Network Support For Scalable And High-Performance Cloud Exchanges

Muhammad Haseeb, Jinkun Geng, Daniel Duclos-Cavalcanti, Xiyu Hao, Ulysses Butler, Radhika Mittal, Srinivas Narayana, Anirudh Sivaraman

New York University, Clockwork Inc., Technical University of Munich, Rutgers University, University of Illinois Urbana-Champaign

Benefits of Public Cloud for Financial Exchanges

- Reducing constraints of physical space around the matching engine
- Unprecedented scale – 1000s of participants can be supported
- Low barriers to entry for launching new global markets
- Various analytics/ML services residing nearby
- And typical benefits of cloud: flexible resource allocation, reduced cost, offloading management etc.

There will be trade-offs.

- Latencies will be higher.
- Fairness guarantees will be coarser.

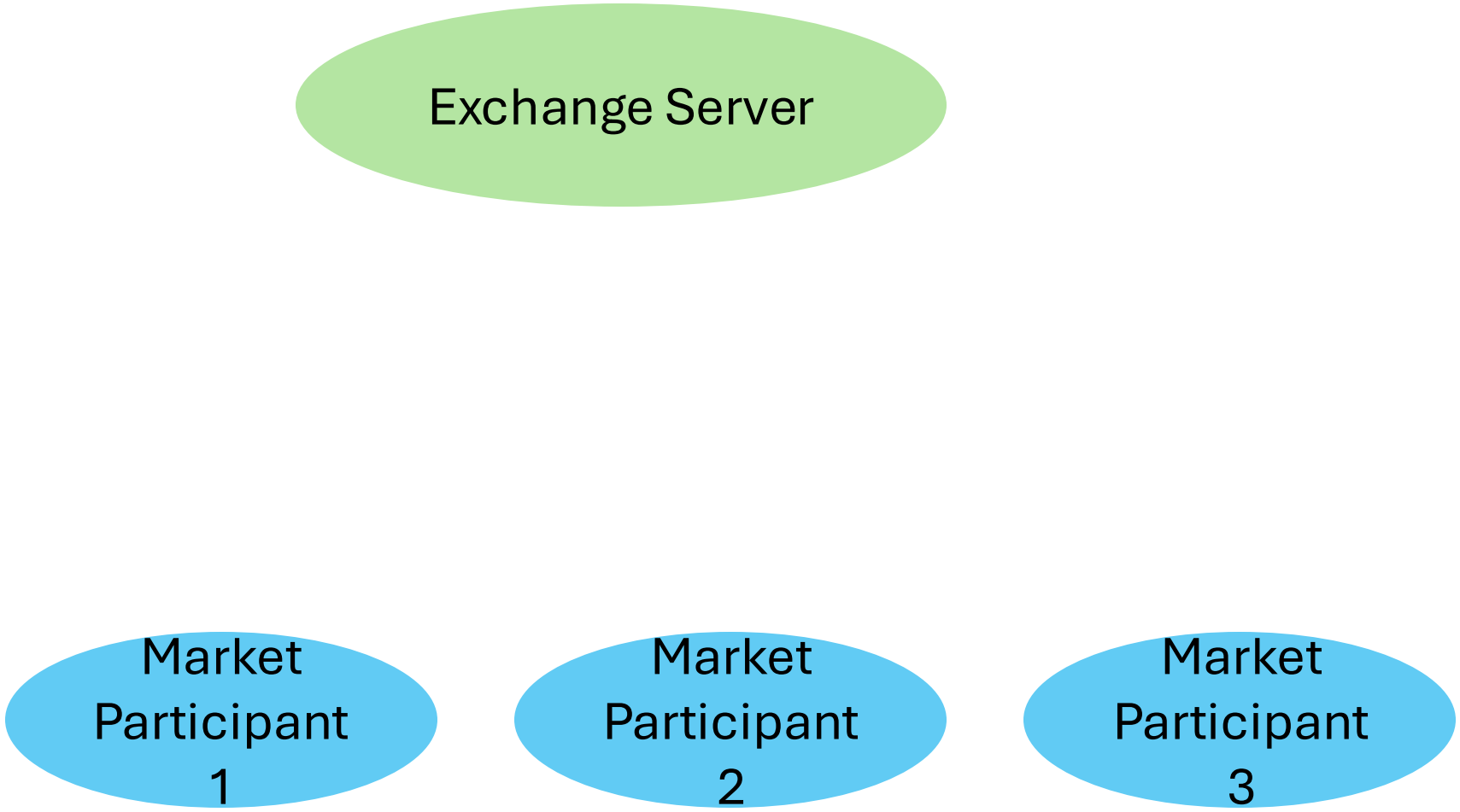
Onyx: Scalable Cloud Financial Exchange

Multicast service that disseminates market data to 1000 participants, each receiving a message within 1-microsecond of each other

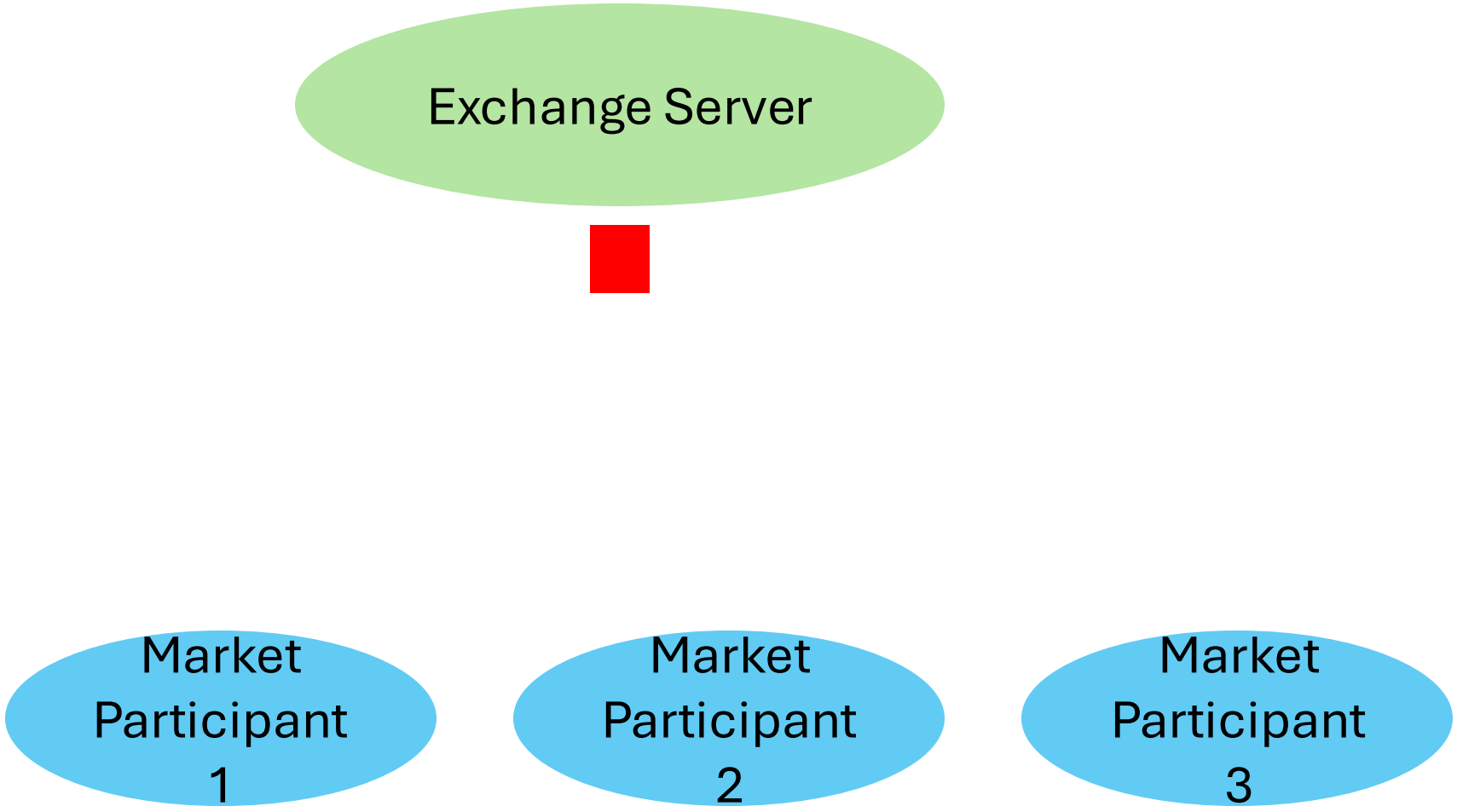
An order submission service which effectively handles bursty traffic while maintaining fairness of competition among the participants

Prototyped on AWS and GCP

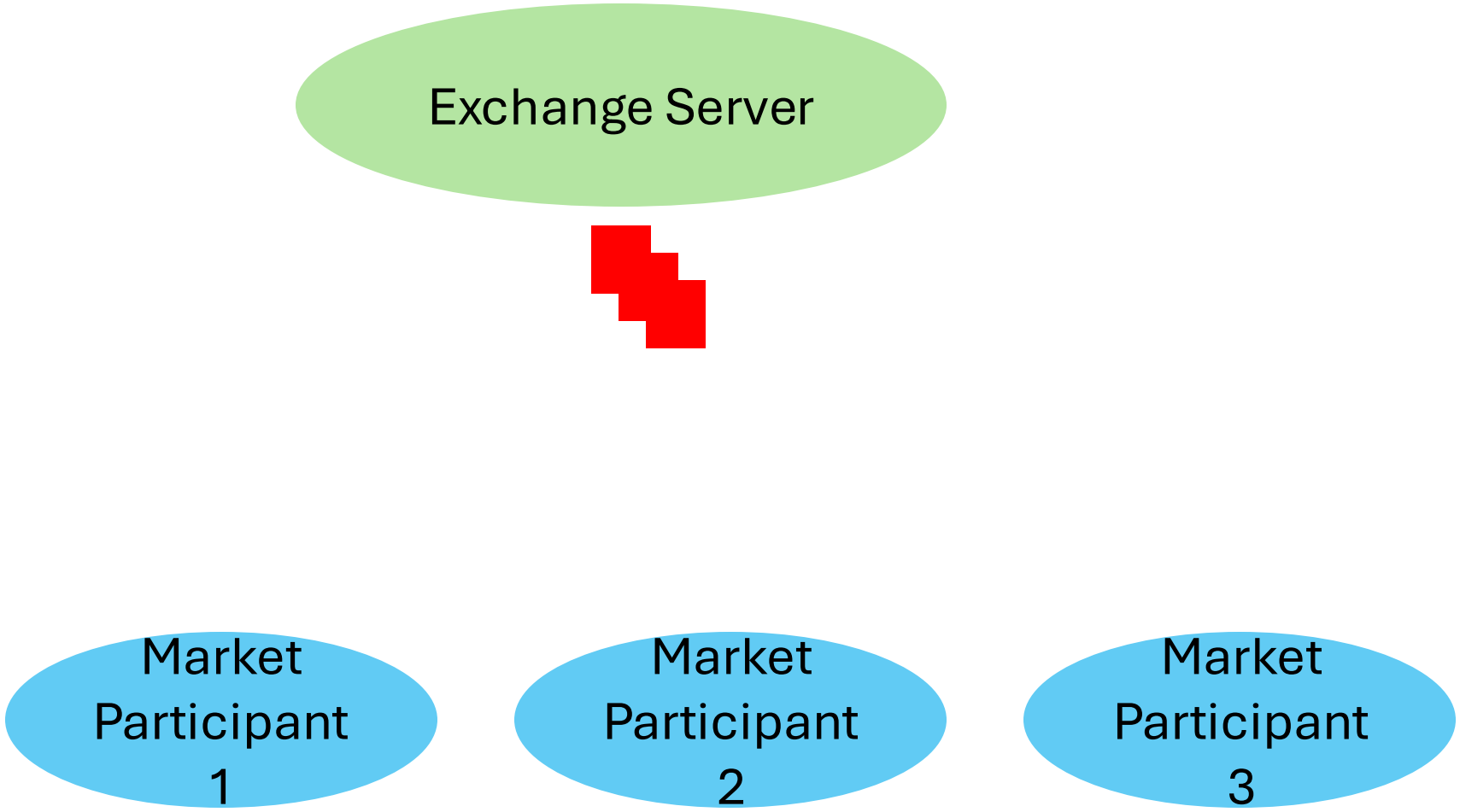
Primer On Financial Exchanges



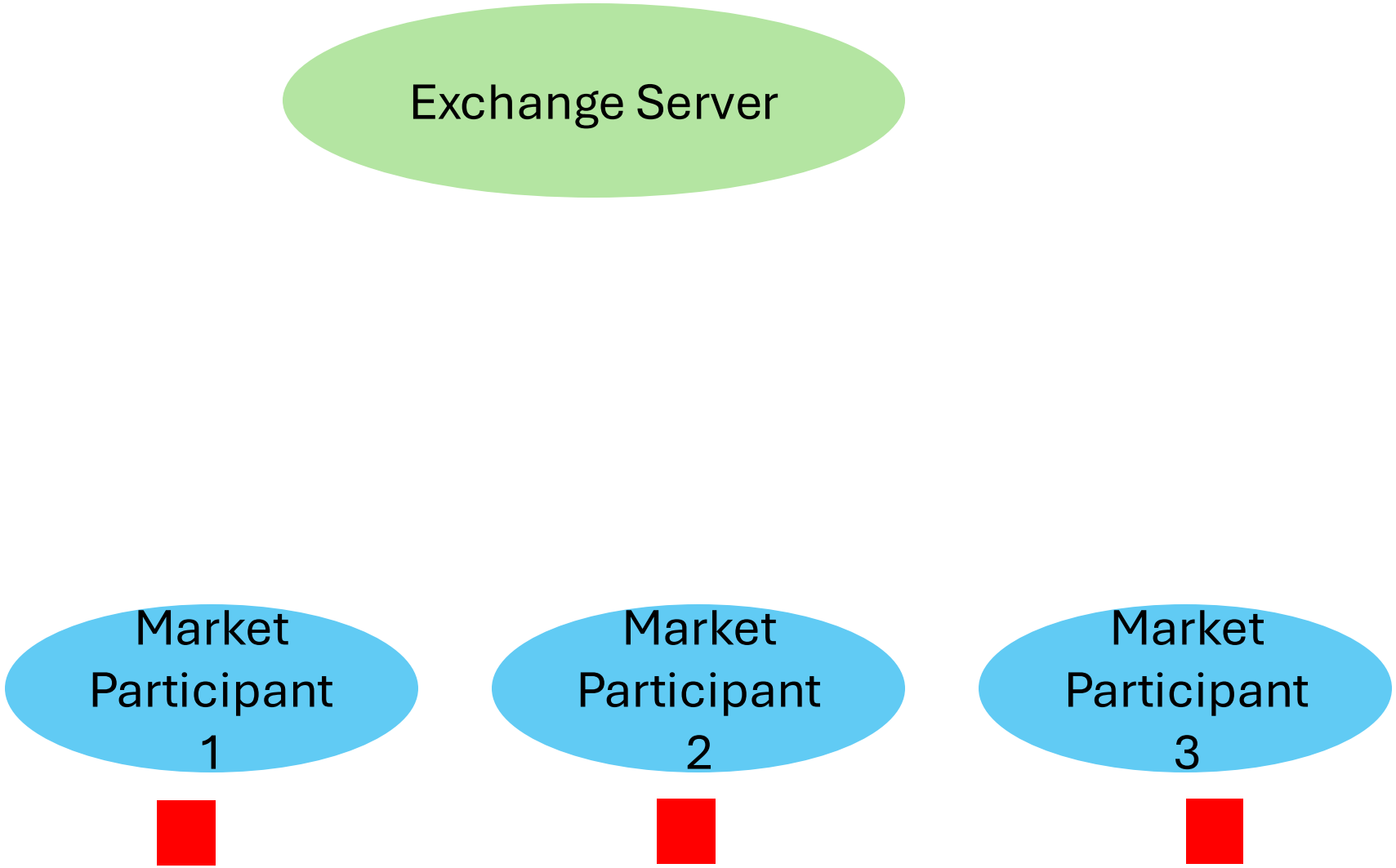
Primer On Financial Exchanges



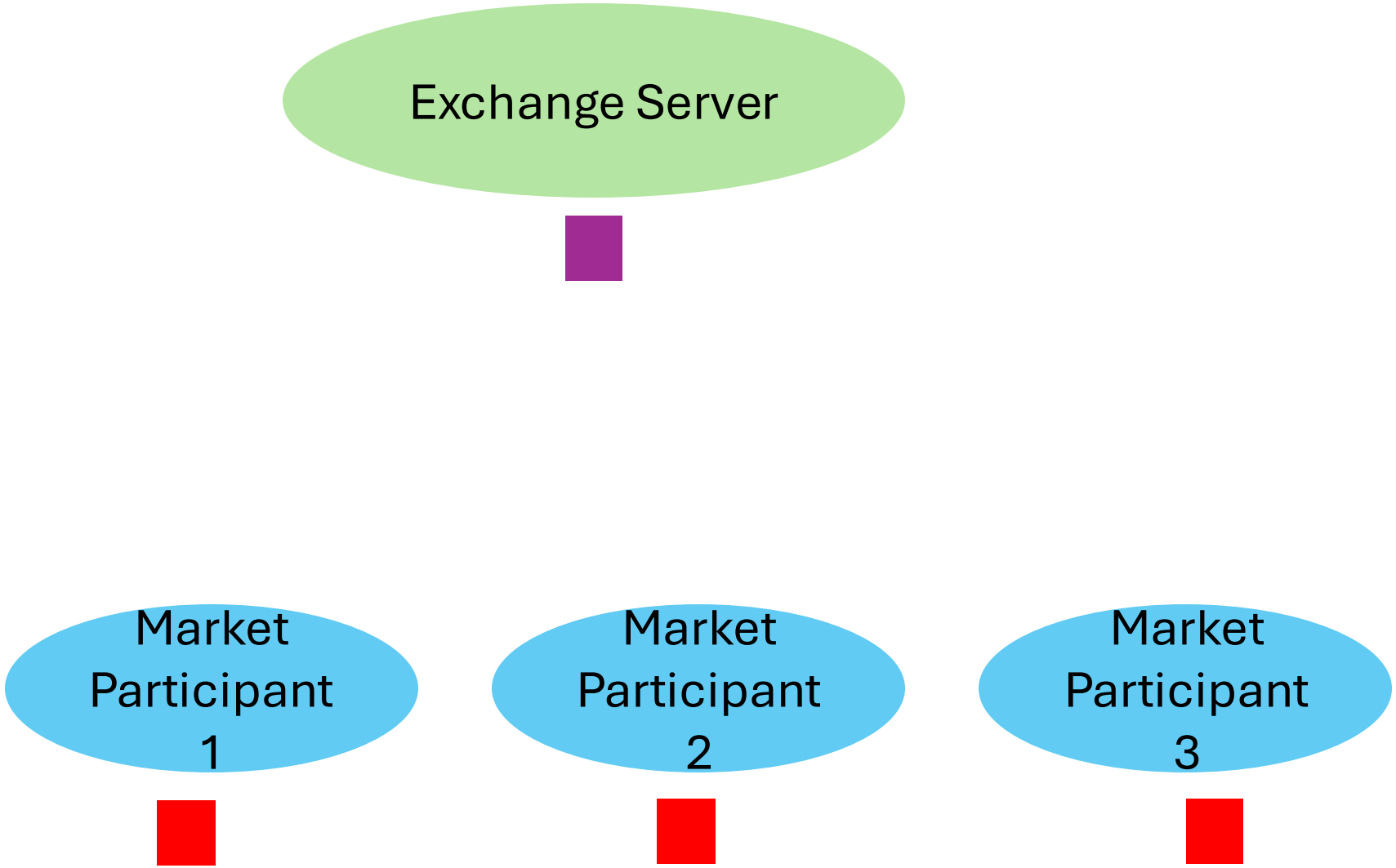
Primer On Financial Exchanges



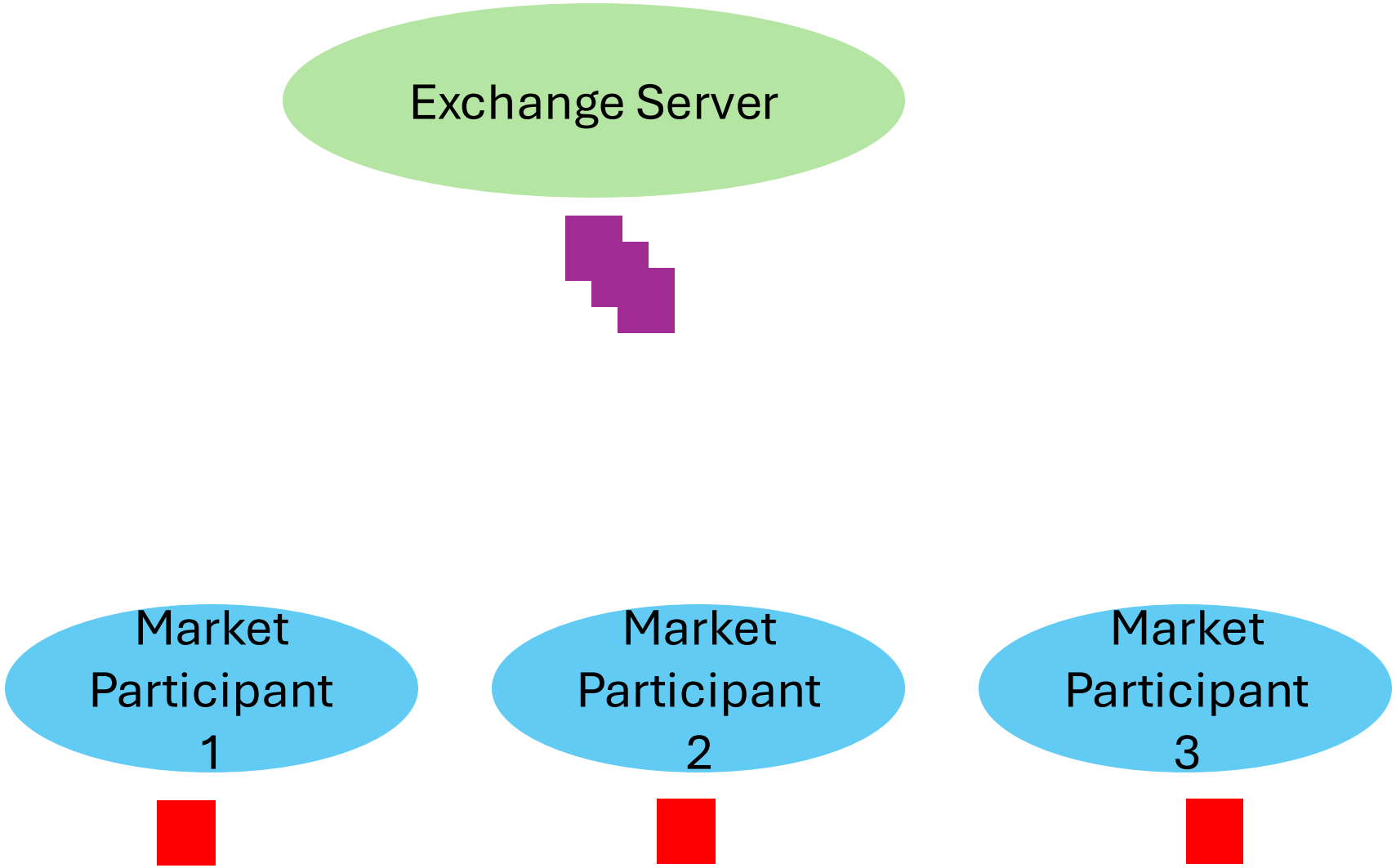
Primer On Financial Exchanges



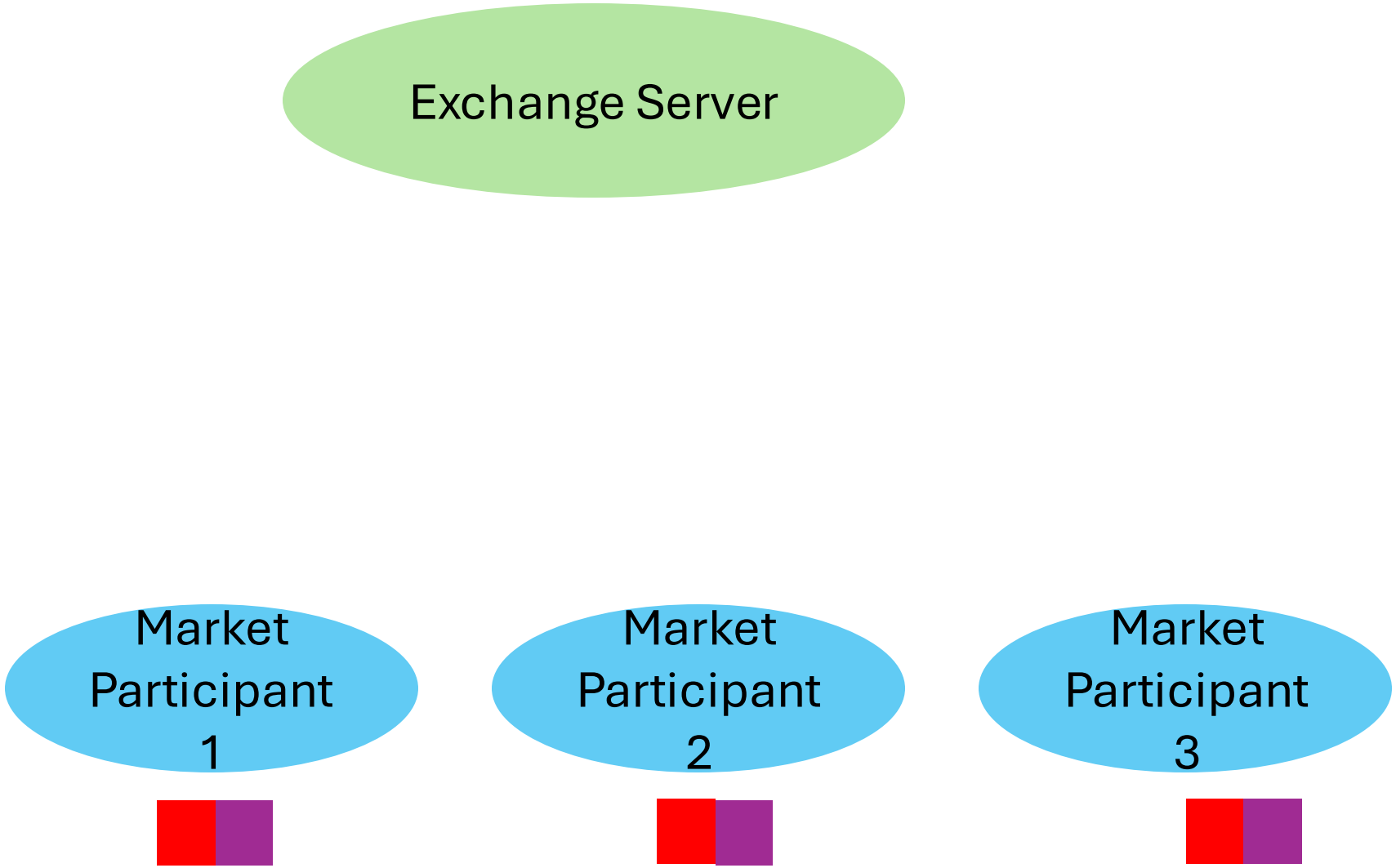
Primer On Financial Exchanges



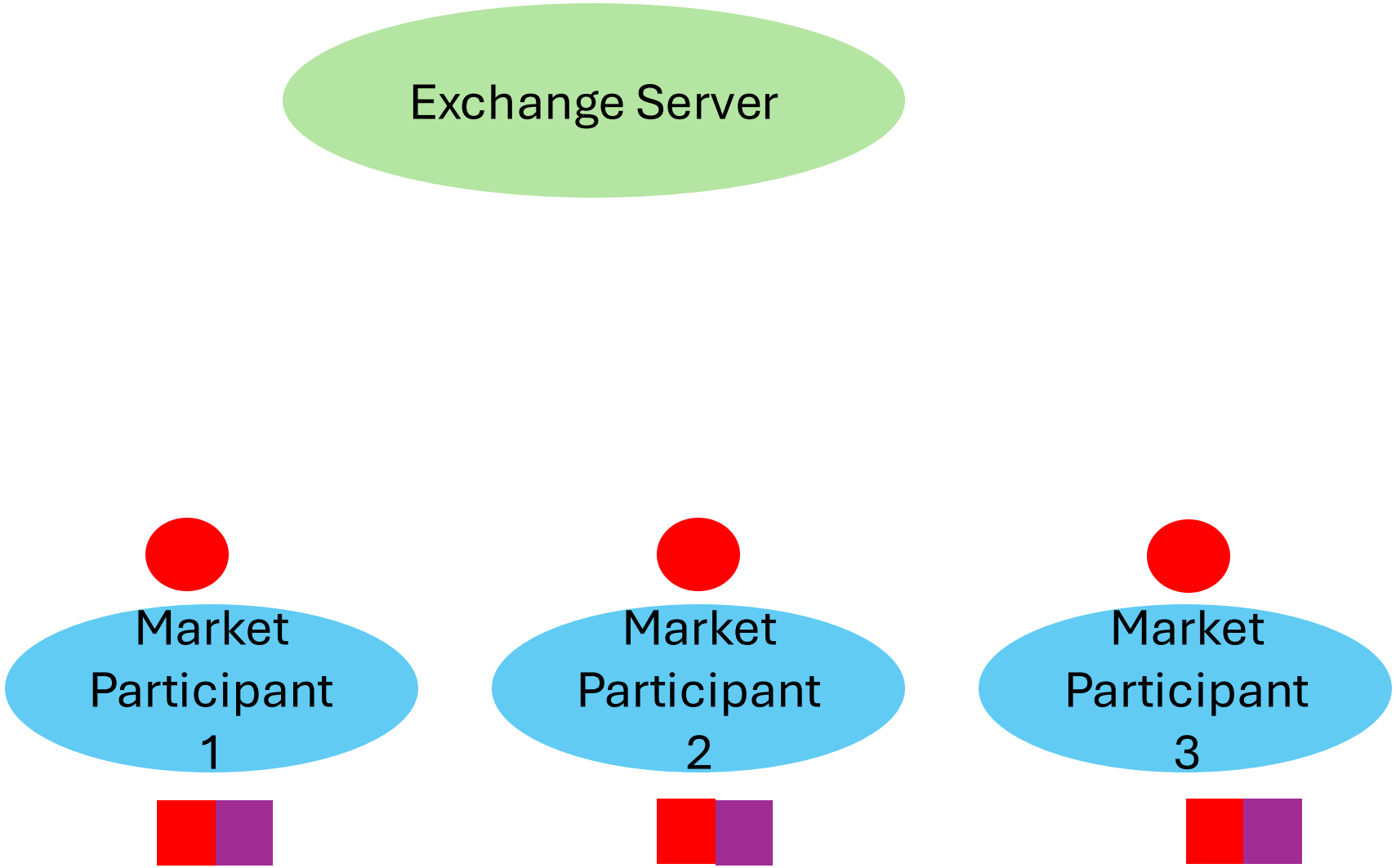
Primer On Financial Exchanges



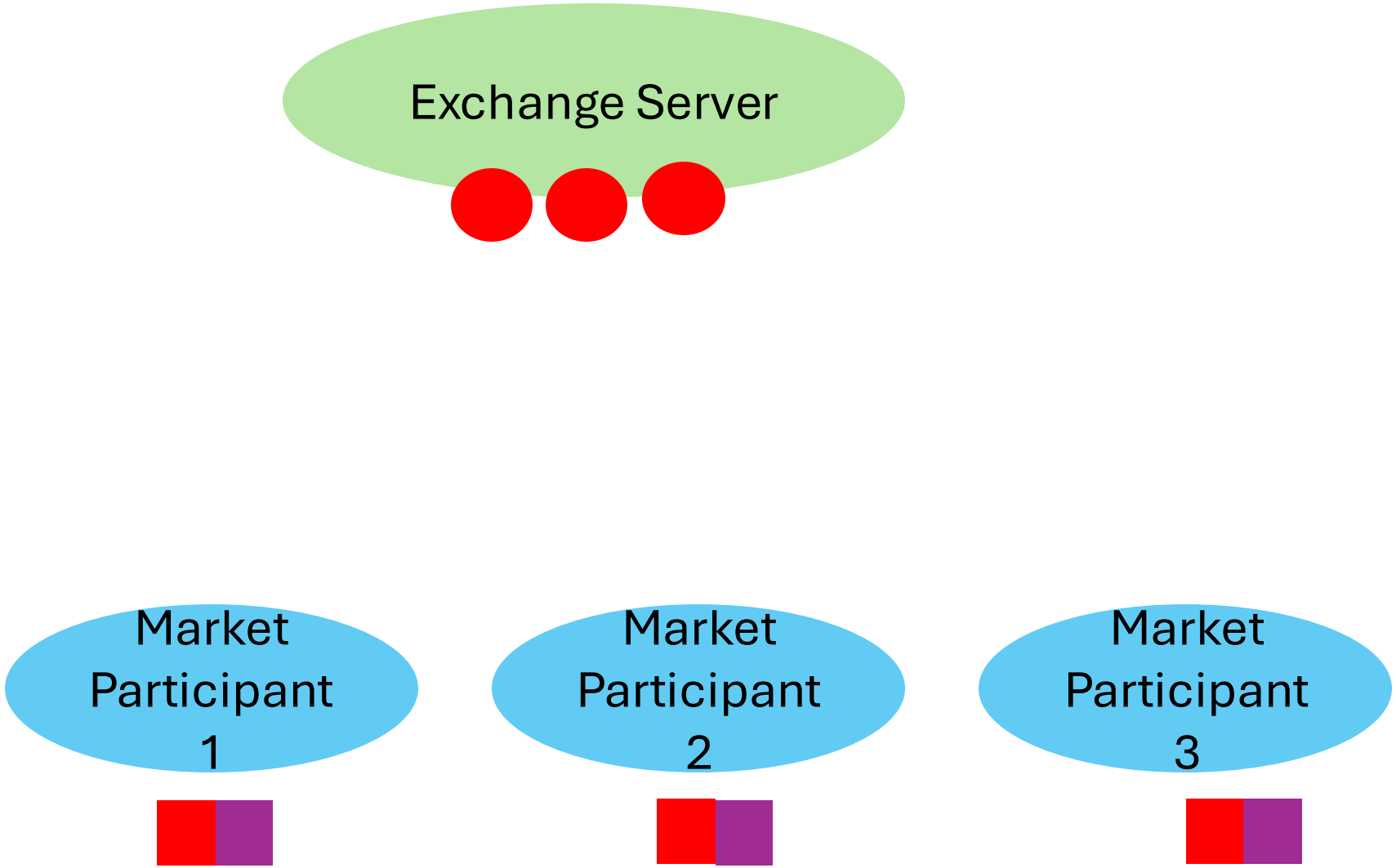
Primer On Financial Exchanges



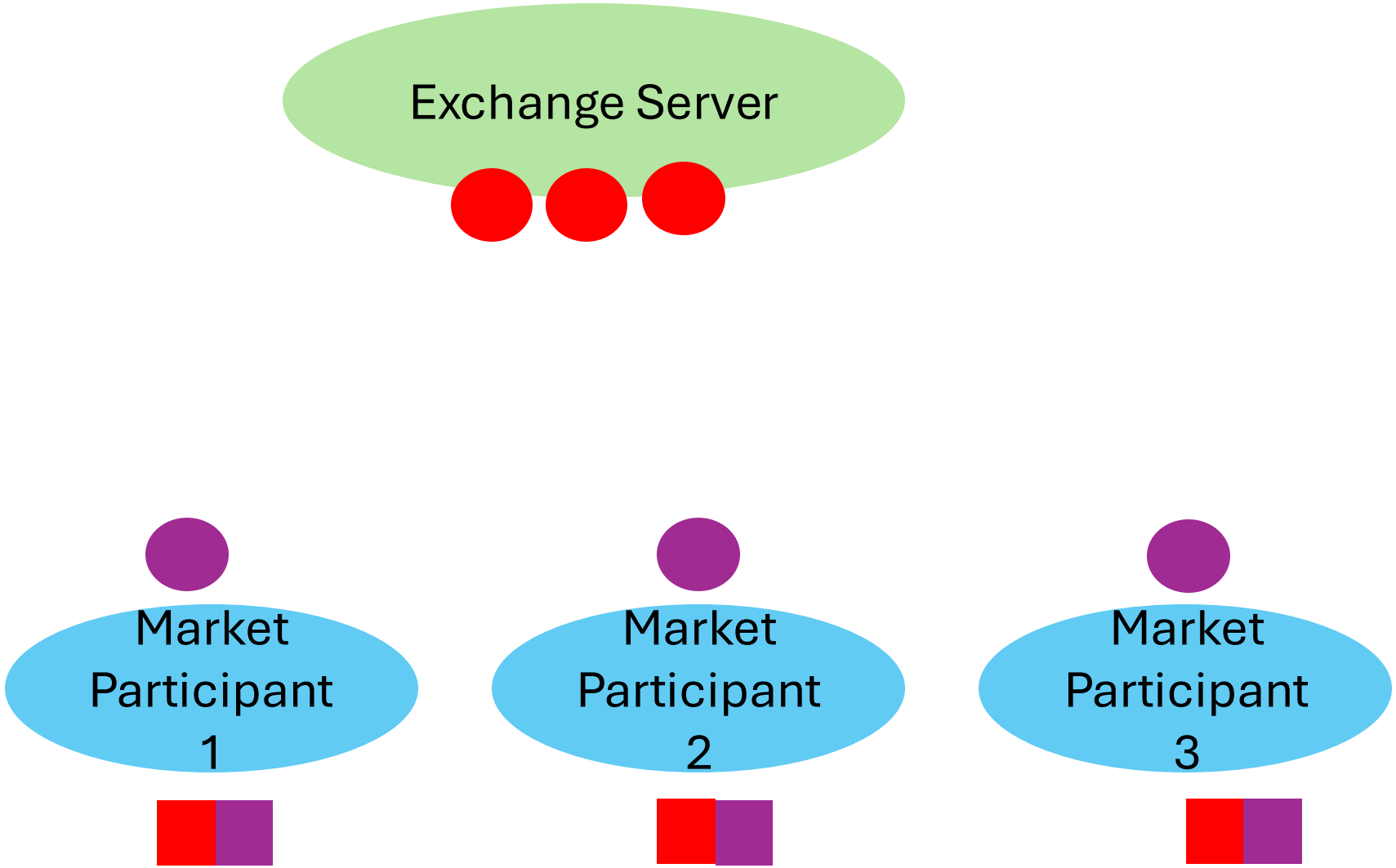
Primer On Financial Exchanges



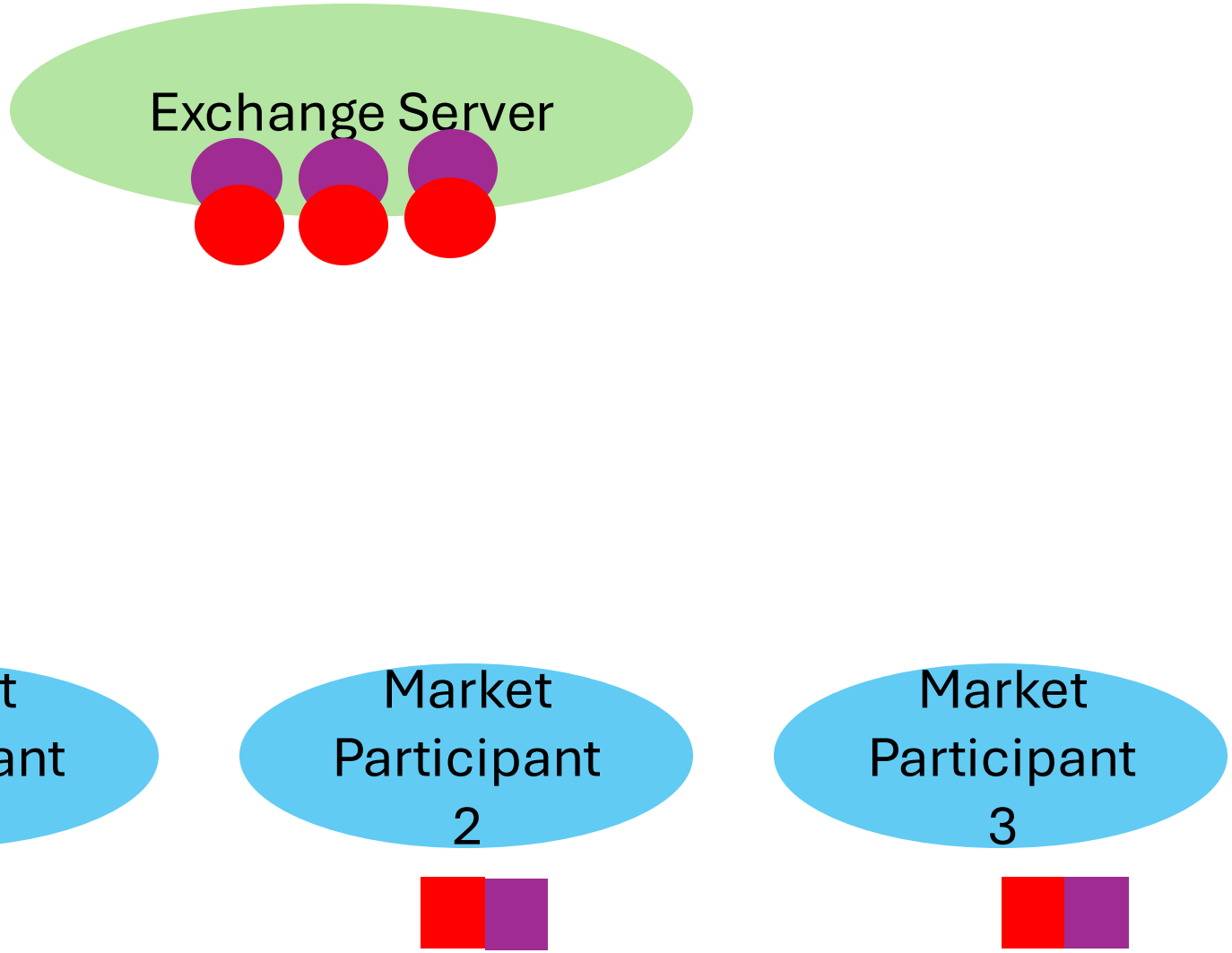
Primer On Financial Exchanges



Primer On Financial Exchanges

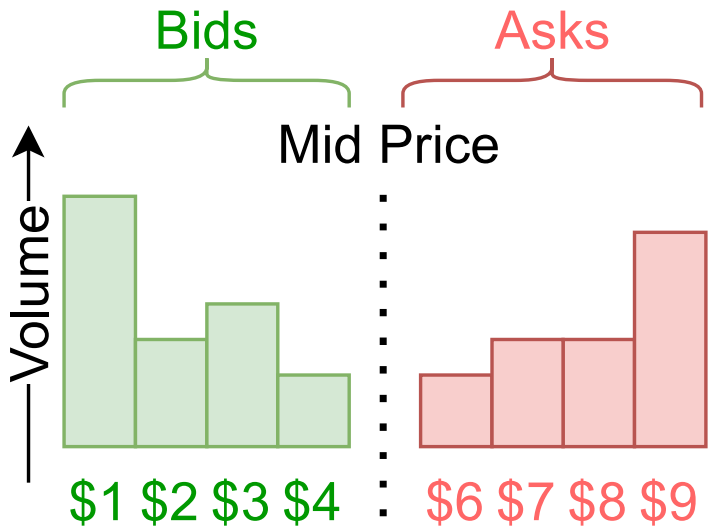


Primer On Financial Exchanges



Primer On Financial Exchanges

Exchange Server



order closer to mid price → Immediate execution/match

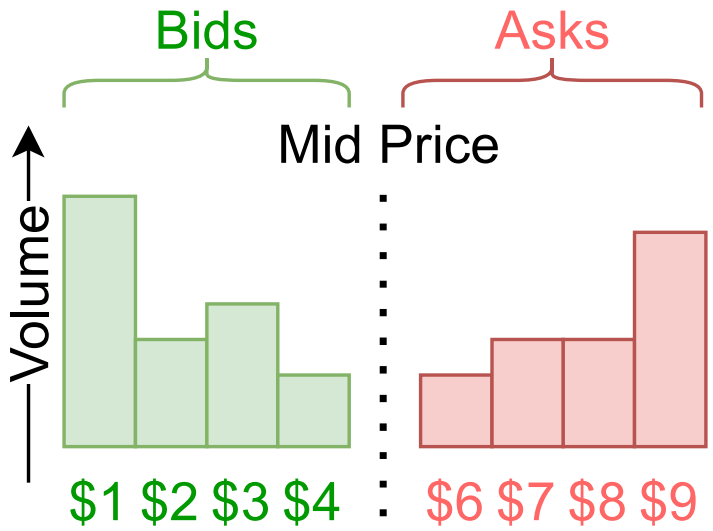
order away from mid price → potential execution/match in future

Position in the queue/price level matters!

Primer On Financial Exchanges

Exchange Server

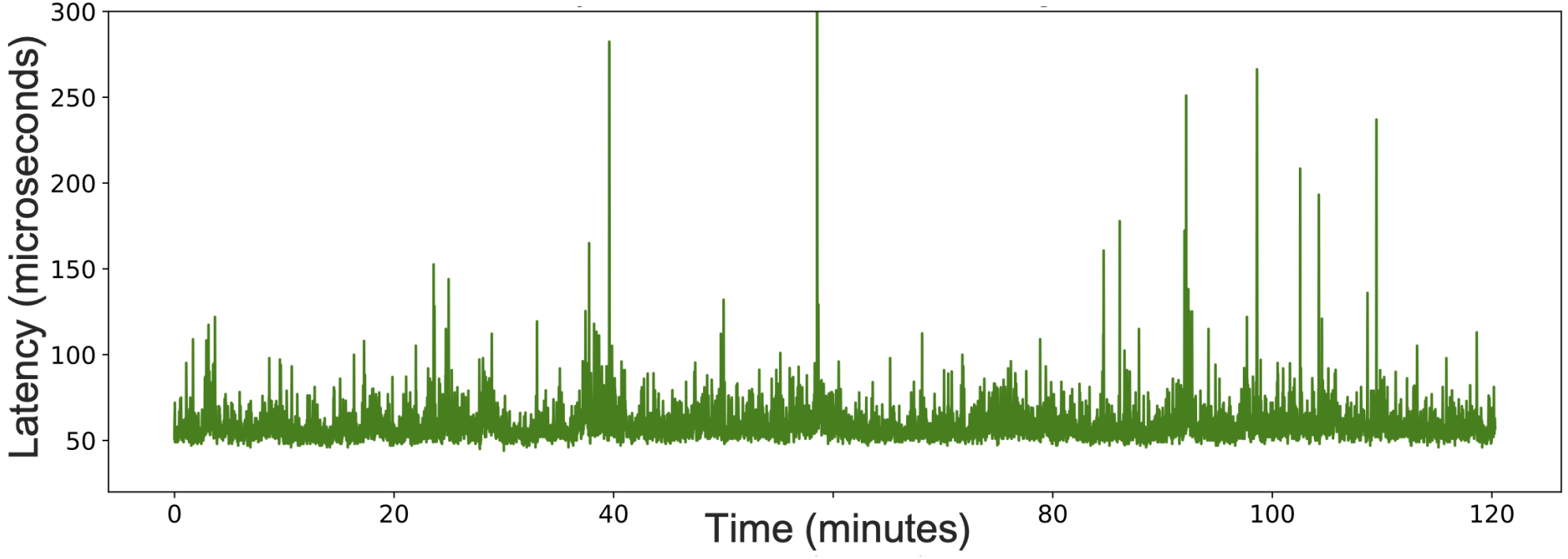
Traditional exchanges built in private data centers or colocation facilities!



order av ential
executi

Competition Among MPs

Position in the queue/price level matters!



Public Cloud Exhibits High Latency Variance

- Latency varies temporally
- Latency varies spatially – across multiple clients
- Unfit for fair competition
- Onyx: mechanisms to achieve fairness + high performance at scale

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness[1]

Outbound Fairness

Inbound Fairness

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness[1]

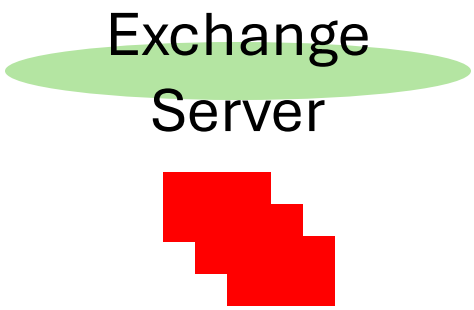
Outbound Fairness

Inbound Fairness

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness[1]

Outbound Fairness



Inbound Fairness

MP1 MP2 MP3

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness[1]

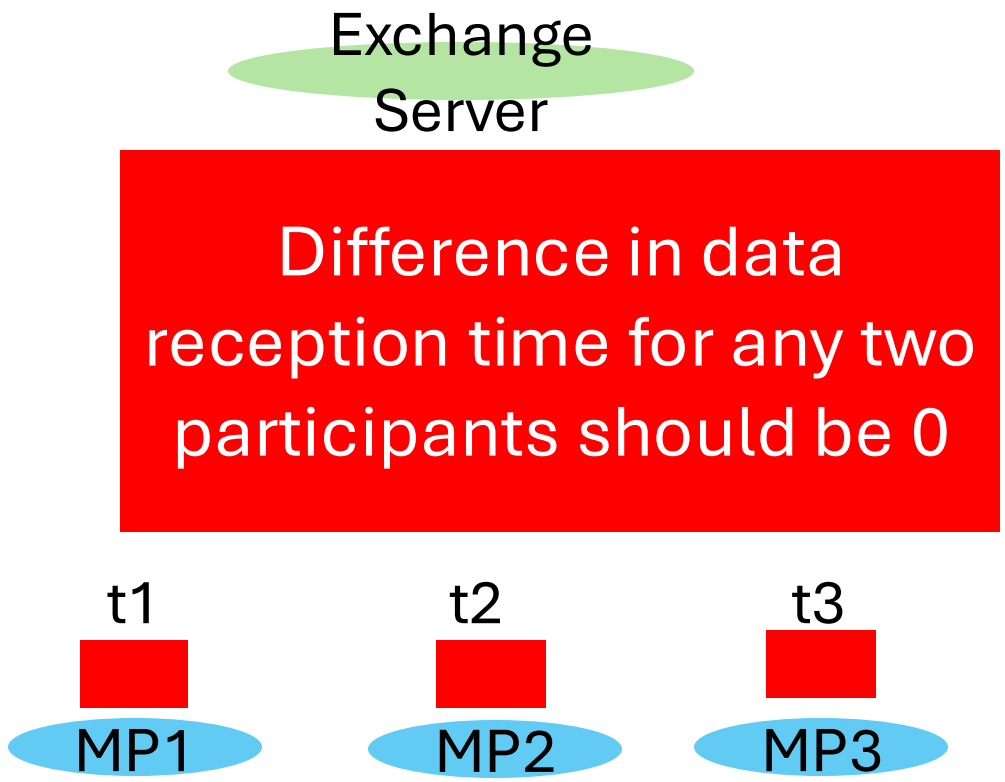
Outbound Fairness

Inbound Fairness

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness[1]

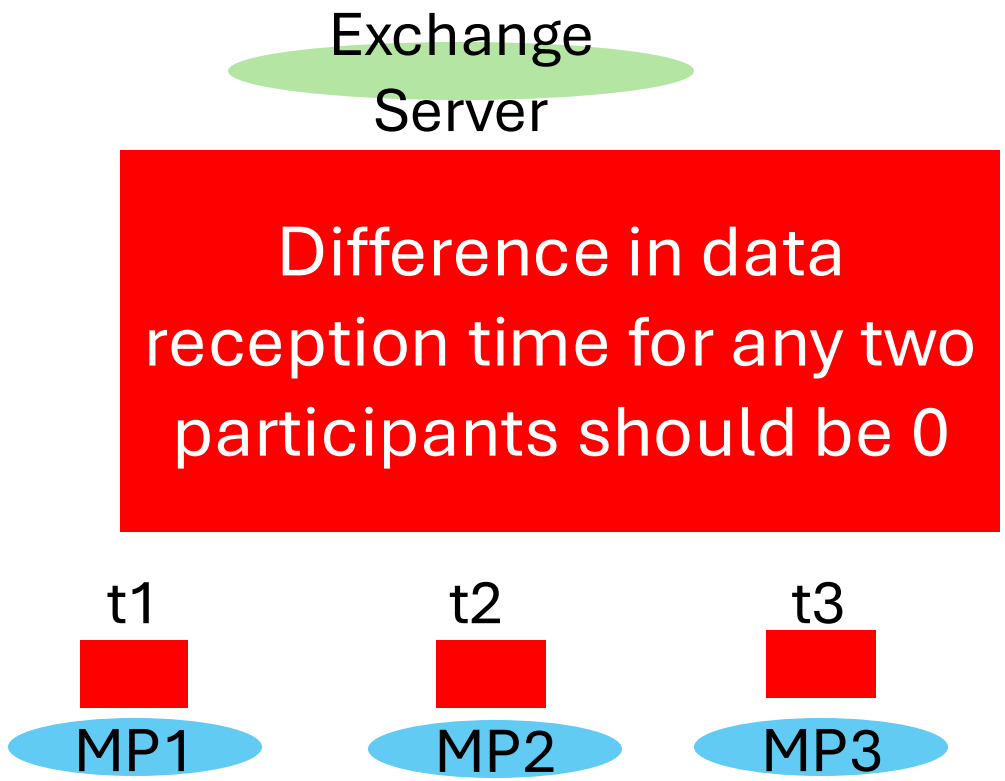
Outbound Fairness

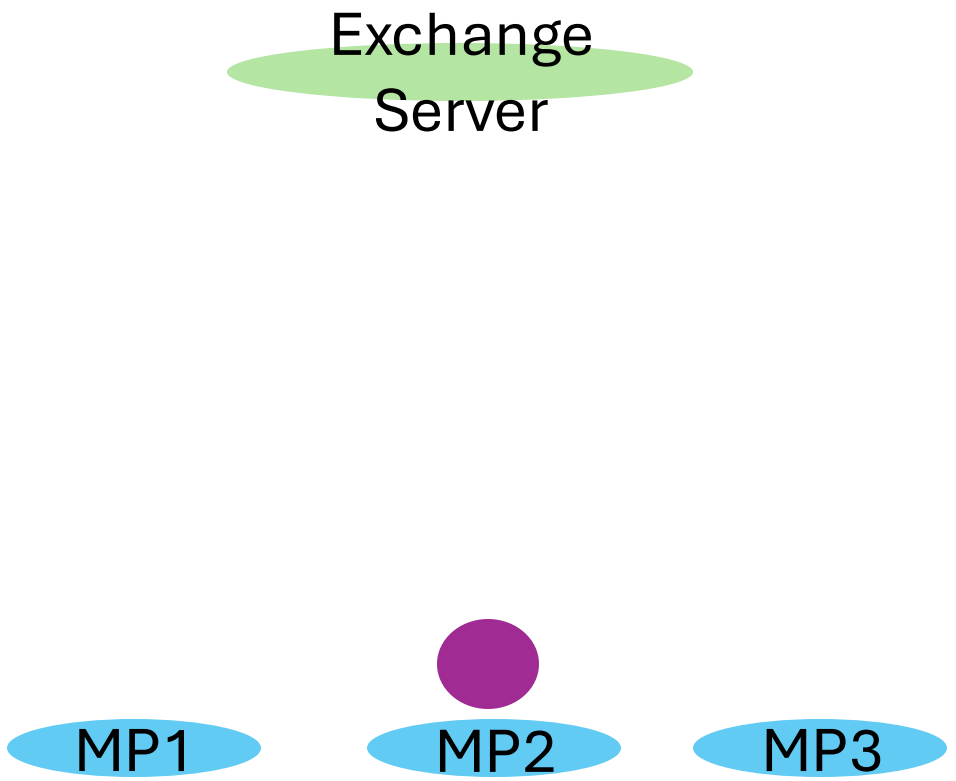


Inbound Fairness

Fairness[1]

Outbound Fairness

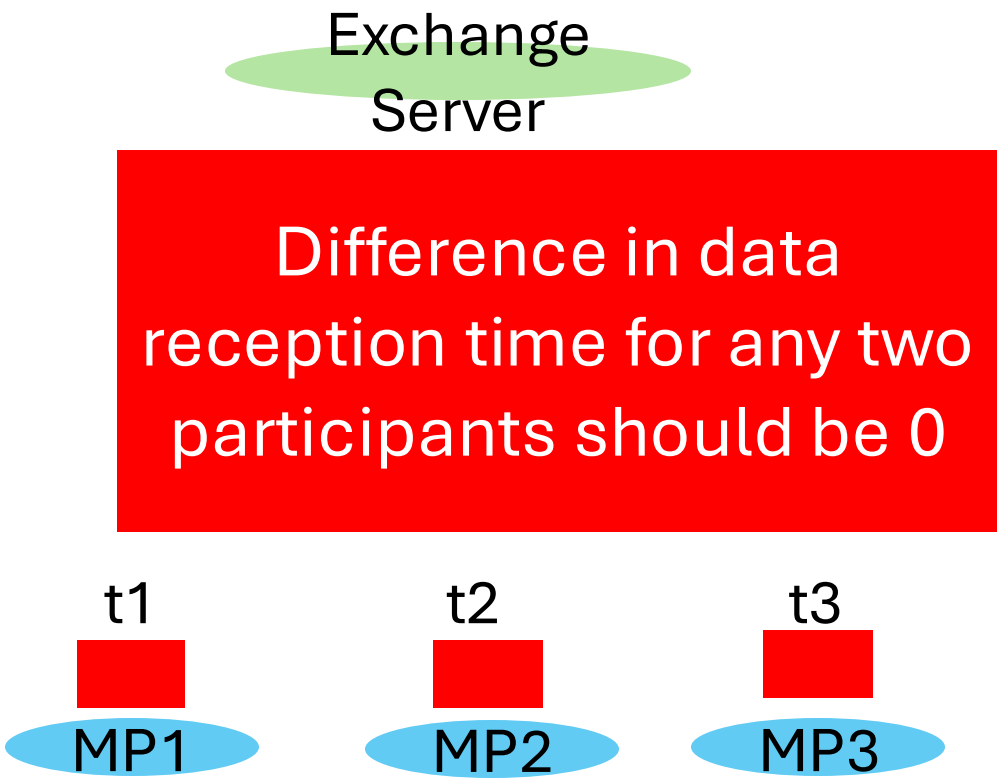


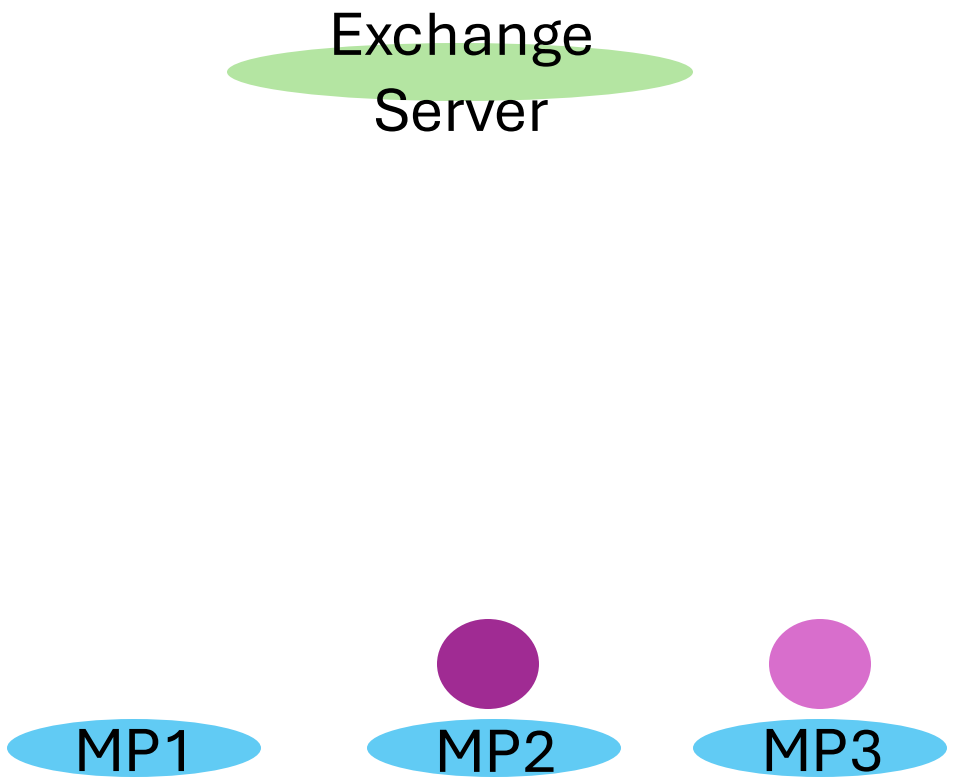


Inbound Fairness

Fairness[1]

Outbound Fairness





Inbound Fairness

Fairness[1]

Exchange Server

Outbound Fairness

Exchange Server

Difference in data reception time for any two participants should be 0

t1 t2 t3

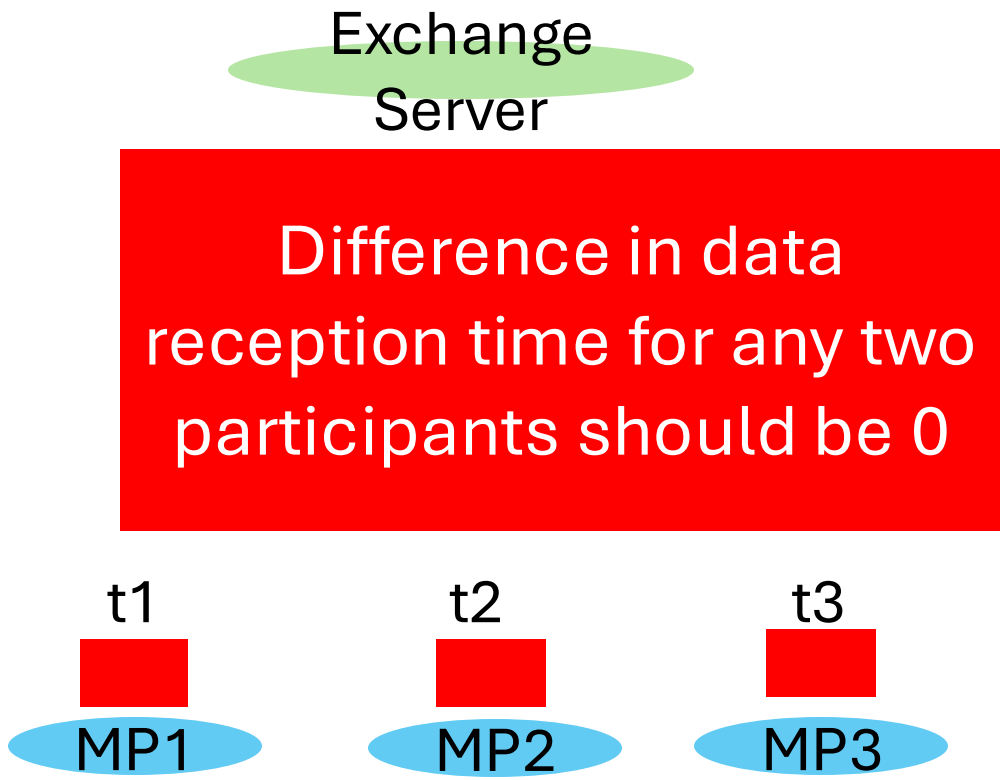
MP1 MP2 MP3

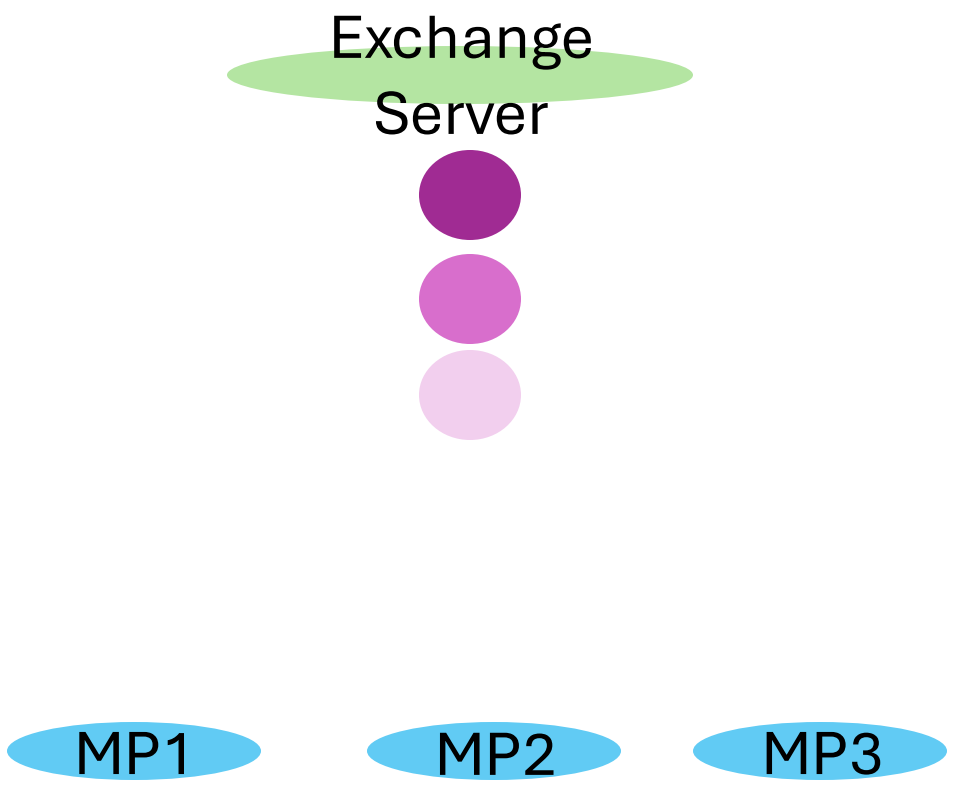
MP1 MP2 MP3

Inbound Fairness

Fairness[1]

Outbound Fairness

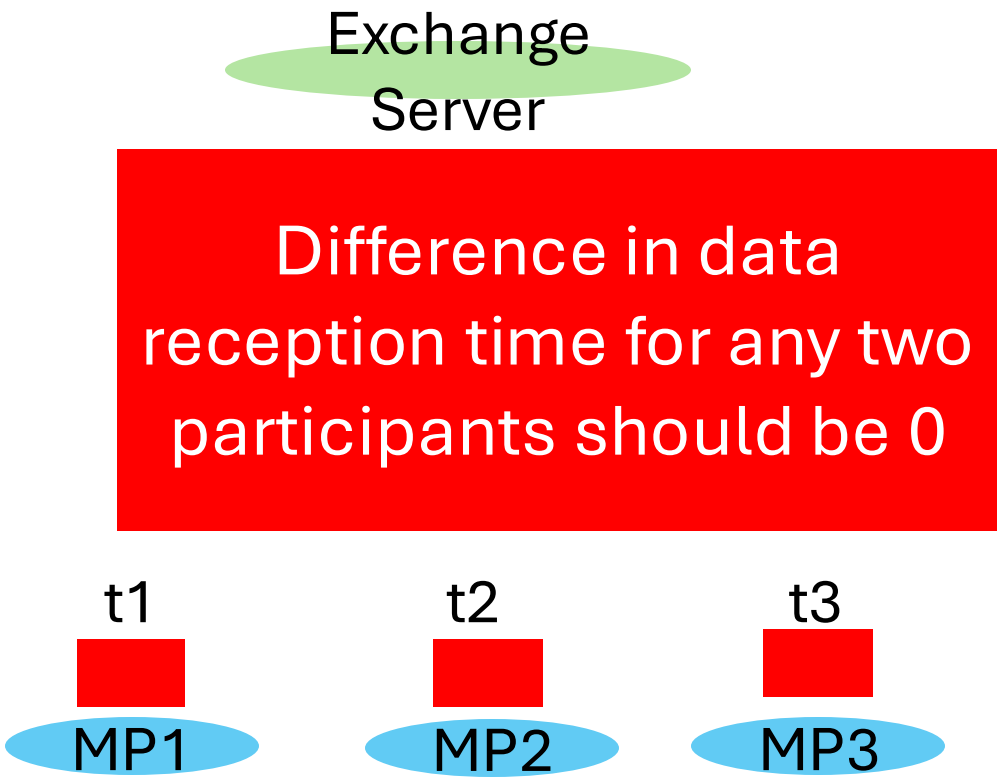


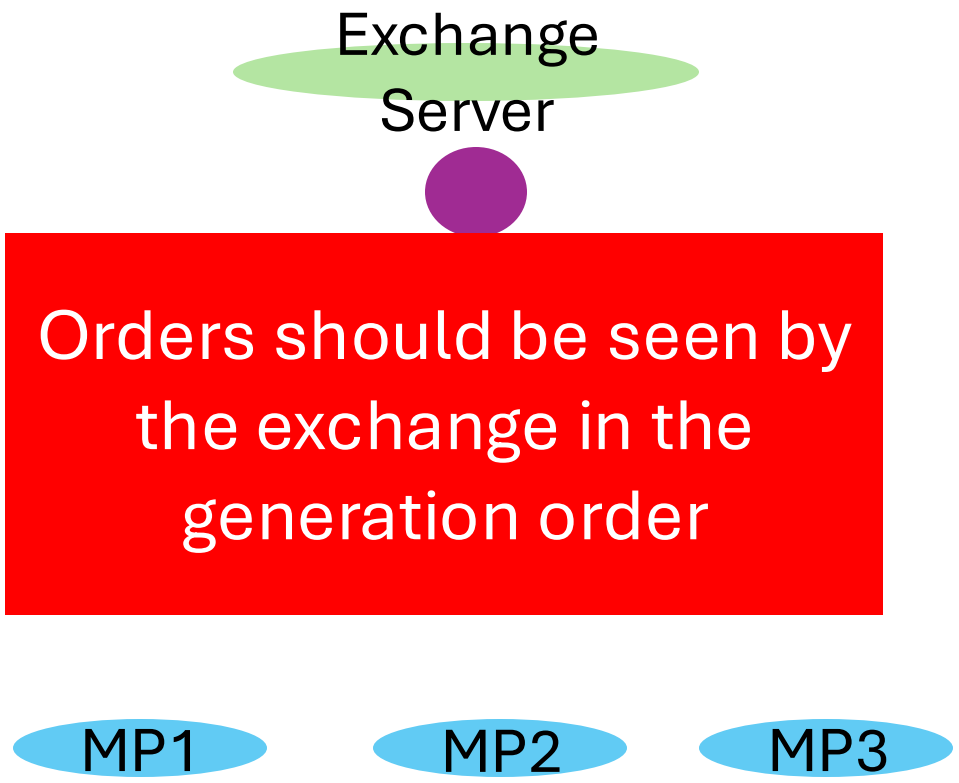


Inbound Fairness

Fairness[1]

Outbound Fairness





Inbound Fairness

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness[1]

Outbound Fairness

Difference in data reception time for any two participants should be 0

Orders should be seen by the exchange in the generation order

Inbound Fairness

[1] CloudEx: A Fair-Access Financial Exchange in the Cloud

Fairness In Practice

Outbound Fairness

Difference in data reception time for any two participants should be 0

Orders should be seen by the exchange in the generation order

Inbound Fairness

Fairness In Practice

Outbound Fairness

Difference in data reception time for any two participants should be ~0

Orders should be seen by the exchange in the generation order

Inbound Fairness

Fairness In Practice

Outbound Fairness

Difference in data reception time for any two participants should be ~0

Orders should be seen by the exchange in the timestamped generation order

Inbound Fairness

Achieving fairness while being performant at scale is our goal.

Fairness In Practice

Outbound Fairness

Difference in data reception time for any two participa

We target 1000 market participants (MPs). Each MP is a separate VM.

ers should be seen by he exchange in the estamped generation order

Inbound Fairness

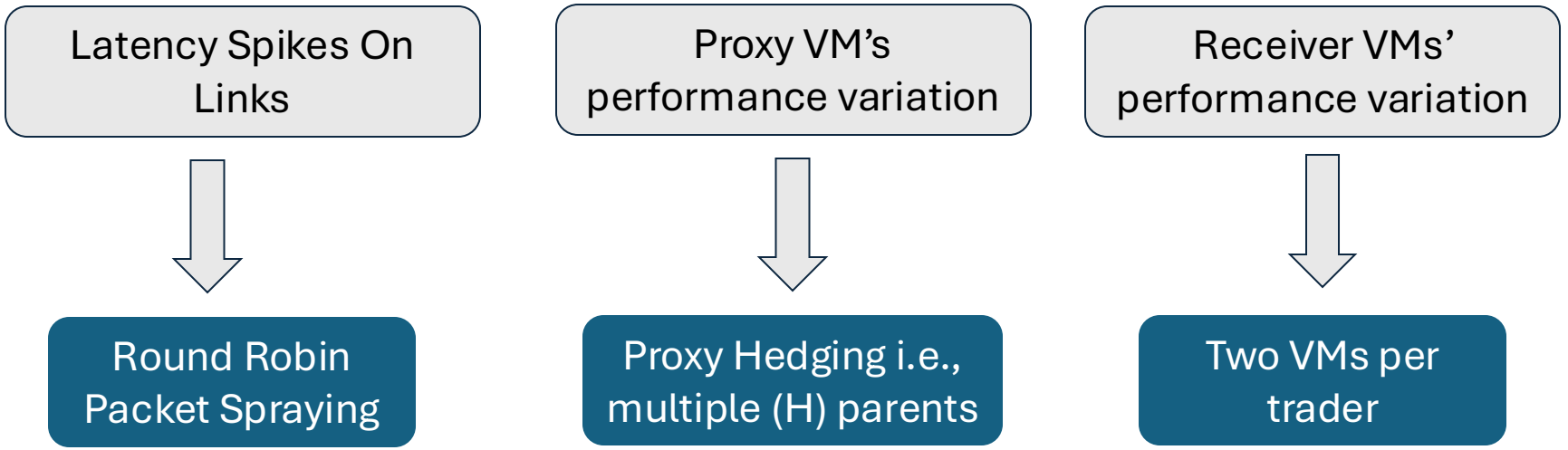
Achieving fairn ss while being performant at scale is our goal.

Scalable Outbound Fairness

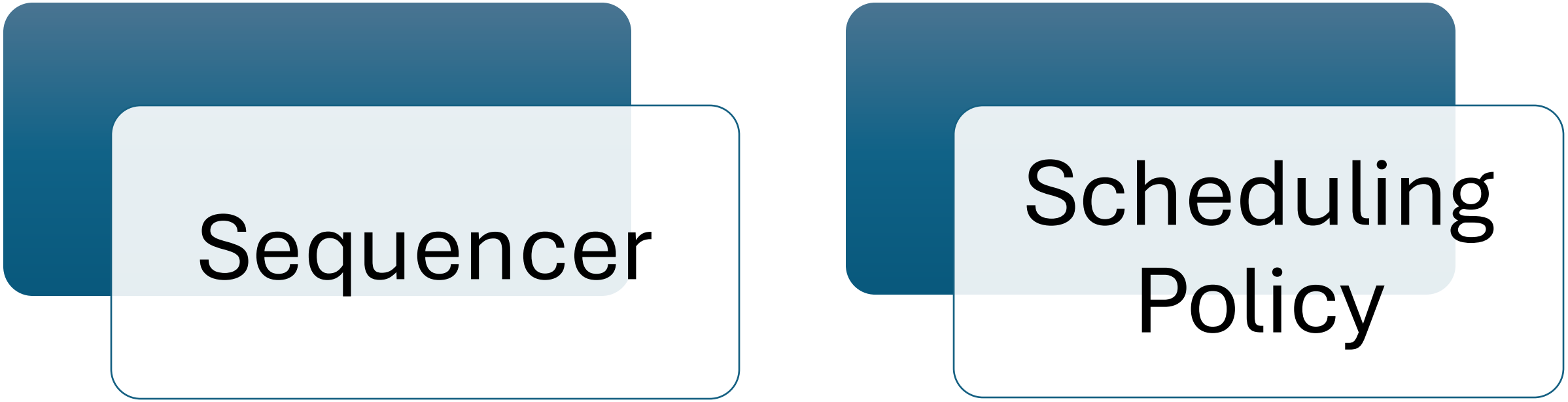
Achieved by a simple “hold-and-release” primitive.

We scale it much further by proposing an efficient market data multicast mechanism.

A tree of proxy nodes is utilized that helps scale multicast.



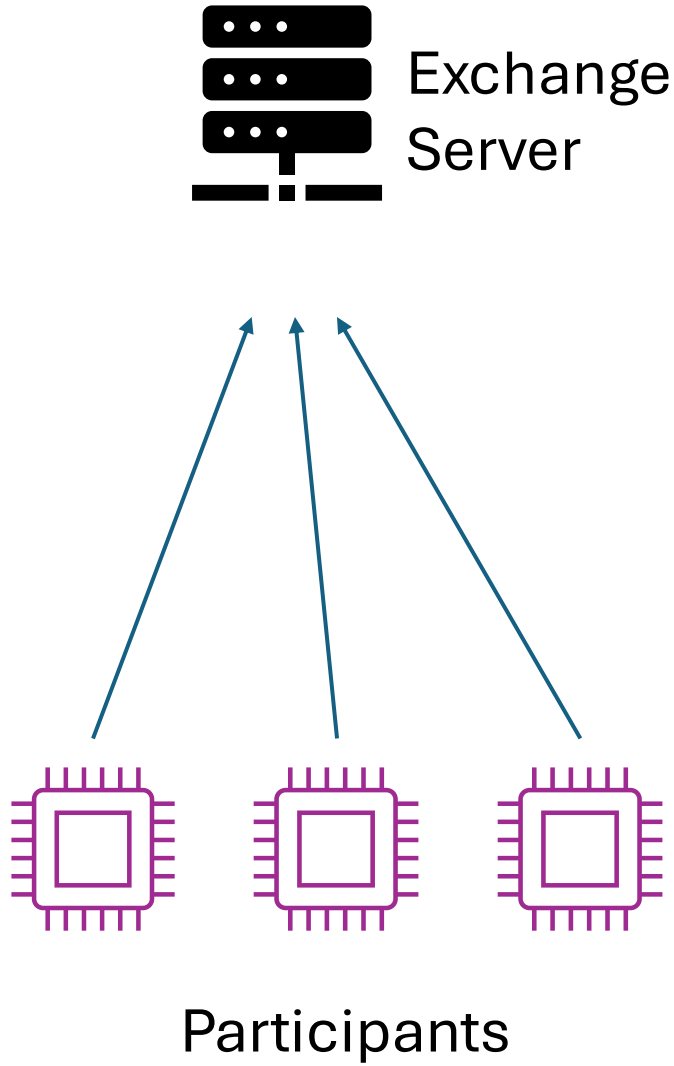
Scalable Inbound Fairness



A sequencer

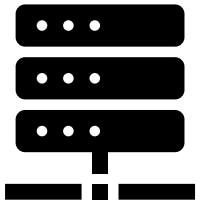
- Guarantees that orders are seen by the exchange in non-decreasing order of their (generation) timestamps

- Assumes clock synchronization



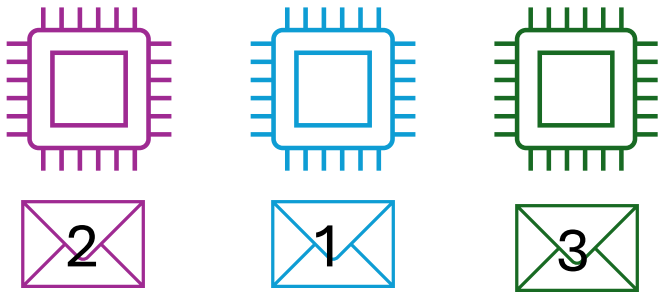
A sequencer

Clients attach message generation timestamps to outgoing messages



Exchange Server

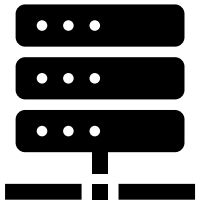
Sequencer waits for at least one message from each client



Clients/Participants

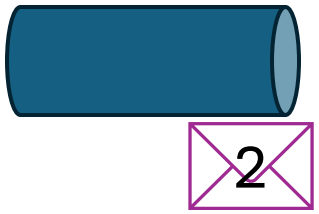
A sequencer

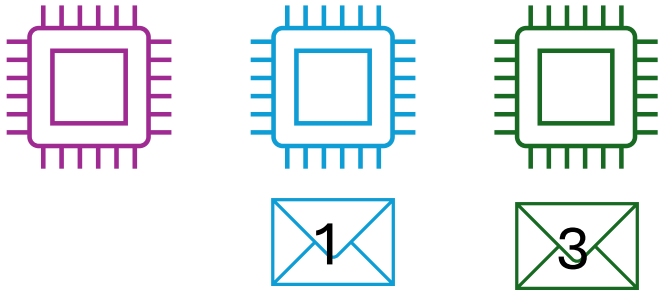
Clients attach message generation timestamps to outgoing messages



Exchange Server

Sequencer waits for at least one message from each client

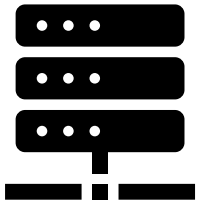




Clients/Participants

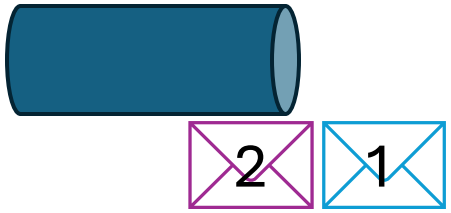
A sequencer

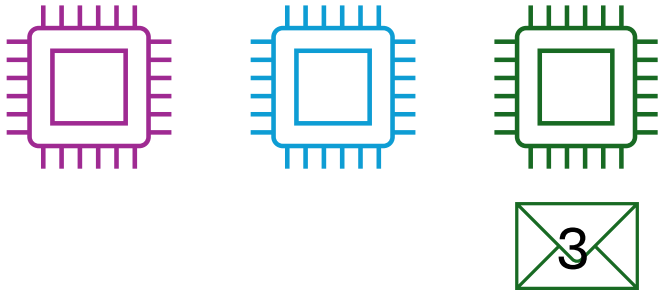
Clients attach message generation timestamps to outgoing messages



Exchange Server

Sequencer waits for at least one message from each client

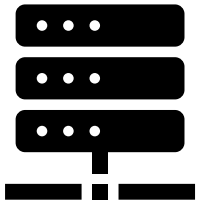




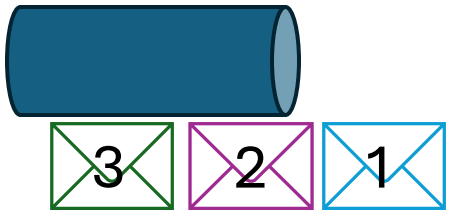
Clients/Participants

A sequencer

Clients attach message generation timestamps to outgoing messages

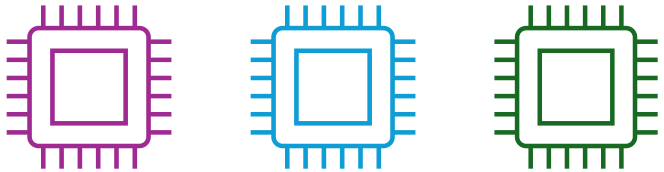


Exchange Server



Sequencer waits for at least one message from each client

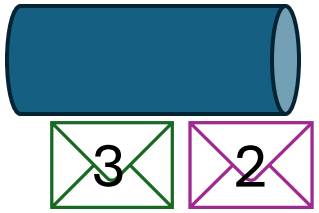
It releases the smallest ts' message and waits again

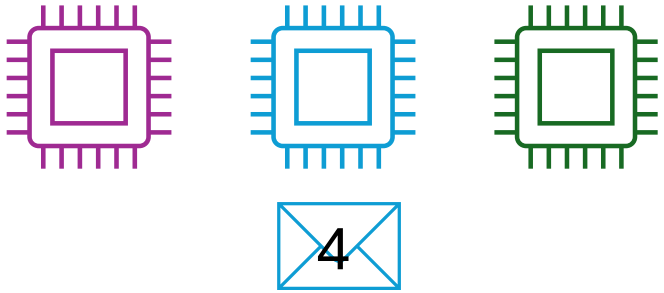


Clients/Participants

A sequencer

Clients attach message generation timestamps to outgoing messages





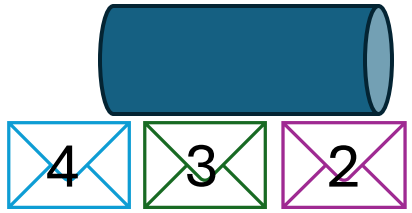
Clients/Participants

Sequencer waits for at least one message from each client

It releases the smallest ts' message and waits again

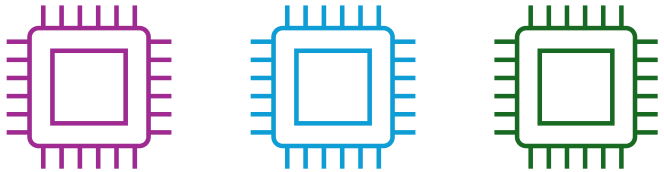
A sequencer

Clients attach message generation timestamps to outgoing messages



Sequencer waits for at least one message from each client

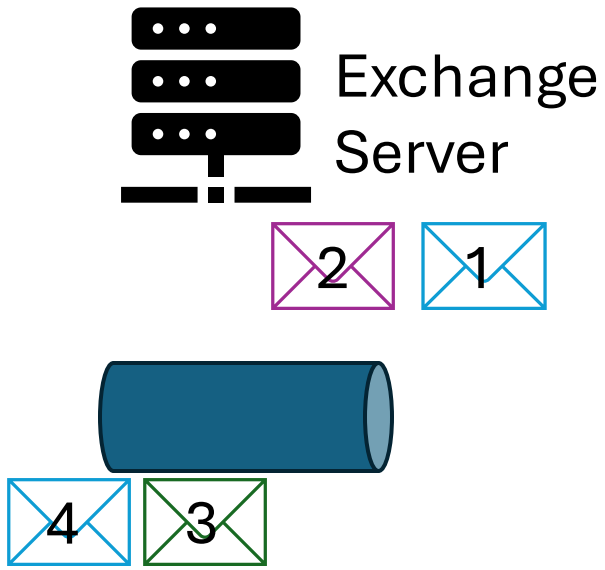
It releases the smallest ts' message and waits again



Clients/Participants

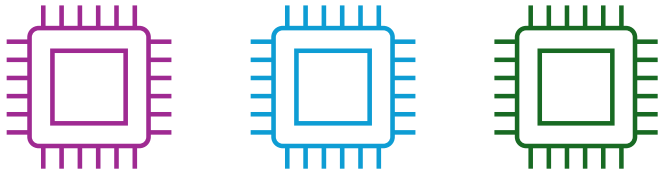
A sequencer

Clients attach message generation timestamps to outgoing messages



Sequencer waits for at least one message from each client

It releases the smallest ts' message and waits again



Clients/Participants

A sequencer

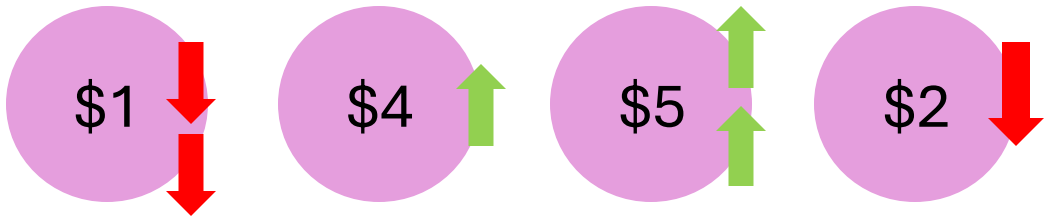
Drawback: liveness (progress) may halt under client failures

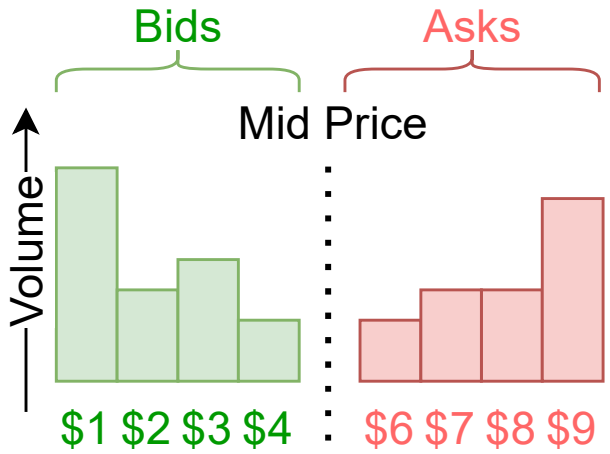
Safety (fairness) is guaranteed if clocks are perfectly synchronized

In practice, we ensure that clock synchronization accuracy is sufficiently high compared to the time granularity of interest.

Limit Order Queue

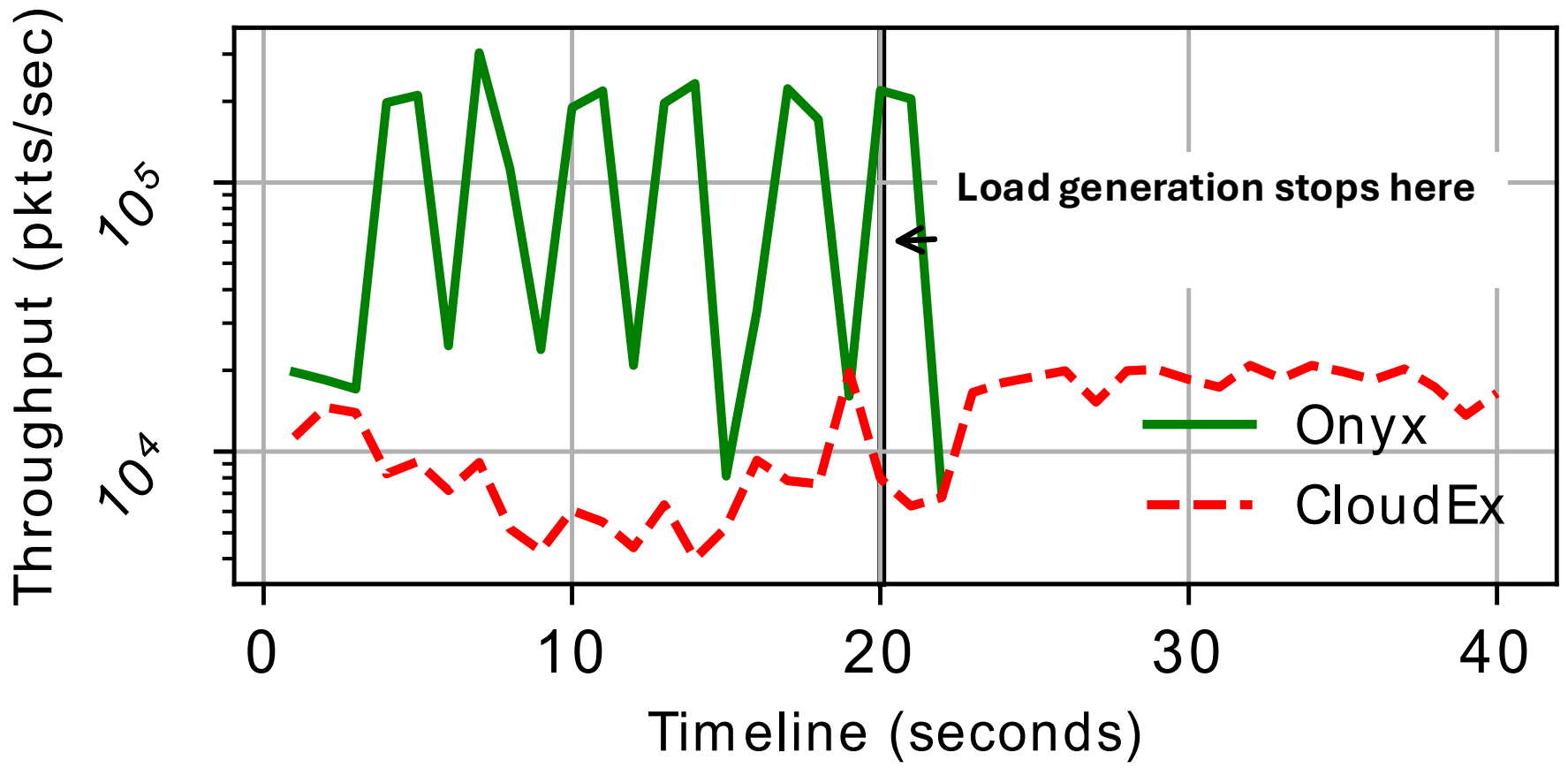
- During bursty market activity, the exchange server may be overwhelmed.
- LOQ schedules order in a way to avoid idling the matching engine, utilizing the knowledge that matching engine uses price time priority algorithm for matching orders



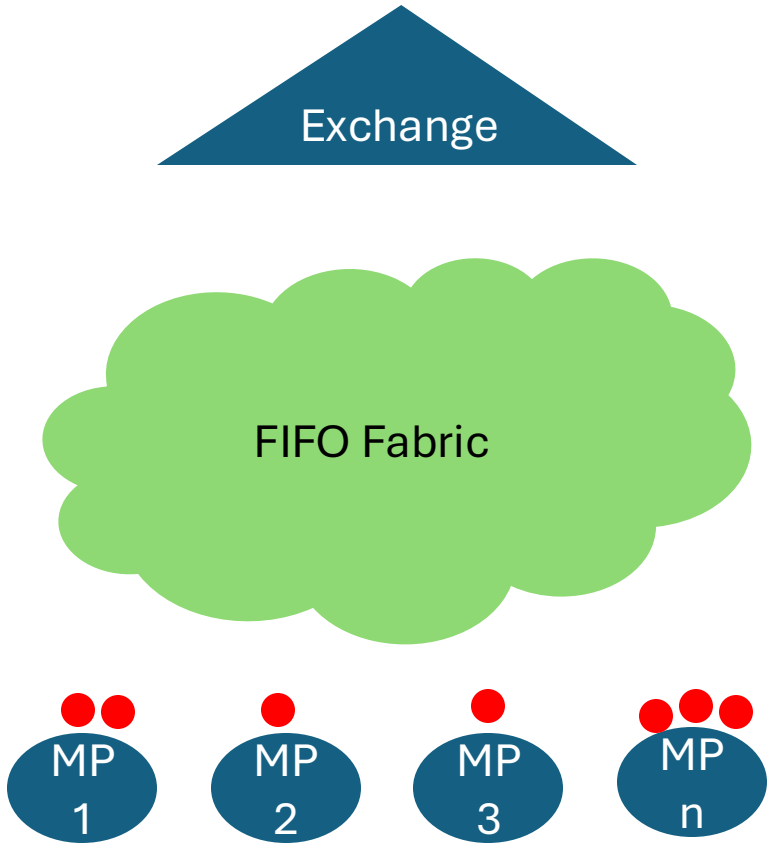


Naively doing such prioritization will not preserve inbound fairness.

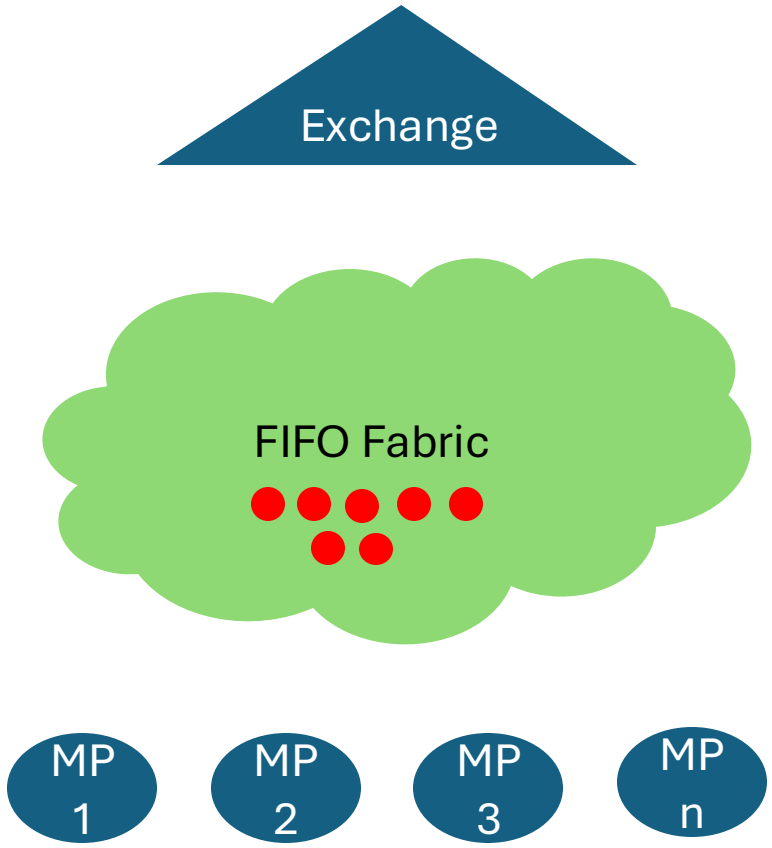
Limit Order Queue Gracefully Handles Bursts



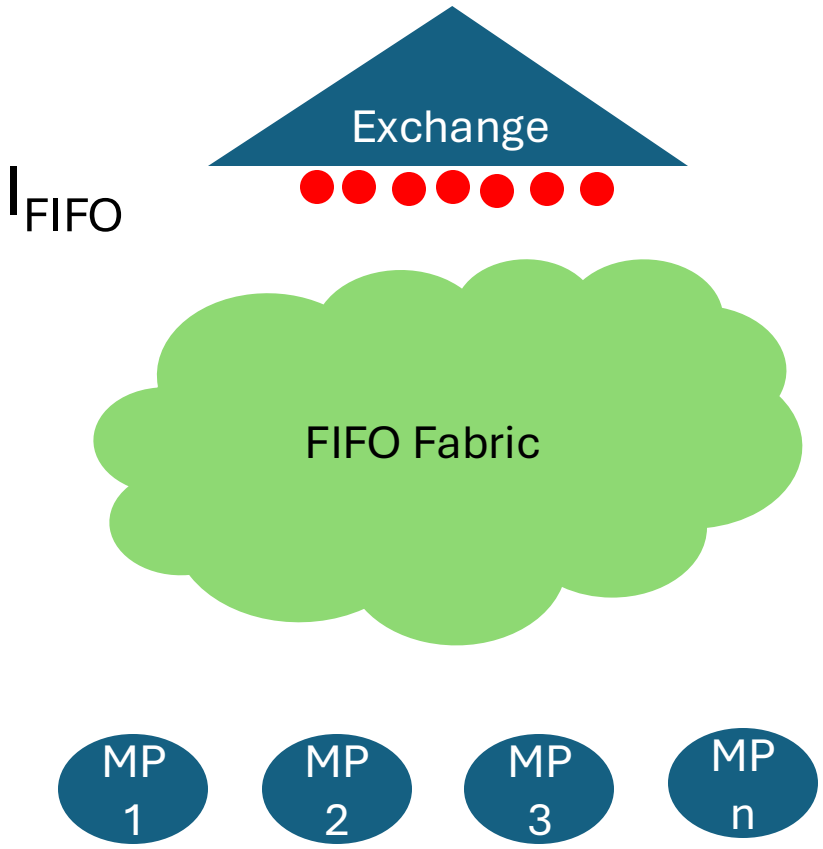
Limit Order Queue Preserves Fairness



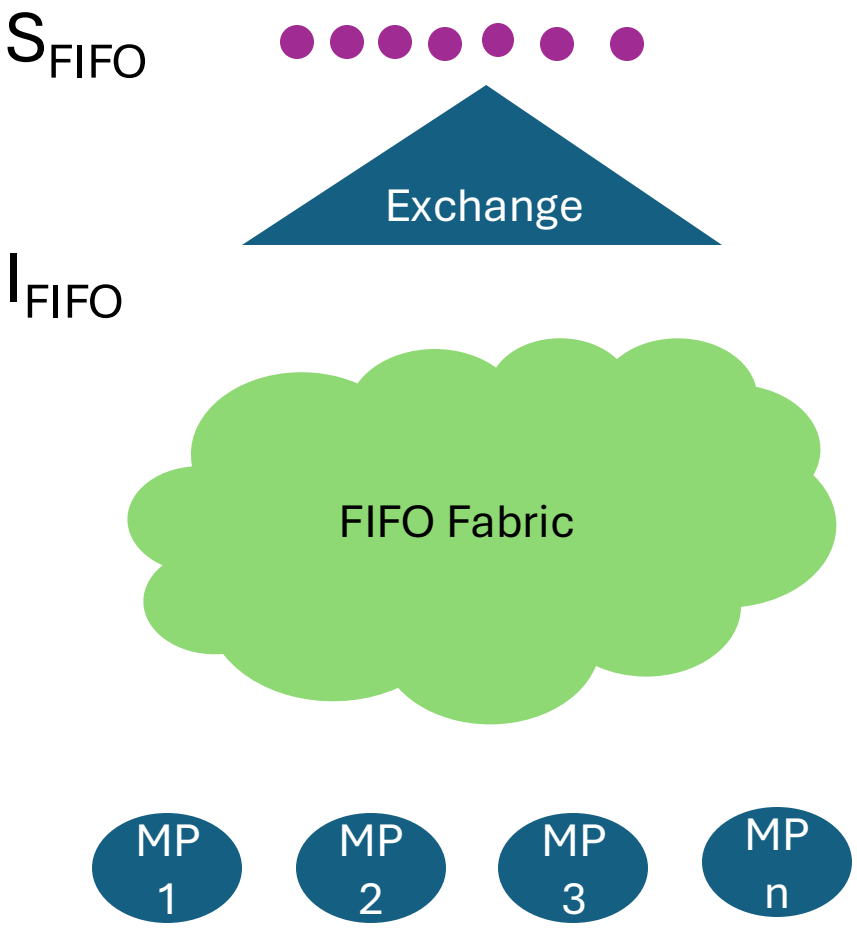
Limit Order Queue Preserves Fairness



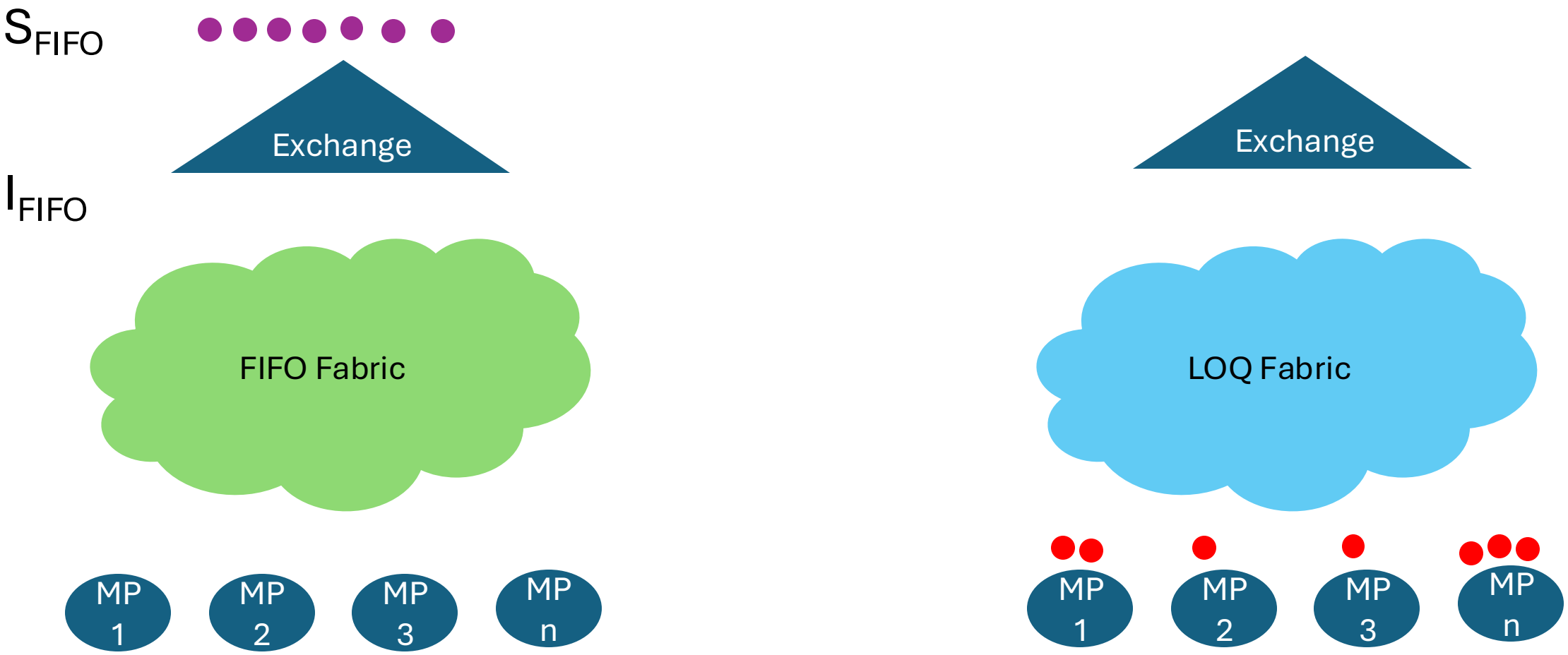
Limit Order Queue Preserves Fairness



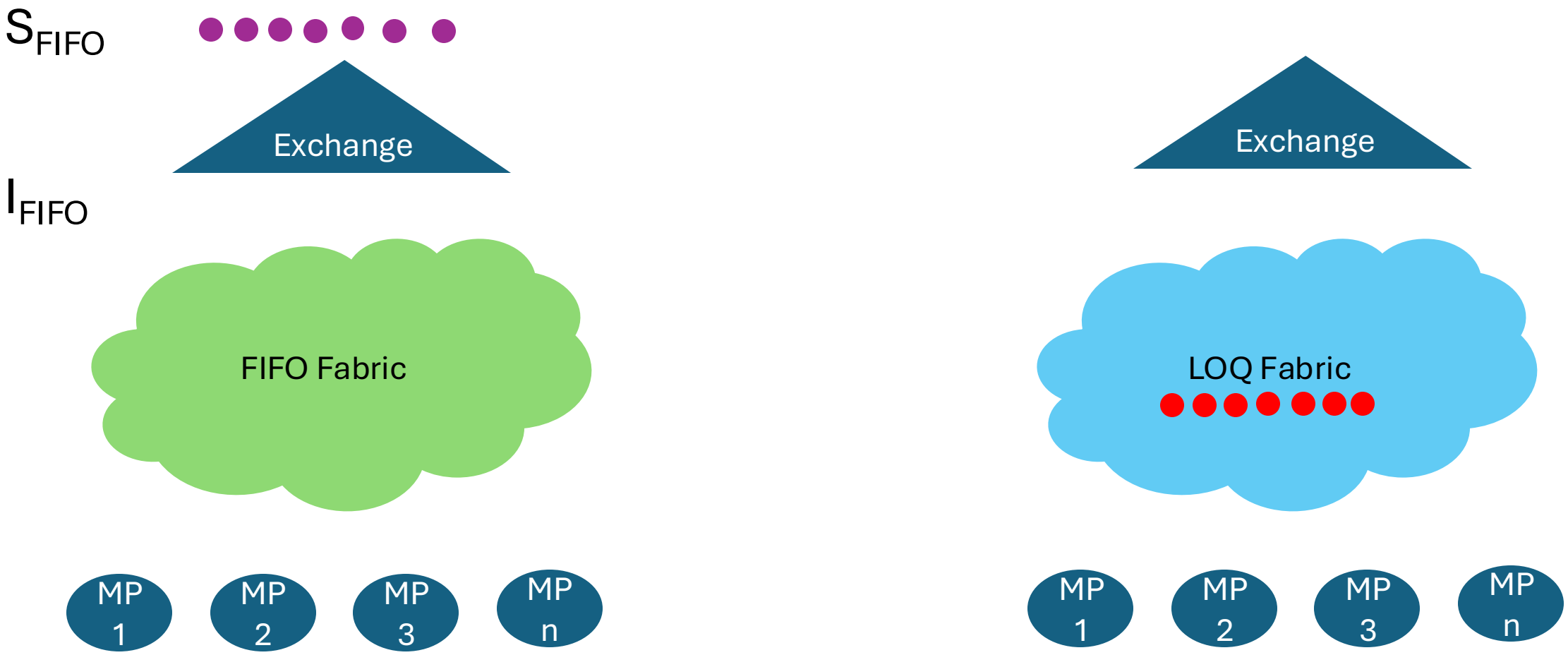
Limit Order Queue Preserves Fairness



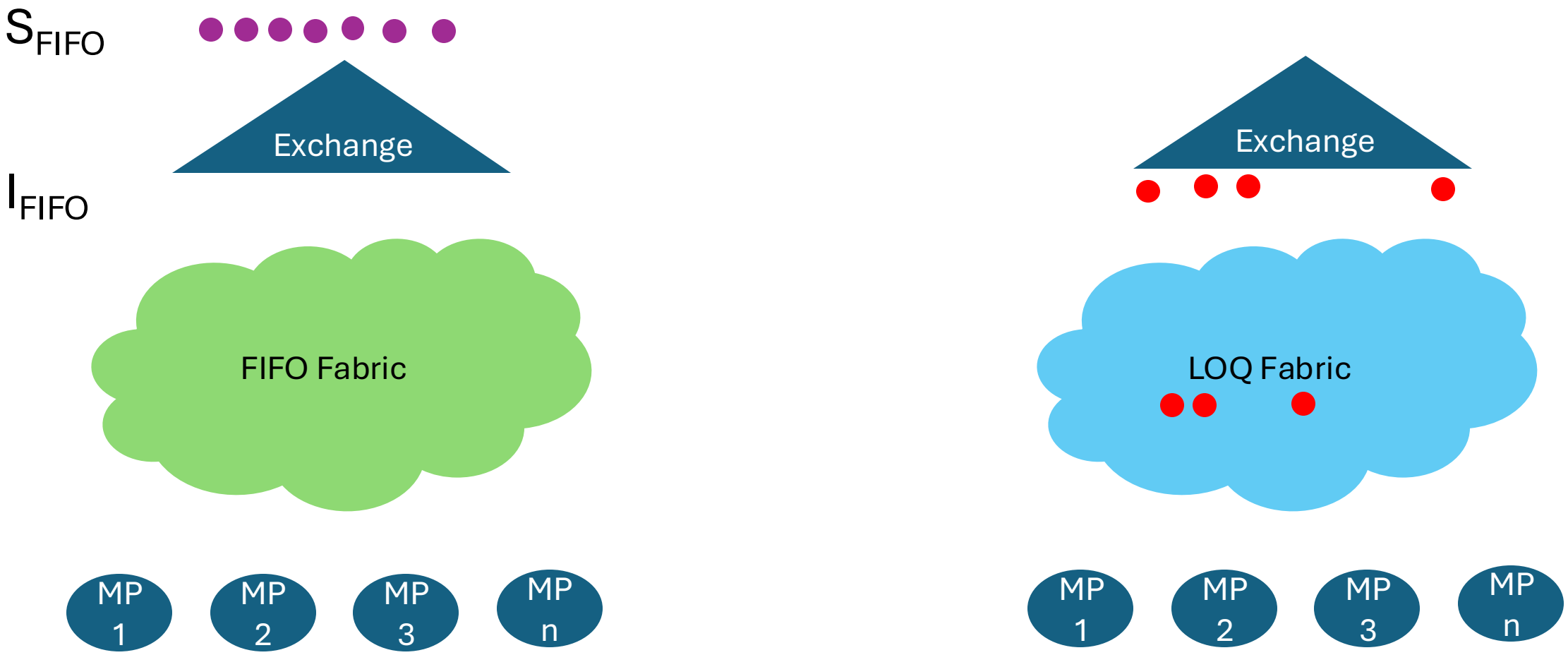
Limit Order Queue Preserves Fairness



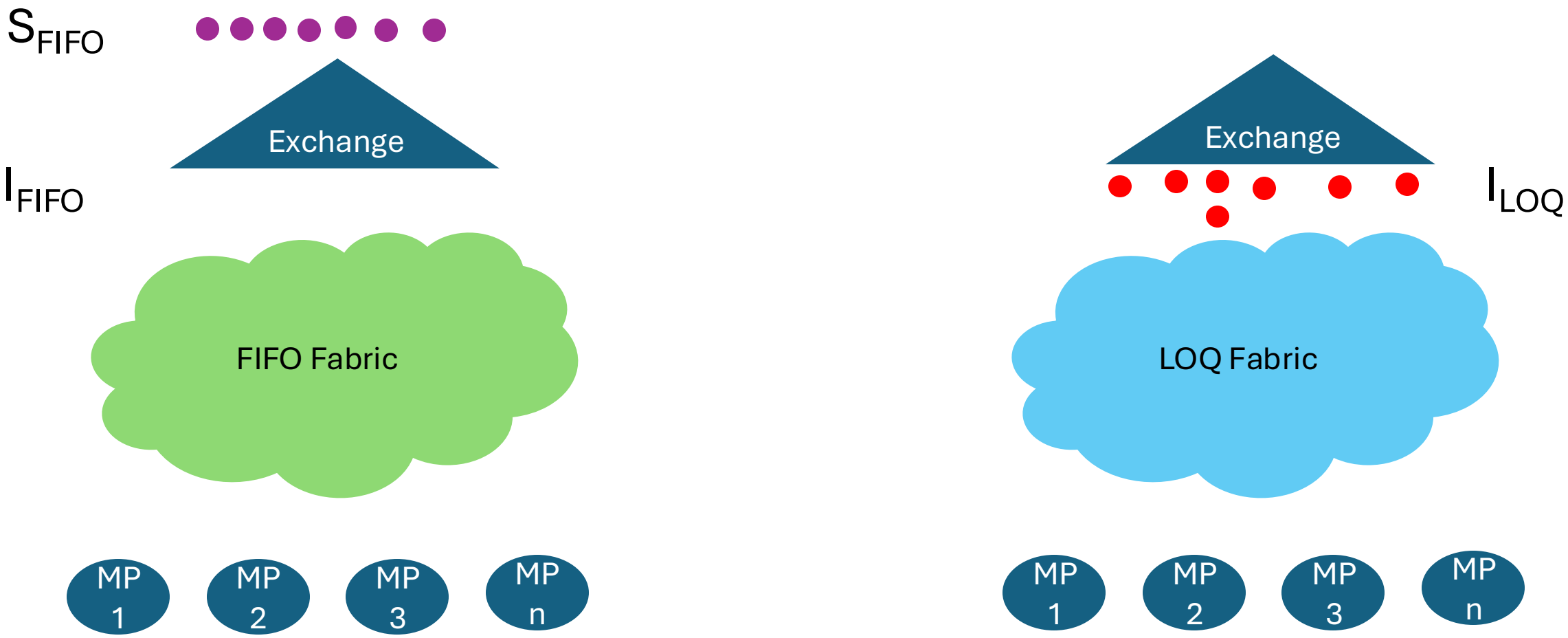
Limit Order Queue Preserves Fairness



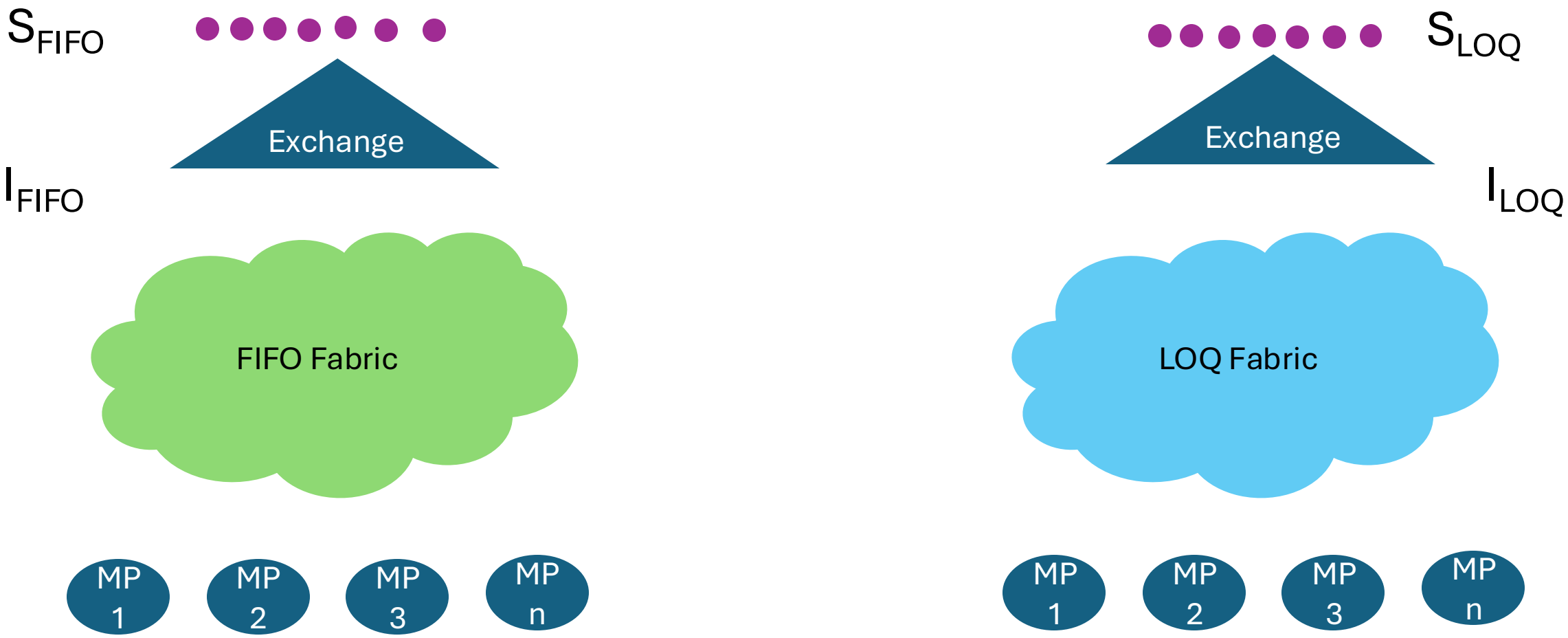
Limit Order Queue Preserves Fairness



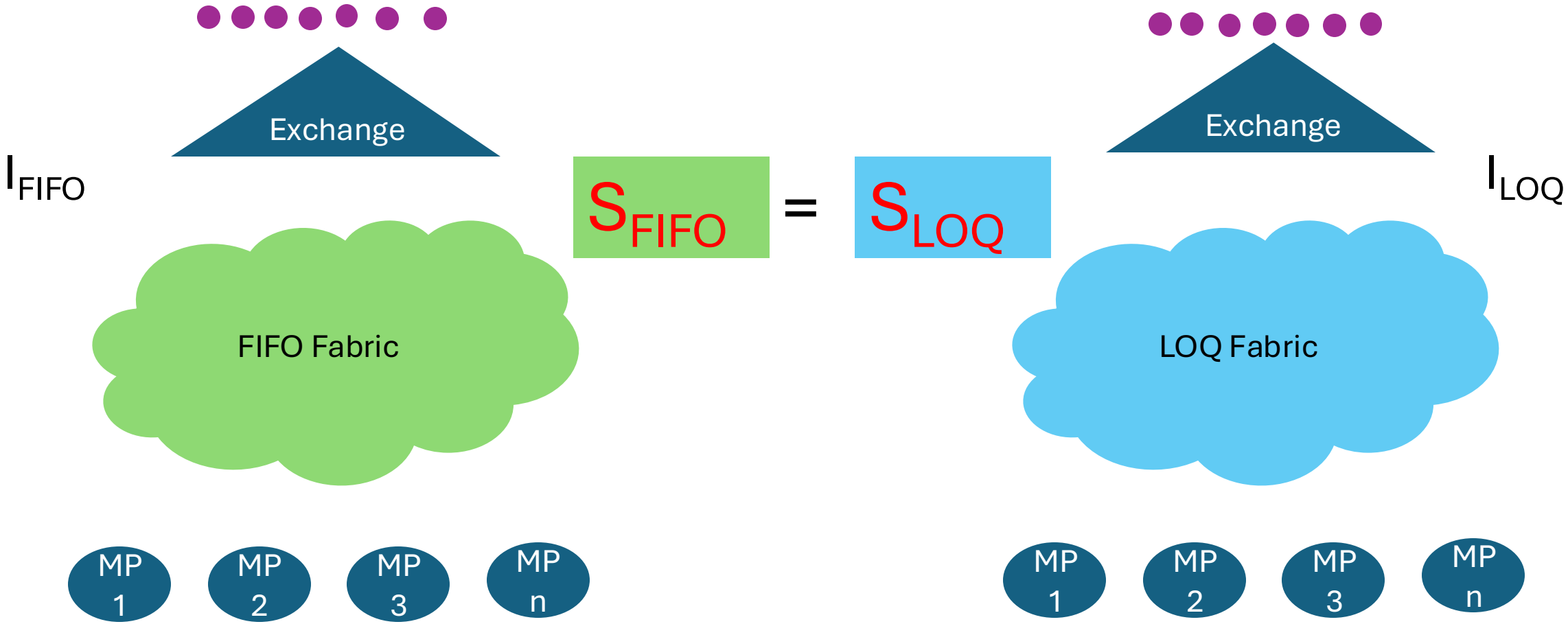
Limit Order Queue Preserves Fairness



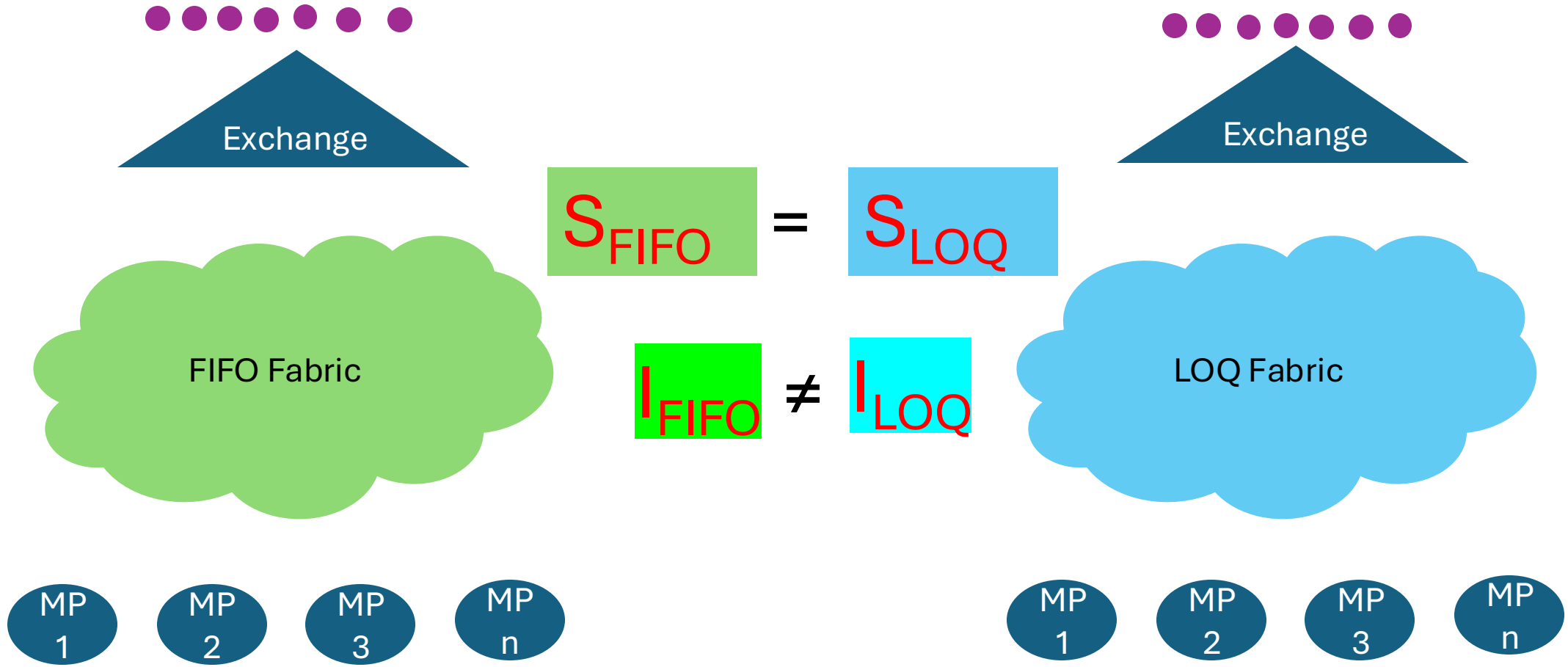
Limit Order Queue Preserves Fairness



Limit Order Queue Preserves Fairness

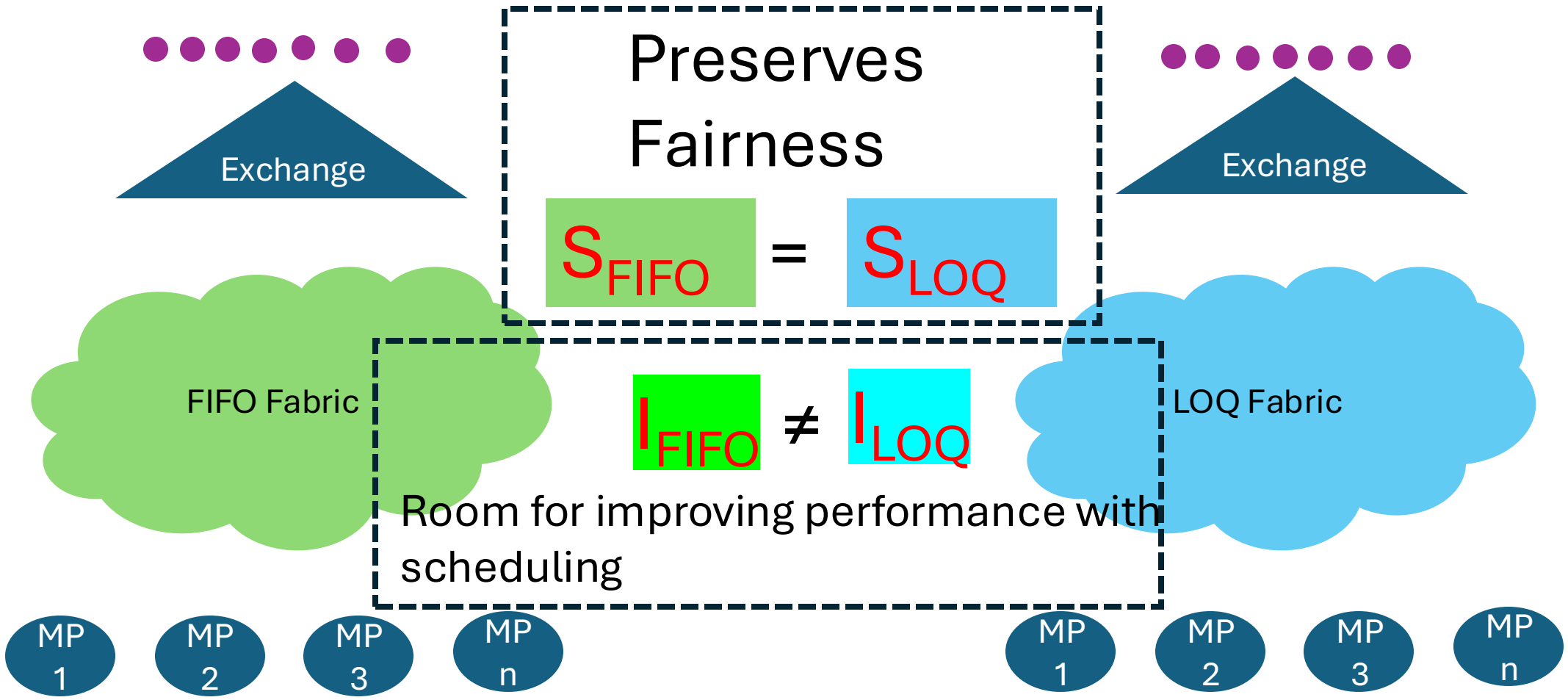


Limit Order Queue Preserves Fairness



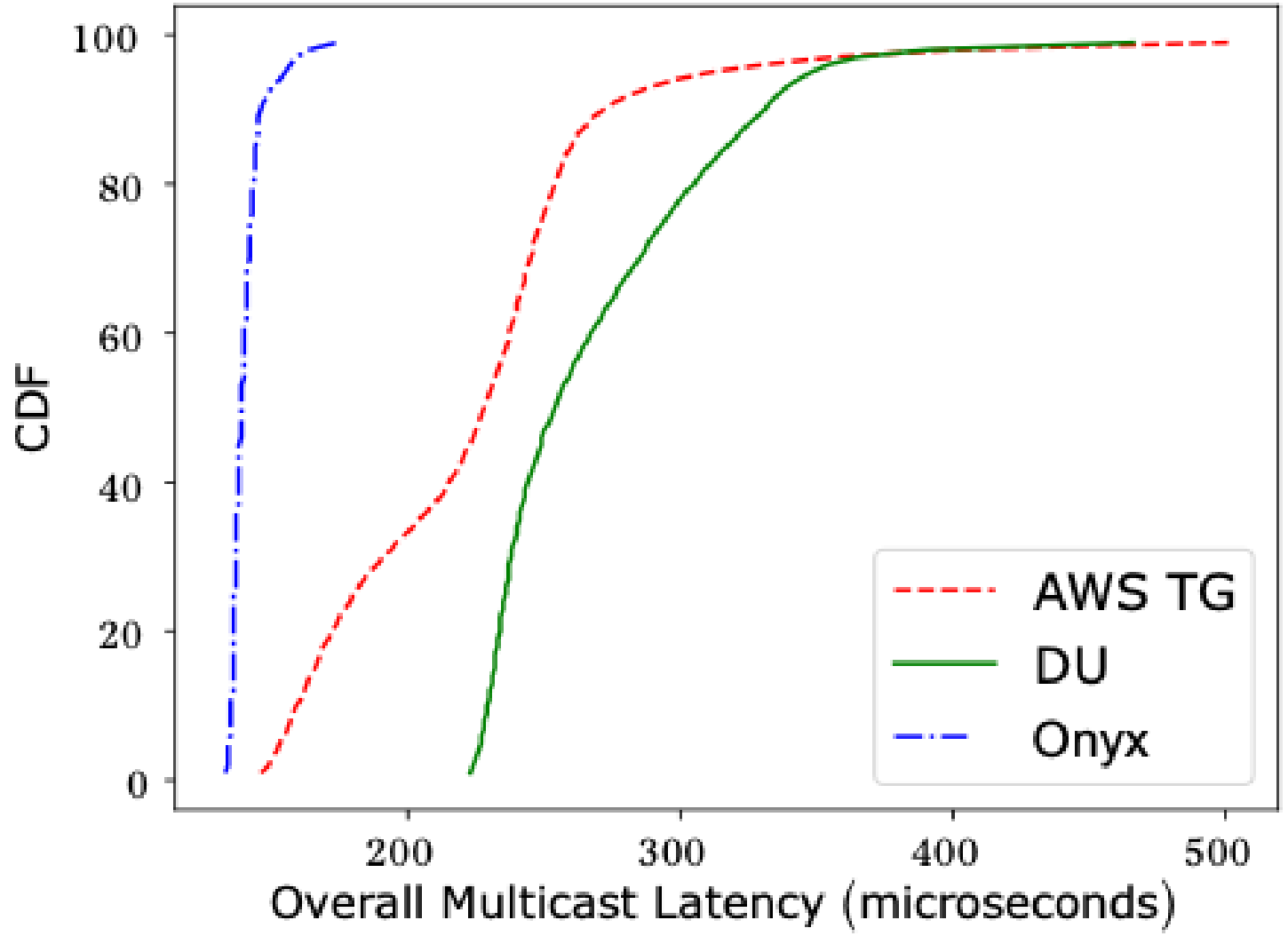
Limit Order Queue Preserves Fairness

Limit Order Queue Preserves Fairness

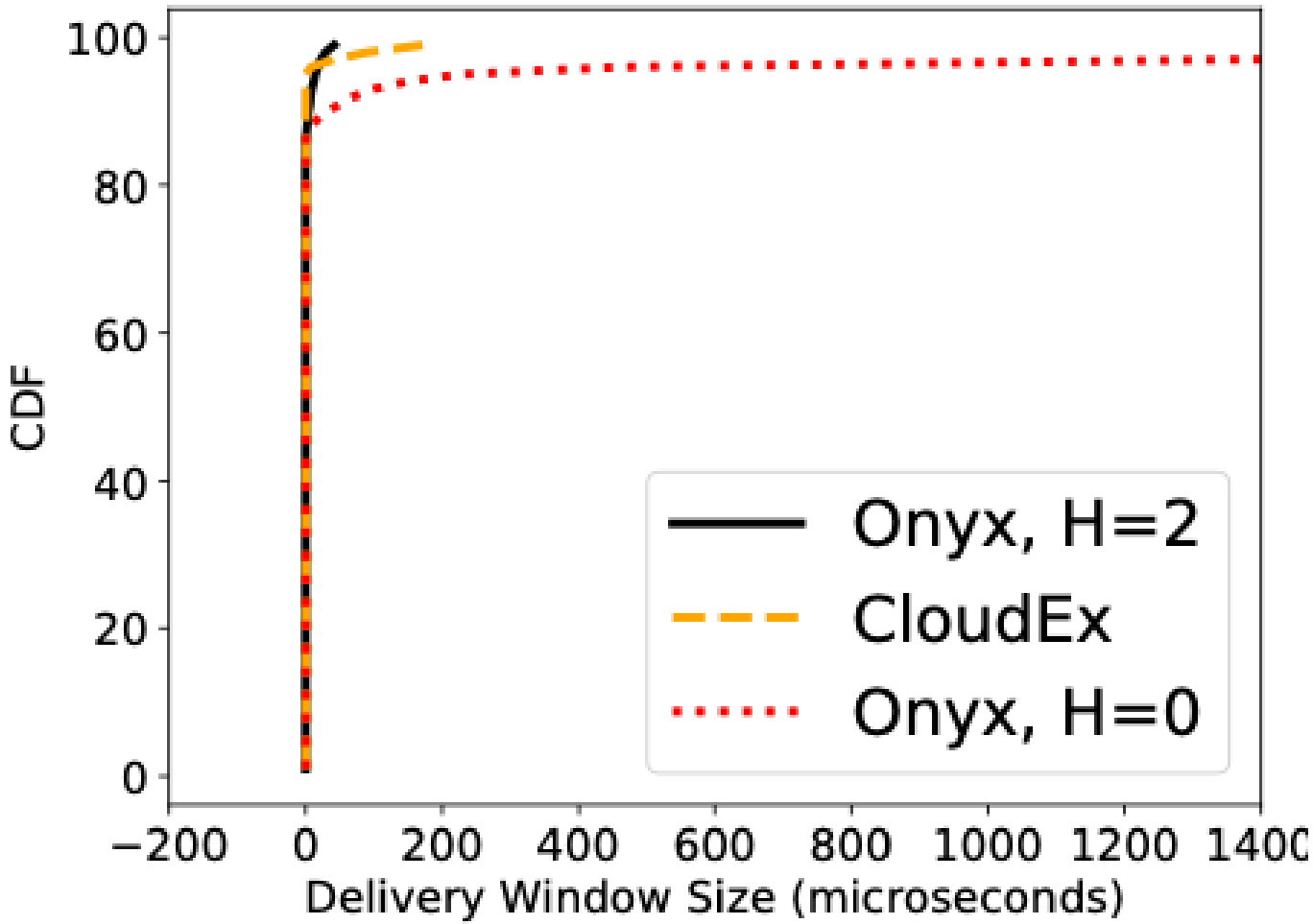


Evaluation

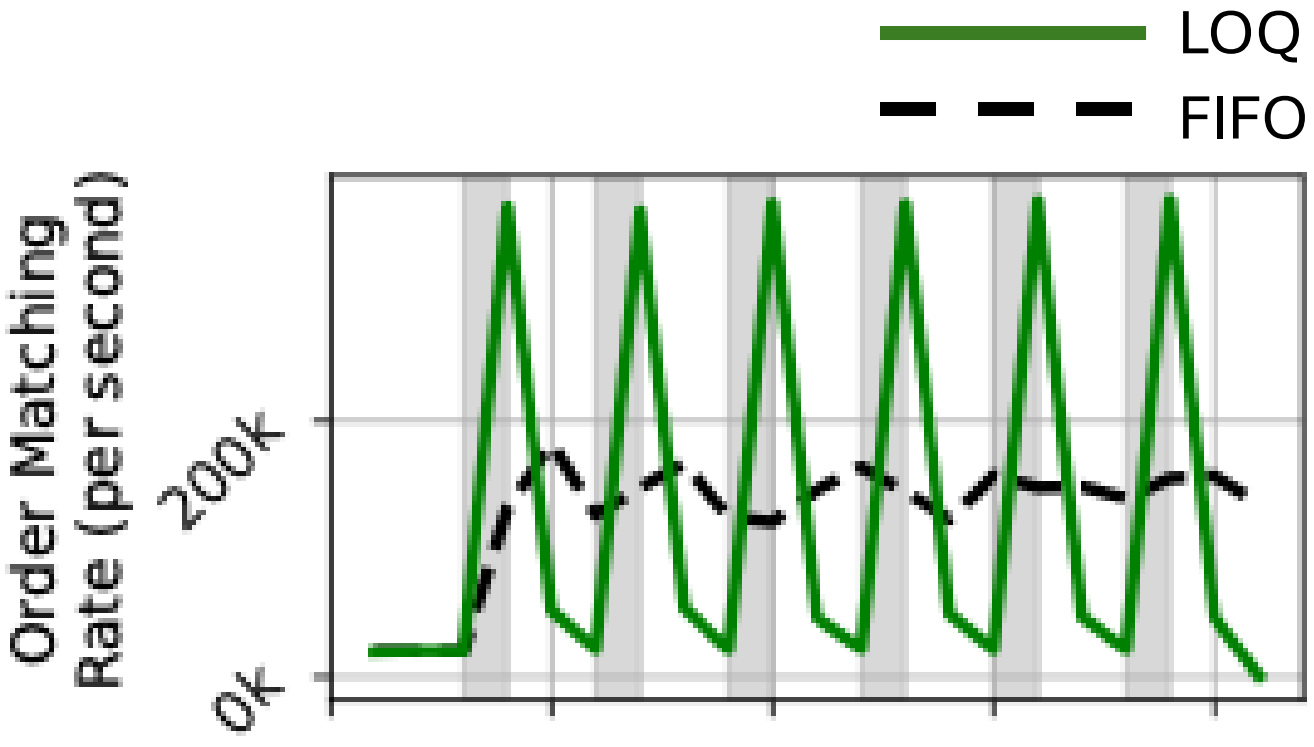
Multicast Service



Multicast Service



Order Submission Service



Concluding Remarks

- Cloud financial exchanges can be realized by developing new primitives, without specialized infrastructure
  - Some fairness/performance guarantees need to be accordingly relaxed

- Such exchanges present a new operating point in the cost-performance curve