ePulsar
Control Plane for Publish-Subscribe Systems on Geo-Distributed Edge Infrastructure

Harshit Gupta, Tyler Landle, and
Dr. Umakishore Ramachandran

Georgia Tech College of Computing
School of Computer Science

Incorporating network proximity at scale for latency-sensitive broker selection.

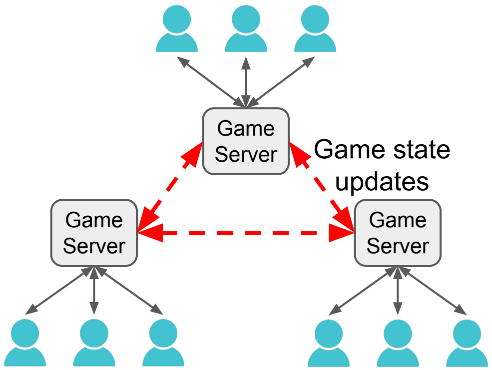
Talk Outline

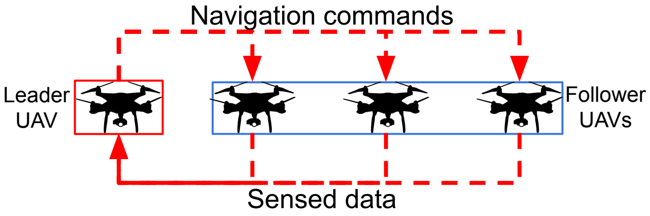
1. **Background**
2. Problem and Challenges
3. Design Principles of ePulsar
4. Architecture
   a. Network Proximity Estimation
   b. Distributed Monitoring
5. Implementation
6. Evaluations
7. Conclusion

Emerging distributed applications need publish-subscribe

- Apps with multiple distributed components
  - Massively Multiplayer Online Games (MMOG)
  - UAV Swarm coordination
  - Collaborative Perception for vehicles

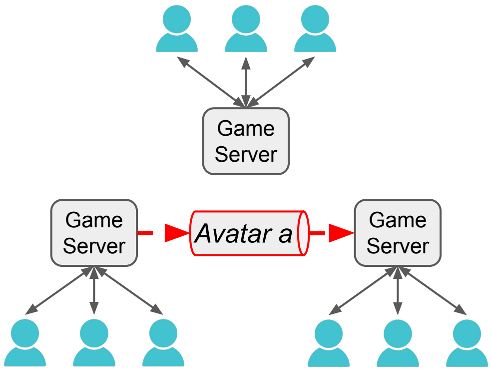
- Share sensed data, state-updates

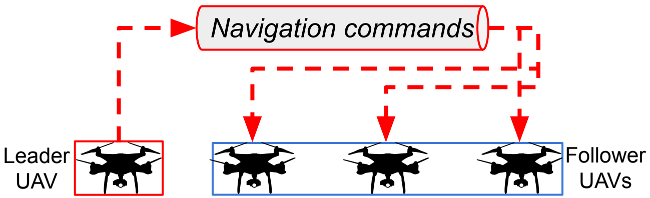




Emerging distributed applications need publish-subscribe

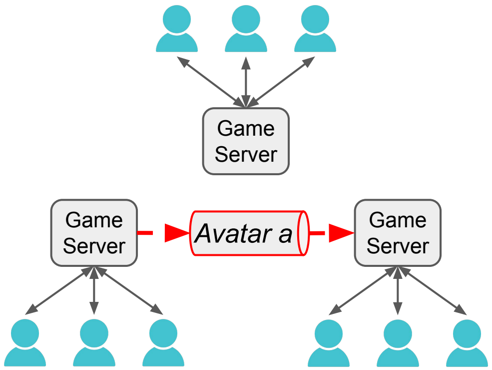
- Apps with multiple distributed components
- Share sensed data, state-updates
- Publish-subscribe model is a suitable abstraction
  - Decouples data ***Producers*** and ***Consumers***
  - Communication through ***Topics***

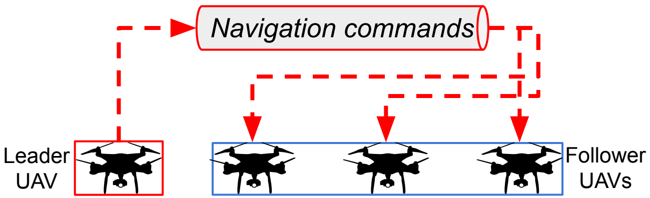


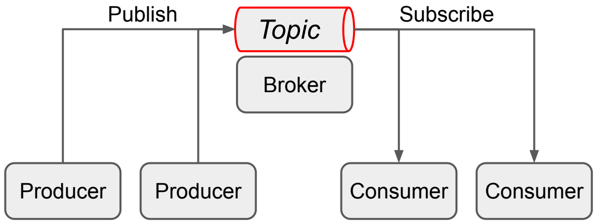


Emerging distributed applications need publish-subscribe

- Apps with multiple distributed components
- Share sensed data, state-updates
- Publish-subscribe model is a suitable abstraction
  - Decouples data ***Producers*** and ***Consumers***
  - Communication through ***Topics***
  - ***Topics*** hosted by ***Broker*** nodes



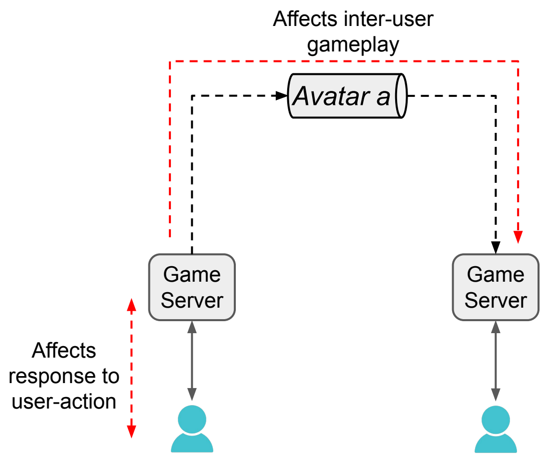




Low latency requirement

- Communication latency affects functionality

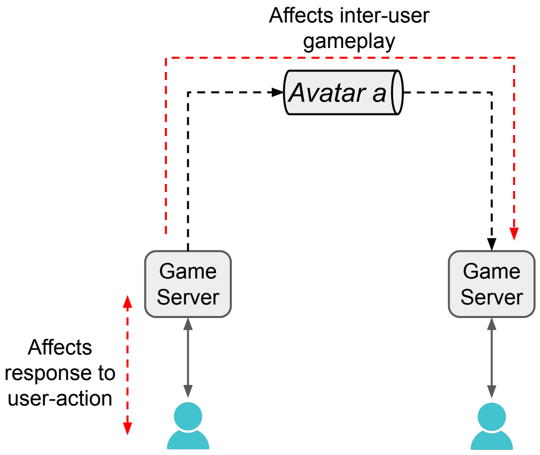
- Stringent latency requirements
    - UAV Swarm coordination: < 40 ms [1]
    - MMOG: < 100ms GS-to-GS [2]



[1]Massive MIMO for Connectivity With Drones: Case Studies and Future Directions
[2] Lag Compensation for First-Person Shooter Games in Cloud Gaming

Low latency requirement

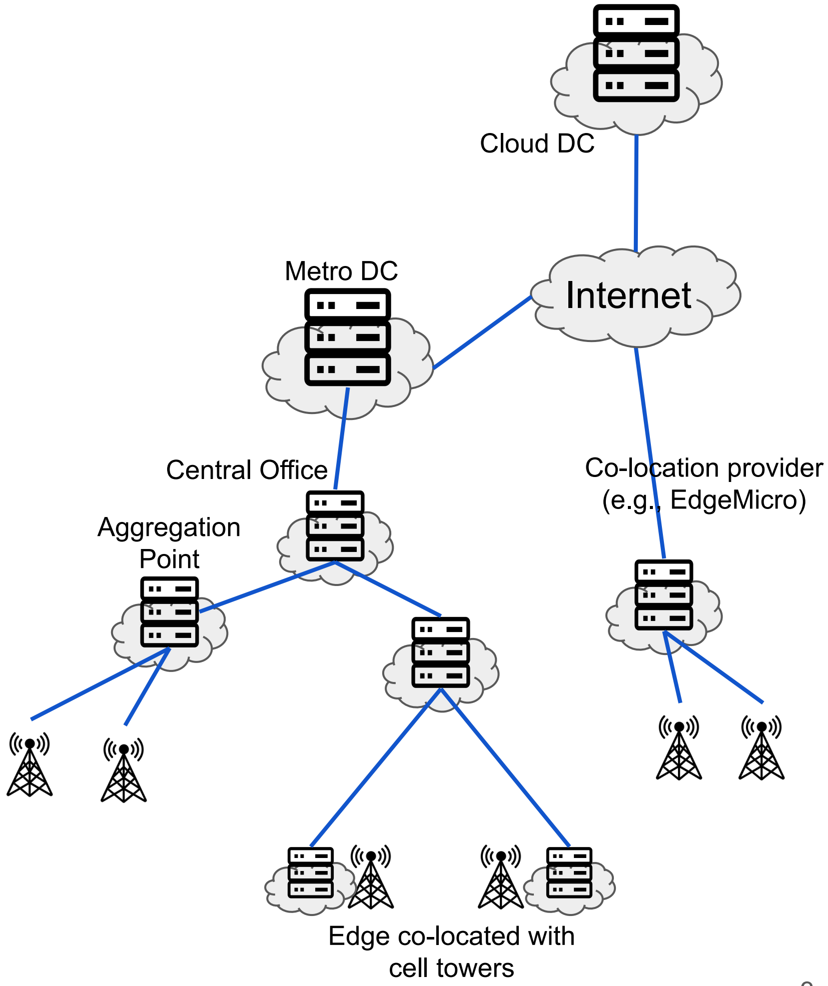
- Communication latency affects functionality
- Stringent latency requirements
  - UAV Swarm coordination: < 40 ms [1]
  - MMOG: < 100ms GS-to-GS [2]
- Cloud-based Pub-Sub systems
  - Offer strong data semantics, but
  - High end-to-end latency due to Wide Area Network



[1]Massive MIMO for Connectivity With Drones: Case Studies and Future Directions
[2] Lag Compensation for First-Person Shooter Games in Cloud Gaming

Edge infrastructure

- Edge-Cloud continuum
  - Multiple providers
  - Multi-city geographical coverage
  - Inter-edge network latency

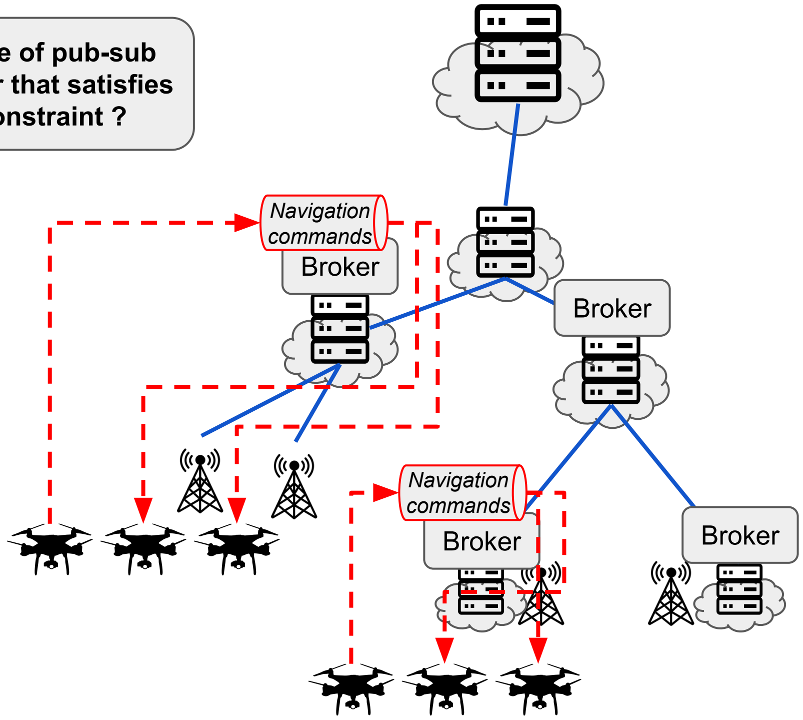


Talk Outline

1. Background
2. **Problem and Challenges**
3. Design Principles of ePulsar
4. Architecture
    a. Network Proximity Estimation
    b. Distributed Monitoring
5. Implementation
6. Evaluations
7. Conclusion

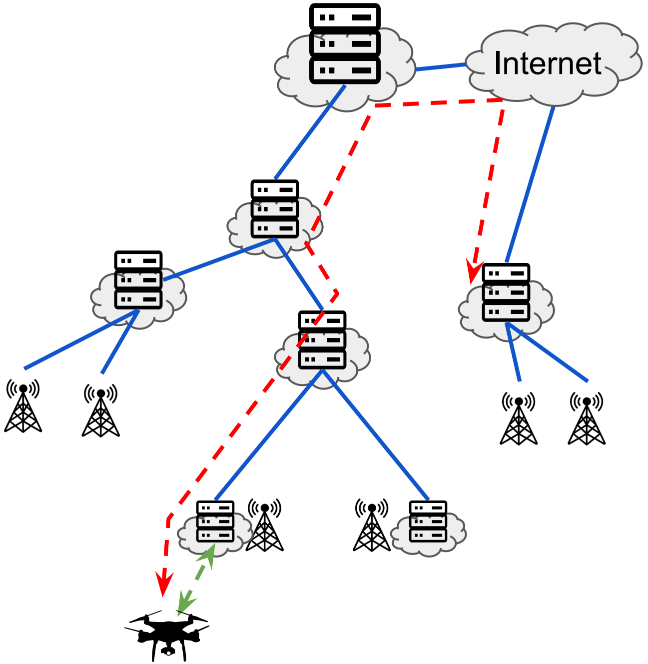
Problem: Serving pub-sub latency requirement of apps

How to aid control-plane of pub-sub system to select a broker that satisfies end-to-end latency constraint ?



Challenges in operating a geo-distributed pub-sub system

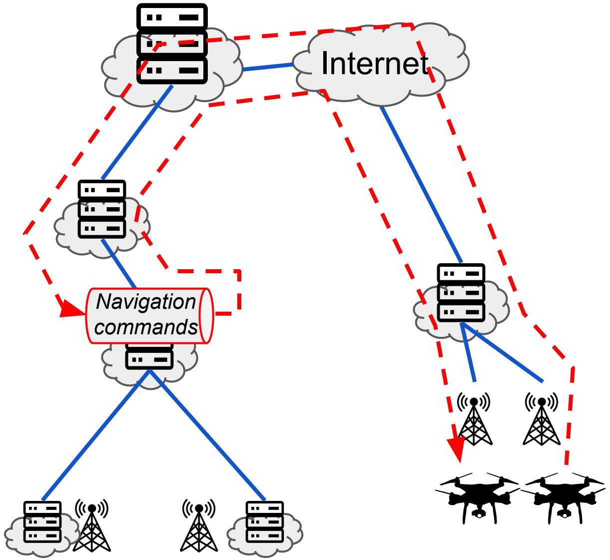
- Topology awareness
  - Edge network topology is highly heterogeneous
  - Latency variation
  - Dense geo-distribution



Challenges in operating a geo-distributed pub-sub system

- Topology awareness
  - Edge network topology is highly heterogeneous
  - Latency variation
  - Dense geo-distribution

- Client mobility
  - Publish-subscribe latency violation



Using cloud-based pub-sub systems on the Edge

- E.g., Apache Pulsar, Apache Kafka

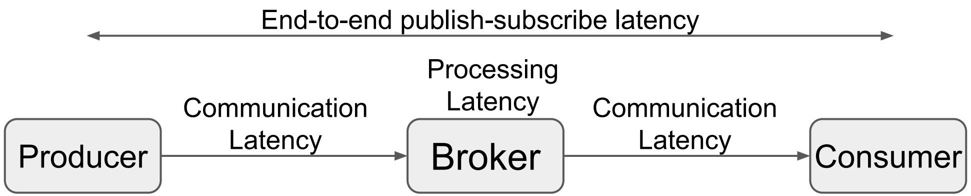
- Control-plane designed for datacenter workloads
    - Focus on even workload distribution, not end-to-end latency
    - Don’t consider high client-edge communication latencies

- Need to provide latency-awareness to broker selection

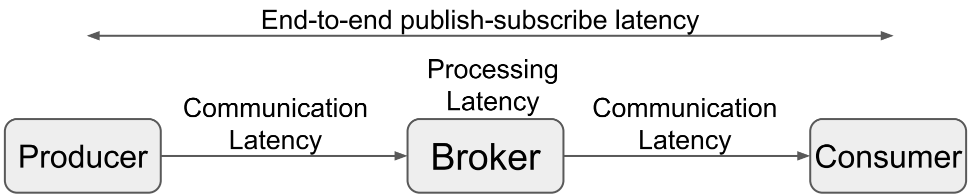
Talk Outline

1. Background
2. Problem and Challenges
3. **Design Principles of ePulsar**
4. Architecture
   a. Network Proximity Estimation
   b. Distributed Monitoring
5. Implementation
6. Evaluations
7. Conclusion

Design principles of ePulsar’s edge-ready control-plane

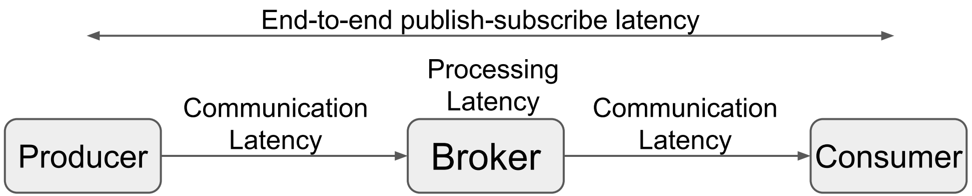


Design principles of ePulsar’s edge-ready control-plane



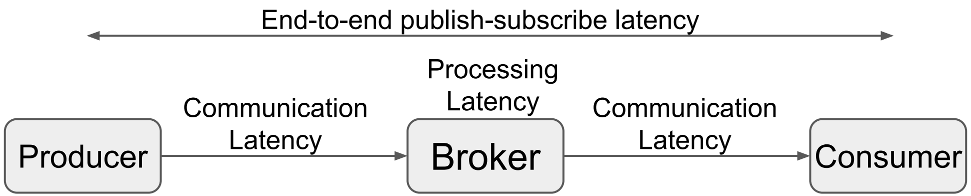
- Scalable Network Proximity Estimation → communication latency

Design principles of ePulsar's edge-ready control-plane



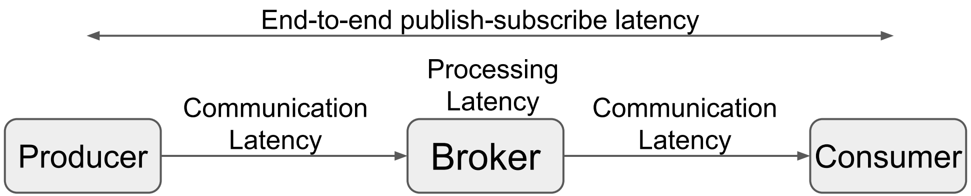
- Scalable Network Proximity Estimation → communication latency
- Distributed Monitoring → reduce monitoring overhead

Design principles of ePulsar’s edge-ready control-plane



- Scalable Network Proximity Estimation → communication latency
- Distributed Monitoring → reduce monitoring overhead
- Agile Reconfiguration → efficiently handling client mobility

Design principles of ePulsar’s edge-ready control-plane



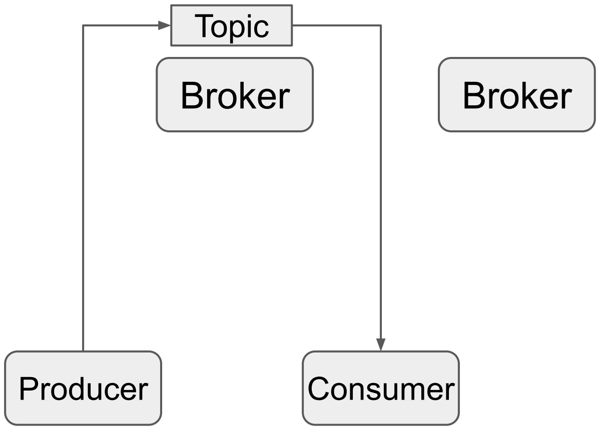
- Scalable Network Proximity Estimation → communication latency
- Distributed Monitoring → reduce monitoring overhead
- Agile Reconfiguration → efficiently handling client mobility

Talk Outline

1. Background
2. Problem and Challenges
3. Design Principles of ePulsar
4. **Architecture**
    a. Network Proximity Estimation
    b. Distributed Monitoring
5. Implementation
6. Evaluations
7. Conclusion

High-level architecture of ePulsar

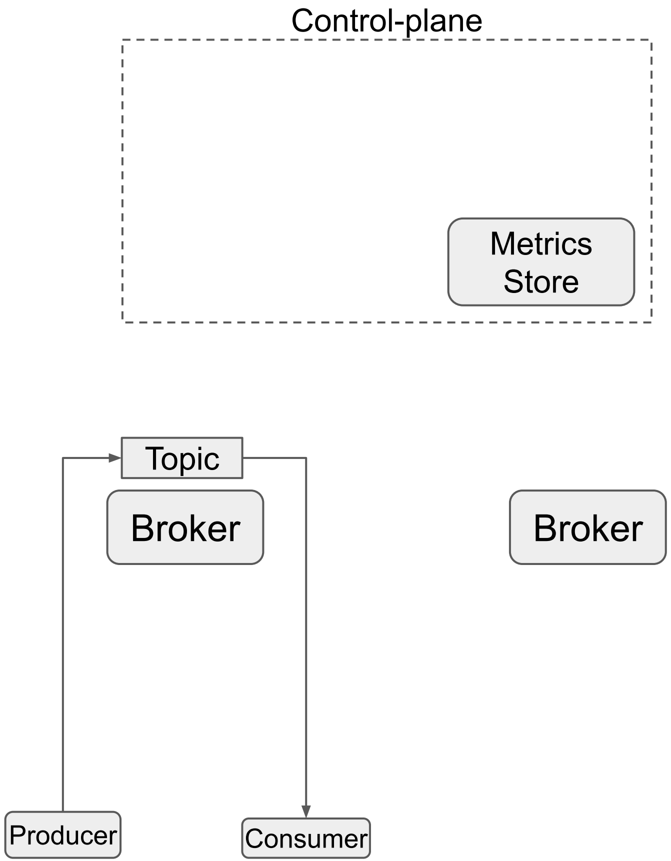
- Geo-distributed Broker nodes
  - Host Topics
- Producers and Consumers share data through topics



High-level architecture of ePulsar

- Geo-distributed Broker nodes
  - Host Topics
- Producers and Consumers share data through topics

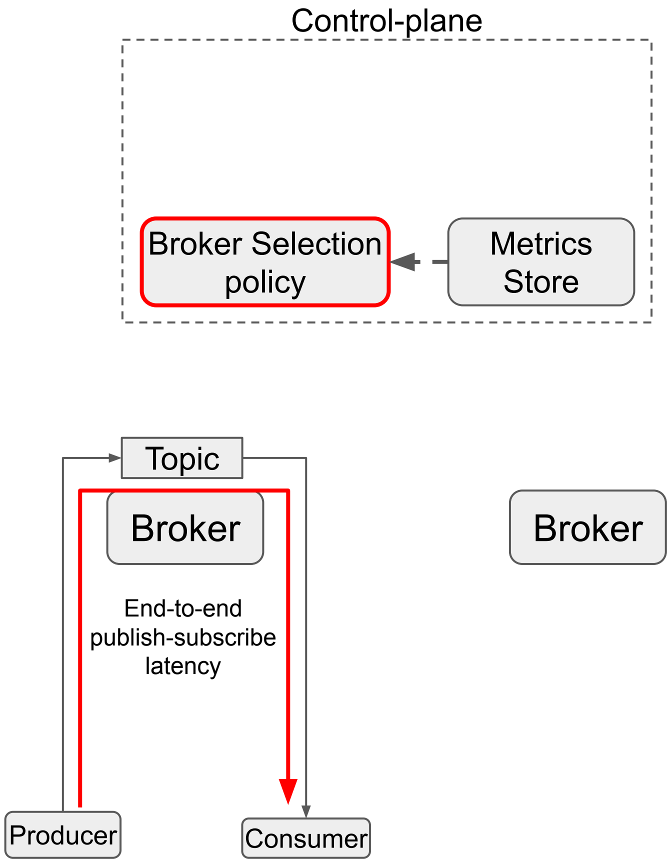
- Components of control-plane
  - **Metrics Store** for storing monitoring data



High-level architecture of ePulsar

- Geo-distributed Broker nodes
  - Host Topics
- Producers and Consumers share data through topics

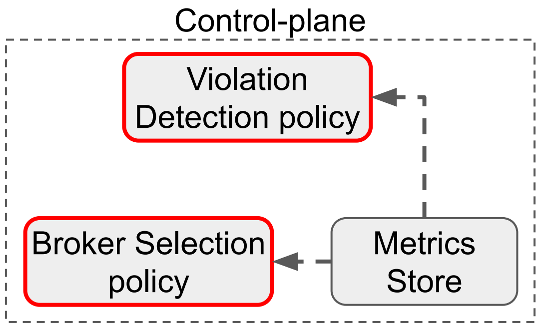
- Components of control-plane
  - **Metrics Store** for storing monitoring data
  - **Broker Selection Policy**
    Latency-aware Topic → Broker mapping

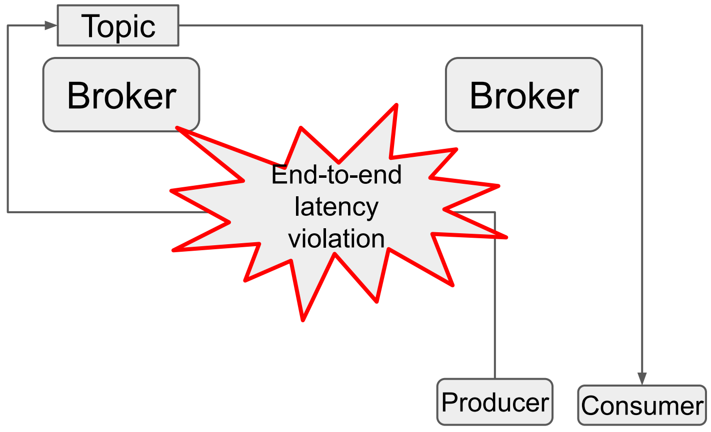


High-level architecture of ePulsar

- Geo-distributed Broker nodes
  - Host Topics
- Producers and Consumers share data through topics

- Components of control-plane
  - **Metrics Store** for storing monitoring data
  - **Broker Selection Policy**
    Latency-aware Topic → Broker mapping
  - **Violation Detection Policy**

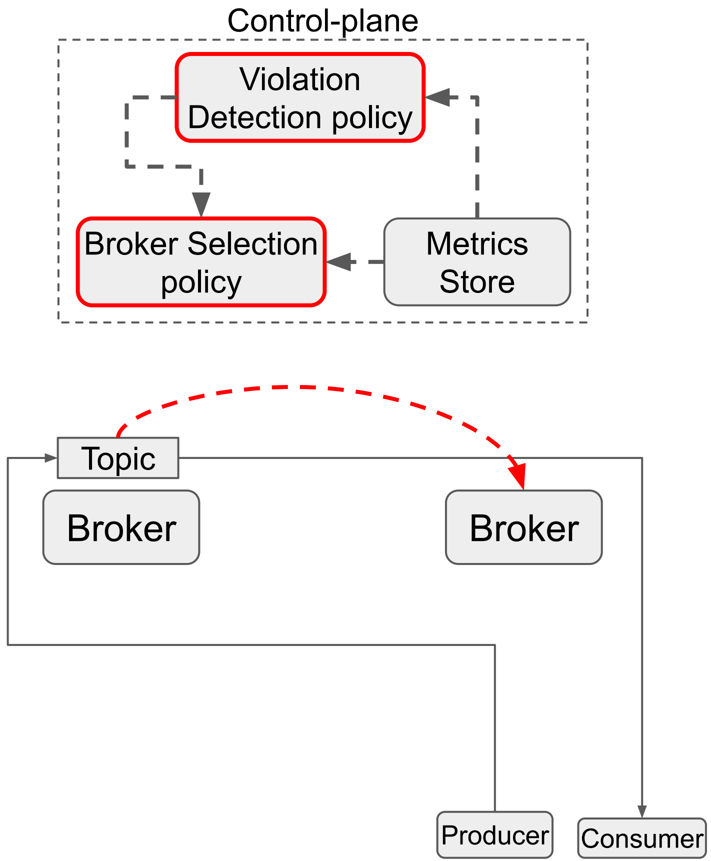




High-level architecture of ePulsar

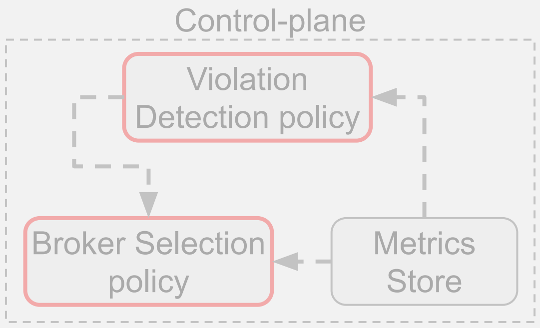
- Geo-distributed Broker nodes
  - Host Topics
- Producers and Consumers share data through topics

- Centralized control-plane
  - **Metrics Store** for storing monitoring data
  - **Broker Selection Policy**
    Latency-aware Topic → Broker mapping
  - **Violation Detection Policy**
    → Triggers topic migration



High-level architecture of ePulsar

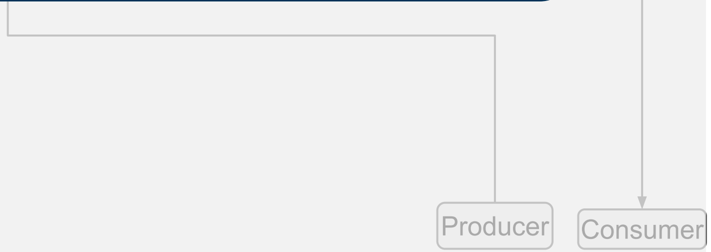
- Geo-distributed Broker nodes
  - Host Topics
- Producers and Consumers share data through topics



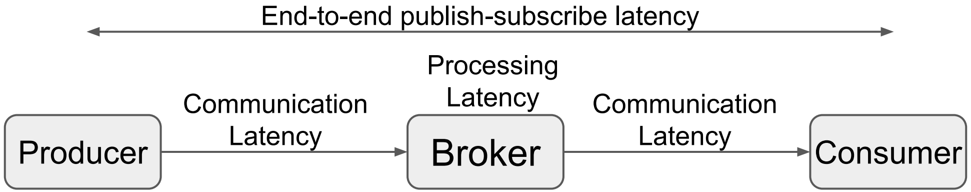
- Ce

End-to-end latency estimation is at the core of control-plane policies.

  - **Broker Selection Policy**
    Latency-aware Topic → Broker mapping
  - **Violation Detection Policy**
    → Triggers topic migration



Estimating end-to-end publish-subscribe latency



- Network Proximity estimations for communication latency

- Message rate + offline profiling for processing latency
    - Khare et al. (SEC 2018)

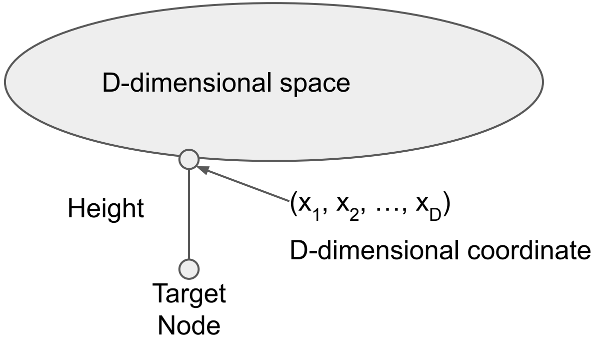
Khare, Shweta, et al. "Scalable edge computing for low latency data dissemination in topic-based publish/subscribe." *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2018.

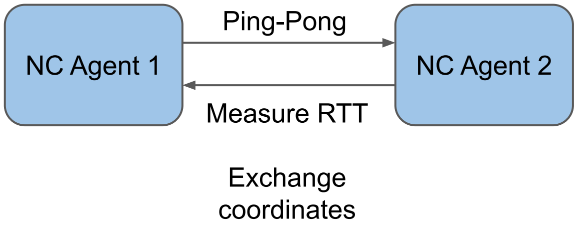
Network Proximity Estimation in ePulsar

- **Network Coordinates (NC)**
  - Arrange nodes in a Euclidean space
  - Euclidean distance b/w nodes equals RTT

- Network Coordinate (NC) Agents

- Decentralized P2P protocol [1]

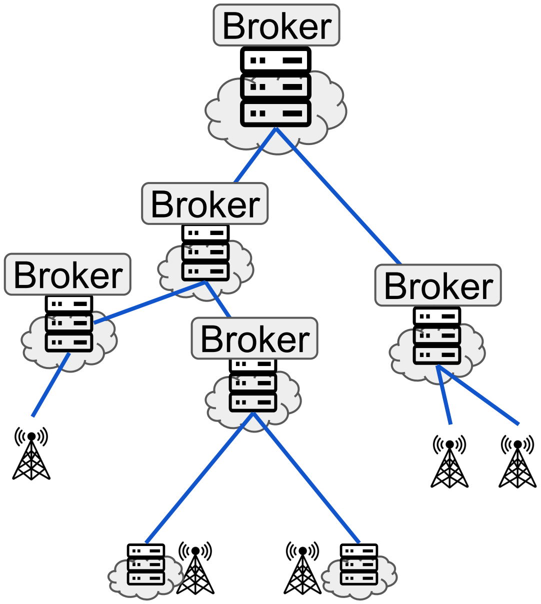




If $|\, d(NC_1, NC_2) - measured\_rtt \,| > error$
Update NC_1

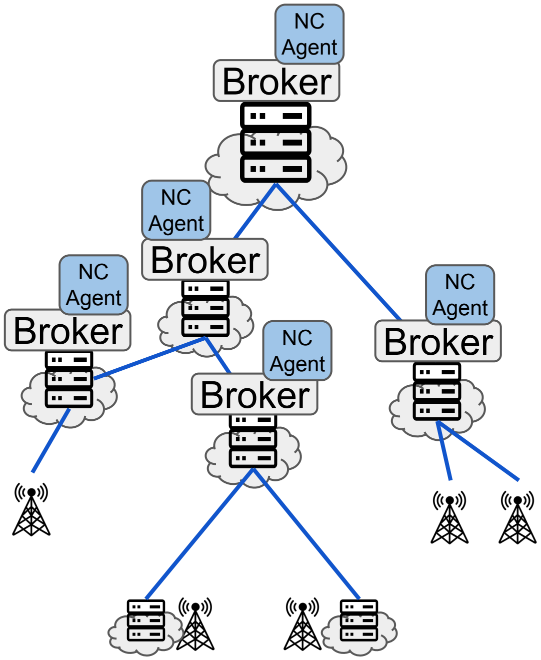
[1] Dabek, Frank, et al. "Vivaldi: A decentralized network coordinate system." ACM SIGCOMM Computer Communication Review 34.4 (2004): 15-26.
[2] Ledlie, Jonathan, Paul Gardner, and Margo I. Seltzer. "Network Coordinates in the Wild." NSDI. Vol. 7. 2007.
[3] Lee, Sanghwan, et al. "On suitability of euclidean embedding for host-based network coordinate systems." IEEE/ACM Transactions on Networking 18.1 (2009): 27-40.

Deployment of Network Coordinate Agents



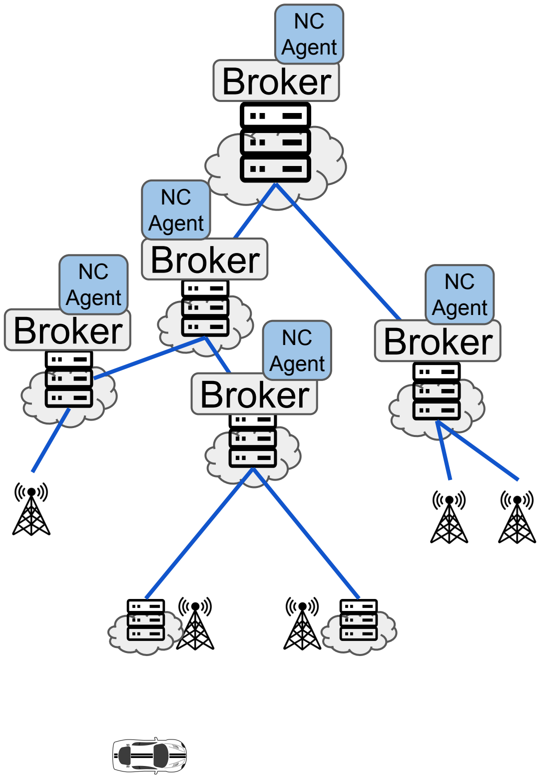
Deployment of Network Coordinate Agents

- NC Agent deployed with Broker



Deployment of Network Coordinate Agents

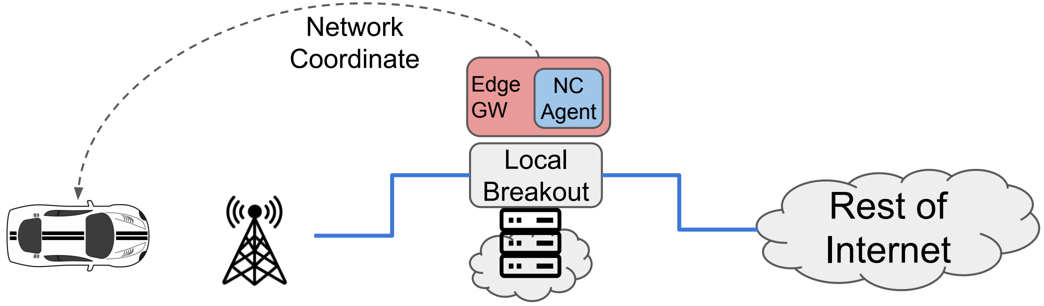
- NC Agent deployed with Broker
- Handling mobile clients

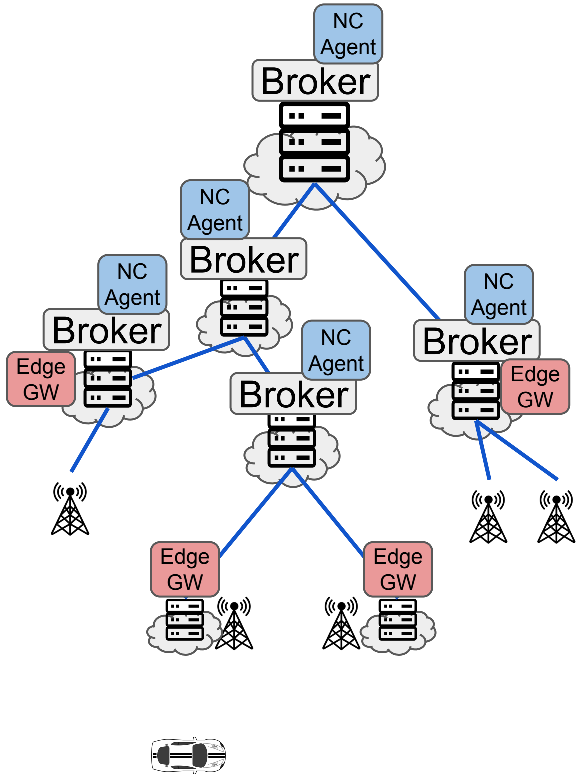


Deployment of Network Coordinate Agents

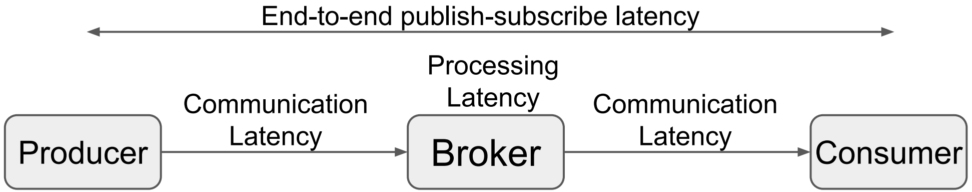
- NC Agent deployed with Broker

- Handling mobile clients
  - Additional Edge Gateway component
  - Located at gateway of access point
  - Adjust for Client-Edge GW RTT

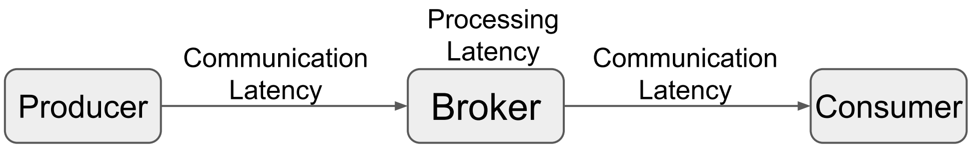




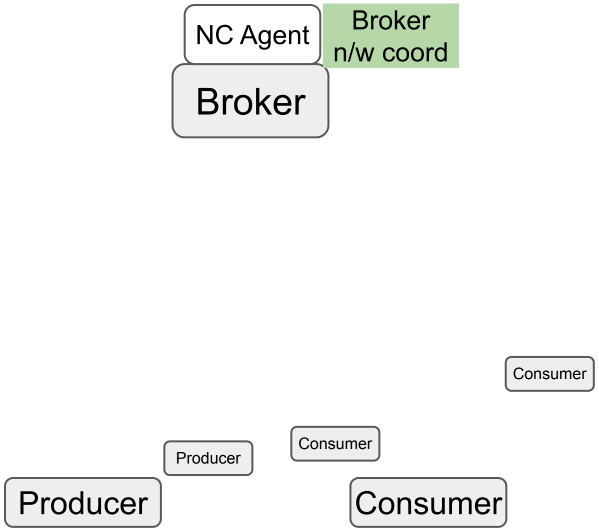
Network proximity for end-to-end latency calculation



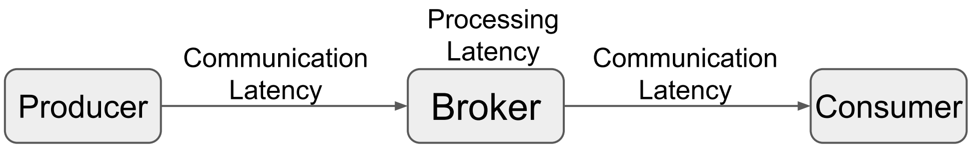
Network proximity for end-to-end latency calculation



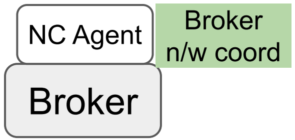
- Network Coordinate (NC) Agents
  - On Brokers

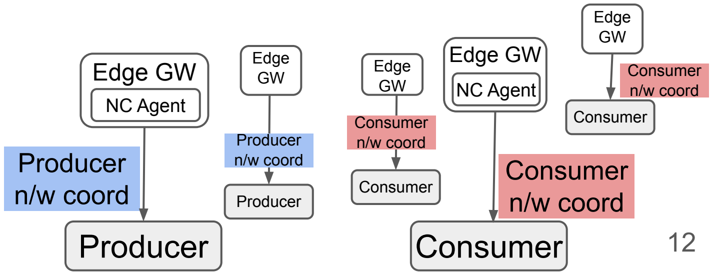


Network proximity for end-to-end latency calculation

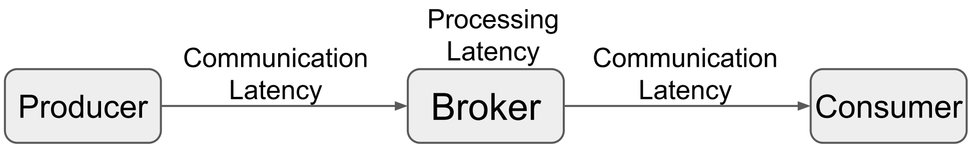


- Network Coordinate (NC) Agents
  - On Brokers
  - On Edge Gateways

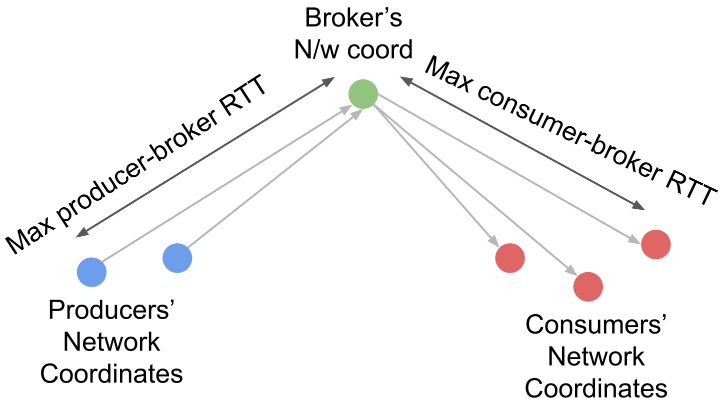




Network proximity for end-to-end latency calculation



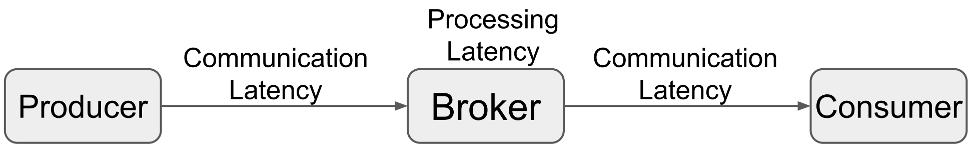
System entities for topic T



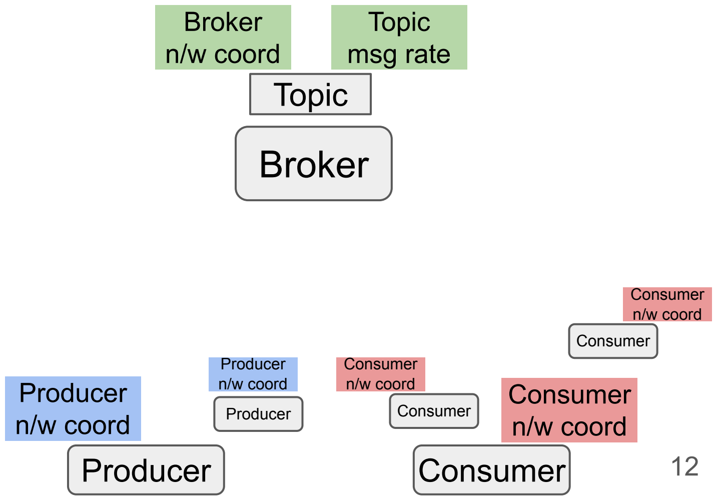
- Network Coordinate (NC) Agents

- Worst-case communication latency for topic
    - Using network coordinates

Network proximity for end-to-end latency calculation

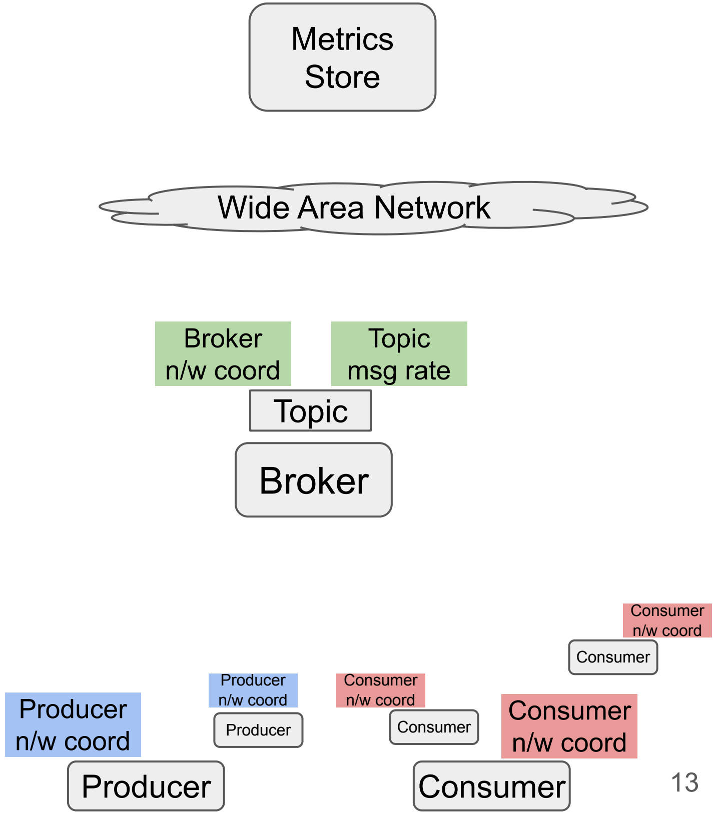


- Network Coordinate (NC) Agents
- Worst-case communication latency for topic
- End-to-end Pub-sub latency estimate
    - Violation Detection policy
    - Broker Selection policy



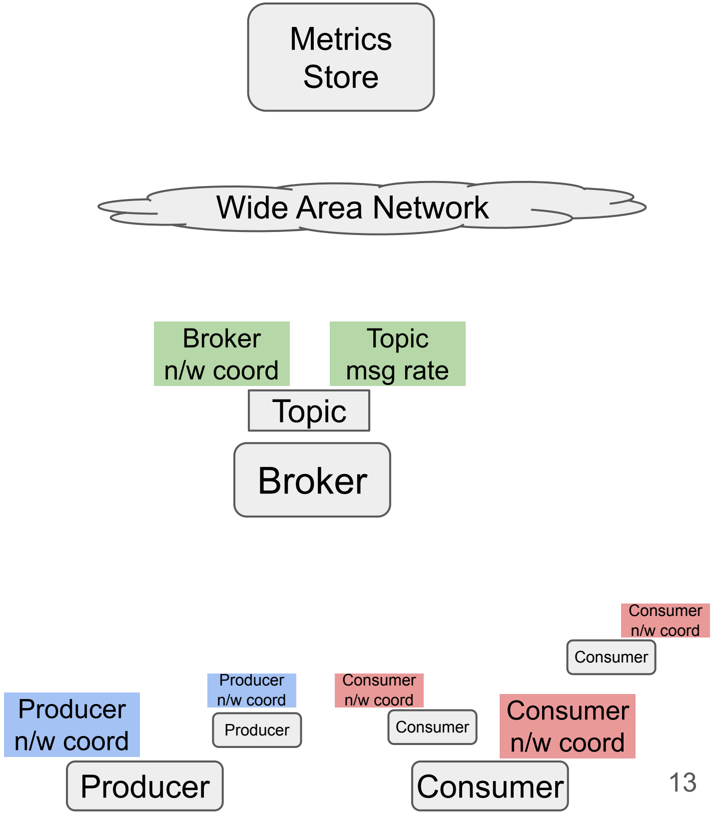
Distributed Monitoring in ePulsar

- Need all topics' metrics at Metrics Store
  - High monitoring traffic through WAN

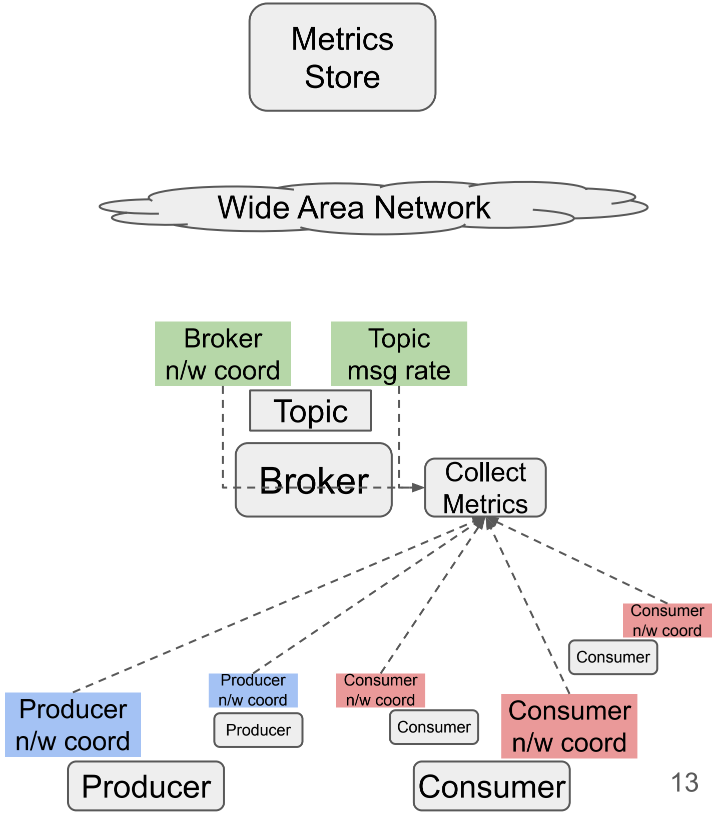


Distributed Monitoring in ePulsar

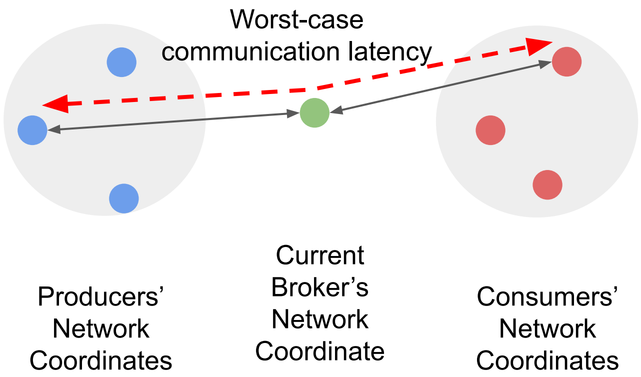
- Need all topics' metrics at Metrics Store
  - High monitoring traffic through WAN
- Distributed metric aggregation
  - Independently per topic

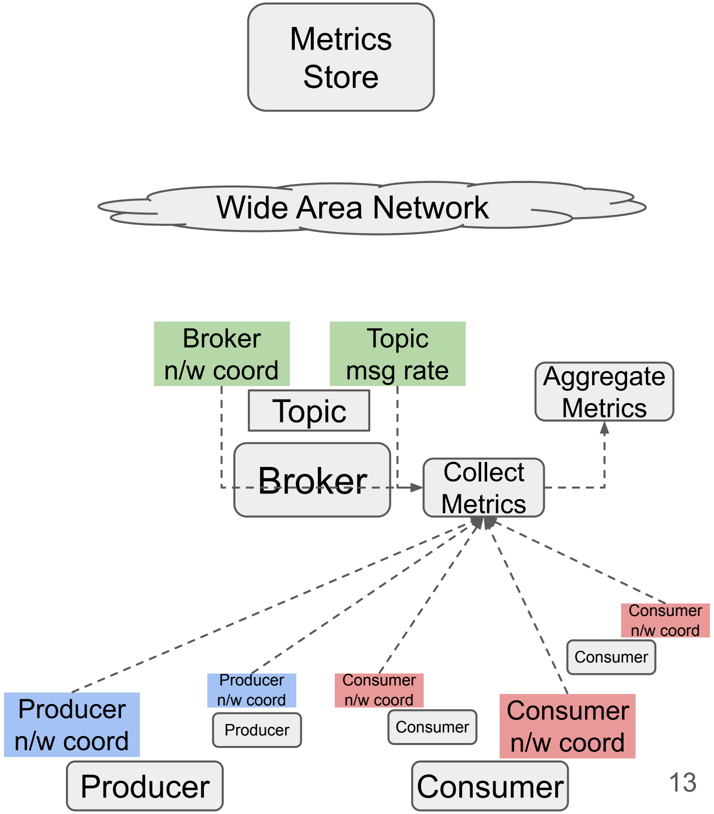


Distributed Monitoring in ePulsar

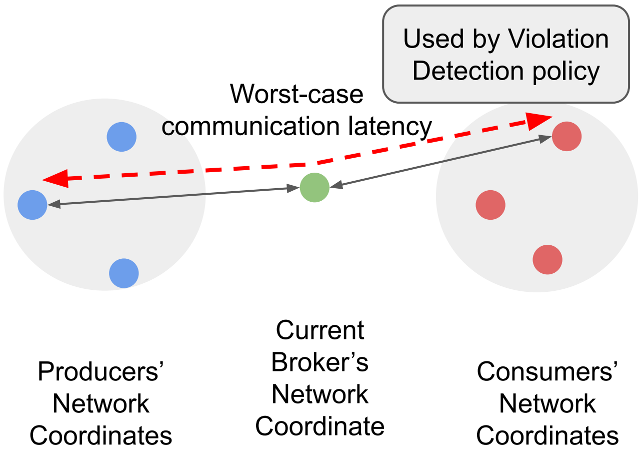


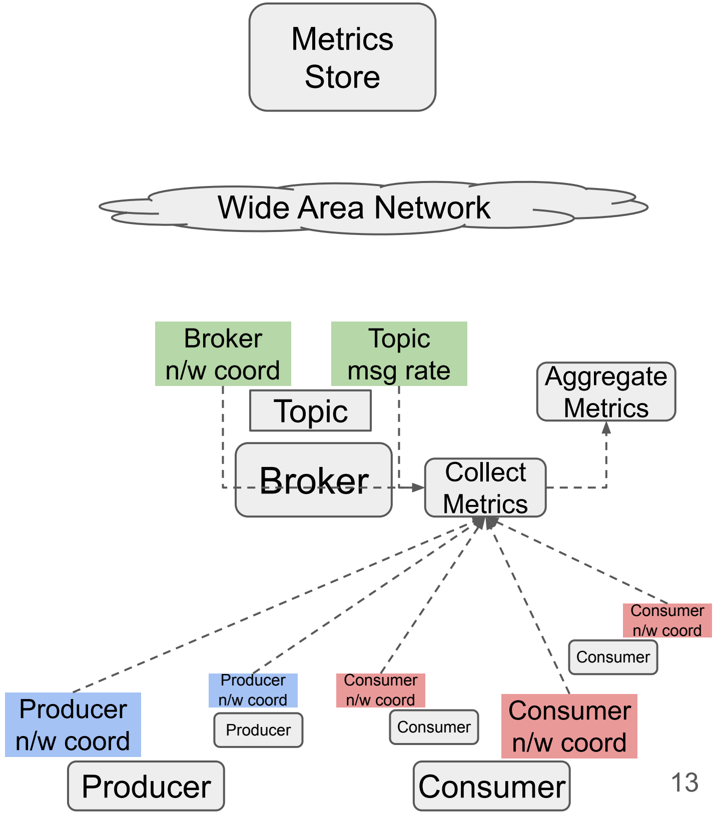
Distributed Monitoring in ePulsar



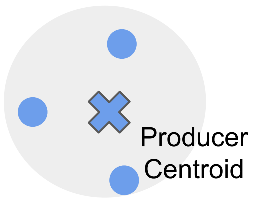


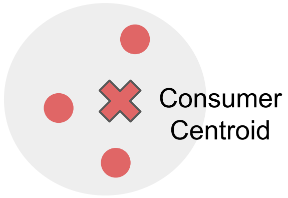
Distributed Monitoring in ePulsar

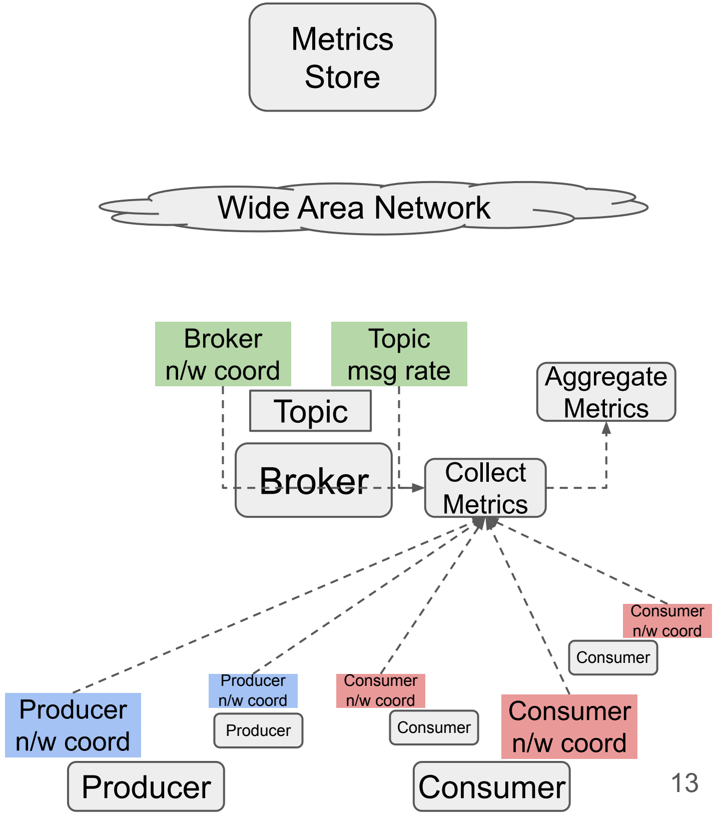




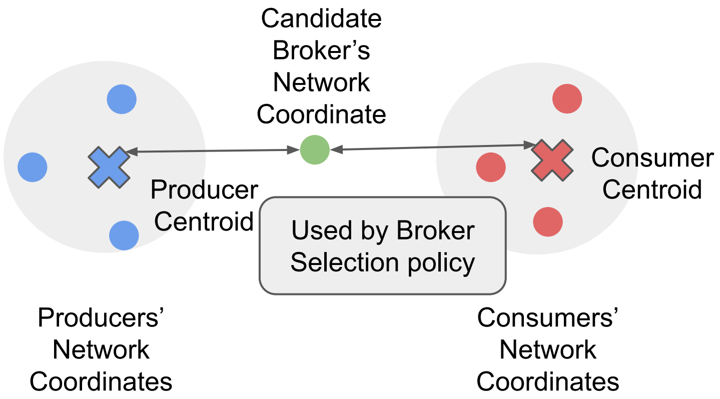
Distributed Monitoring in ePulsar

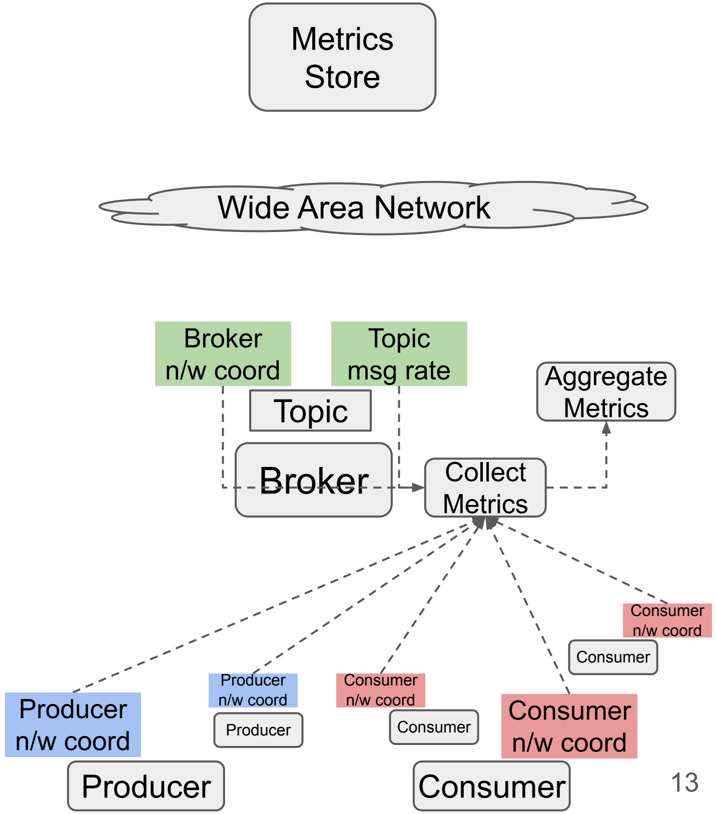




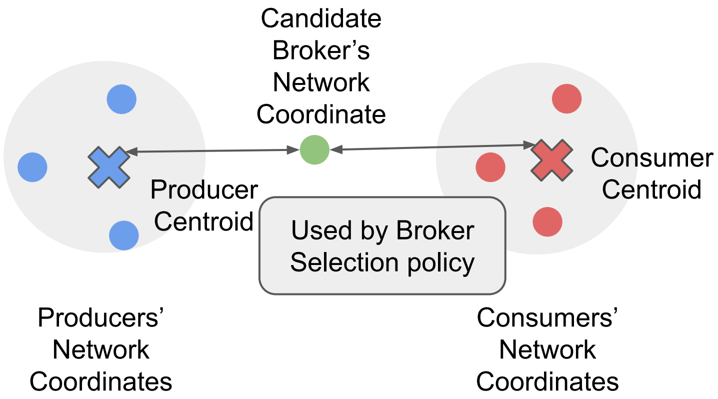


Distributed Monitoring in ePulsar

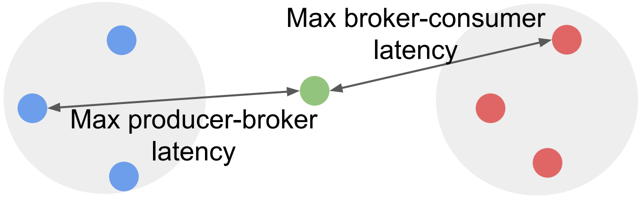


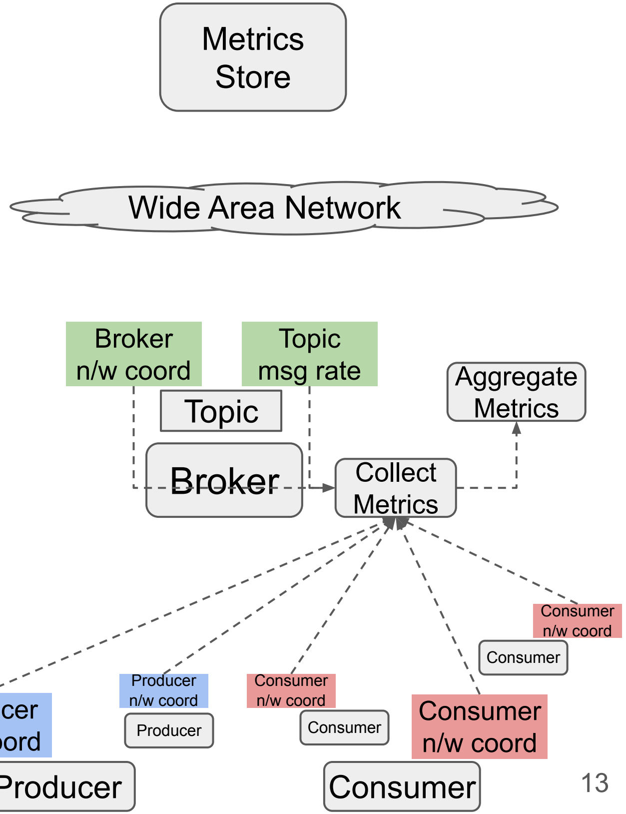


Distributed Monitoring in ePulsar

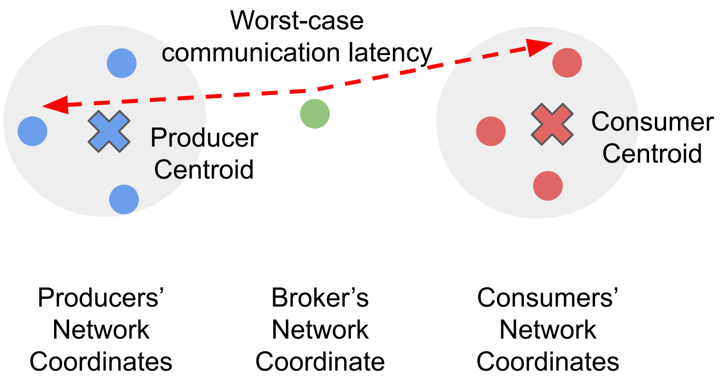


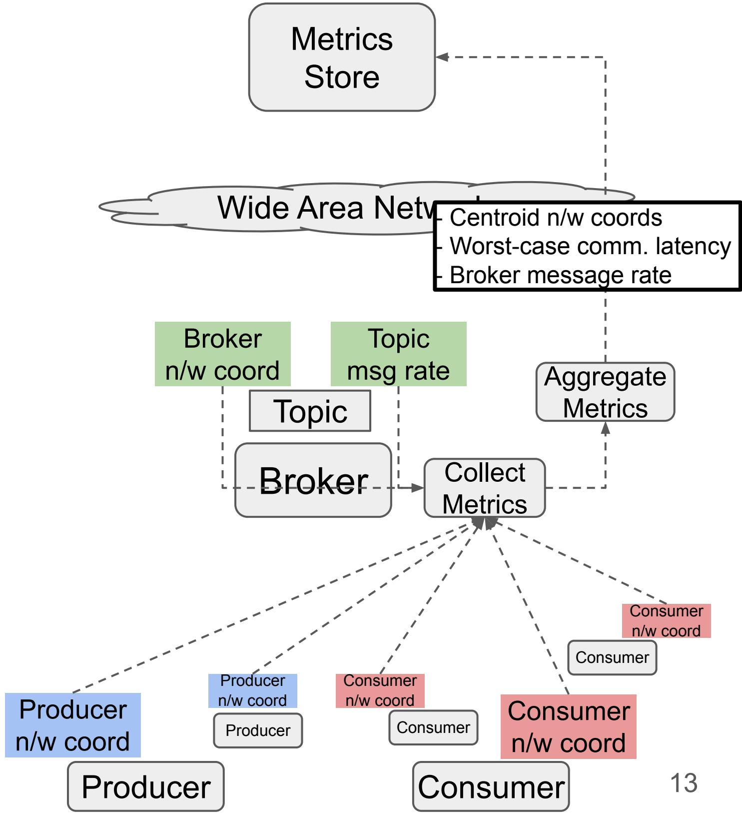
AllPairs Broker Selection policy



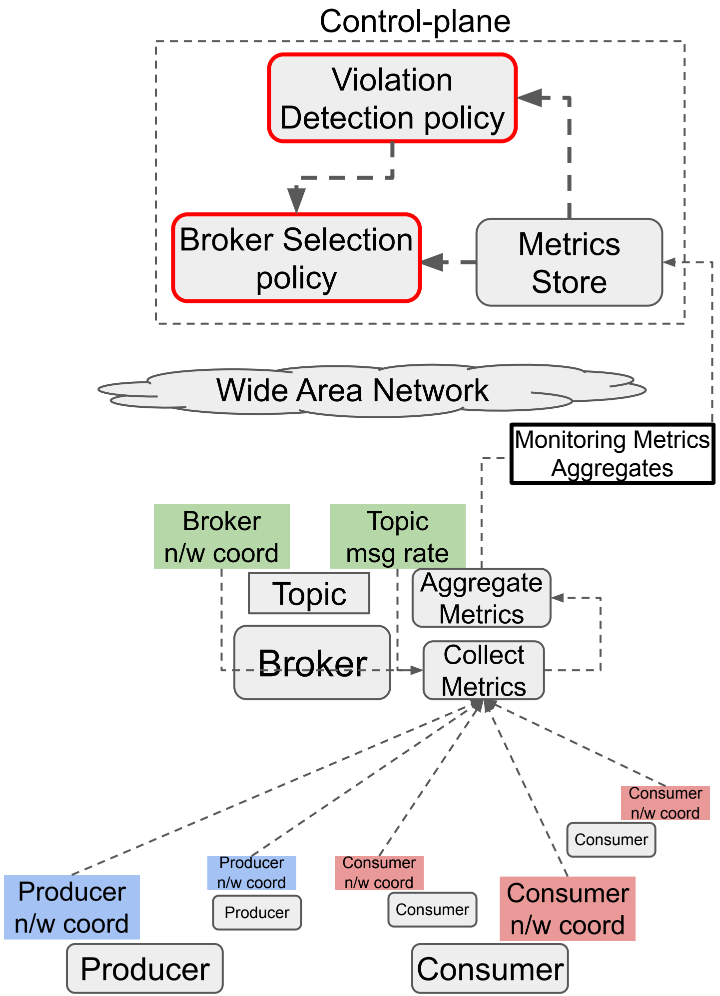


Distributed Monitoring in ePulsar





Putting it all together



Talk Outline

1. Background
2. Problem and Challenges
3. Design Principles of ePulsar
4. Architecture
    a. Network Proximity Estimation
    b. Distributed Monitoring
5. **Implementation**
6. Evaluations
7. Conclusion

Implementation of enhanced control-plane

- Apache Pulsar PULSAR
    - Edge-aware Load Manager module
        - Broker Selection Policy
        - Violation Detection Policy
    - Per-topic monitoring and placement → no bundling of topics
    - ZooKeeper as Metrics Store (as in vanilla Pulsar)

Image source: https://dev.to/alvsanand/life-beyond-kafka-with-apache-pulsar-3mcg

Implementation of enhanced control-plane

- Apache Pulsar
- Serf as Network Coordinate Agent

Image source: https://www.serf.io/

Talk Outline

1. Background
2. Problem and Challenges
3. Design Principles of ePulsar
4. Architecture
    a. Network Proximity Estimation
    b. Distributed Monitoring
5. Implementation
6. **Evaluations**
7. Conclusion

Evaluations

Hypotheses being evaluated

1. Network Coordinates protocol has low error and resource overheads.

2. ePulsar’s Broker Selection satisfies end-to-end latency constraint.

3. ePulsar’s Distributed Monitoring reduces monitoring overhead with increasing scale.

4. ePulsar is able to dynamically detect and mitigate latency violations.

Evaluations

Hypotheses being evaluated

1. Network Coordinates protocol has low error and resource overheads.

2. ePulsar’s Broker Selection satisfies end-to-end latency constraint.

3. ePulsar’s Distributed Monitoring reduces monitoring overhead with increasing scale.

4. ePulsar is able to dynamically detect and mitigate latency violations.

<u>Evaluation methodology</u>

- Emulated clients and edge topology
  - Containernet (Open vSwitch + Docker)
  - Linux Traffic Control (*tc*) for synthetic latency

- Client mobility

Evaluations

Hypotheses being evaluated

1. Network Coordinates protocol has low error and resource overheads.

2. ePulsar’s Broker Selection satisfies end-to-end latency constraint.

3. ePulsar’s Distributed Monitoring reduces monitoring overhead with increasing scale.

4. ePulsar is able to dynamically detect and mitigate latency violations.

<u>Evaluation methodology</u>

- Emulated clients and edge topology
    - Containernet (Open vSwitch + Docker)
    - Linux Traffic Control (*tc*) for synthetic latency
- Client mobility
- Workload
    - UAV Swarm Coordination
    - Massively Multiplayer Online Gaming

Evaluations

Hypotheses being evaluated

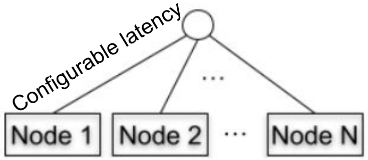
1. Network Coordinates protocol has low error and resource overheads.
2. ePulsar's Broker Selection satisfies end-to-end latency constraint.
3. ePulsar's Distributed Monitoring reduces monitoring overhead with increasing scale.
4. ePulsar is able to dynamically detect and mitigate latency violations.

<u>Evaluation methodology</u>

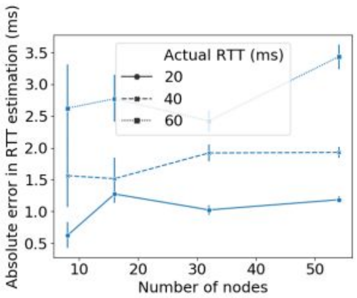
- Emulated clients and edge topology
  - Containernet (Open vSwitch + Docker)
  - Linux Traffic Control (*tc*) for synthetic latency
- Client mobility
- Workload
  - UAV Swarm Coordination
  - Massively Multiplayer Online Gaming

Evaluation of network coordinates for measuring proximity

- Low error in RTT estimation < 3.5ms
- Low CPU and memory overhead
  - < 1% CPU util on AMD EPYC 7501
  - < 15 MB memory usage



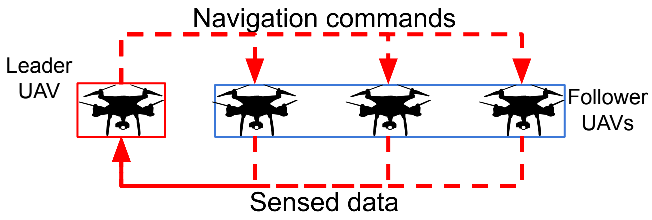
(a) NC Cluster Topology.

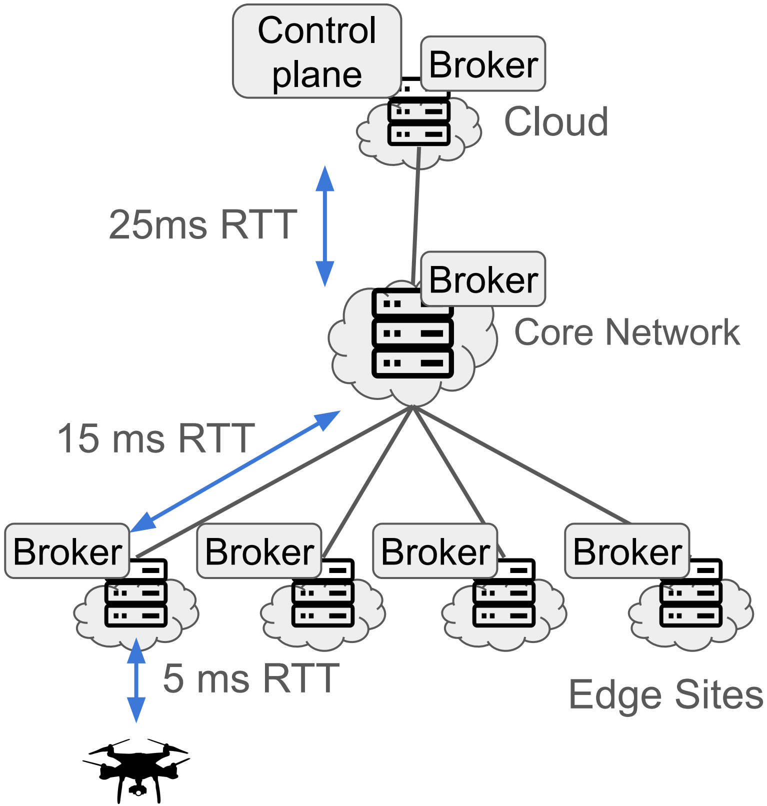


(b) RTT estimation error.

Evaluation of UAV Swarm scenario

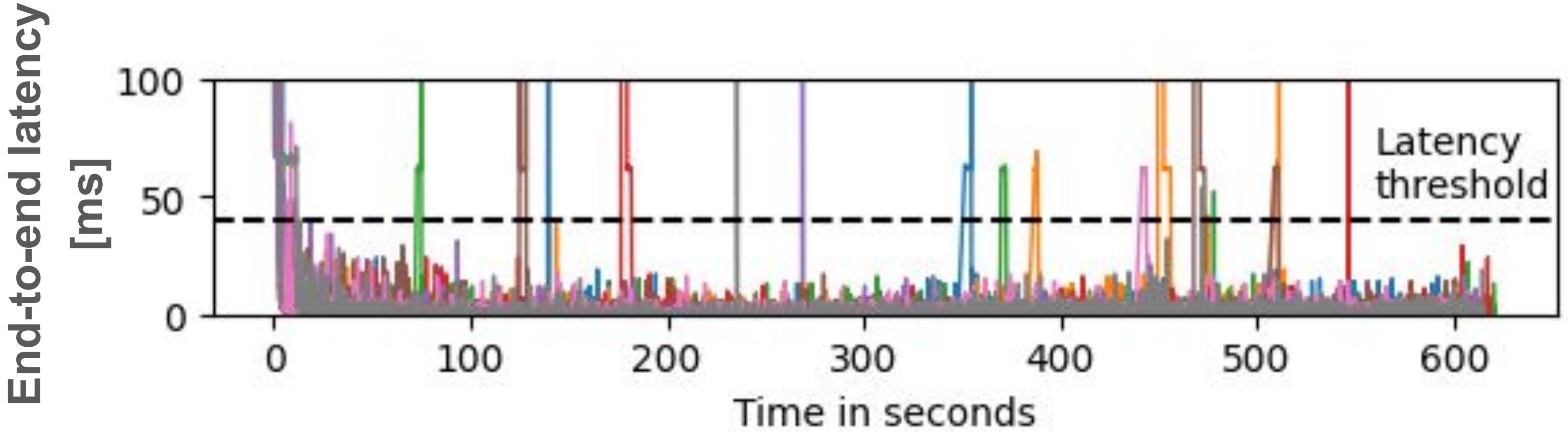
UAV Swarm Coordination application





Dynamic violation detection and topic migration

- 16 drone swarms - each with 8 drones
- Random Waypoint mobility model in a city with 8 edge sites



Talk Outline

1. Background
2. Problem and Challenges
3. Design Principles of ePulsar
4. Architecture
   a. Network Proximity Estimation
   b. Distributed Monitoring
5. Implementation
6. Evaluations
7. **Conclusion**

Conclusion

- Control-plane architecture for geo-distributed publish-subscribe system
- Ensures end-to-end publish-subscribe latency
- Latency-aware broker selection and topic migration
- Network Proximity Estimation
- Distributed Monitoring

Conclusion

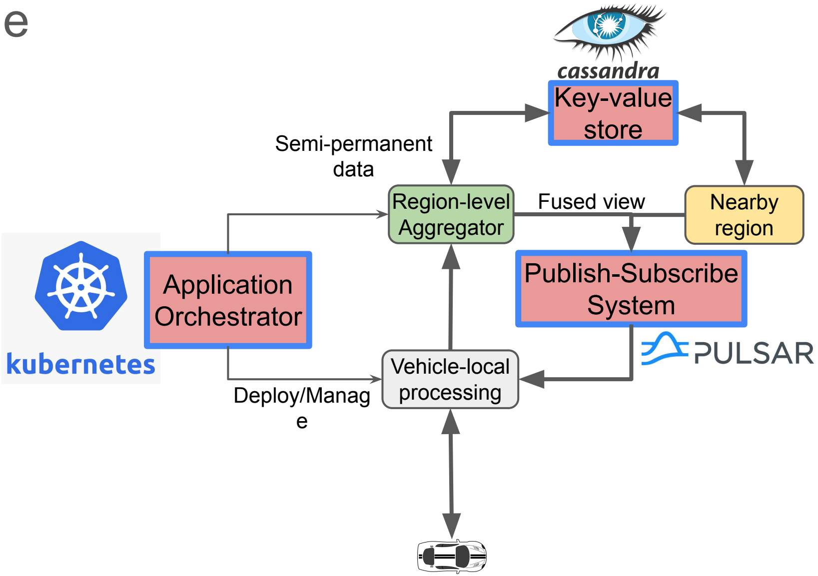
- Control-plane architecture for geo-distributed publish-subscribe system
- Ensures end-to-end publish-subscribe latency
- Latency-aware broker selection and topic migration
- Network Proximity Estimation
- Distributed Monitoring

Check out the paper for

- Optimizations for agile reconfigurations
- Support for persistent topics
- Evaluation of MMOG application scenario

Future Work

- Decentralize publish-subscribe control-plane (Desousis et al. ICDCS 2018)
- Enable other edge-ready platform services
  - Network Proximity Estimation and Distributed Monitoring as independent services



Dedousis, Dimitris, Nikos Zacheilas, and Vana Kalogeraki. "On the fly load balancing to address hot topics in topic-based pub/sub systems." 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2018.

Thank you!

ePulsar

- Control-plane architecture for geo-distributed publish-subscribe system
- Ensures end-to-end publish-subscribe latency
- Latency-aware broker selection and topic migration
- Network Proximity Estimation
- Distributed Monitoring

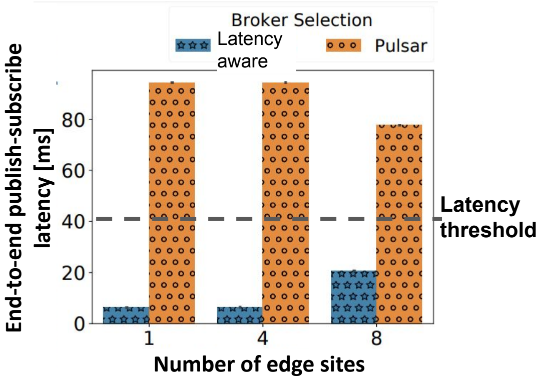
Georgia Tech College of Computing
School of Computer Science

Harshit Gupta
harshitg@gatech.edu

Backup Slides

Limitations of state-of-the-art in geo-distributed pub-subs

- Off-the-shelf Cloud-based pub-subs on the Edge
  - E.g., Apache Pulsar, Apache Kafka
  - Focus on even workload distribution, not end-to-end latency
  - Don’t consider high client-edge communication latencies



Comparing end-to-end publish-subscribe latency of off-the-shelf Apache Pulsar running on the Edge vs. a topology-aware approach.

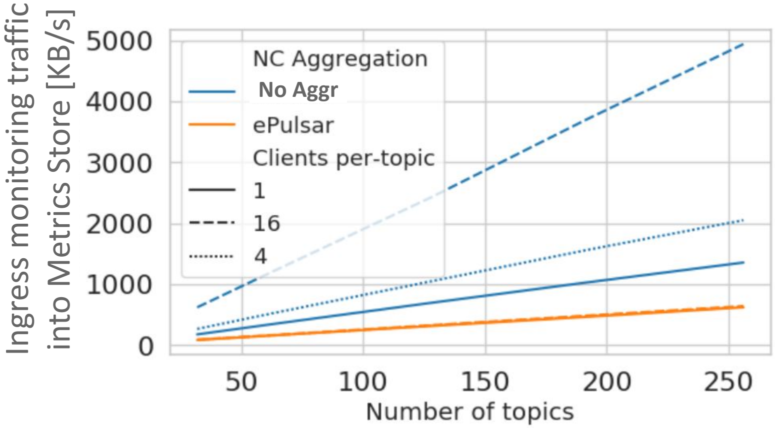
- Pub-subs designed for the Edge
  - E.g., EMMA [1], MultiPub [2]
  - Active measurements for topology awareness [1]
    ⇒ High monitoring overhead
  - Require latency between each client-broker pair [2]
    ⇒ Scales poorly

[1] Rausch, Thomas, Stefan Nastic, and Schahram Dustdar. "Emma: Distributed qos-aware mqtt middleware for edge computing applications." 2018 IEEE International Conference on Cloud Engineering (IC2E). IEEE, 2018.
[2] Gascon-Samson, Julien, Jörg Kienzle, and Bettina Kemme. "Multipub: Latency and cost-aware global-scale cloud publish/subscribe." 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2017.

Distributed Monitoring: monitoring traffic analysis

- **ePulsar** vs. **NoAggr**

- ePulsar: Lower monitoring overhead
  - With increasing scale of workload



Broker selection policy with network proximity awareness

- Systems compared
  - **Pulsar**: No network proximity awareness
  - **NoAggr**: Same as ePulsar (w/o network coord. aggr.)
- Metric: per-topic worst-case publish-subscribe latency
  - Across producer-consumer pairs
- 16 UAV swarms
  - Each with 8 drones
- Drones in a swarm move together
  - Randomized swarm locations