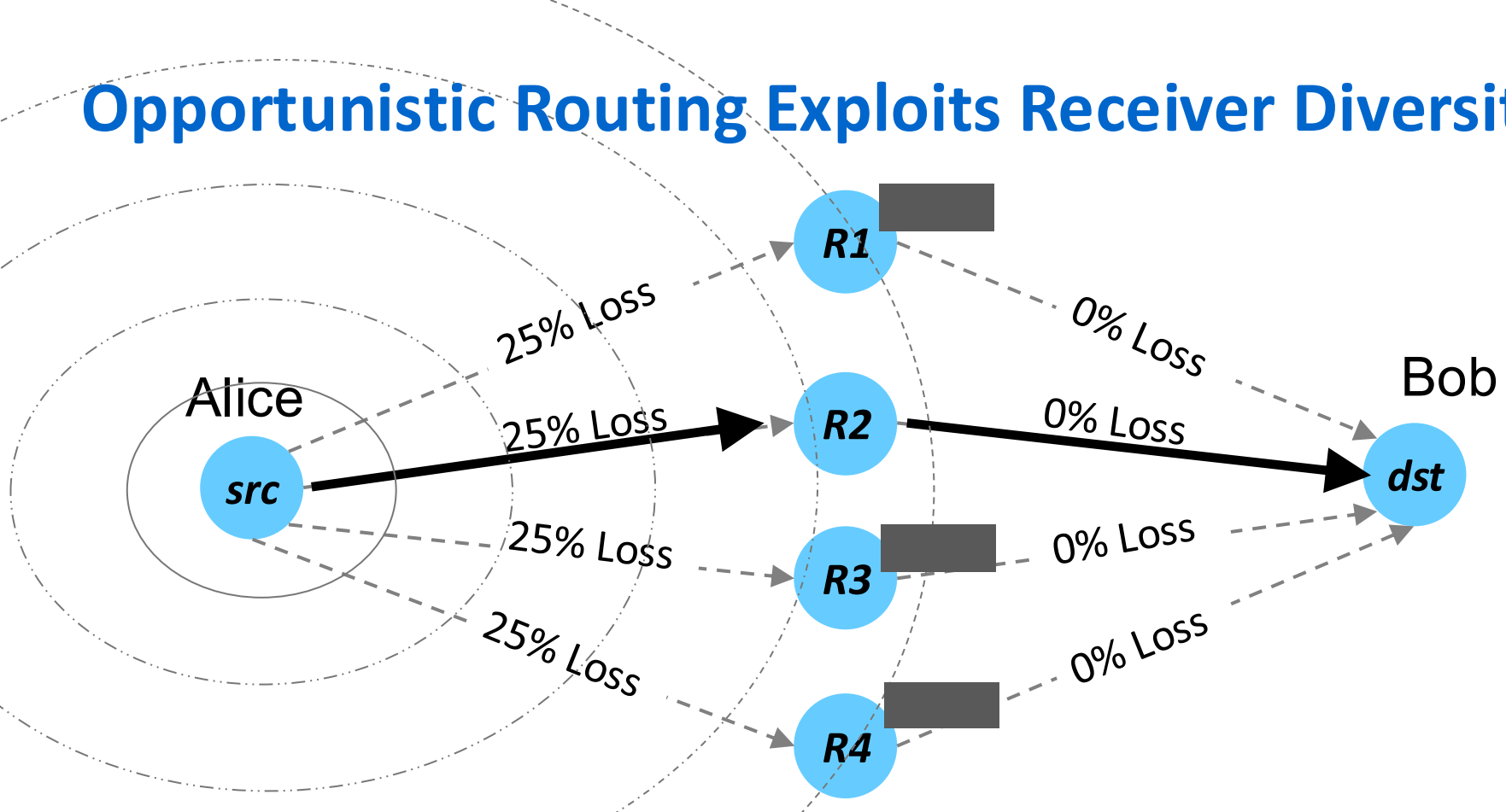
- Broadcast
- Unlikely paths to all receivers are attenuated at the same time

Receiver Diversity Underlies Many Systems

- All Opportunistic Routing Protocols
  - ExOR, MORE, MIXIT, ROMER, SOAR, …
- WLAN Diversity
  - MRD, SOFT, Link-Alike, Vi-Fi, …

Opportunistic Routing Exploits Receiver Diversity

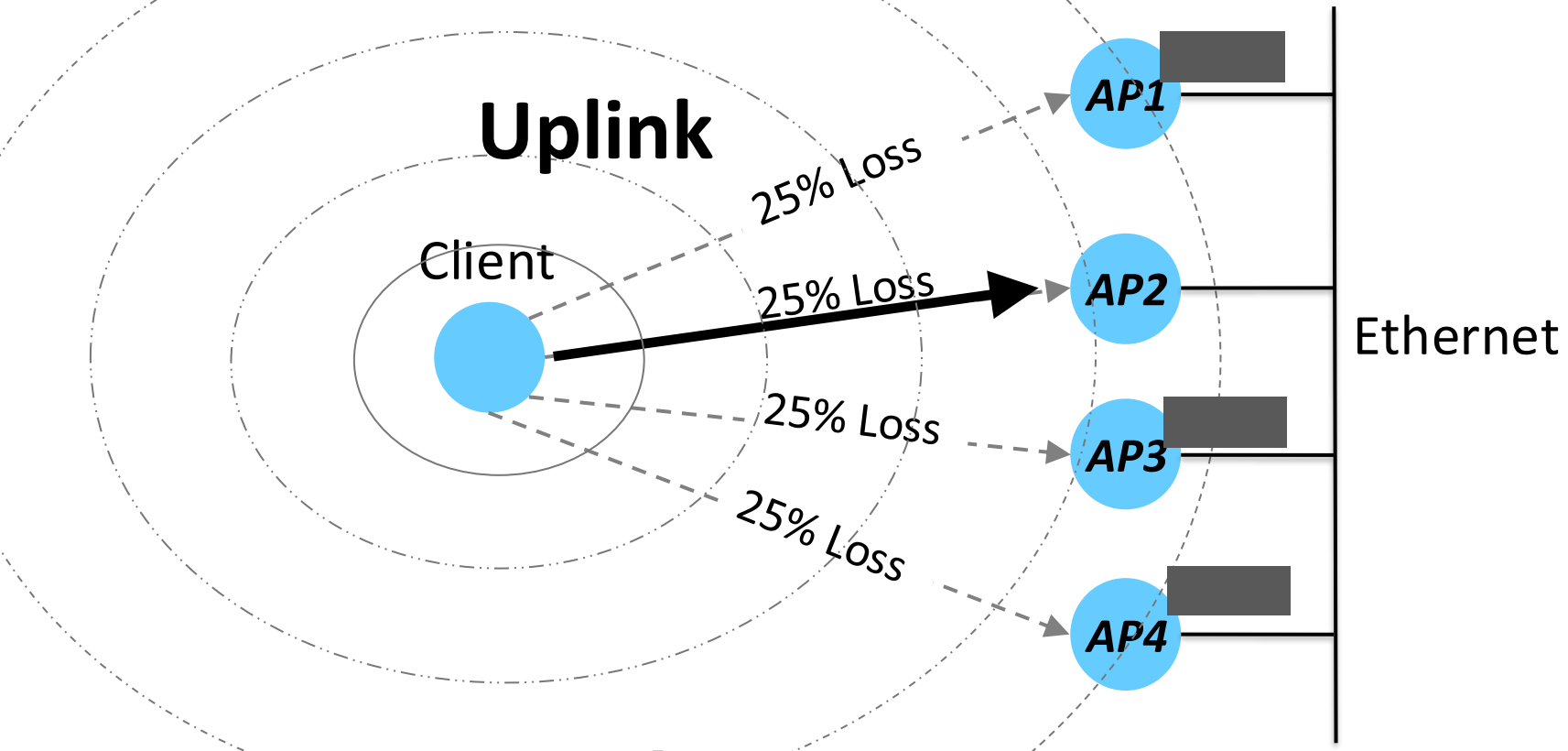
Opportunistic Routing Exploits Receiver Diversity



Best single path → Loss Probability is 25%

Any router can forward → Receiver Diversity → Unlikely all routers see a loss → Loss probability $(0.25)^4 = 1\%$

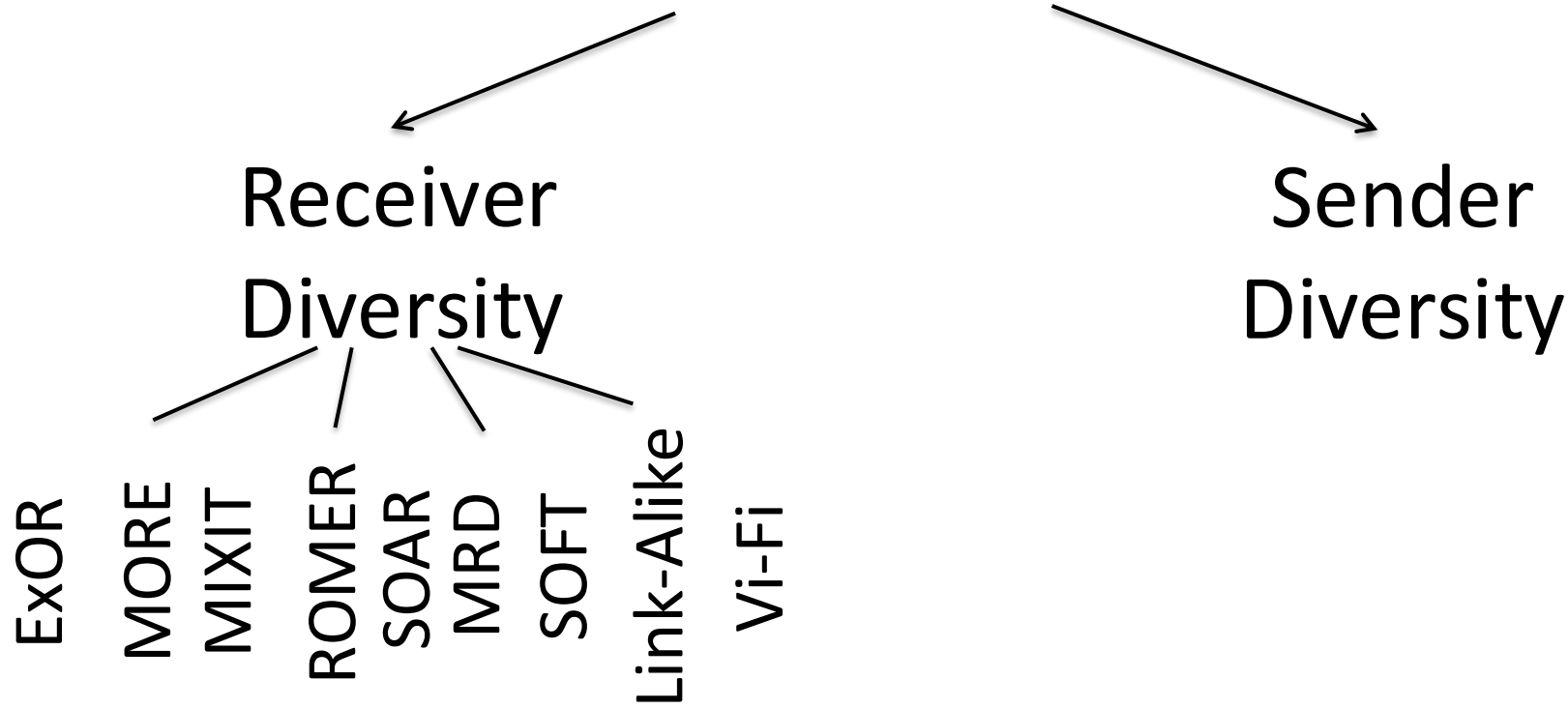
WLAN Diversity Exploits Receiver Diversity



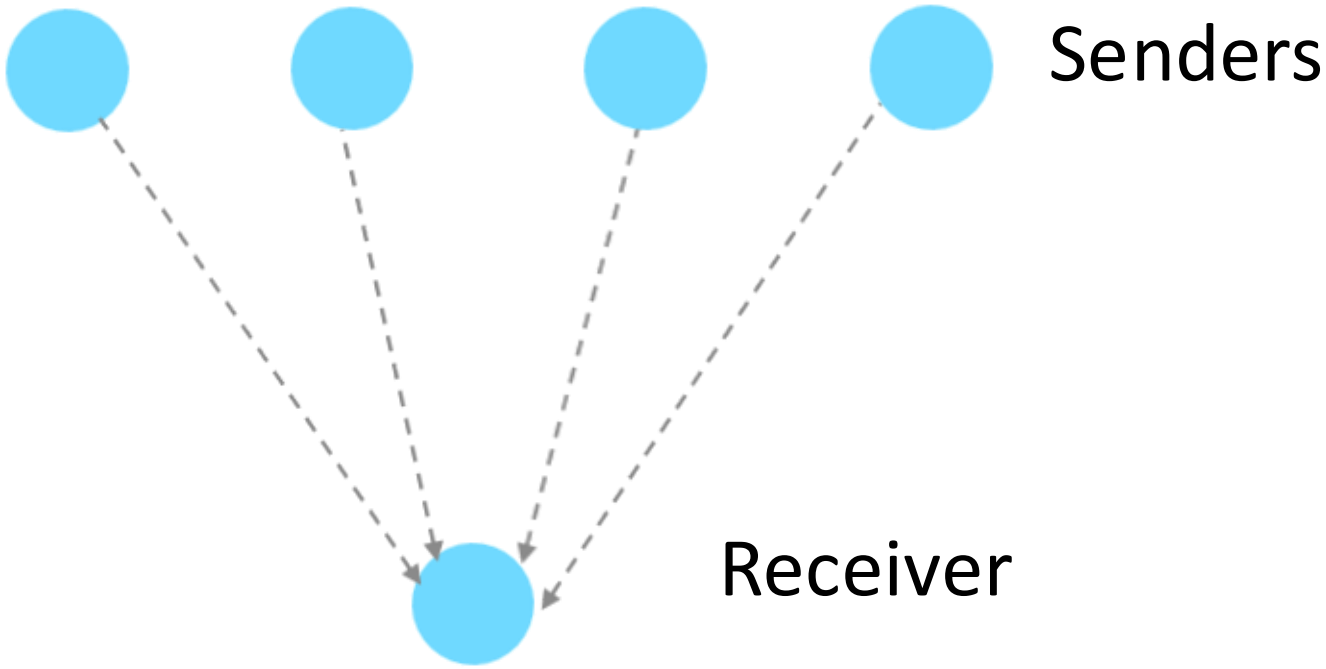
Best Single AP → Uplink Loss Probability is 25%

Any AP can forward → Uplink Loss Prob. $(0.25)^4 < 1\%$

Diversity is a fundamental property of wireless networks

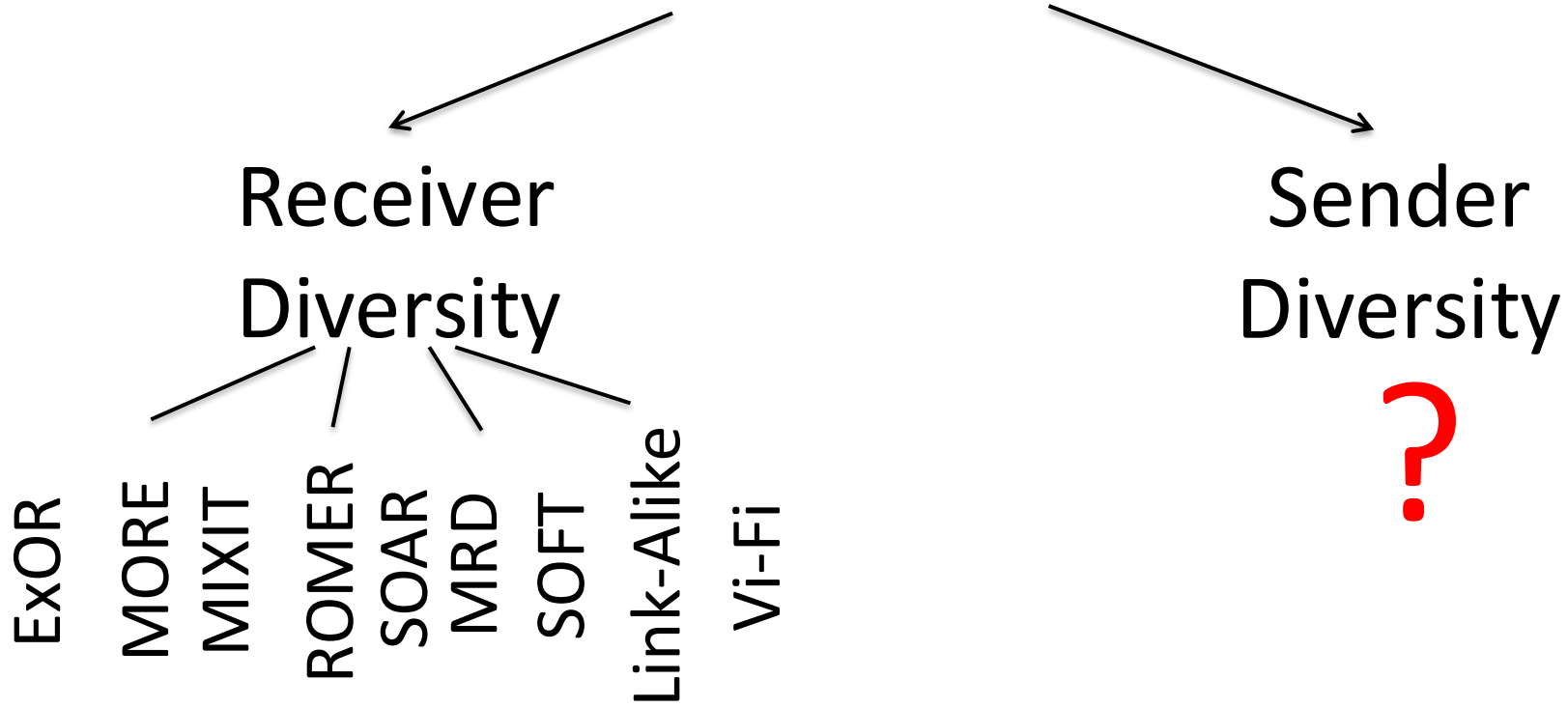


Sender Diversity



- Transmit simultaneously
- Unlikely paths from all senders are attenuated at the same time

Diversity is a fundamental property of wireless networks



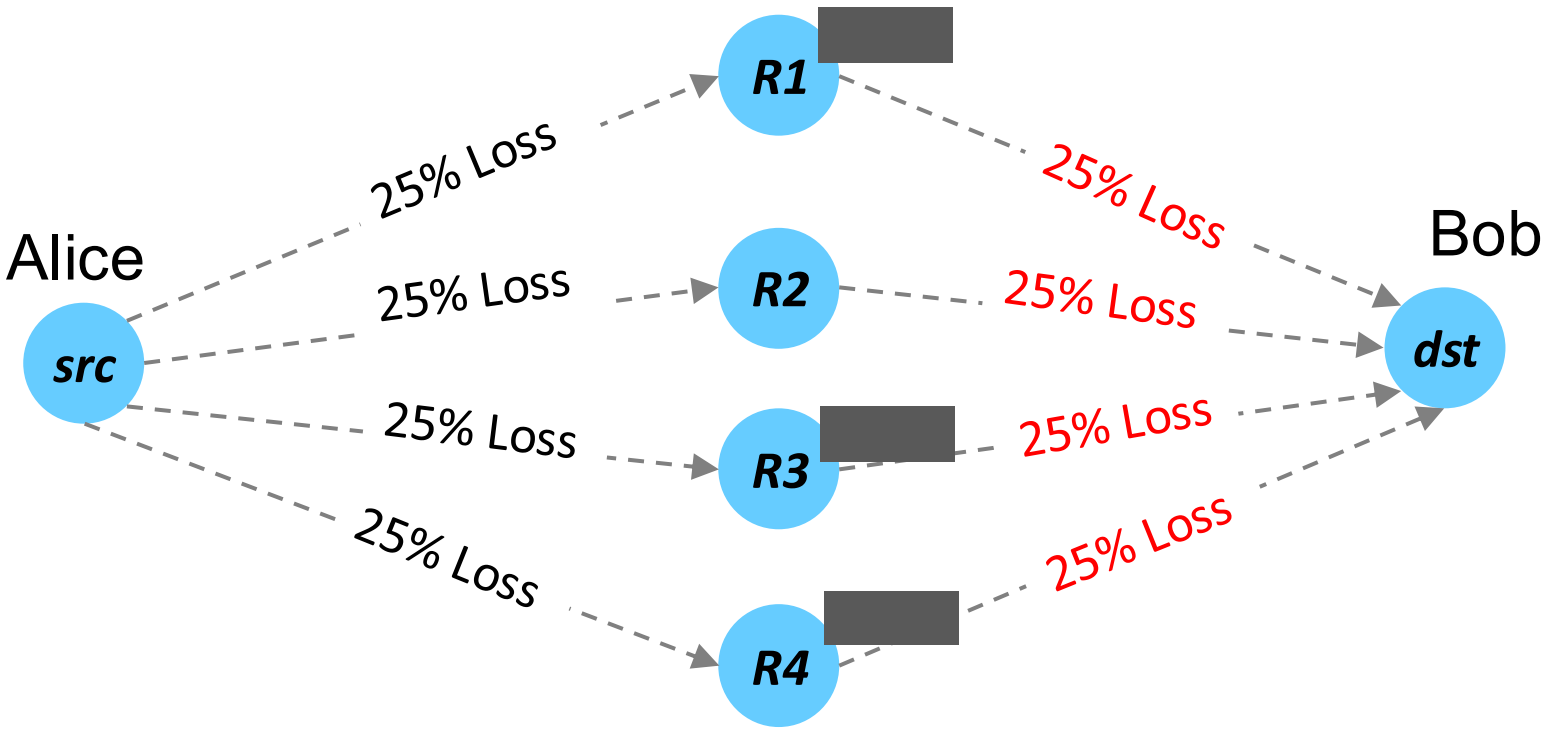
Is it because Sender Diversity has no benefits?

No

Sender Diversity has analogous benefits to Receiver Diversity

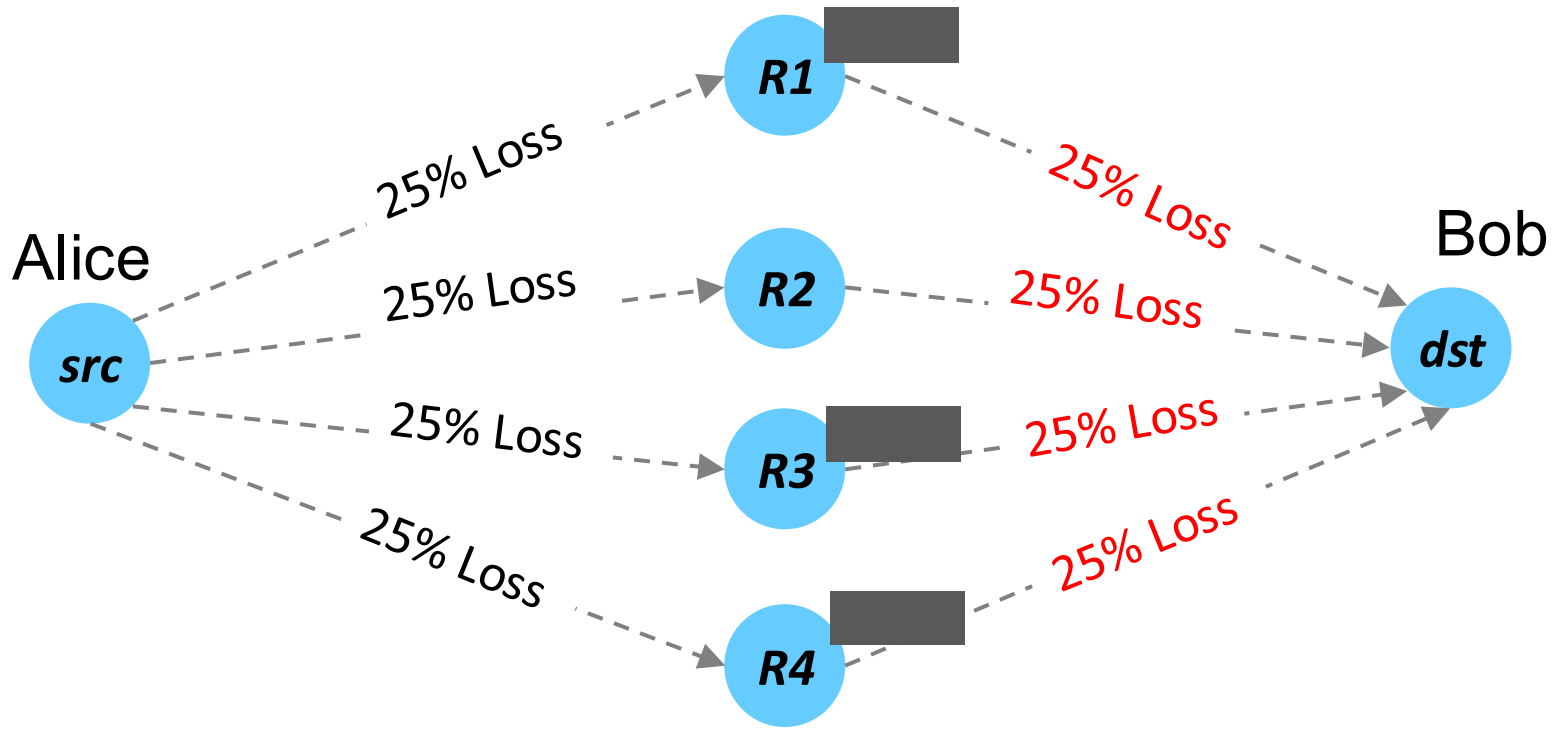
Sender Diversity Improves Opportunistic Routing

Sender Diversity Improves Opportunistic Routing



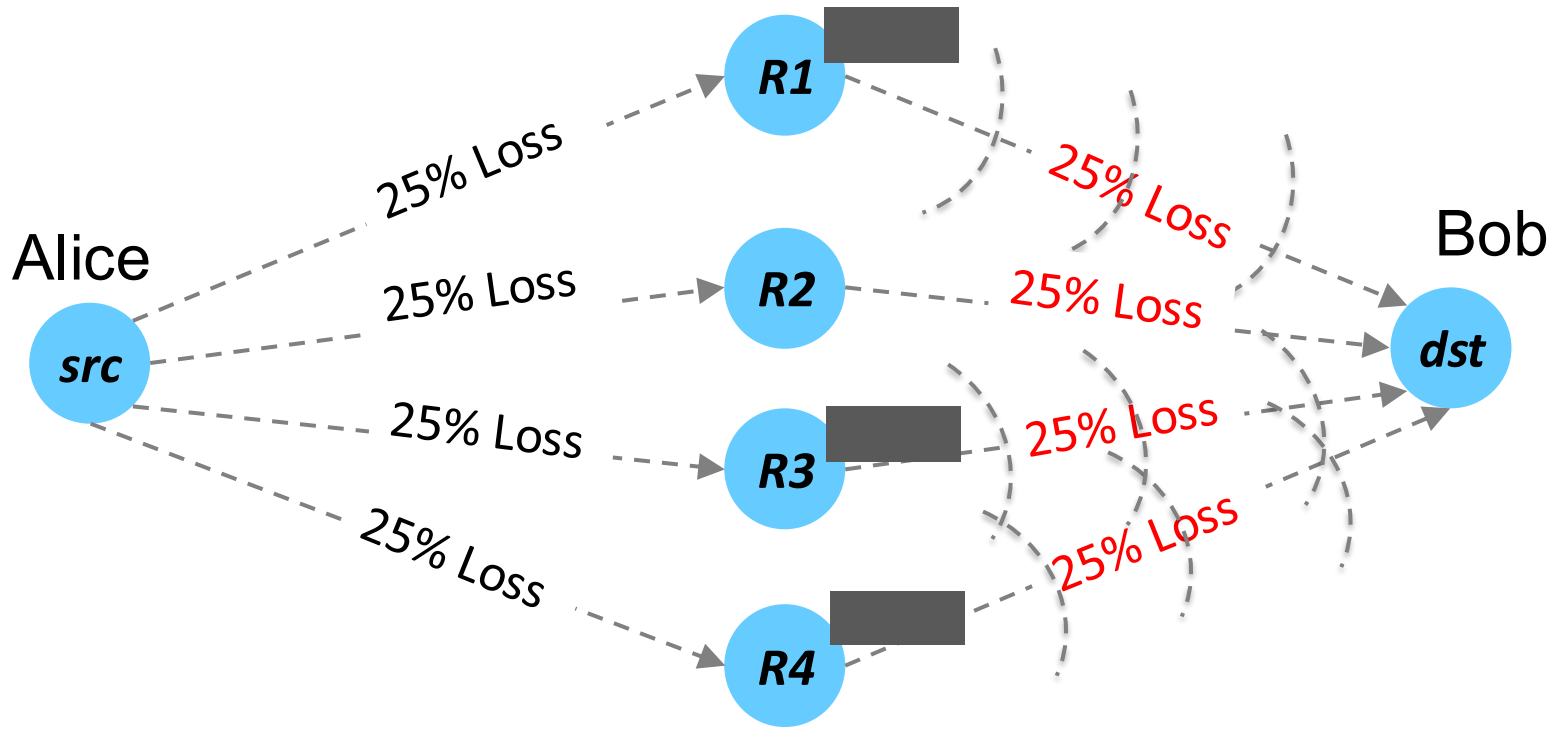
Opportunistic Routing picks single router to forward

Sender Diversity Improves Opportunistic Routing



Opportunistic Routing → Loss to Bob is 25%

Sender Diversity Improves Opportunistic Routing



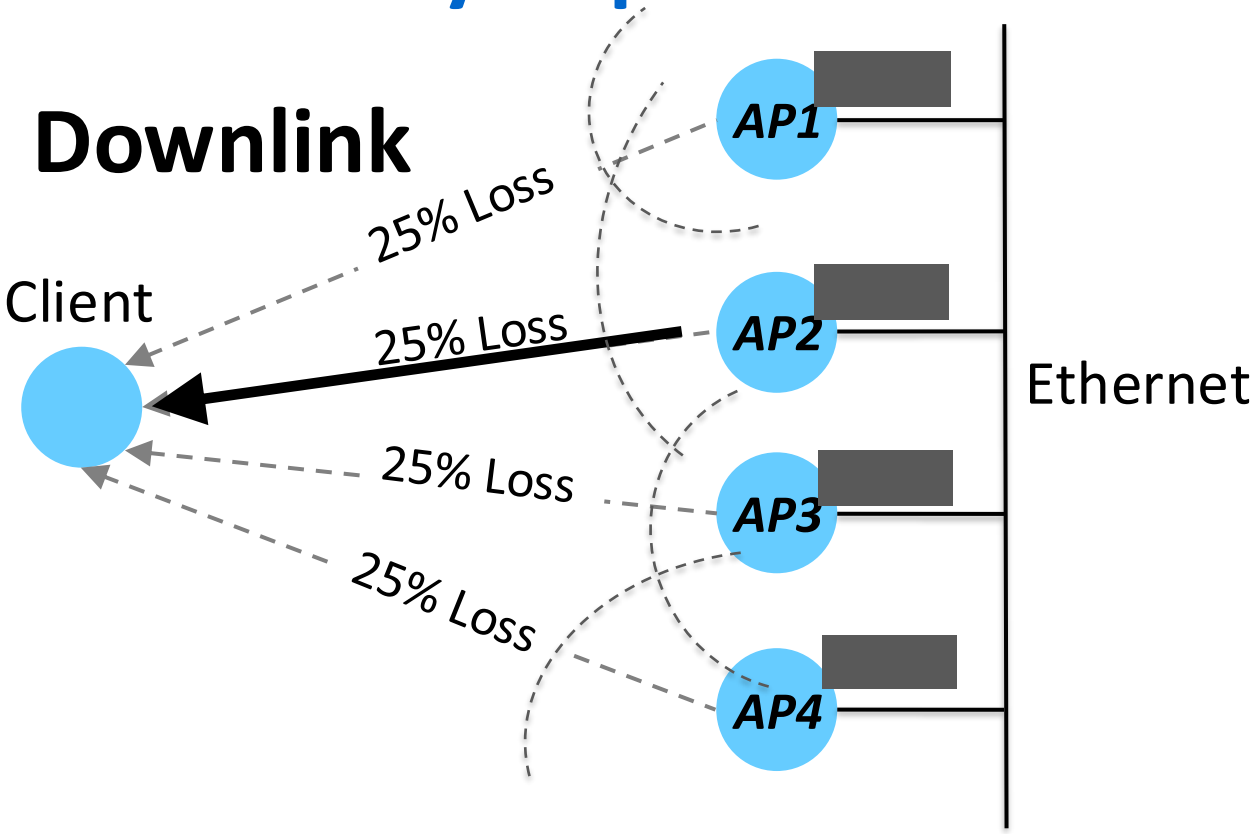
Opportunistic Routing → Loss to Bob is 25%

Forward simultaneously →

Sender Diversity → Unlikely Loss to Bob is $(0.25)^3 < 2\%$

attenuated

Sender Diversity Improves WLANs



WLAN Diversity→ Downlink Loss is 25%

Forwarder Diversity → Downlink Loss (0.25^4) < 1%

Sender Diversity → Unlikely path loss (0.25) are attenuated

Is it because Sender Diversity has no benefits?

No

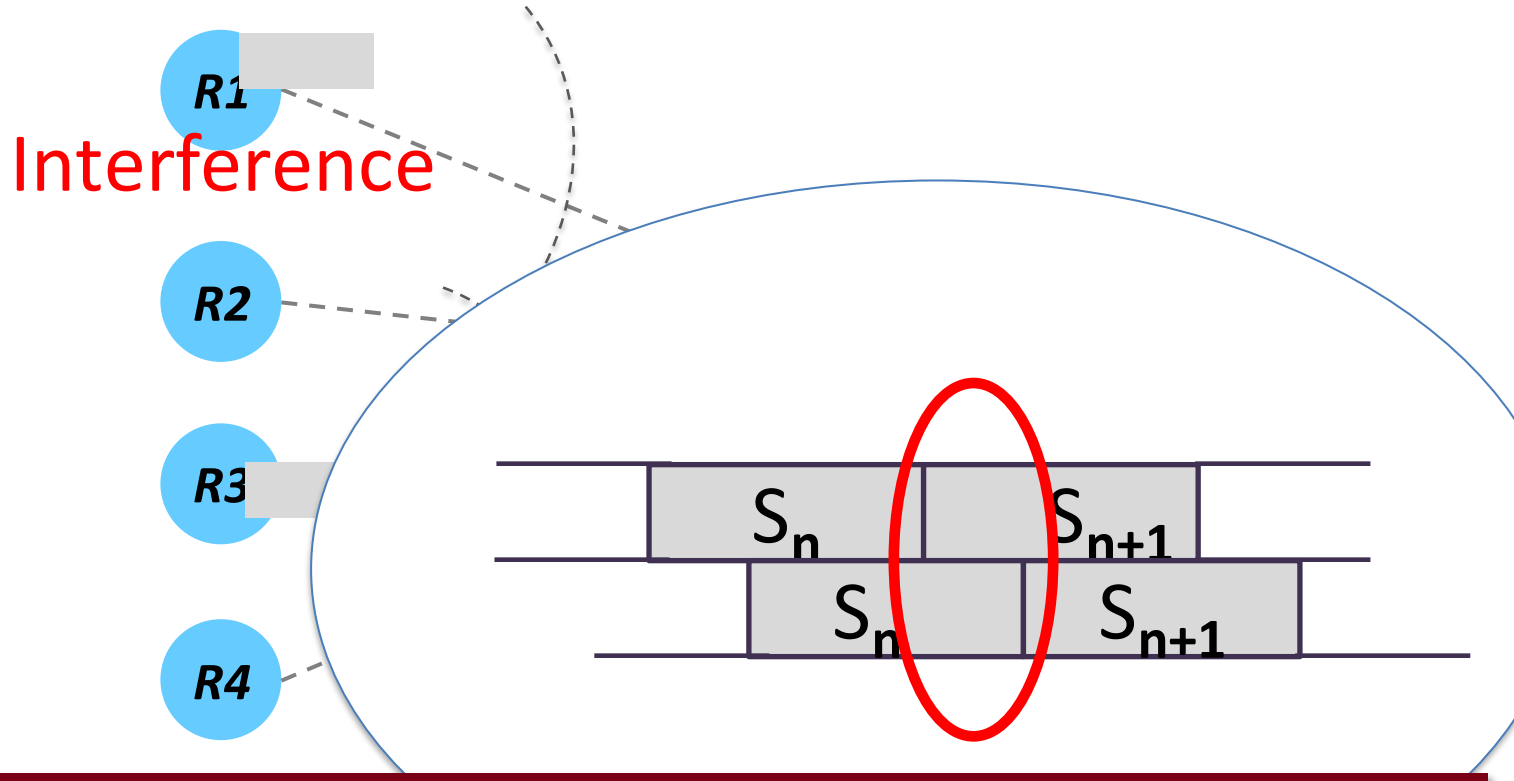
What Has Prevented Us from Using Sender Diversity?

Challenge

Today, simultaneous transmissions don’t strengthen each other

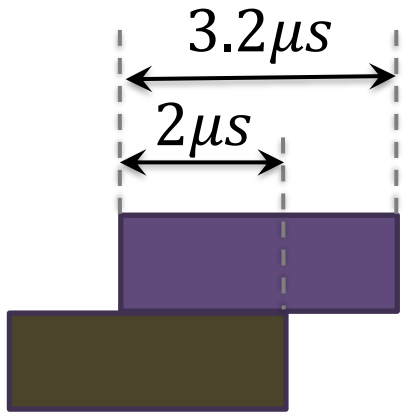
Challenge

Today, simultaneous transmissions don't strengthen each other



Need Distributed Symbol-Level Synchronization

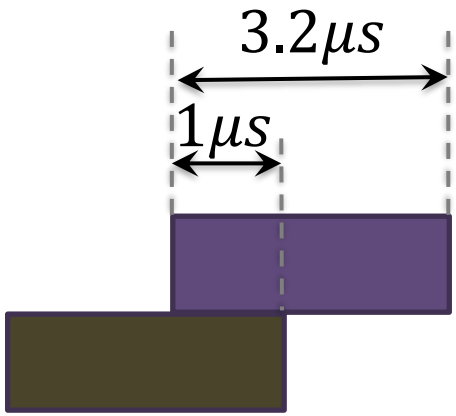
How Accurately Need We Synchronize?



- 802.11 Symbol Time is $3.2\mu s$
- Synchronization Error of $2\mu s$

→Maximum Possible SNR is $10log_{10}(\frac{3.2}{2})$ = 2 dB

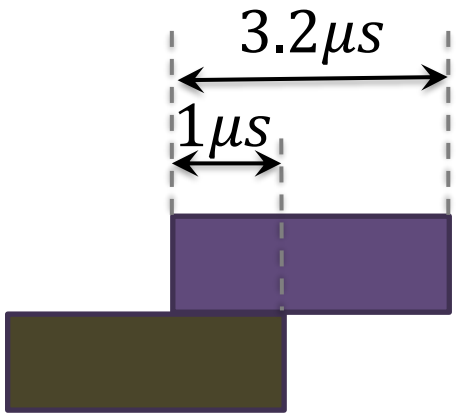
How Accurately Need We Synchronize?



- 802.11 Symbol Time is $3.2\mu s$
- Synchronization Error of $1\mu s$

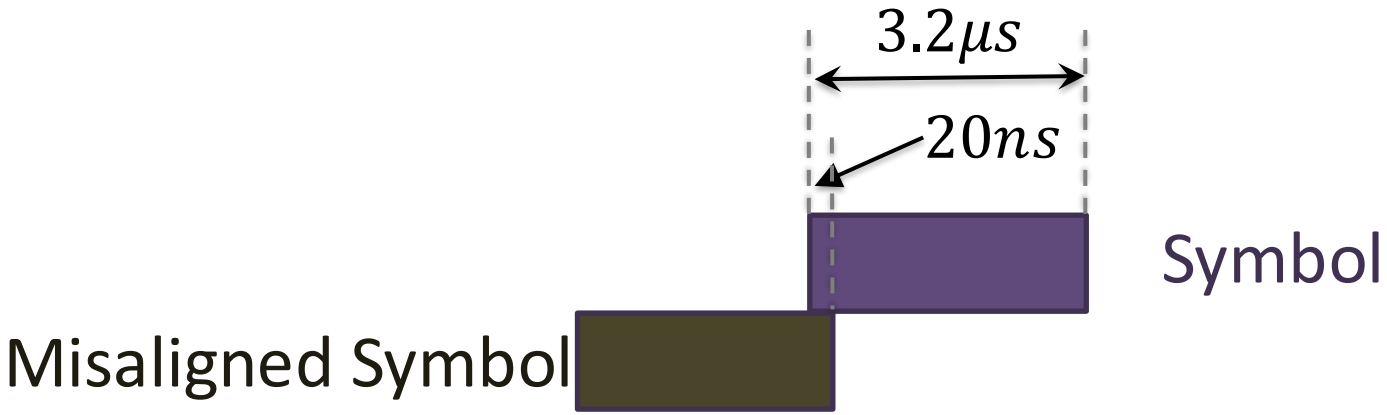
→Maximum Possible SNR is $10log_{10}(\frac{3.2}{1})$ = 5 dB

How Accurately Need We Synchronize?



- 802.11 Symbol Time is $3.2\mu s$
- For max. bitrate, 802.11 needs an SNR of $\sim$22dB

How Accurately Need We Synchronize?



- 802.11 Symbol Time is $3.2\mu s$
- For max. bitrate, 802.11 needs an SNR of ~22dB
  → Maximum Synchronization Error is 20 ns

Need to Synchronize Symbols to within 20 ns

SourceSync

- Provides distributed and accurate symbol-level synchronization (within 20 ns)
- Complements Opportunistic Routing by harnessing sender diversity gains
- Complements WLAN Diversity by reducing losses on the downlink
- Implemented in FPGA and evaluated in a wireless testbed

Talk is in the context of Opportunistic Routing

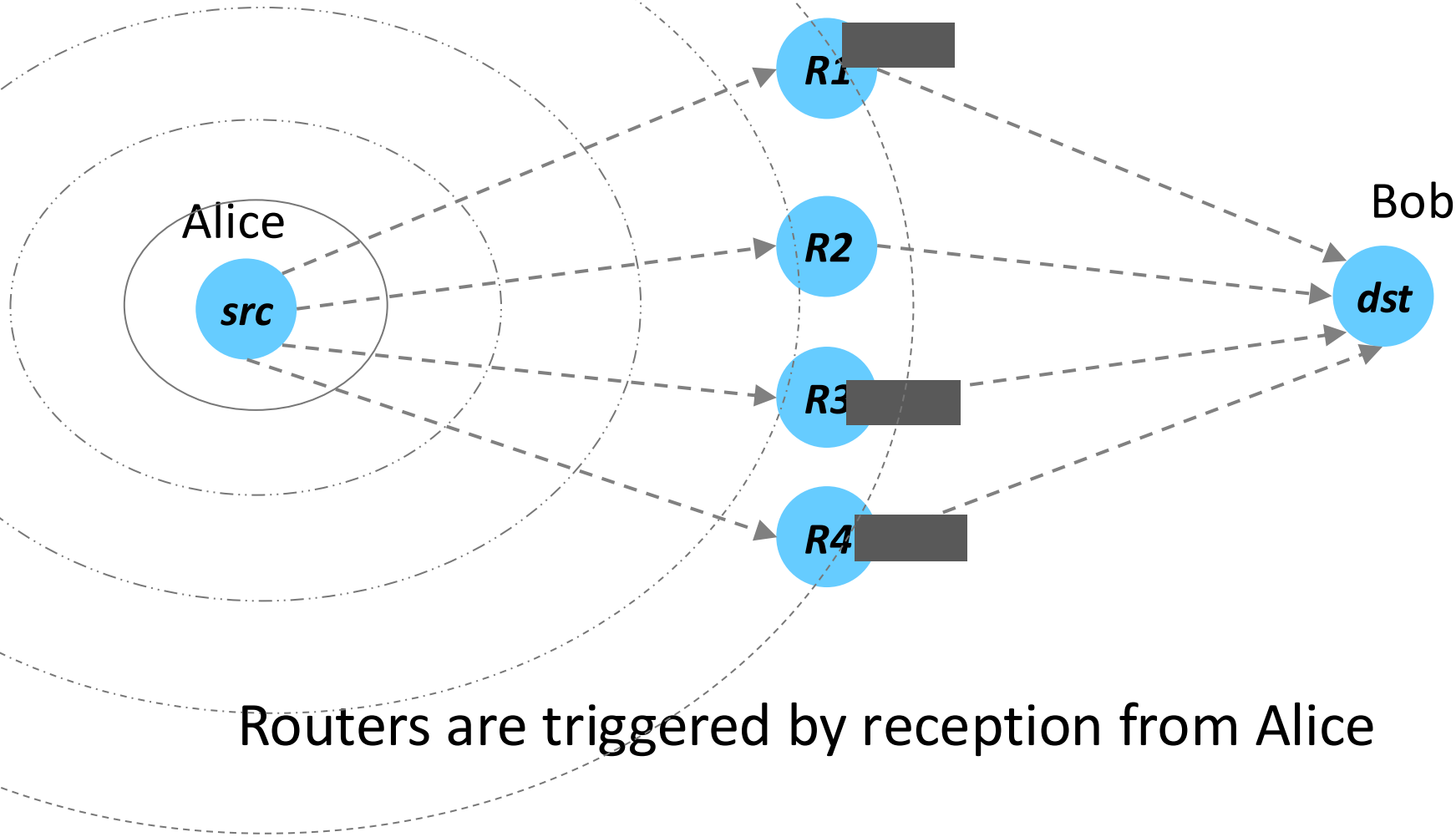
Results apply to both Opportunistic Routing and WLANs

How Do We Synchronize Distributed Transmitters?

- Synchronize transmitters by reception

How Do We Synchronize Distributed Transmitters?

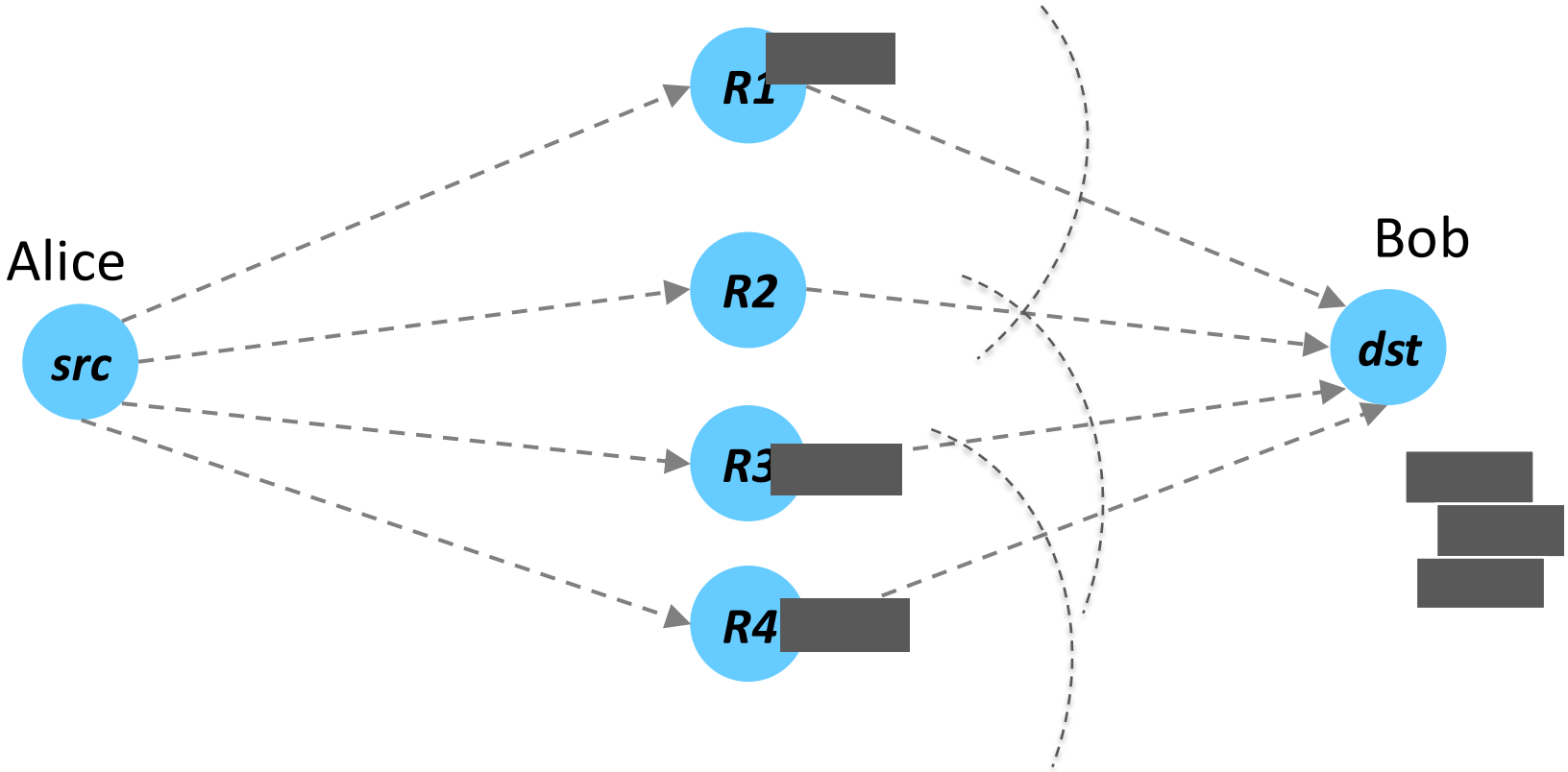
- Synchronize transmitters by reception



Routers are triggered by reception from Alice

How Do We Synchronize Distributed Transmitters?

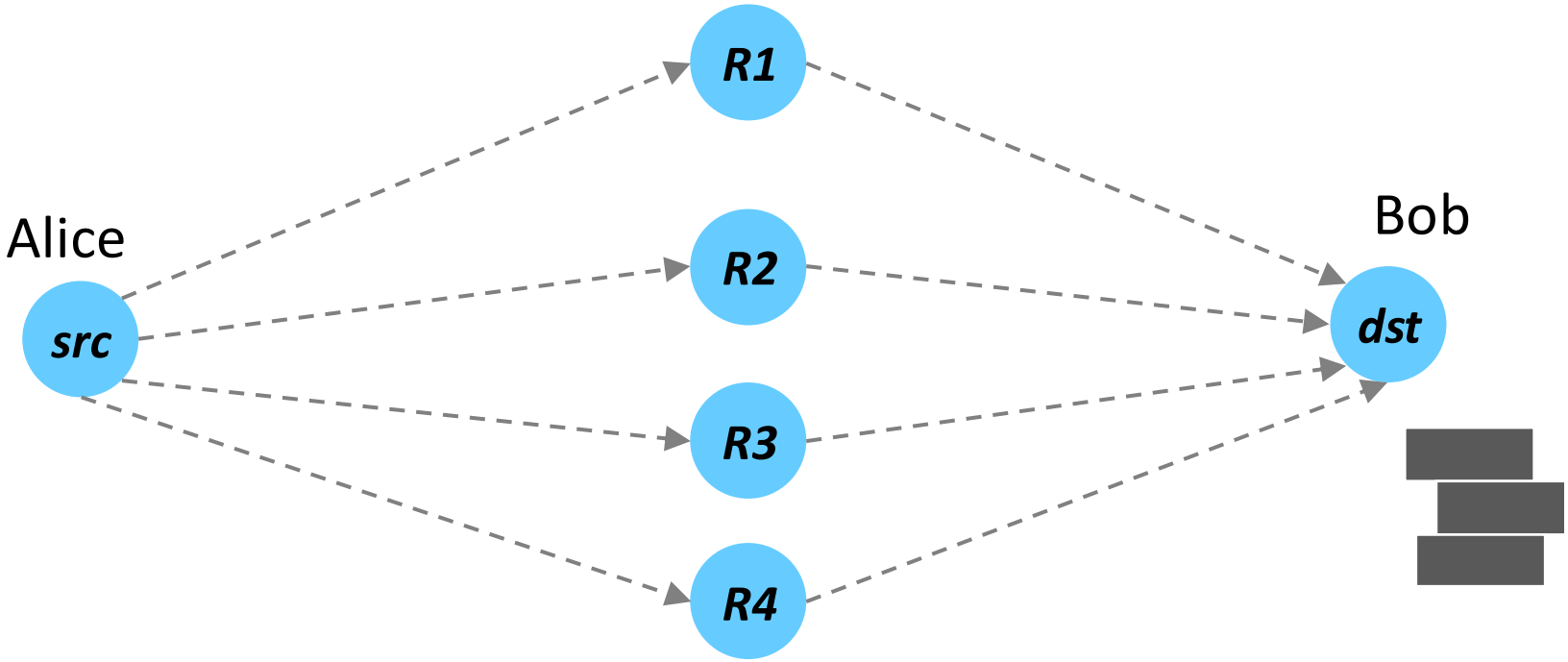
- Synchronize transmitters by reception



Routers transmit jointly to Bob

How Do We Synchronize Distributed Transmitters?

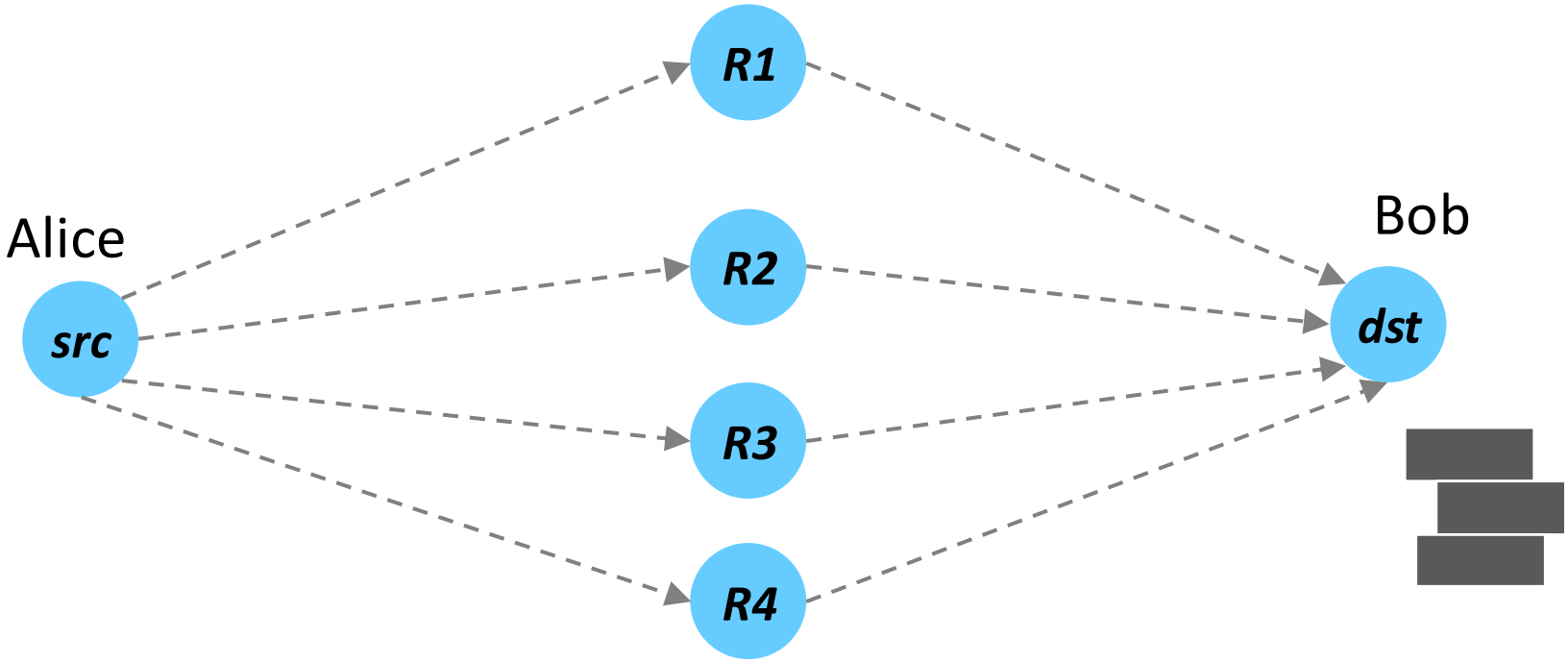
- Synchronize transmitters by reception



Paths have different delays → Signals arrive out of sync

How Do We Synchronize Distributed Transmitters?

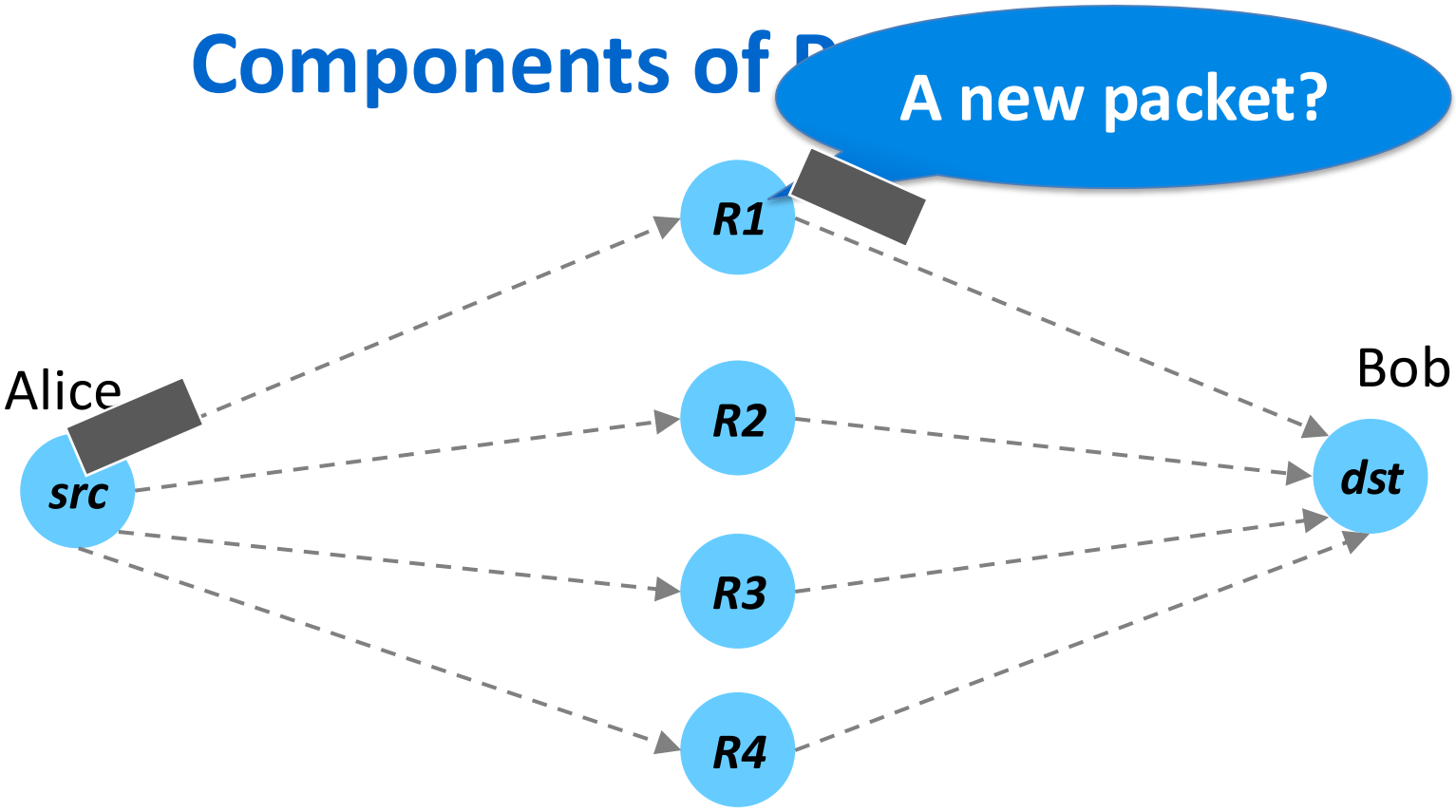
- Synchronize transmitters by reception



- Routers measure path delays and compensate for delay differences

How do we measure path delays?

Components of D



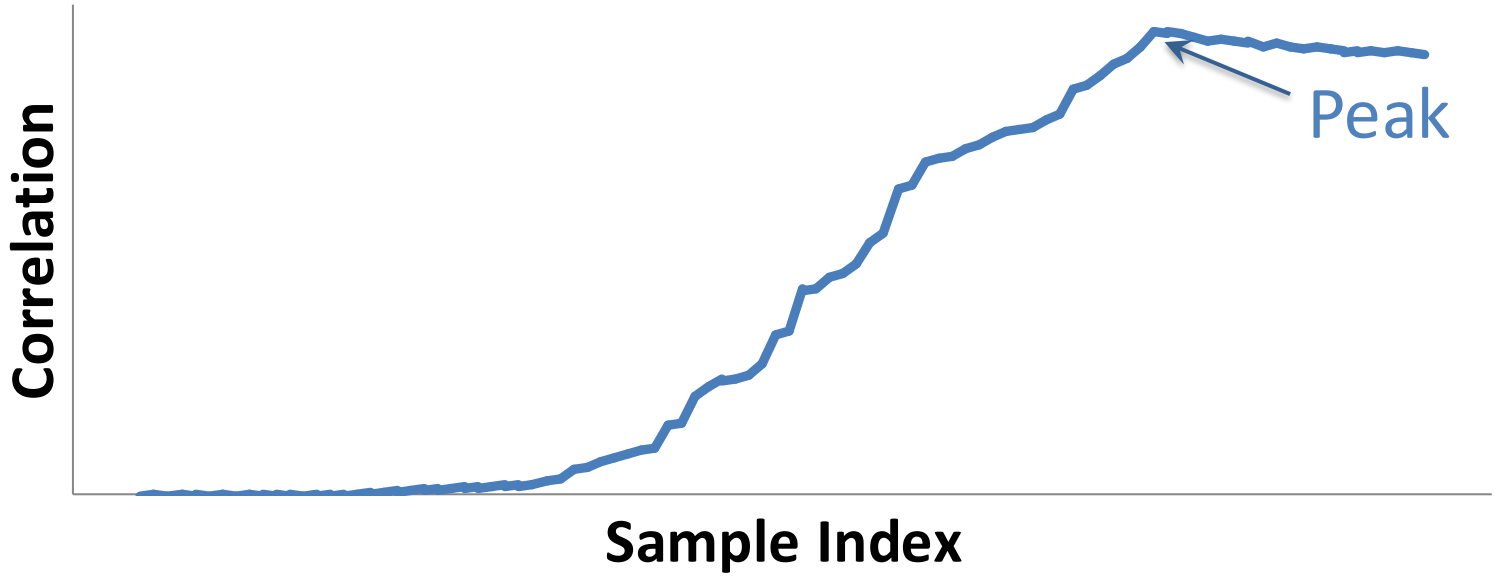
- Propagation Delay
- Packet Detection Delay
- Hardware Turnaround Time from Rx → Tx

Estimating Path Delays

- Propagation Delay
- Packet Detection Delay
- Hardware Turnaround Time from Rx → Tx

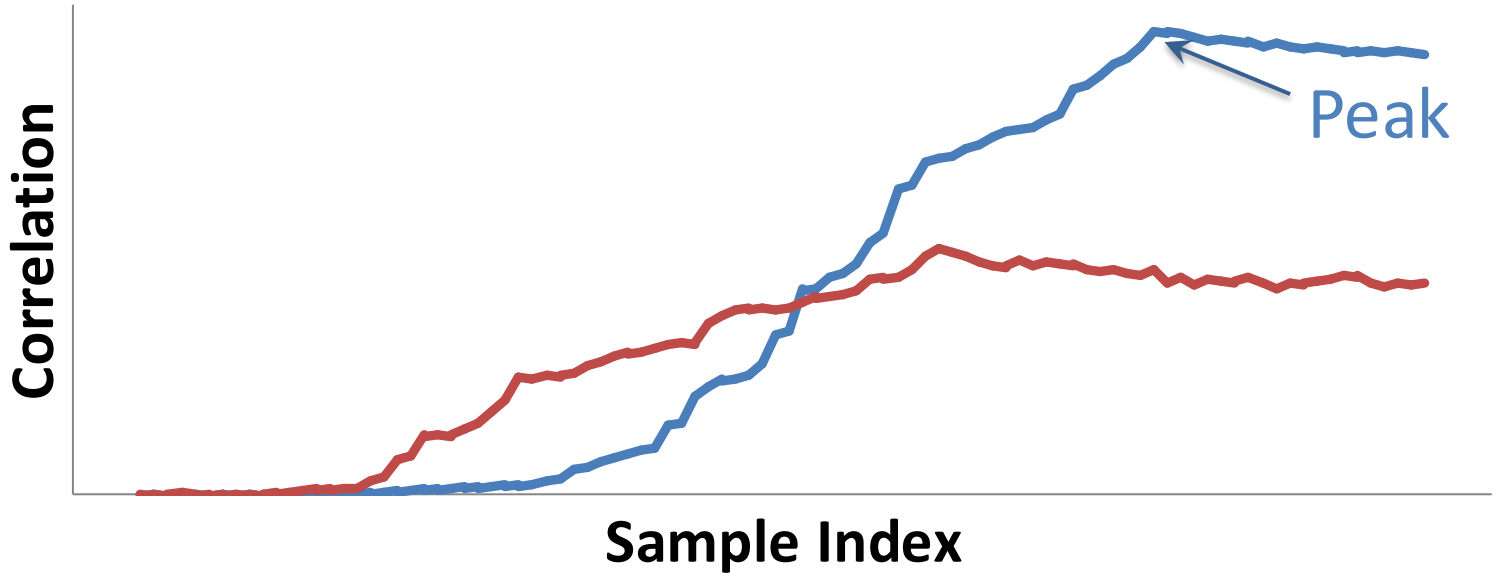
Packet Detection Delay

- Receivers detect packet using correlation
- Random noise → Do not detect packet on first sample



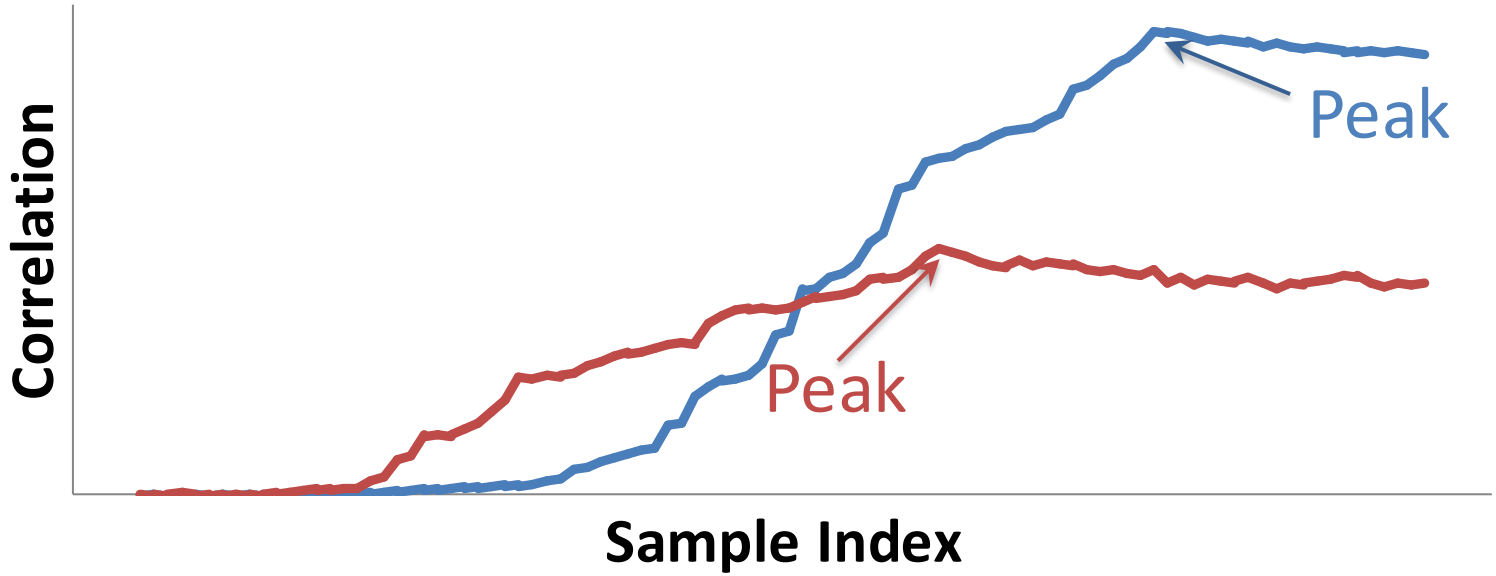
Packet Detection Delay

- Receivers detect packet using correlation
- Random noise → Do not detect packet on first sample
- Different receivers → different noise



Packet Detection Delay

- Receivers detect packet using correlation
- Random noise → Do not detect packet on first sample
- Different receivers → different noise
  → different detection delay



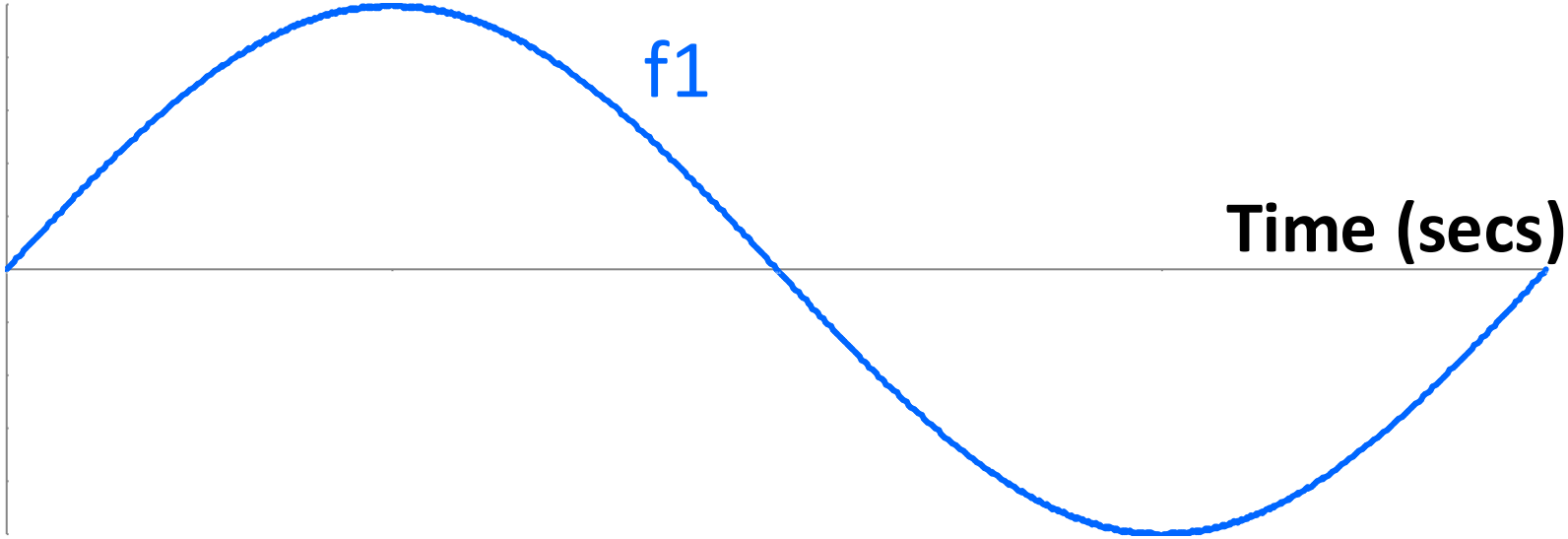
Packet Detection Delay

Routers estimate packet detection delay

Compensate for detection delay by syncing to first sample

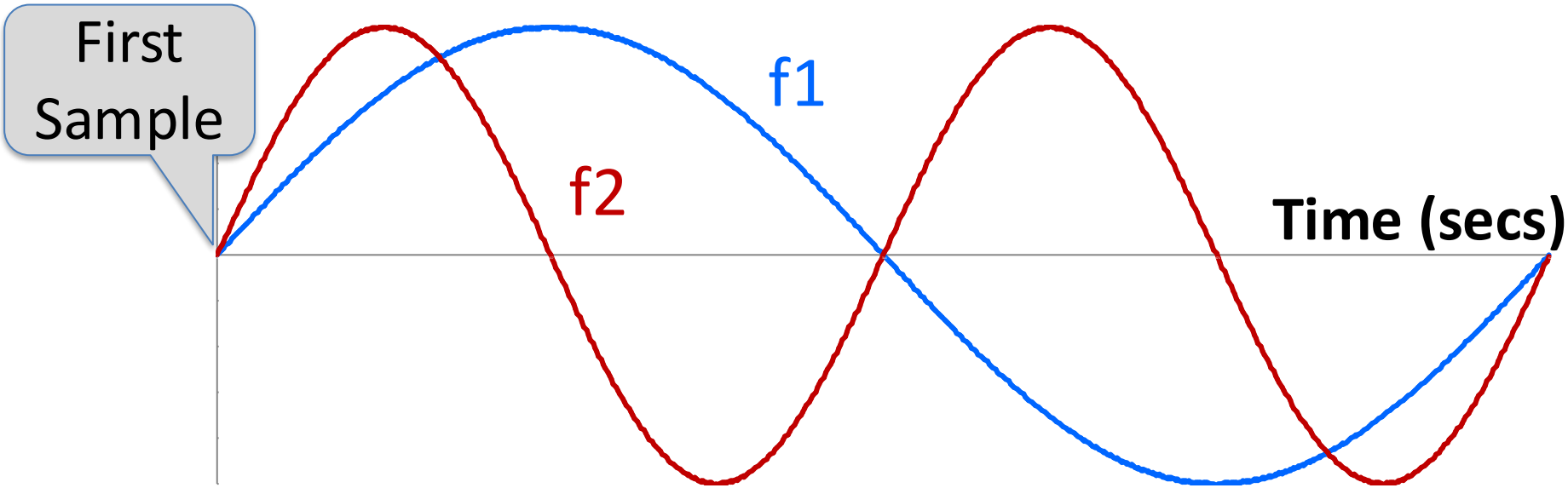
Estimating Packet Detection Delay

OFDM transmits signal over multiple frequencies



Estimating Packet Detection Delay

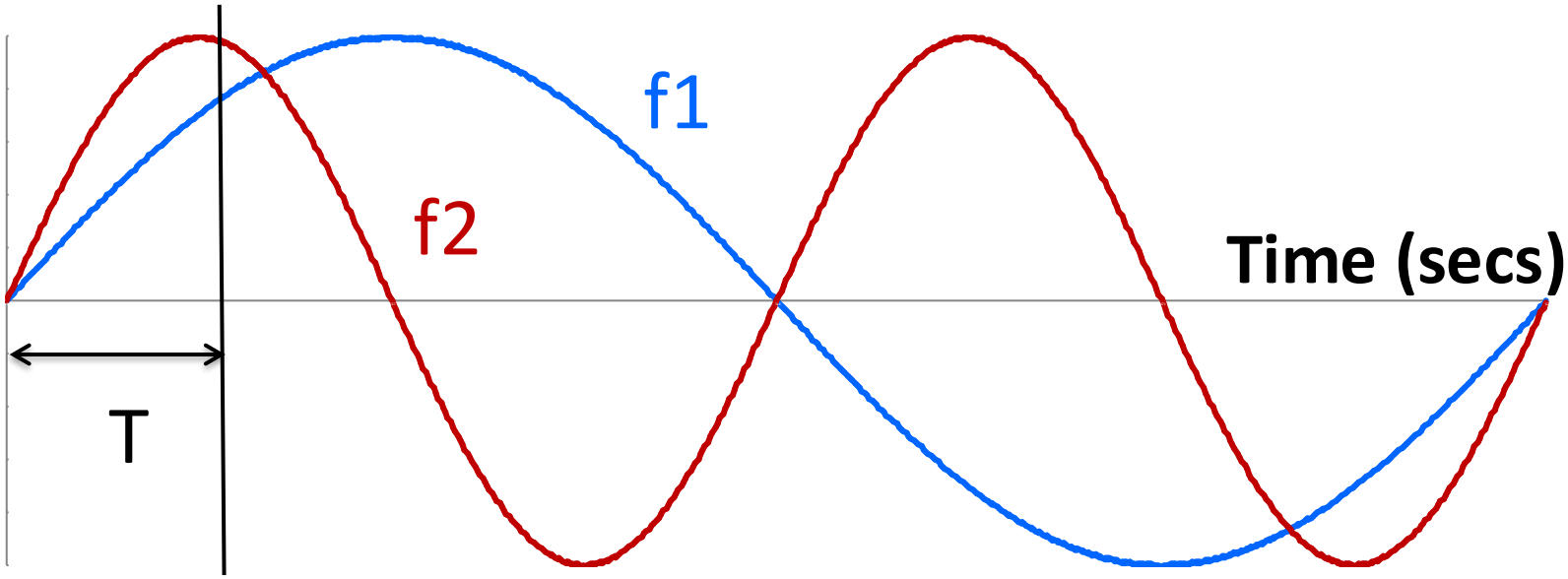
OFDM transmits signal over multiple frequencies



Detect on first sample → Same phase

Estimating Packet Detection Delay

OFDM transmits signal over multiple frequencies

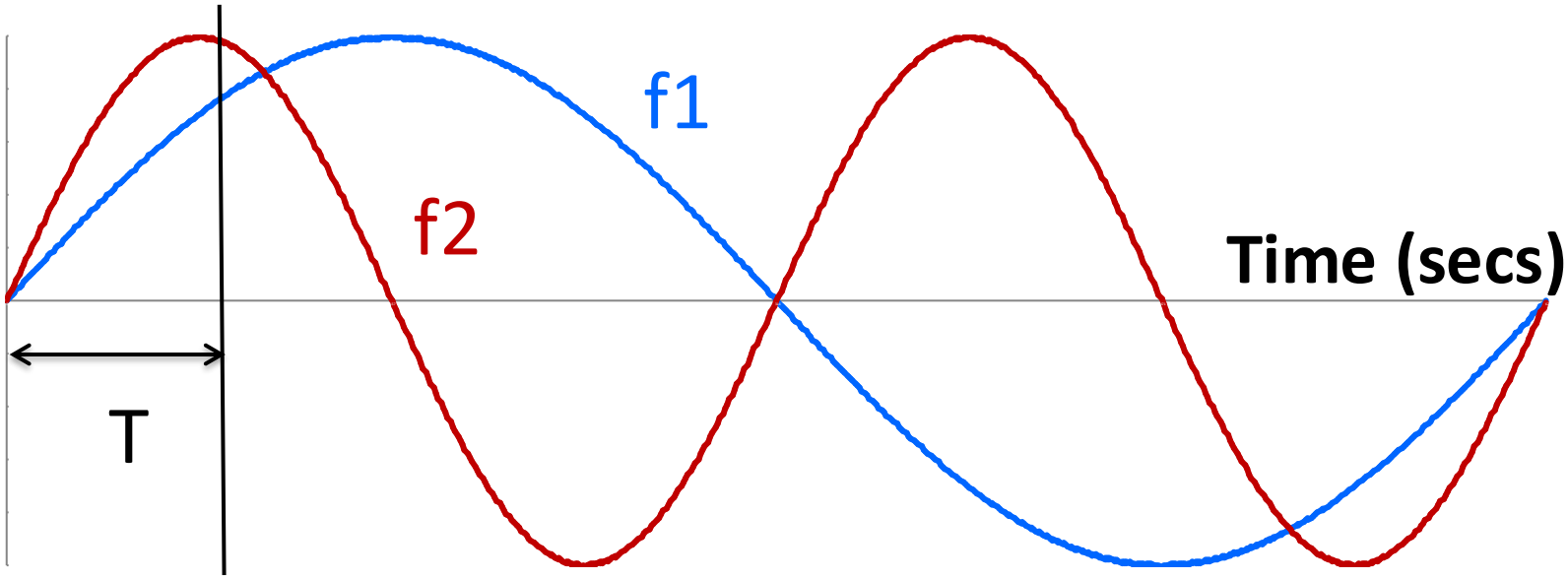


Detect after T

Frequencies rotate at different speeds

Estimating Packet Detection Delay

OFDM transmits signal over multiple frequencies

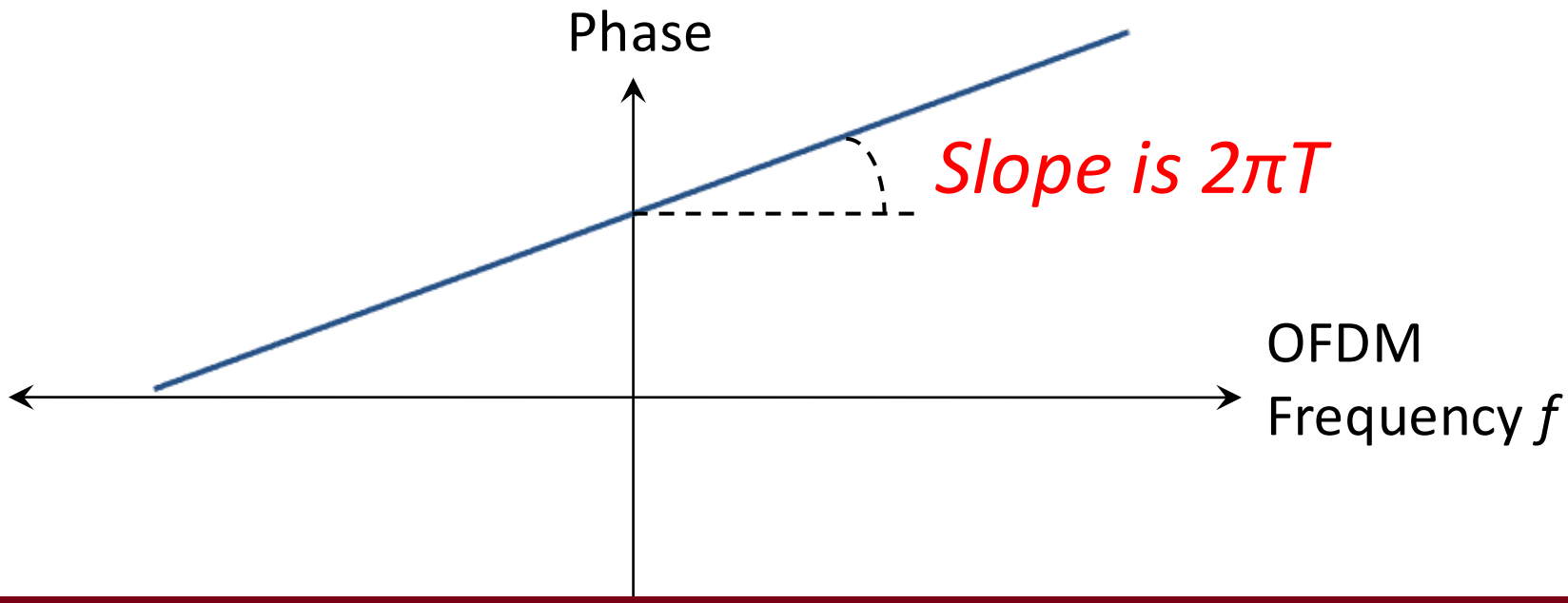


Detect after T

Different frequencies exhibit different phases

$$Phase = 2\pi fT$$

Estimating Packet Detection Delay



- Each router estimates packet detection delay
- Estimate uses every symbol in packet → Robust to noise

Estimating Path Delays

- Propagation Delay
- Packet Detection Delay
- Hardware Turnaround Time from Rx → Tx

Hardware Turnaround Time

- Turnaround time is hardware dependent
  - Different hardware pipelines
  - Different radio frontends

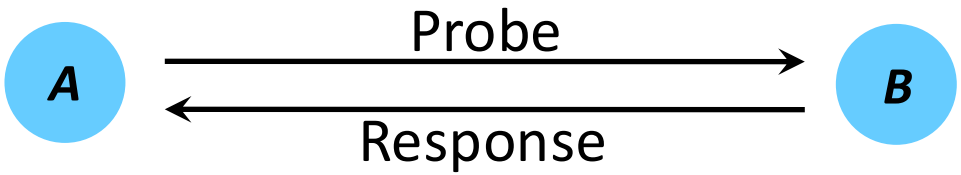
- Each router locally calibrates its turnaround by counting the clock ticks

Estimating Path Delays

- Propagation Delay
- Packet Detection Delay
- Hardware Turnaround Time from Rx → Tx

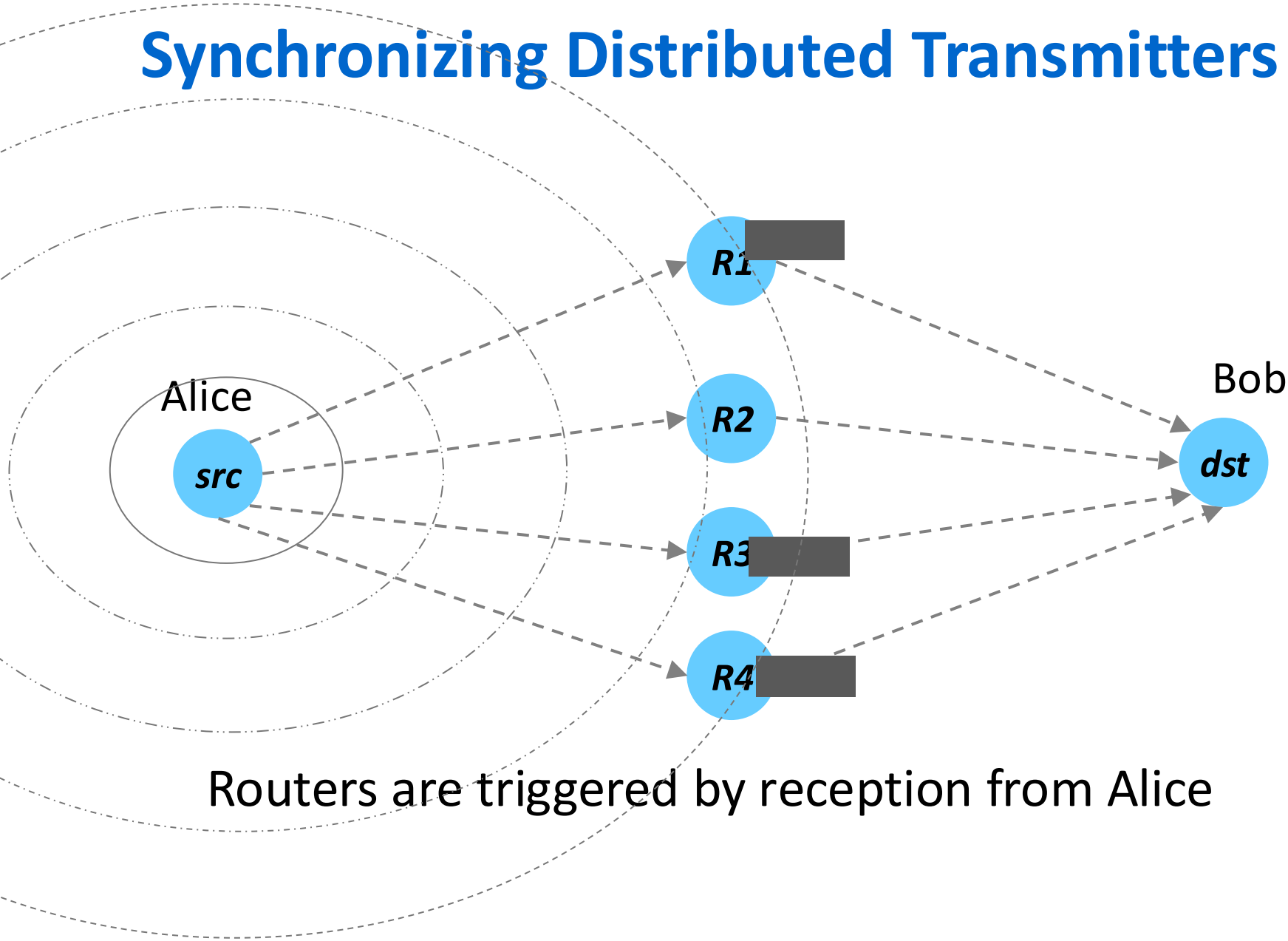
Measuring Propagation Delays

Use Probe-Response between node pairs



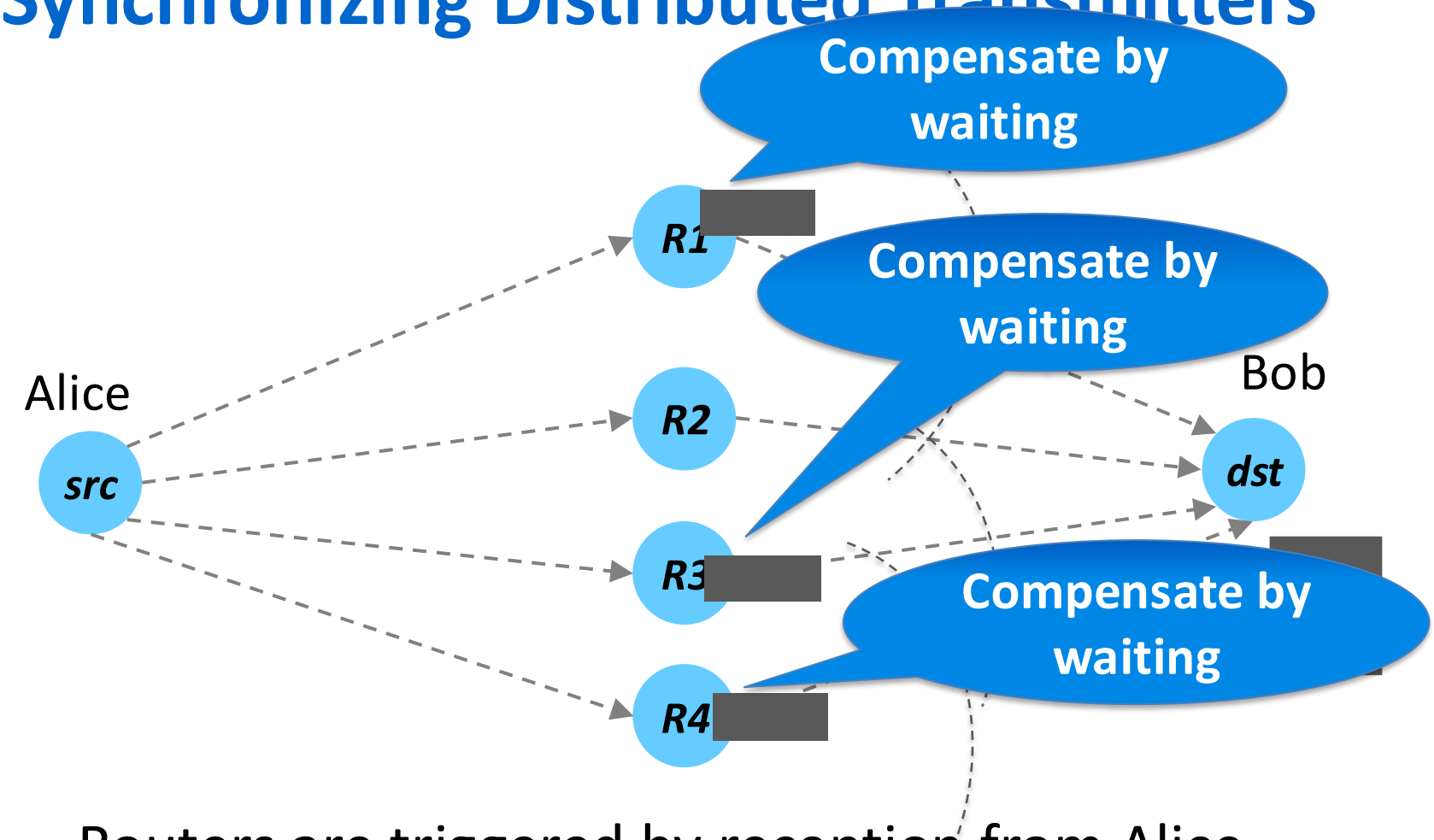
✓ RTT = 2 x Propagation Delay
+ Turnaround time at B ✓
+ Packet Detection Delay at B ✓
+ Packet Detection Delay at A ✓

Synchronizing Distributed Transmitters



Routers are triggered by reception from Alice

Synchronizing Distributed Transmitters

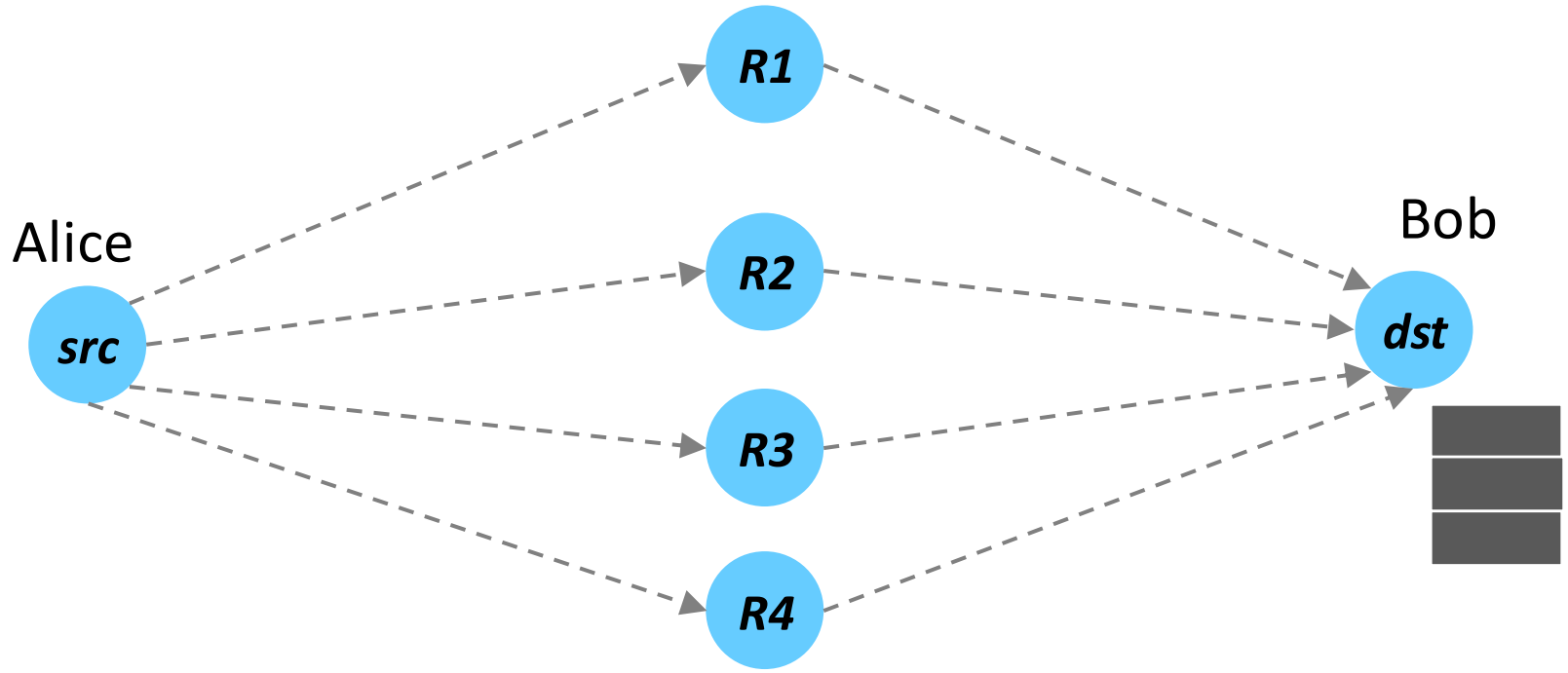


Routers are triggered by reception from Alice

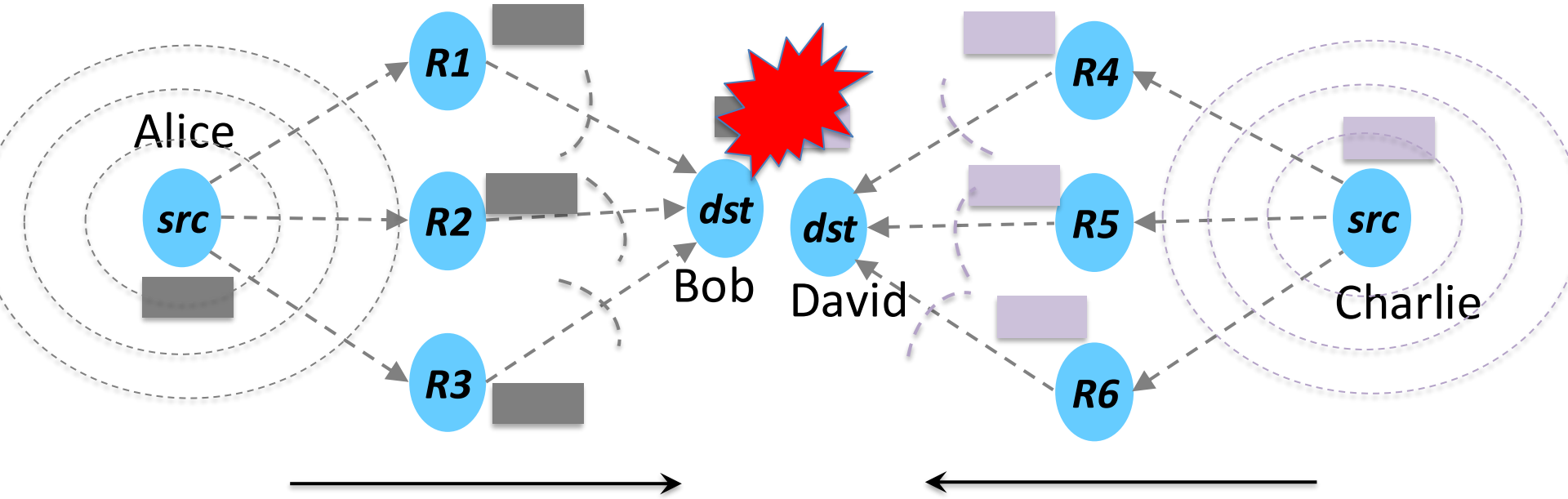
Routers insert wait times to compensate for delay differences

Routers transmit after waiting

Synchronizing Distributed Transmitters



What about the MAC?



Can We Synchronize While Using Carrier Sense?

SourceSync MAC

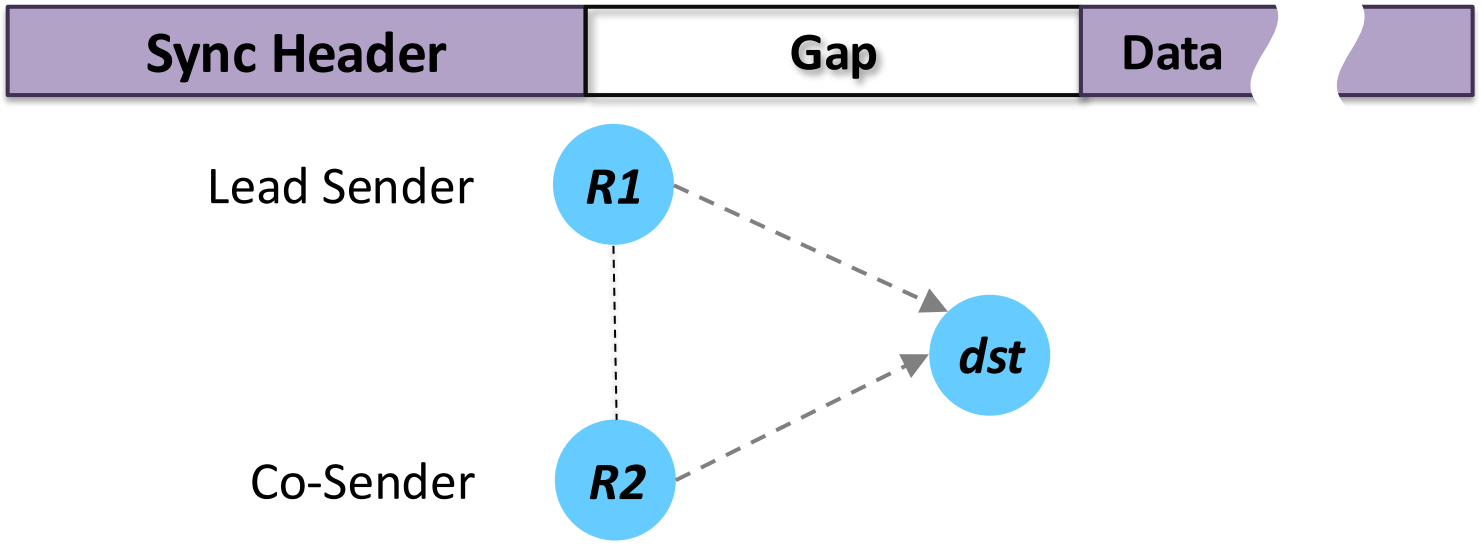
Instead of triggering by reception from the previous hop, we trigger by transmission from one of the joint senders.

All nodes in the network use CSMA.

One of the nodes wins the contention and begins transmitting, just like in CSMA.

Other nodes hear the transmission; join the transmission if they have the packet after inserting wait time.

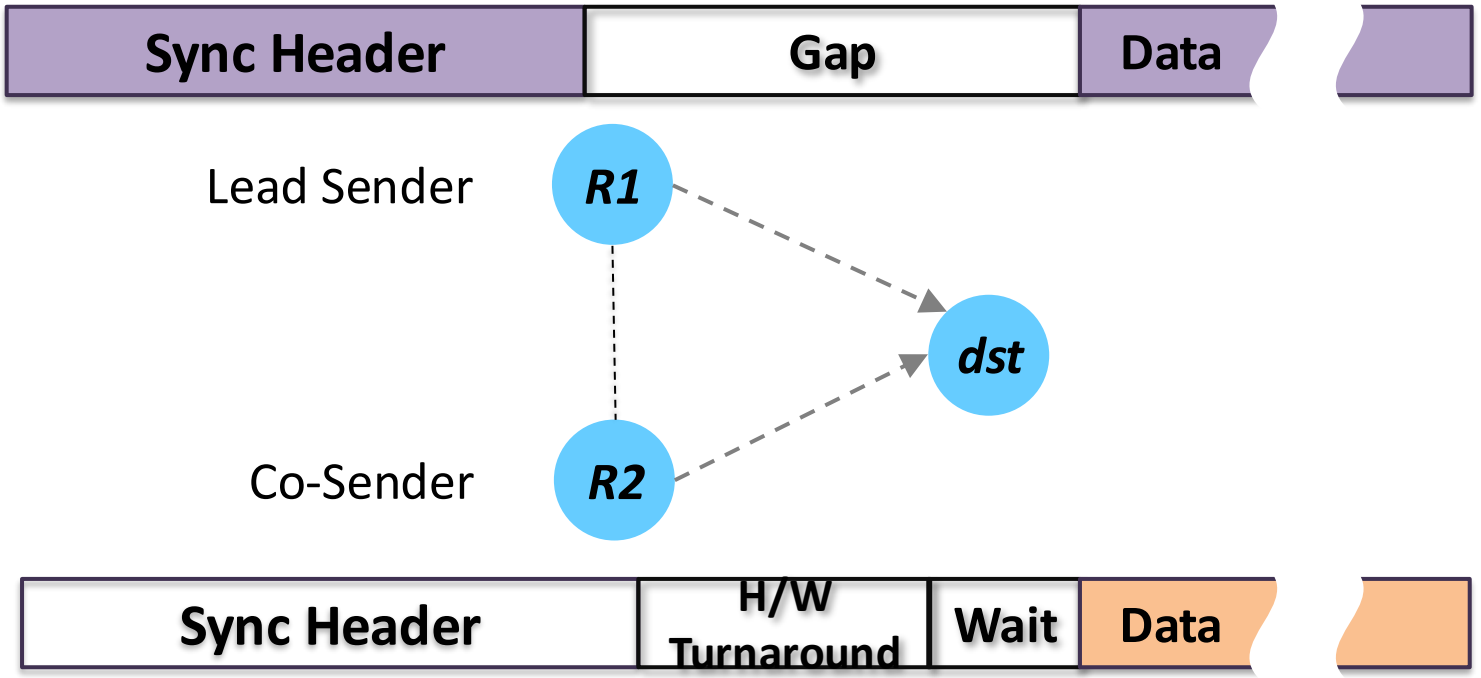
Synchronizing Multiple Transmitters



Lead sender:

- Transmits Synchronization Header
- Waits for known fixed gap
- Transmits data

Synchronizing Multiple Transmitters



Co-Sender:

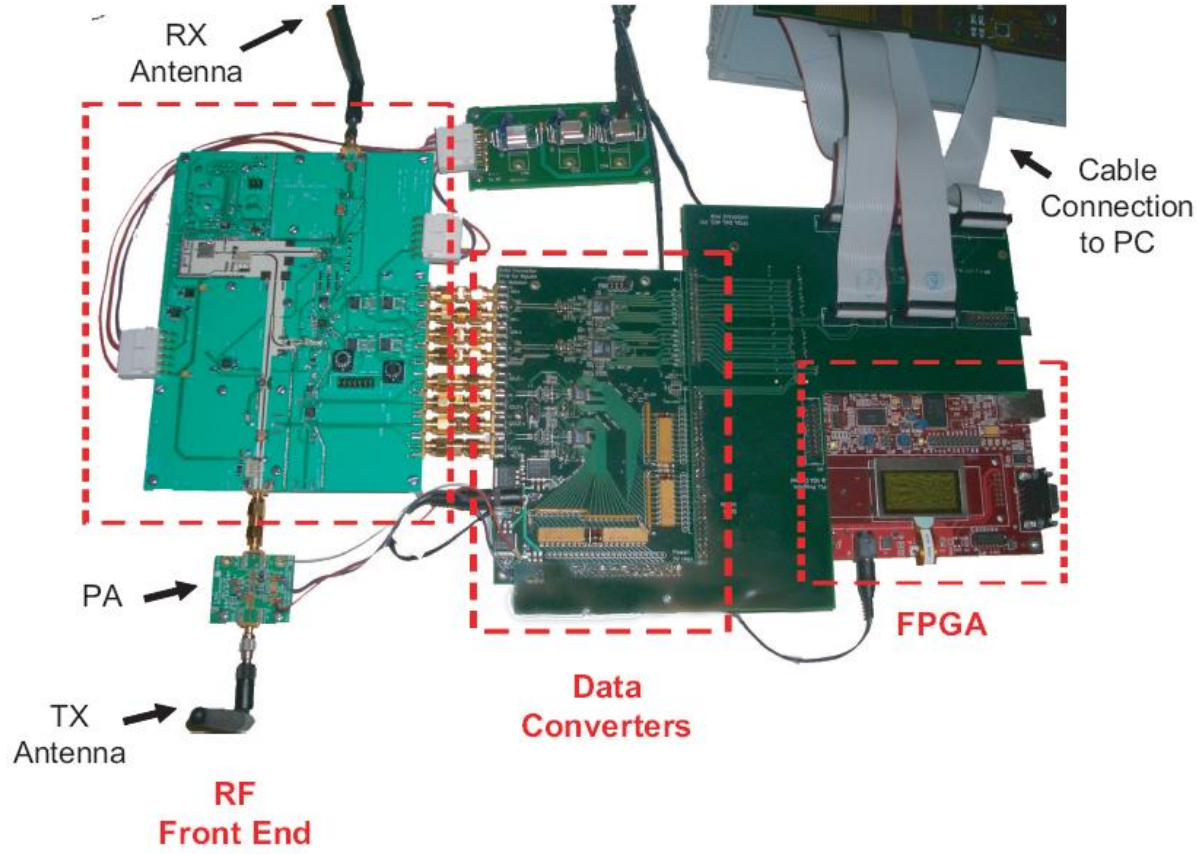
- Listens to Synchronization Header
- Turns around from receive to transmit
- Waits to compensate for differences in path delays
- Transmits data

Performance

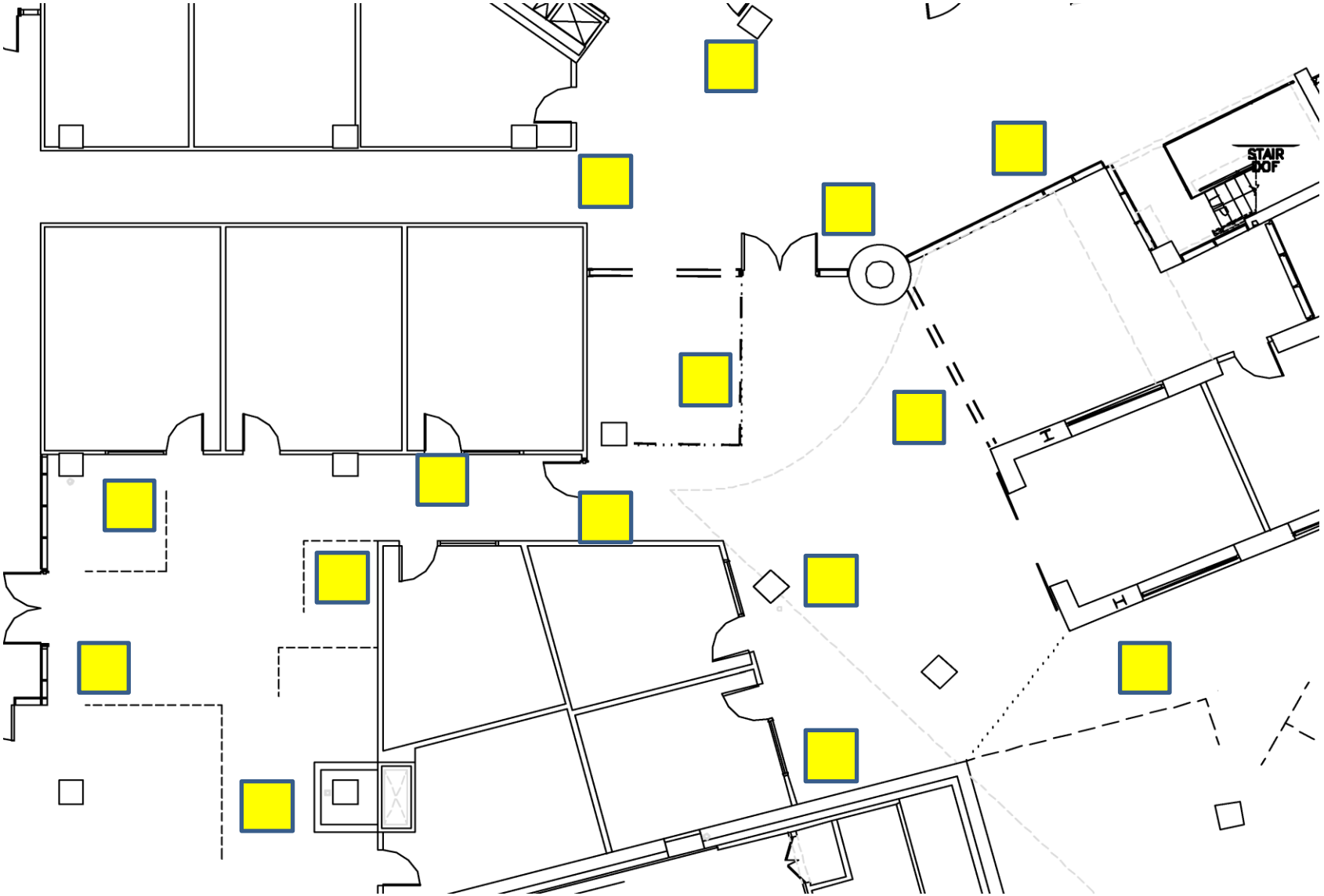
Implementation

Implemented in FPGA of WiGLAN radio

- Carrier Frequency: 5.247 GHz
- Bandwidth: 20 MHz



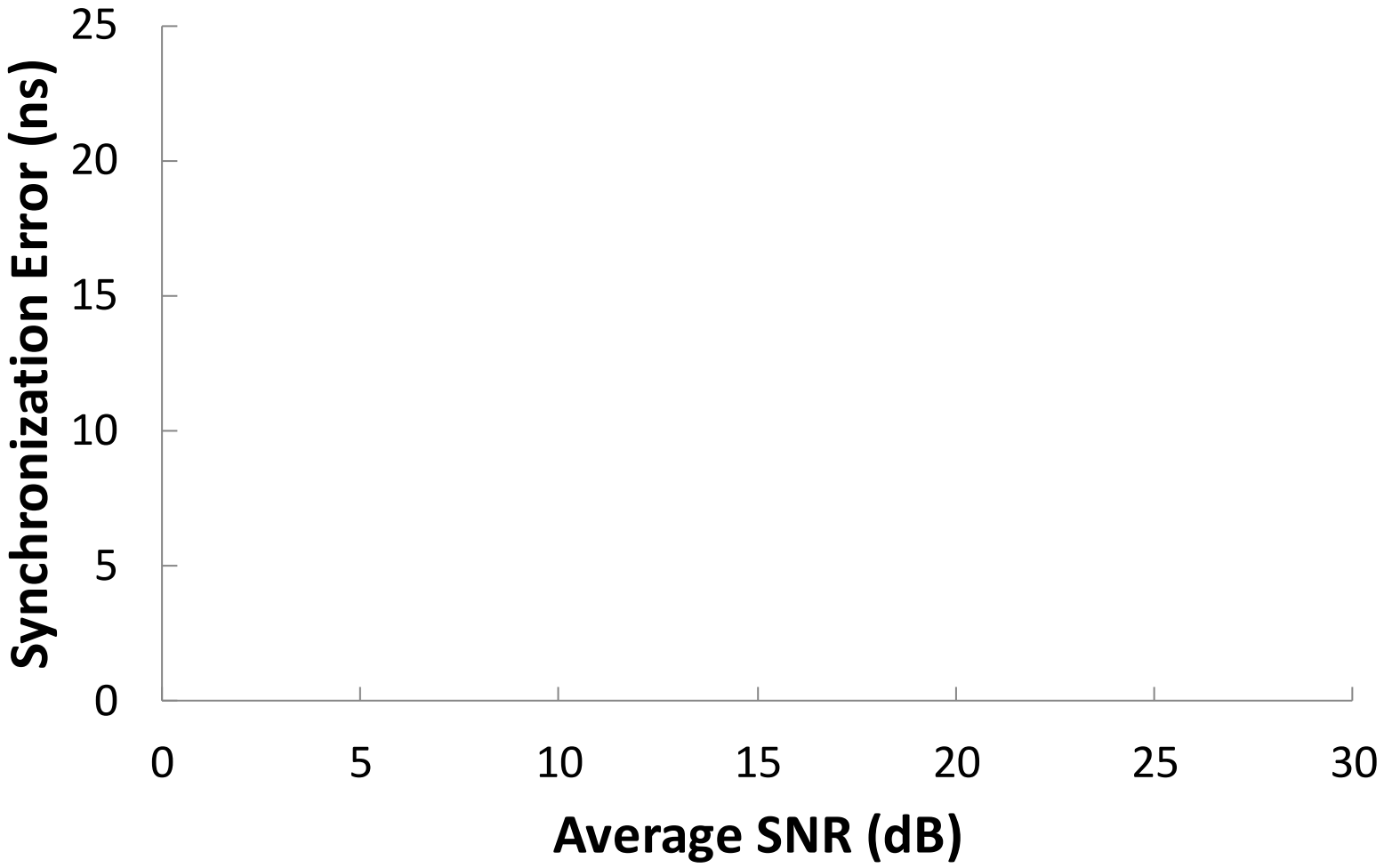
Node Locations in our Testbed



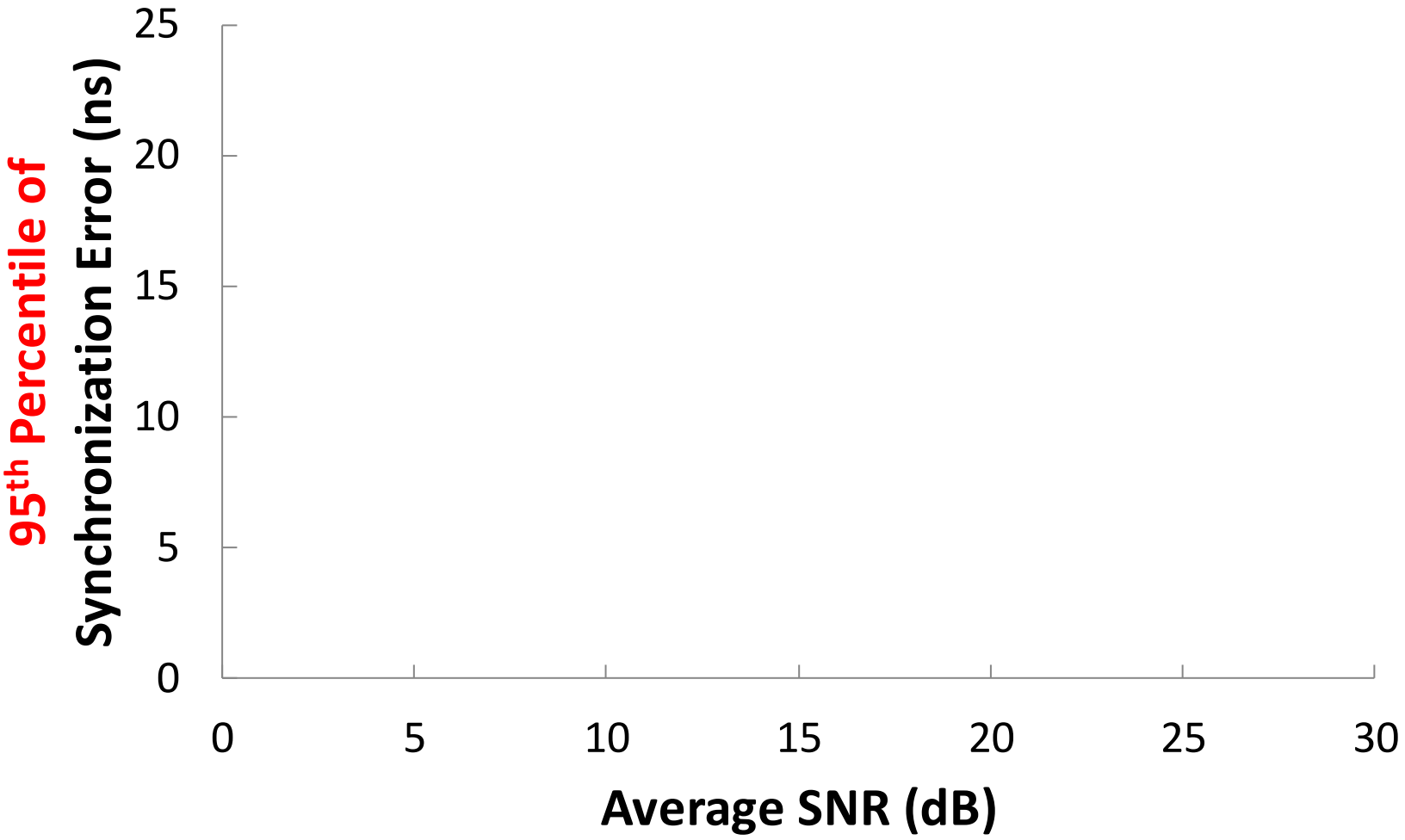
Can SourceSync Achieve Tight Synchronization?

- Randomly pick a pair of nodes to transmit simultaneously to a receiver
- Measure synchronization error
- Repeat for different nodes

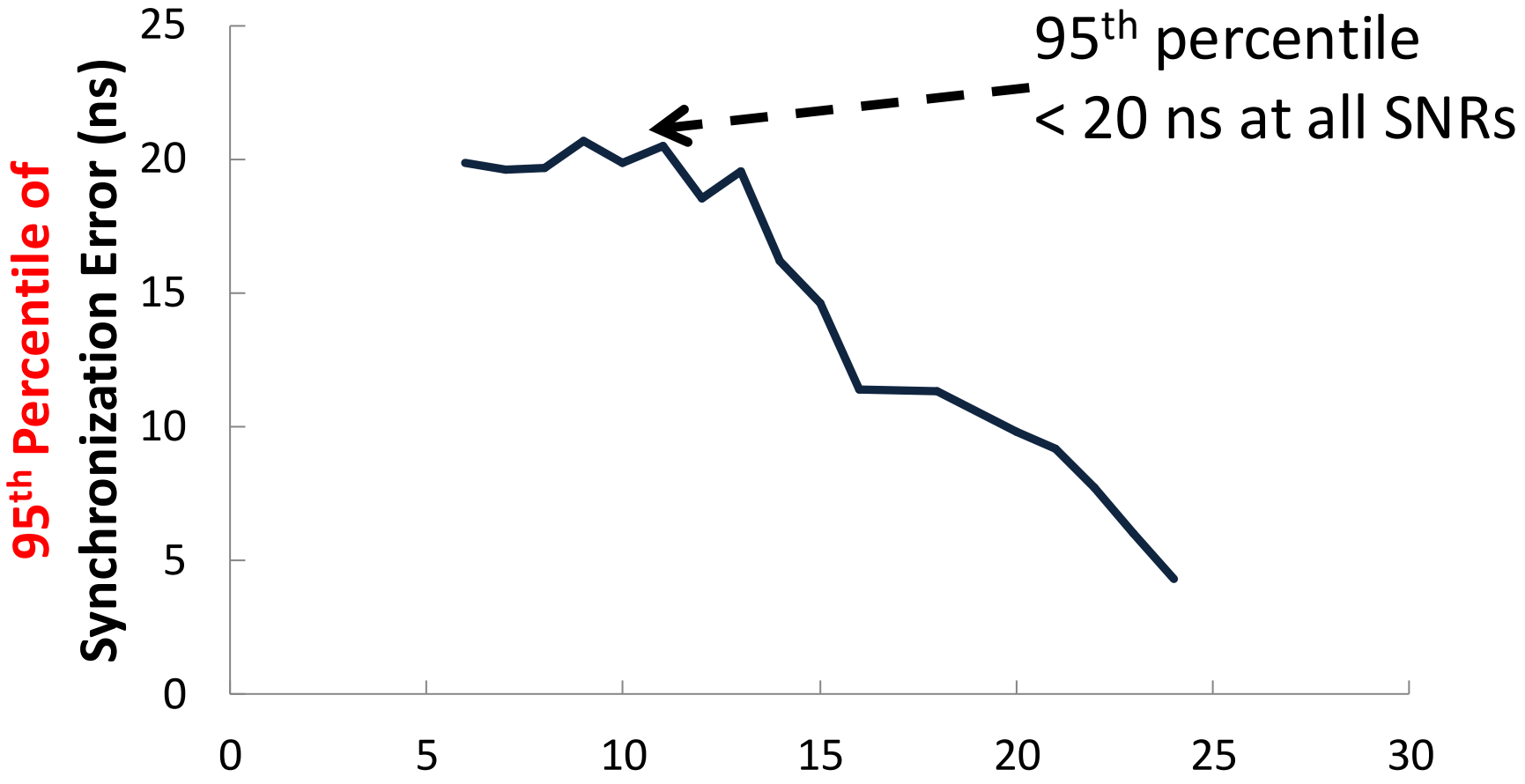
Can SourceSync Achieve Tight Synchronization?



Can SourceSync Achieve Tight Synchronization?



Can SourceSync Achieve Tight Synchronization?

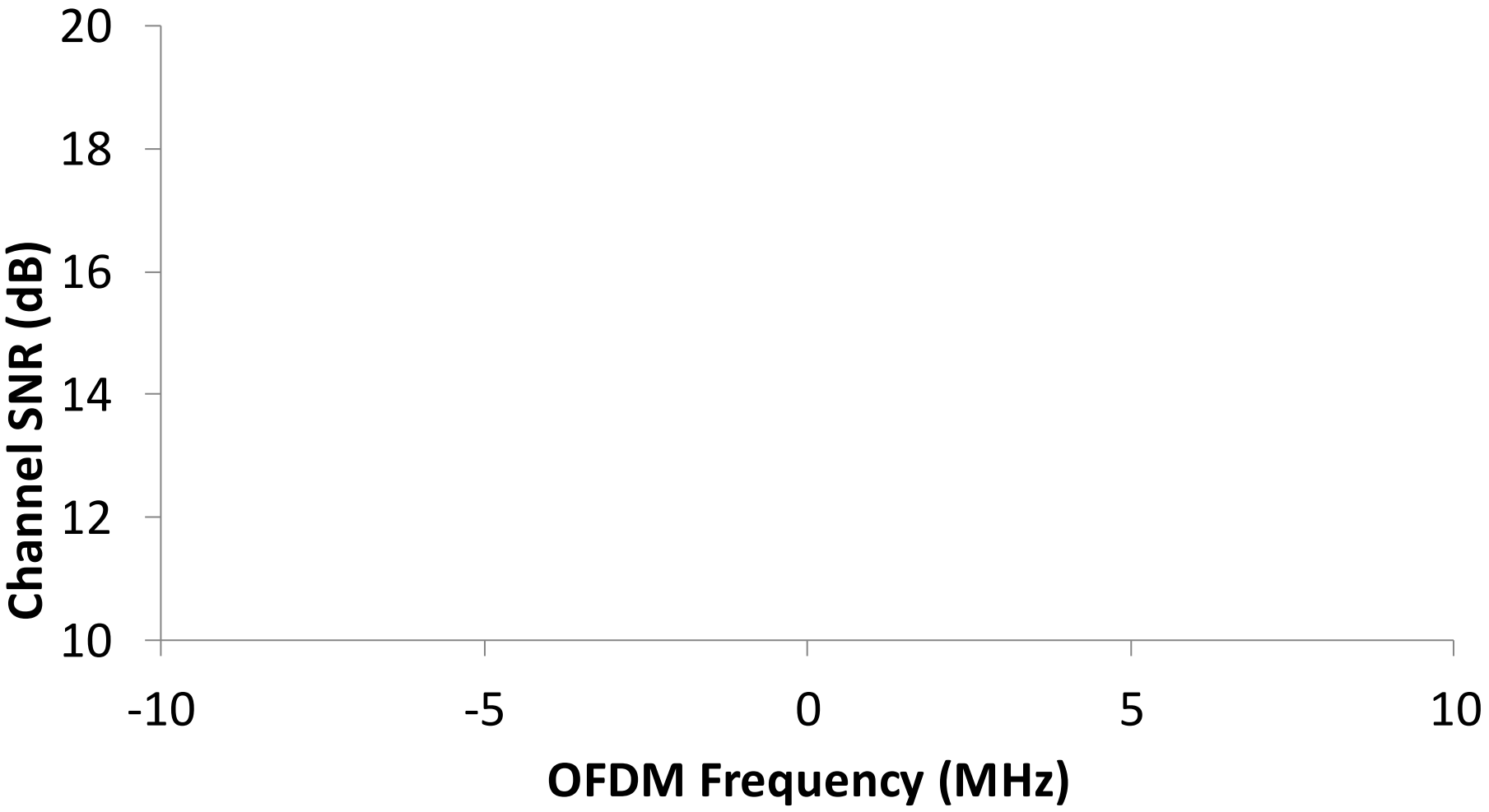


SourceSync achieves distributed synchronization within 20 ns

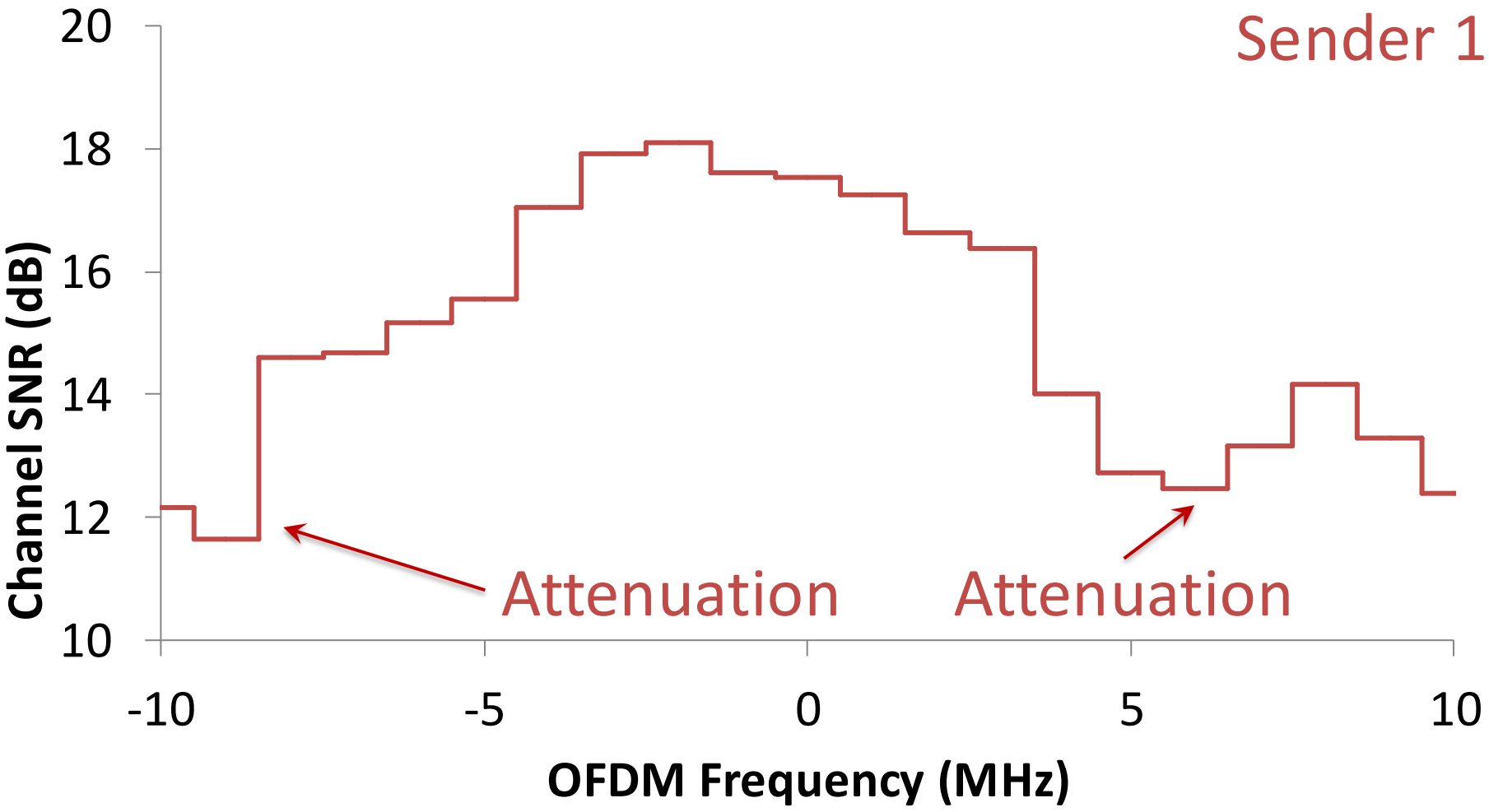
Can SourceSync Achieve Sender Diversity Gains?

- Again, transmit simultaneously to a receiver
- Check:
  - Two senders exhibit different channel SNRs
  - SourceSync is better than either sender

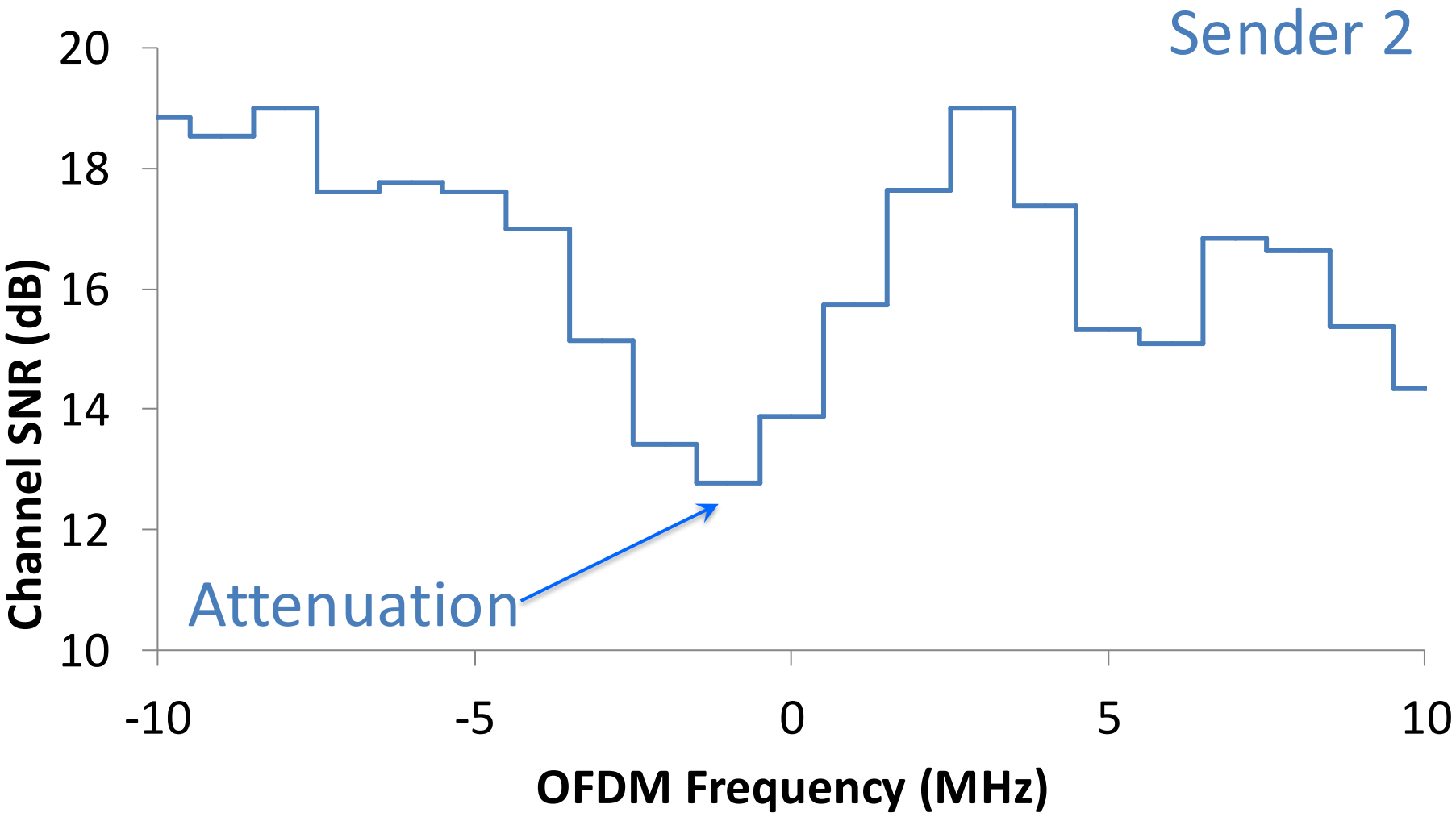
Can SourceSync Achieve Diversity Gains?



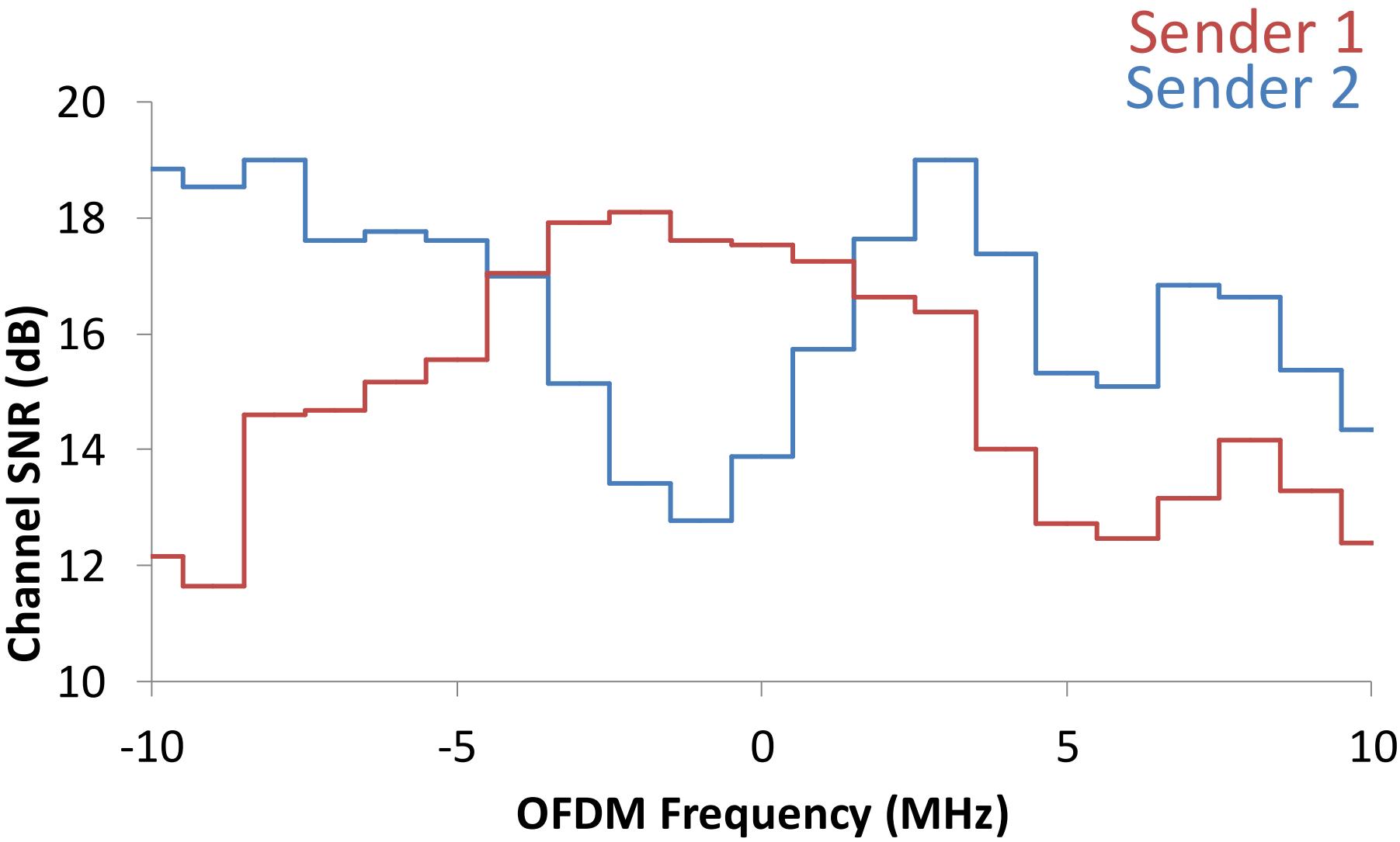
Can SourceSync Achieve Diversity Gains?



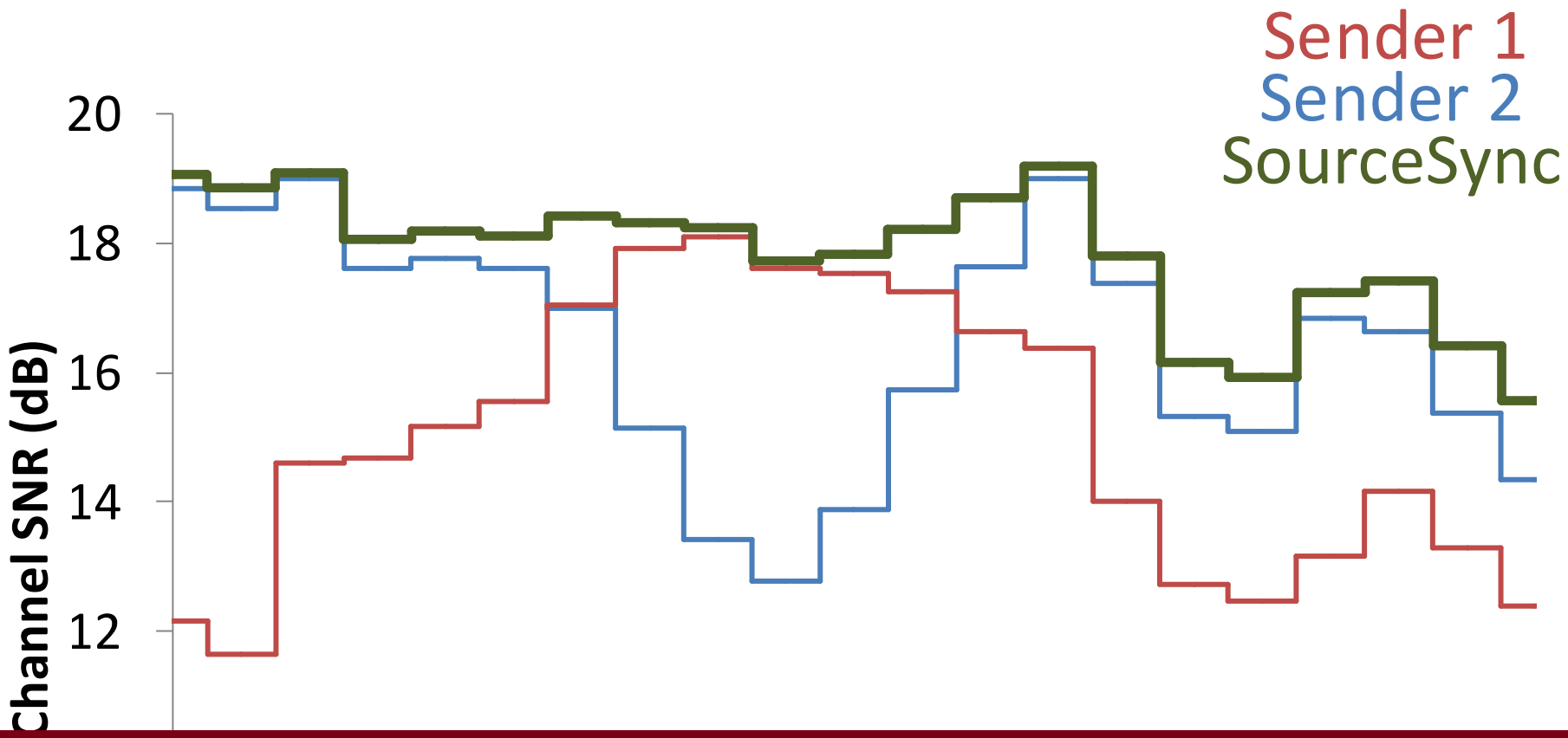
Can SourceSync Achieve Diversity Gains?



Can SourceSync Achieve Diversity Gains?



Can SourceSync Achieve Diversity Gains?

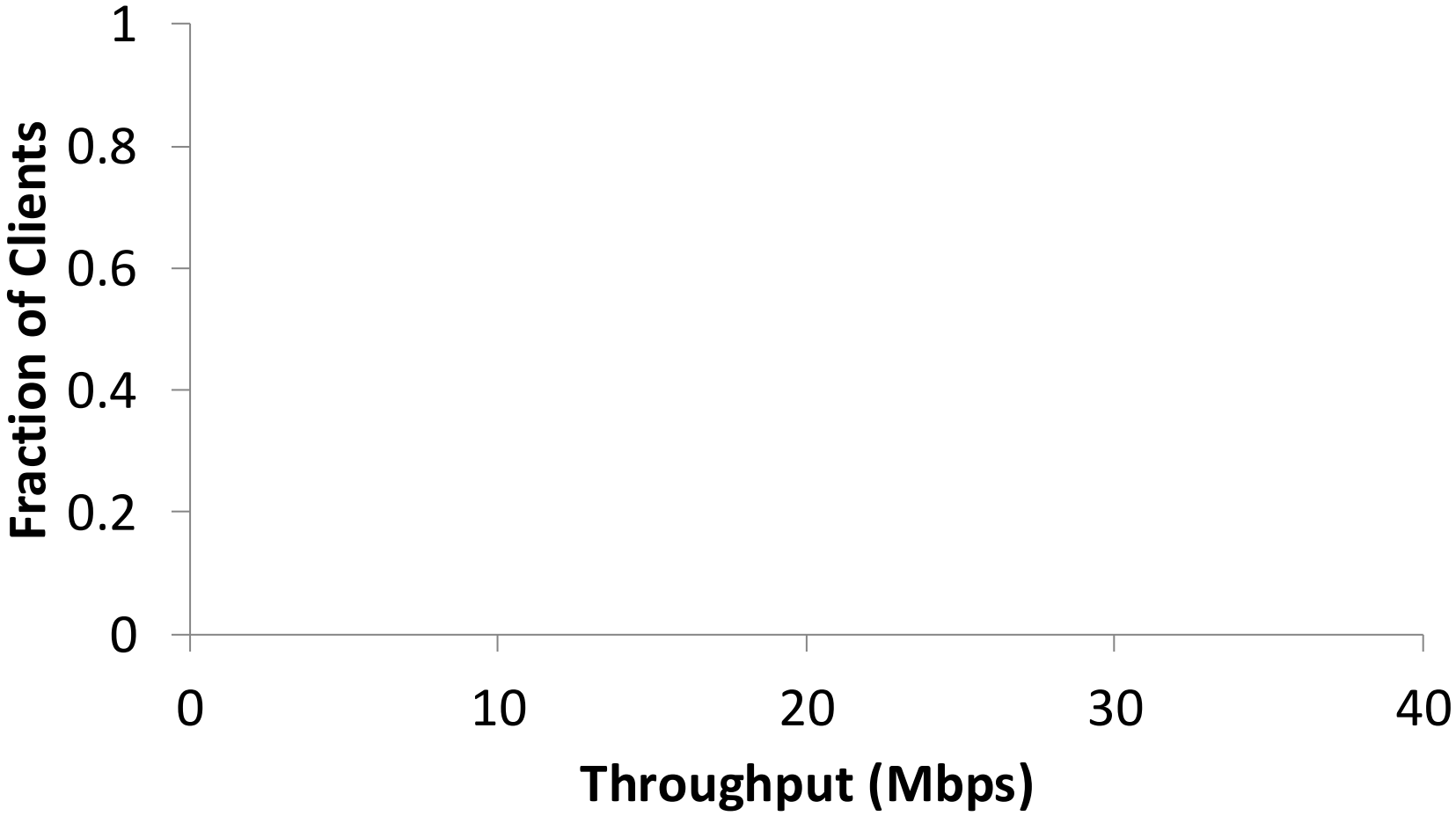


SourceSync Harnesses Sender Diversity

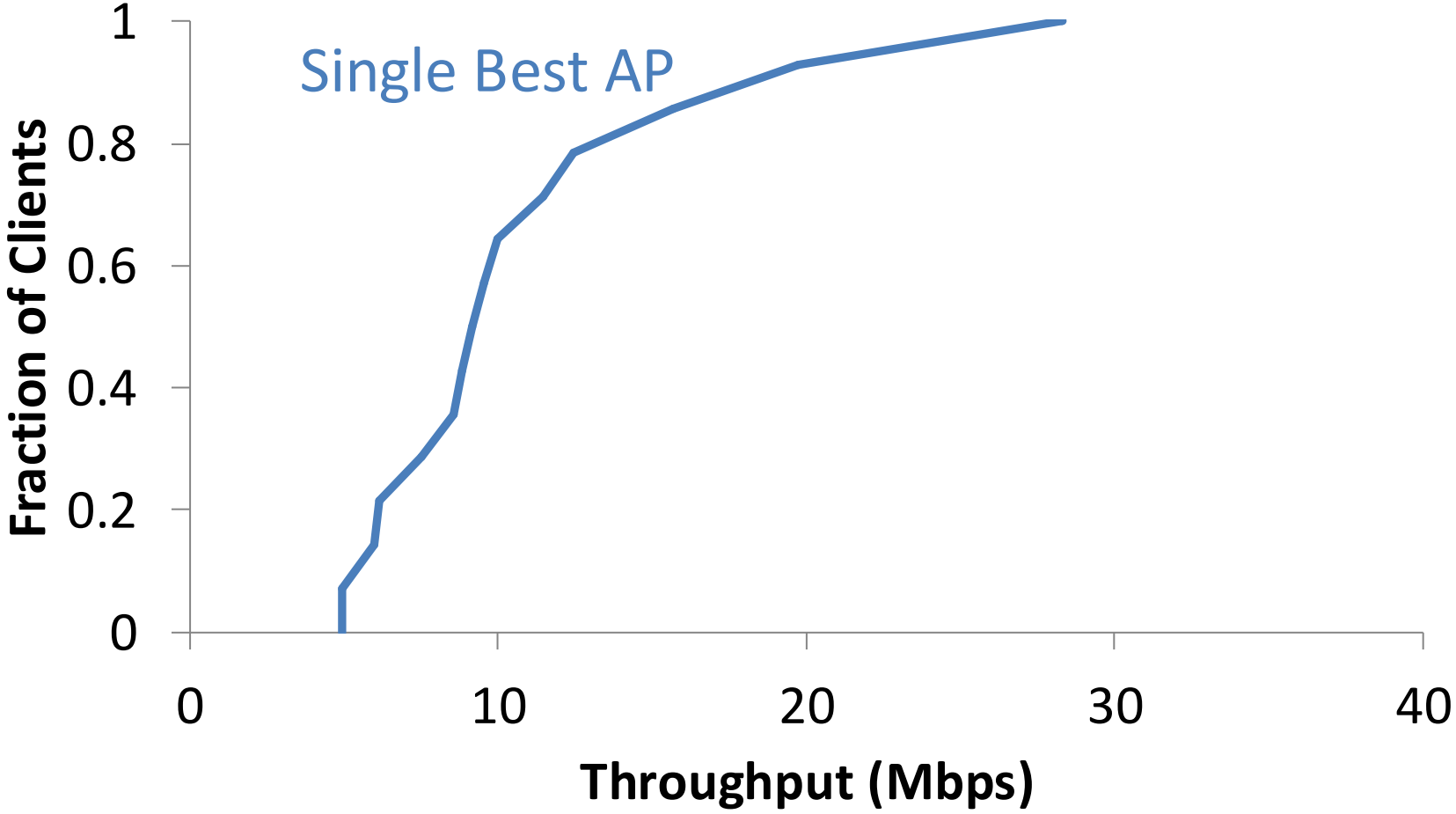
Can SourceSync Improve WLAN Downlink Performance?

- Two APs and one client
- We compare
  - Best Single AP
  - SourceSync
- Repeat for different nodes

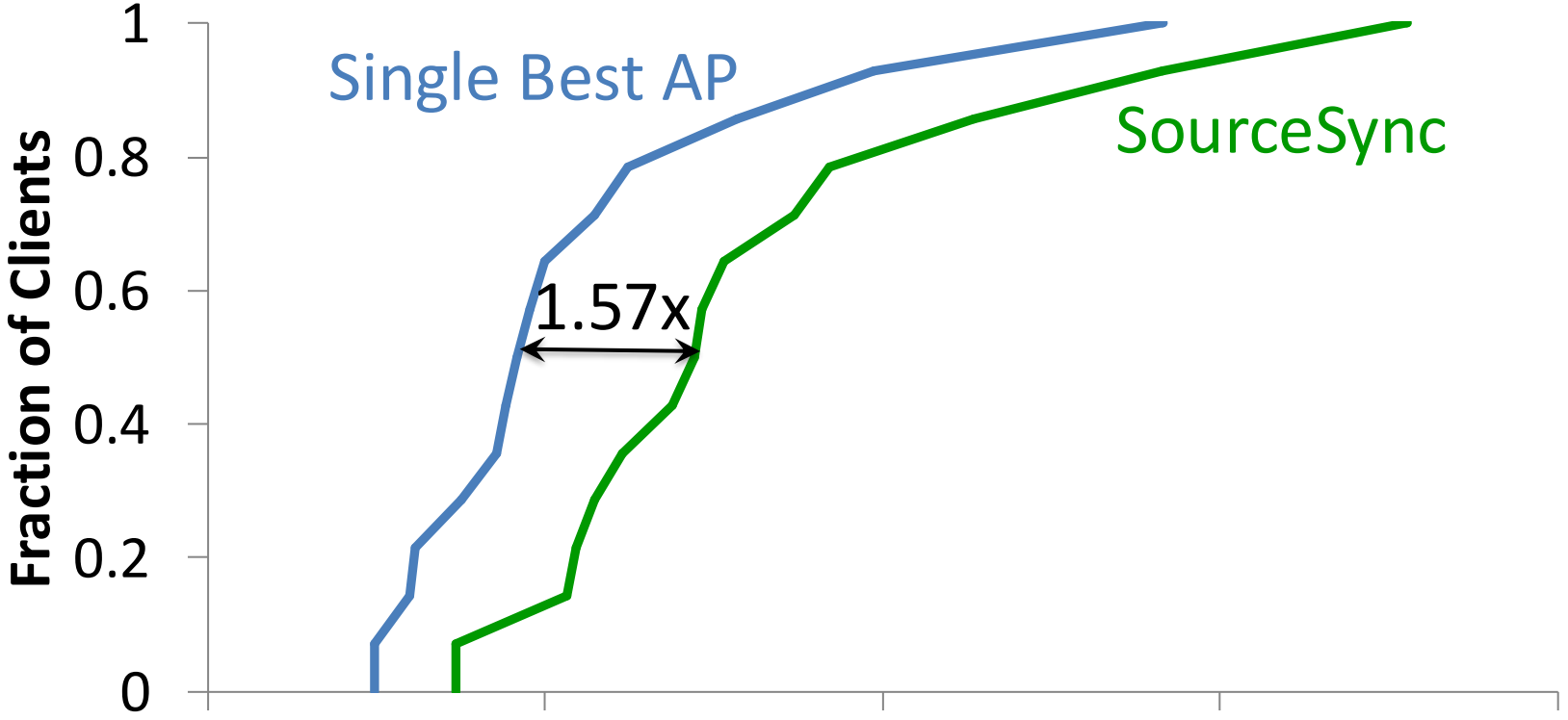
Can SourceSync Improve WLAN Downlink Performance?



Can SourceSync Improve WLAN Downlink Performance?



Can SourceSync Improve WLAN Downlink Performance?

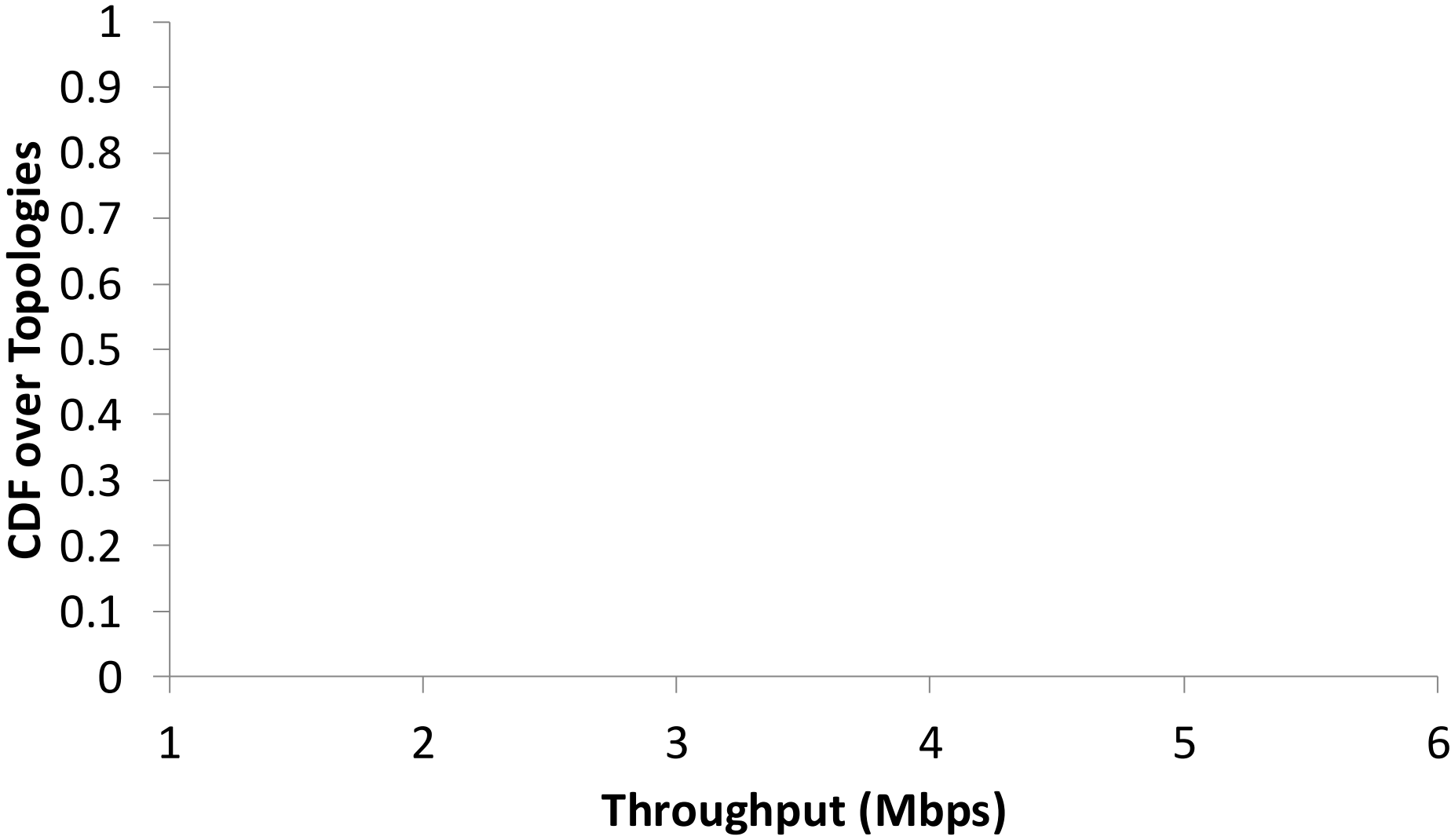


SourceSync improves throughput over Single Best AP by 57%

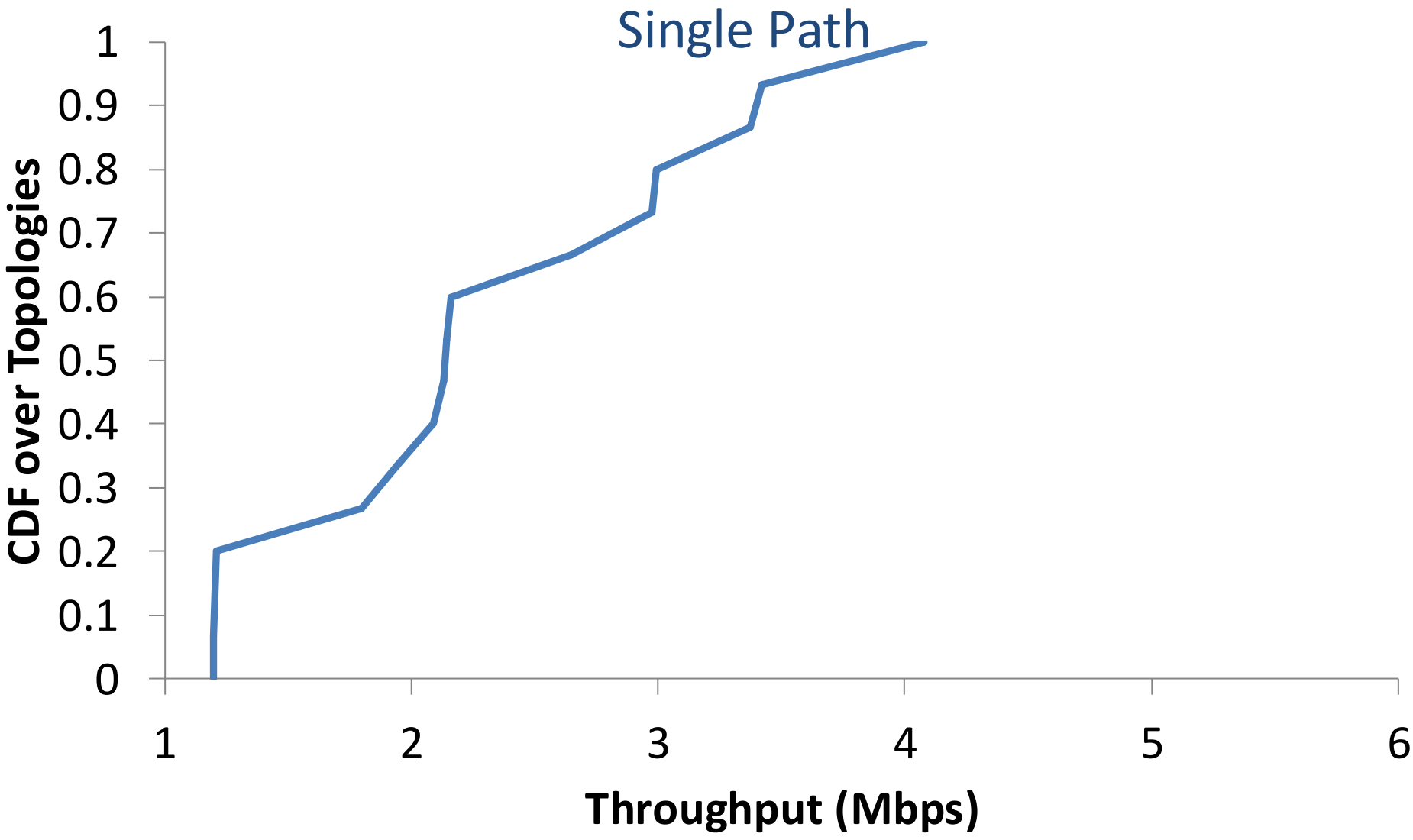
SourceSync With Opportunistic Routing

- We compare
  - Single Path Routing
  - ExOR
  - SourceSync+ExOR

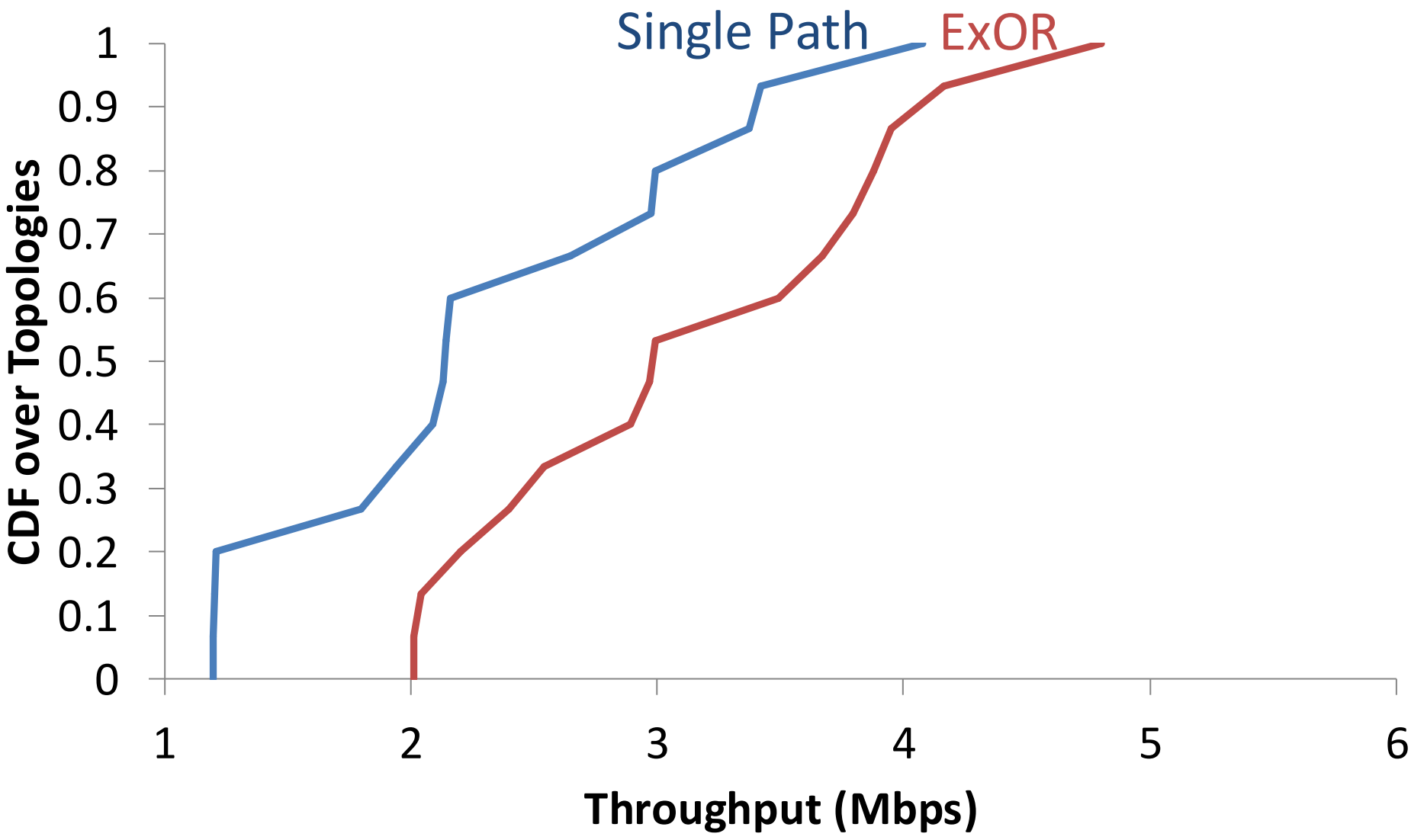
Throughput Gains of SourceSync



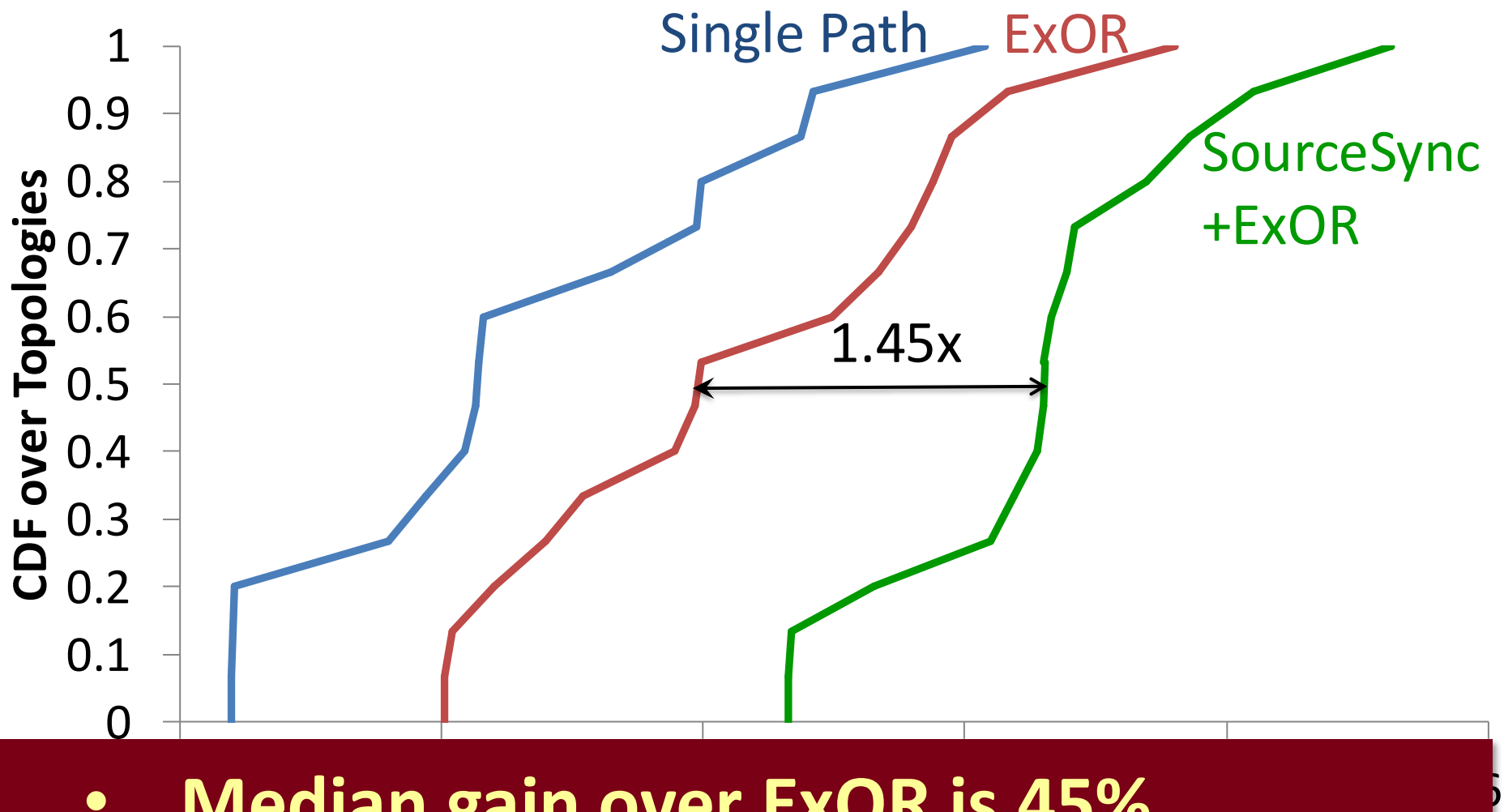
Throughput Gains of SourceSync



Throughput Gains of SourceSync

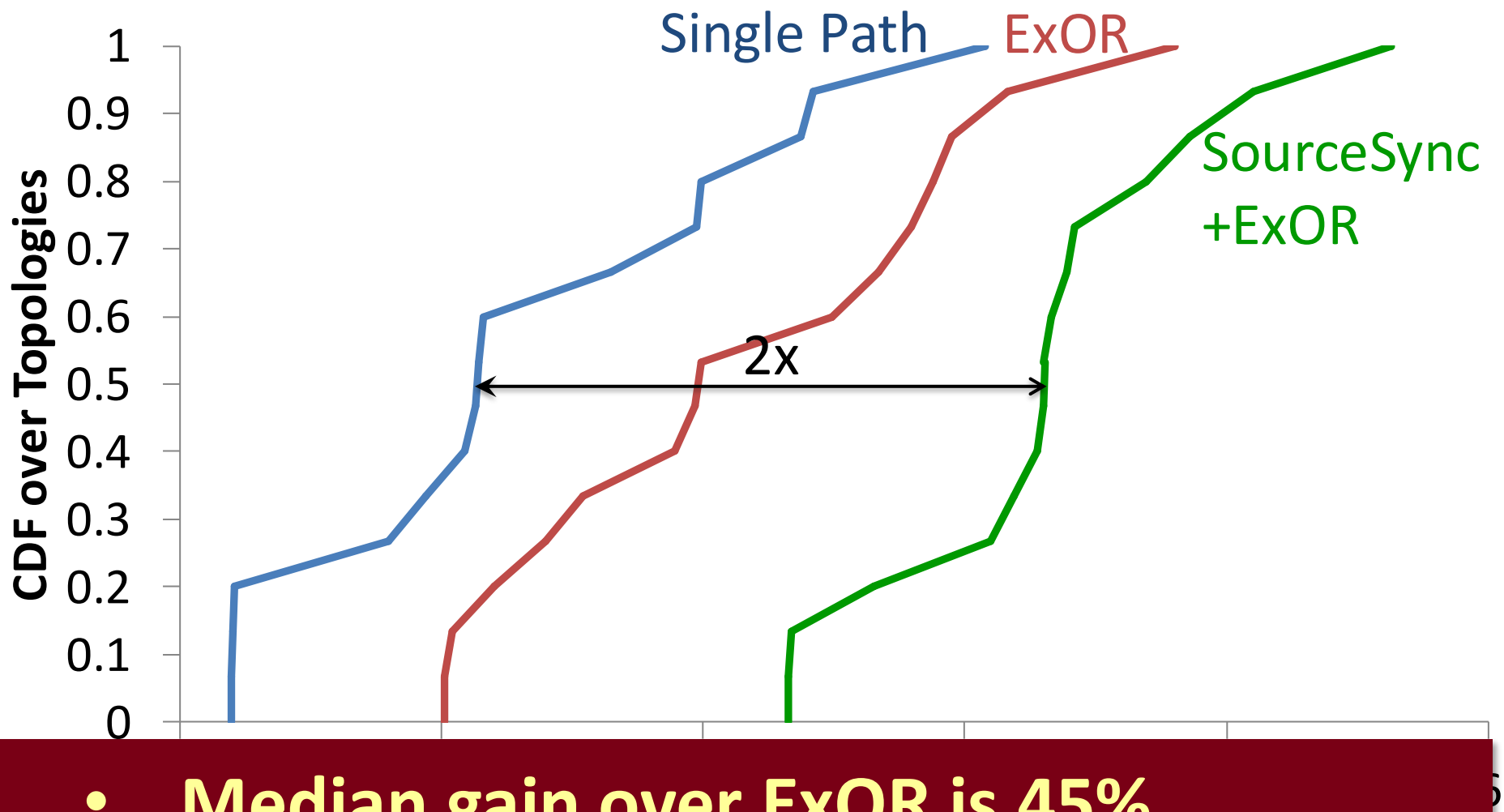


Throughput Gains of SourceSync



- **Median gain over ExOR is 45%**

Throughput Gains of SourceSync



- **Median gain over ExOR is 45%**
- **Median gain over Single Best Path is 2x**

Conclusion

- Synchronizes distributed senders within 20ns
- Adds sender diversity gains to opportunistic routing
- Adds downlink diversity gains to WLANs
- Brings a large body of theory closer to practice
  - Distributed Beamforming, Virtual MIMO, Lattice Codes …