

**IETF 117 – RTG WG**

# **Signaling In-Network Computing operations (SINC)**

**~~draft-zhou-rtgwg-sinc-00~~**

**draft-lou-rtgwg-sinc-00**

**Zhe Lou, Luigi Iannone, Yizhou Li, Cuimin Zhang, Kehan Yao**

**IETF 117 – San Francisco**

# Since IETF 116

## Signaling In-Network Computing operations (SINC)

draft-lou-rtgwg-sinc-00

Status [Email expansions](#) [History](#)

### Versions:

00

draft-zhou-sfc-sinc

00

draft-zhou-rtgwg-sinc

00

draft-lou-rtgwg-sinc

00

Oct 2022

Feb 2023

Jun 2023

- Split the original draft into 2 drafts,
  - draft-zhou-rtgwg-sinc depicts the main spec
  - draft-zhou-rtgwg-sinc-deployment-considerations covers the deployment considerations
- New co-author: Jinze Yang

- Added control plane considerations
- New co-author: Yizhou LI (Huawei), Kehan YAO (China Mobile)

# Updates since IETF 116

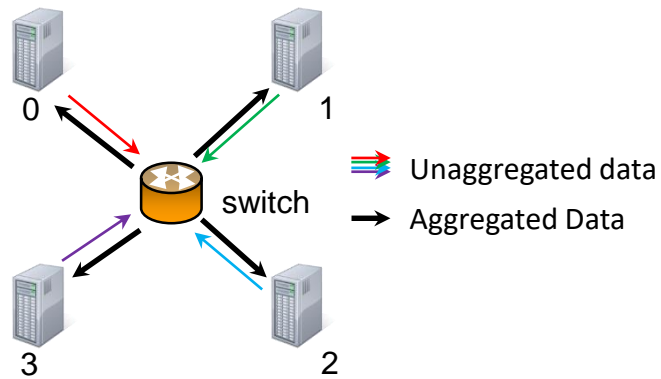
## Table of Contents

1. Introduction . . . . .	2
2. Requirements Language . . . . .	3
3. SINC Relevant Use Cases . . . . .	3
3.1. NetReduce . . . . .	3
3.2. NetDistributedLock . . . . .	4
3.3. NetSequencer . . . . .	5
4. In-Network Generic Operations . . . . .	5
5. SINC Framework Overview . . . . .	6
6. Data Operation Mode . . . . .	8
6.1. Individual Computing Mode . . . . .	8
6.2. Batch Computing Mode . . . . .	9
7. SINC Header . . . . .	9
8. Control Plane Considerations . . . . .	10
9. Security Considerations . . . . .	12
10. IANA Considerations . . . . .	13
Acknowledgements . . . . .	13
References . . . . .	13
Normative References . . . . .	13
Informative References . . . . .	13
Contributors . . . . .	16
Authors' Addresses . . . . .	16

- Clarify the difference between CATS and SINC
- A brief comparison from SHARP, SwithML and NetReduce
- Specify how does a SINC switch/router exam the SINC header (framework)
- Specify a tree topology is required for batch operation (data operation mode)
- Control plane requirements -> control plane consideration, to include the rationales and basic requirements
- Removed the computing capability operation abstraction
- Overall text refinement

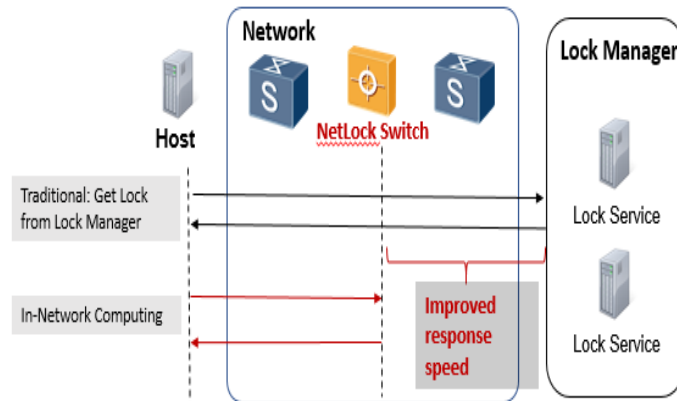
# SINC Use Cases (recap)

## NetReduce



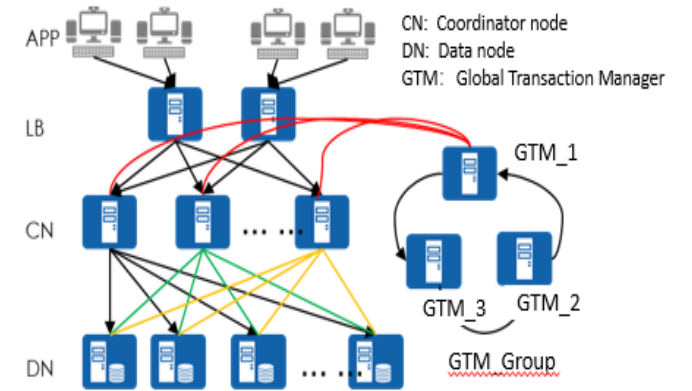
- ❖ Comparing with the host oriented solutions, in-network aggregation could potentially reduce nearly half the aggregation data
- ❖ NetReduce is >1.5x faster, and has better scalability than ring all-reduce.

## NetLock



- ❖ The lock manager can be abstracted as Compare And Swap (CAS) or Fetch Add (FA) operations.
- ❖ The test results in NetLock[1] show that the lock manager running on a switch is able to answer 100 million requests per second, nearly 10 times more than what a lock server can do.

## NetSequencer



- ❖ Switches could realize the sequencer[2] by using a "Fetch-and-Add" operation.
- ❖ Compared with Gbps-level throughput of servers, network devices have Tbps-level throughput and line-rate processing capabilities

**An explicit and general mechanism is required to tell the switch how to process the packet**

# SINC Framework

Clarify the difference between CATS and SINC

## **1. Introduction**

It is different from what the IETF Computing-Aware Traffic Steering (CATS) working group is chartered for service instance selection based on network and compute metrics between clients of a service and sites offering the service. The in-network computing is more about "light" data calculation/operation performed in the network to reduce the computation work load for the end hosts.

Brief comparison among SwitchML, SHArP, NetReduce

## **3.1 NetReduce**

Comparing to host-oriented solutions, in-network aggregation approaches like SwitchML, SHArP, and NetReduce can potentially reduce to nearly half the bandwidth needed for data aggregation, by offloading gradients aggregation from the host to network switches. However, they are limited to one single specific operation, namely aggregation...

SwitchML... SHArP... NetReduce...

Ways of checking the SINC header

## **5. SINC Framework Overview**

The SINC switch/router is the node equipped with in-network computing capabilities. It MUST look for the SINC header and perform the required operations if any. It could be done from the encapsulation protocols that contain a field of "next protocols". Otherwise, the SINC switch/router should be able to perform a deep packet inspection to identify the location of the SINC header.

Tree topology creation for batch computing mode

## **6.2 Batch Computing Mode**

In this mode, the data operation is collective. The data coming from multiple sources may be aggregated in multiple aggregation nodes in a hierarchy. Hence a tree topology should be created from the control plane for each batch computing request, which will be dismissed once the batch computing is done. A message is required to signal the start and the end of the operation.

# Control Plane Considerations

- ❖ Topology creation/deletion
  - ❖ Topology computation: When receiving the computing request from the Host, the SINC control plane needs to compute a set of feasible paths with SINC capable nodes to support individual or batch computations.
  - ❖ Topology establishment: The topology has to be sent/configured/signaled to the network device, so SINC packets could pass through the right SINC capable nodes to perform the required data computing in the network.
  - ❖ Topology deletion: Once the application finishes the action, it will inform the control plane to delete the topology and release the reserved resources for other applications and purposes.
- ❖ Resource (e.g. computing resource, buffer resource, and bandwidth resource) reservation to avoid traffic and computing congestion
- ❖ Performance monitoring to detect any potential issues during the data operation
- ❖ Service protection
  - ❖ The in-network computing service must be protected. If a SINC node of an in-network operation fails, the impact should be minimized by guaranteeing as much as possible that the packets are at least delivered to the end node, which will perform the requested operation. The control plane will take care to recover the failure, possibly using a different SINC node and re-routing the traffic.
  - ❖ The network service must be able to deliver packet to the designated SINC nodes even in case of partial network failures (e.g. link failures).

# Control Plane Requirements

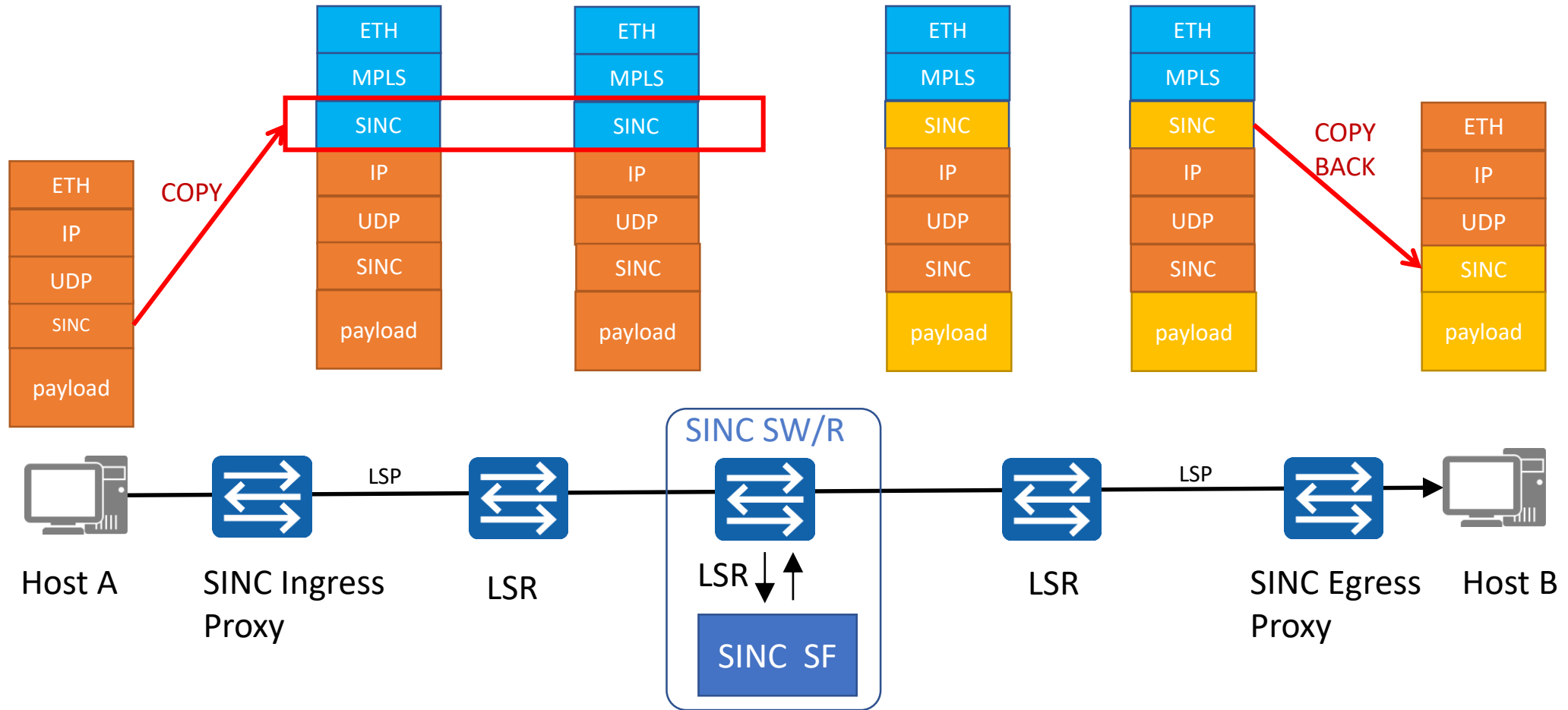
- ❖ The ability to exchange computing requirements (e.g. computing tasks, performance, bandwidth, etc.) and execution status with the application (e.g., via a User Network Interface). SINC tasks should be carefully coordinated with (other) host tasks.
- ❖ The ability to gather the resources available on SINC-capable devices, which demands regular advertisement of node capabilities and link resources to other network nodes or to network controller(s).
- ❖ The ability to dynamically create, modify and delete computing network topologies based on application requests and according to defined constraints.
- ❖ The ability to monitor the performance of SINC nodes and link status to ensure that they meet the requirements.
- ❖ The ability to provide failover mechanism in order to handle errors and failures, and improves the resilience of the system. A fallback mechanism is required in case that in-network resources are not sufficient for processing SINC tasks, in which case, end host might provide some complementary computing capabilities.

# **Signaling In-Network Computing operations (SINC) deployment considerations**

**~~draft-zhou-rtgwg-sinc-deployment-considerations-00~~**

**~~draft-lou-rtgwg-sinc-deployment-considerations-00~~**

# SINC over MPLS (recap)



The SINC SW/R will further identify the location of the SINC header by checking the Next Hop Label Forwarding Entry (NHLFE) as defined in the [RFC3031].

## Next Steps

- Encourage discussion on the mailing lists of the RTG WG
- Update the draft based on comments and remarks
- Welcome to contributions and co-authors

# THANKS!