Consistency in SDN
Aurojit Panda, Wenting Zheng, Xiaohe Hu, Arvind Krishnamurthy, Scott Shenker
Distributed SDN Today

Consistency Layer

Replicated Controller

Switch

Replicated Controller

Switch

Replicated Controller

Switch
Distributed SDN Today

Consistency Layer

Today: Paxos, Raft, etc. used to implement serializability
Our Approach

Consistent Policy Database

Consistency Layer

Independent Controller

Switch

Independent Controller

Switch

Independent Controller

Switch
Our Approach

Consistent Policy Database

Consistency Layer

Independent Controller

Switch

Independent Controller

Switch

Independent Controller

Switch

Respond instantaneously
Our Approach

Consistent Policy Database

Consistency Layer

- Independent Controller
- Switch

Eventual Correctness

Respond instantaneously
Our Approach

Consistent Policy Database

Consistency Layer

Independent Controller

Switch

Switch

Switch

Switch

Independent Controller

Independent Controller

Independent Controller

Consistent view of policy

Eventual Correctness

Respond instantaneously
Performance

• Allows greater scalability and resilience.
Performance

• Allows greater scalability and resilience.

• Faster convergence: we do better than when consistency is used.
Performance

• Allows greater scalability and resilience.

• Faster convergence: we do better than when consistency is used.

Convergence Time in Data Centers
Performance

- Allows greater scalability and resilience.
- Faster convergence: we do better than when consistency is used.
Correctness

- Our approach ensures:
Correctness

- Our approach ensures:
  - Eventually all controllers agree on the sequence of network events seen.
Correctness

- Our approach ensures:
  - Eventually all controllers agree on the sequence of network events seen.
  - Eventually each controller and network agree on state of the network.
Correctness

- Our approach ensures:
  - Eventually all controllers agree on the sequence of network events seen.
  - Eventually each controller and network agree on state of the network.
  - Therefore eventually computed and installed states are correct.
Correctness

- Our approach ensures:
  - Eventually all controllers agree on the sequence of network events seen.
  - Eventually each controller and network agree on state of the network.
  - Therefore eventually computed and installed states are correct.
- Assuming **deterministic controllers** and **idempotent switch updates**.
What about Consistency?
What about Consistency?

- **Correctness**: Needed to ensure flow tables, controllers are correct.
What about Consistency?

- **Correctness**: Needed to ensure flow tables, controllers are correct.
- **Programmability**: Needed to make it easier to program networks.
What about Consistency?

- **Correctness**: Needed to ensure flow tables, controllers are correct.
- **Programmability**: Needed to make it easier to program networks.
- **Performance**: Needed for faster convergence.
What about Consistency?

- **Correctness**: Needed to ensure flow tables, controllers are correct.

- **Programmability**: Needed to make it easier to program networks.

- **Performance**: Needed for faster convergence.
Why Does This Work?

• Networks are open world systems.
Why Does This Work?

- Networks are **open world** systems.

- **Open World**: Truth resides in an external entity (e.g., network).
Why Does This Work?

- Networks are **open world** systems.

- **Open World**: Truth resides in an external entity (e.g., network).

- **Closed World**: Truth resides in the system itself (e.g., a database).
Why Does This Work?

- Networks are **open world** systems.
- **Open World**: Truth resides in an external entity (e.g., network).
- **Closed World**: Truth resides in the system itself (e.g., a database).
- With open world systems
Why Does This Work?

• Networks are **open world** systems.

• **Open World**: Truth resides in an external entity (e.g., network).

• **Closed World**: Truth resides in the system itself (e.g., a database).

• With open world systems
  • Truth can be recovered from the external system.
Why Does This Work?

- Networks are **open world** systems.

- **Open World**: Truth resides in an external entity (e.g., network).

- **Closed World**: Truth resides in the system itself (e.g., a database).

- With open world systems
  - Truth can be recovered from the external system.
  - Consistency with ground truth is more important than within the system.
Why is this relevant?
Sources of Network Updates

- Planned Updates
- Network Events
Sources of Network Updates

**Planned Updates**
Policy updates, link recovery, etc.

**Network Events**
Link failures, switch failure, etc.
Sources of Network Updates

**Planned Updates**
Policy updates, link recovery, etc.

**Network Events**
Link failures, switch failure, etc.

Working Network $\rightarrow$ Working Network

Broken Network $\rightarrow$ Working Network
Sources of Network Updates

**Planned Updates**
Policy updates, link recovery, etc.

Working Network → Working Network

**Network Events**
Link failures, switch failure, etc.

Broken Network → Working Network

**Goal**
Sources of Network Updates

Planned Updates
Policy updates, link recovery, etc.
Working Network $\rightarrow$ Working Network

Network Events
Link failures, switch failure, etc.
Broken Network $\rightarrow$ Working Network

Goal
Maintain correctness during transition
Minimize time to connectivity restored.
Sources of Network Updates

**Planned Updates**
Policy updates, link recovery, etc.

Working Network → Working Network

**Network Events**
Link failures, switch failure, etc.

Broken Network → Working Network

**Goal**
Maintain correctness during transition
Consistency helps (required?)

Minimize time to connectivity restored.
Consistency adds latency.
Edge-Core Separation

Fabric
Provides connectivity

Routing, Traffic Engineering
Edge-Core Separation

Fabric
Provides connectivity

Edge
Richer Policies

Endhost

ACLs
Traffic Priorities
Conclusion

• Existence proof that controller consistency is not necessary.
  • In fact slows down network recovery in response to failures.
• Should we require consistency for SDN controllers?
  • Question is similar to the ACID vs NoSQL debate in data stores.
Open Questions

• What about data plane consistency?
  • Ensures each packet processed according to consistent policy.

• Do we need data plane consistency?
  • For **planned updates**: Helps with correctness during policy changes.
  • For **network events**: Adds latency before connectivity is restored.