Floodless in SEATTLE :
A Scalable Ethernet ArchiTecTure for Large Enterprises

Changhoon Kim, Matthew Caesar,

and Jenifer Rexford

Presented by Chi-Yao Hong

Adapted from slides by Changhoon Kim

Oct. 1, 2009

In *ACM SIGCOMM*, Seattle, WA, Aug. 2008

Challenges of Layer-2 networks

- Facing unprecedented scale
- Highly requirements in terms of efficiency and availability
- Large data centers might comprise 100,000+ computers within a single facility
- What's the possible solution?

Ethernet has substantial benefits

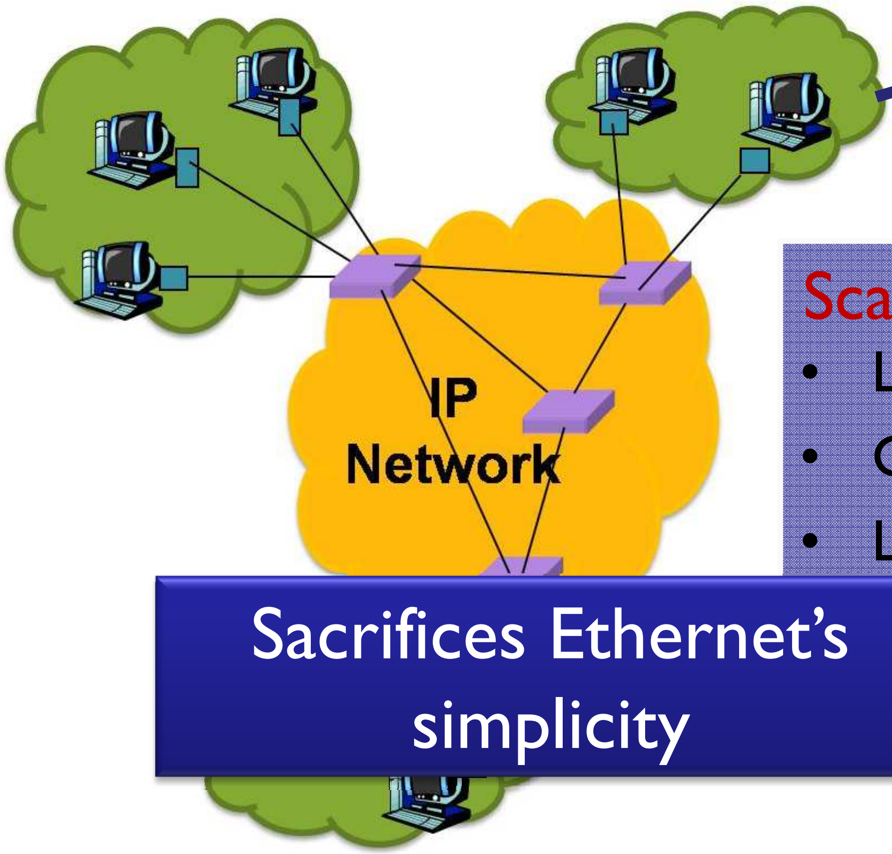
- Because of its simplicity!
  - Plug-and-play via flat addressing
  - Simplifies the handling of topology changes and host mobility
- IP networks require massive effort to configure and manage
  - Why do we still use IP routing inside a single network?

What's Wrong With Ethernet?

- Ethernet is not scalable!
  - Network-wide flooding
  - Frequent broadcasting
  - Unbalanced link utilization due to tree-based forwarding
- Scalability requirement is growing very fast

Current Practice

- Multiple small Ethernet-based IP subnets interconnected by routers



IP subnet == Ethernet broadcast domain

Scalable but…

- Loss of self-configuring capability
- Complexity in implementing policies
- Limited mobility support
- exib

Sacrifices Ethernet's simplicity

Loses IP's efficiency

SEATTLE: The best of IP and Ethernet

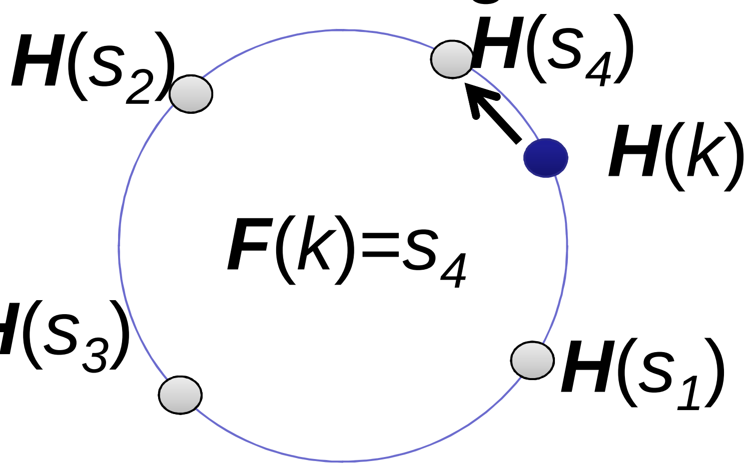
Feature	Ethernet Bridging	IP Routing	SEATTLE
Ease of configuration	👍	👎	👍
Optimality in addressing	👍	👎	👍
Host mobility	👍	👎	👍
Scalability	👎	👍	👍
Path efficiency	👎	👍	👍
Tolerance to loop	👎	👍	👍

Solutions

Objective	Approach	Solution
Avoiding flooding	Never broadcast unicast traffic	Network-layer one-hop DHT
Restraining broadcasting	Bootstrap hosts via unicast	
Reducing routing state	Populate host info only when and where it is needed	Traffic-driven resolution with caching
Shortest-path forwarding	Allow switches to learn topology	L2 link-state routing (switch-level topology)

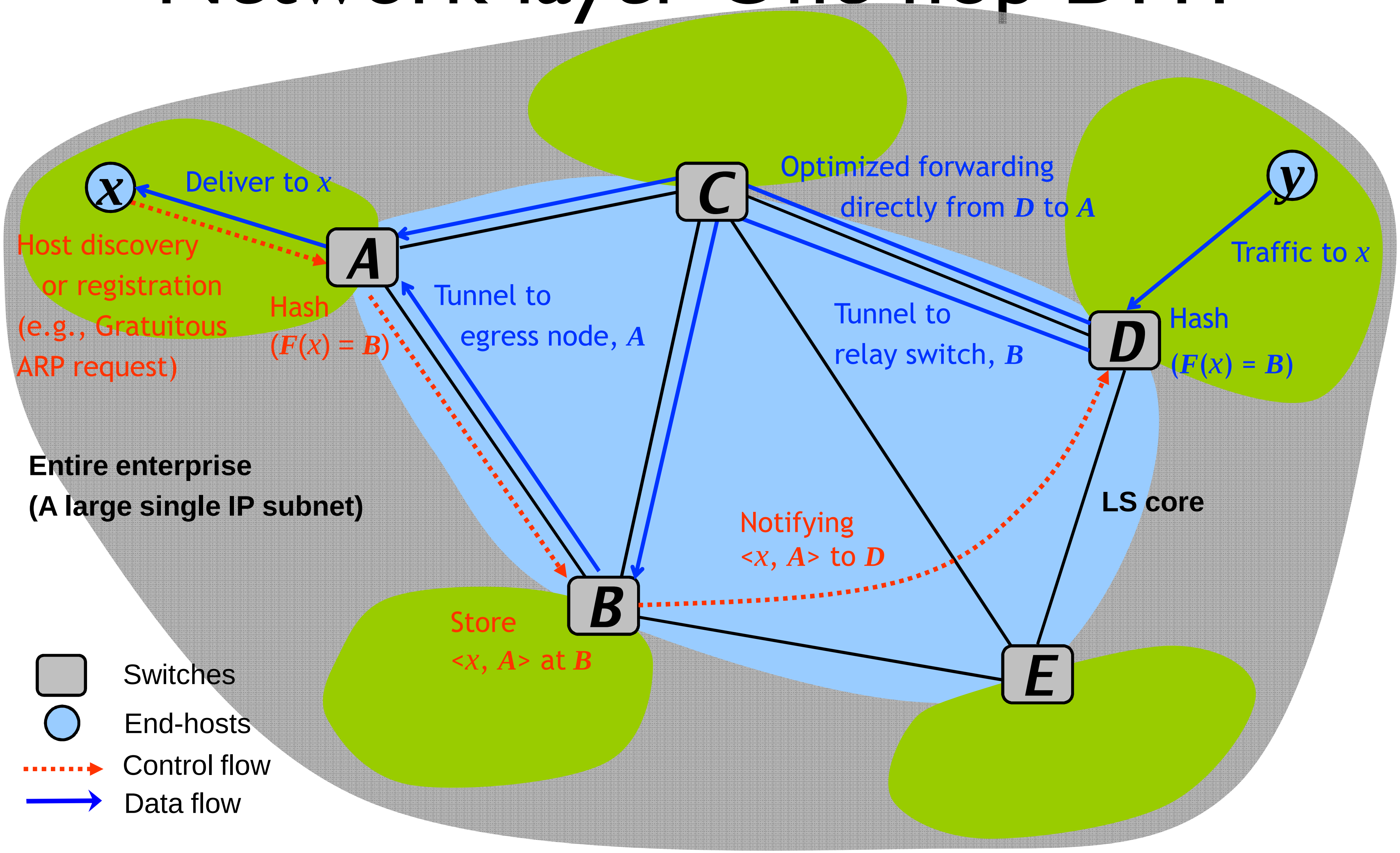
Network-layer One-hop DHT

- Switches maintain <key, value> pairs by commonly using a hash function ***F***
  - ***F*** : Consistent hash mapping a key to a switch
  - Link-state routing ensures each switch knows about all the other live switches, thus enabling one-hop DHT operations



- Benefits
  - Reducing lookup complexity
  - Fast and efficient reaction to changes

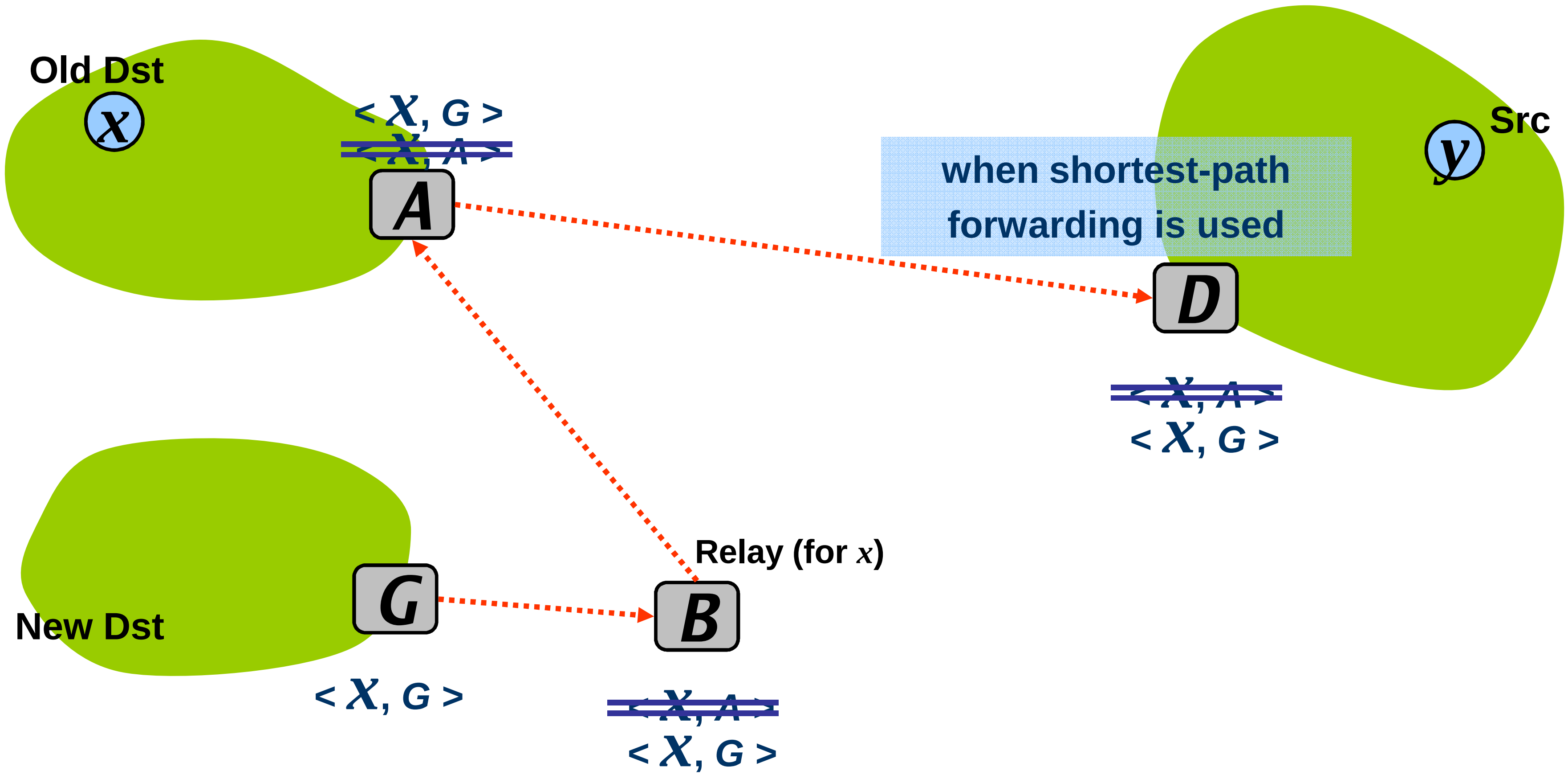
Network-layer One-hop DHT



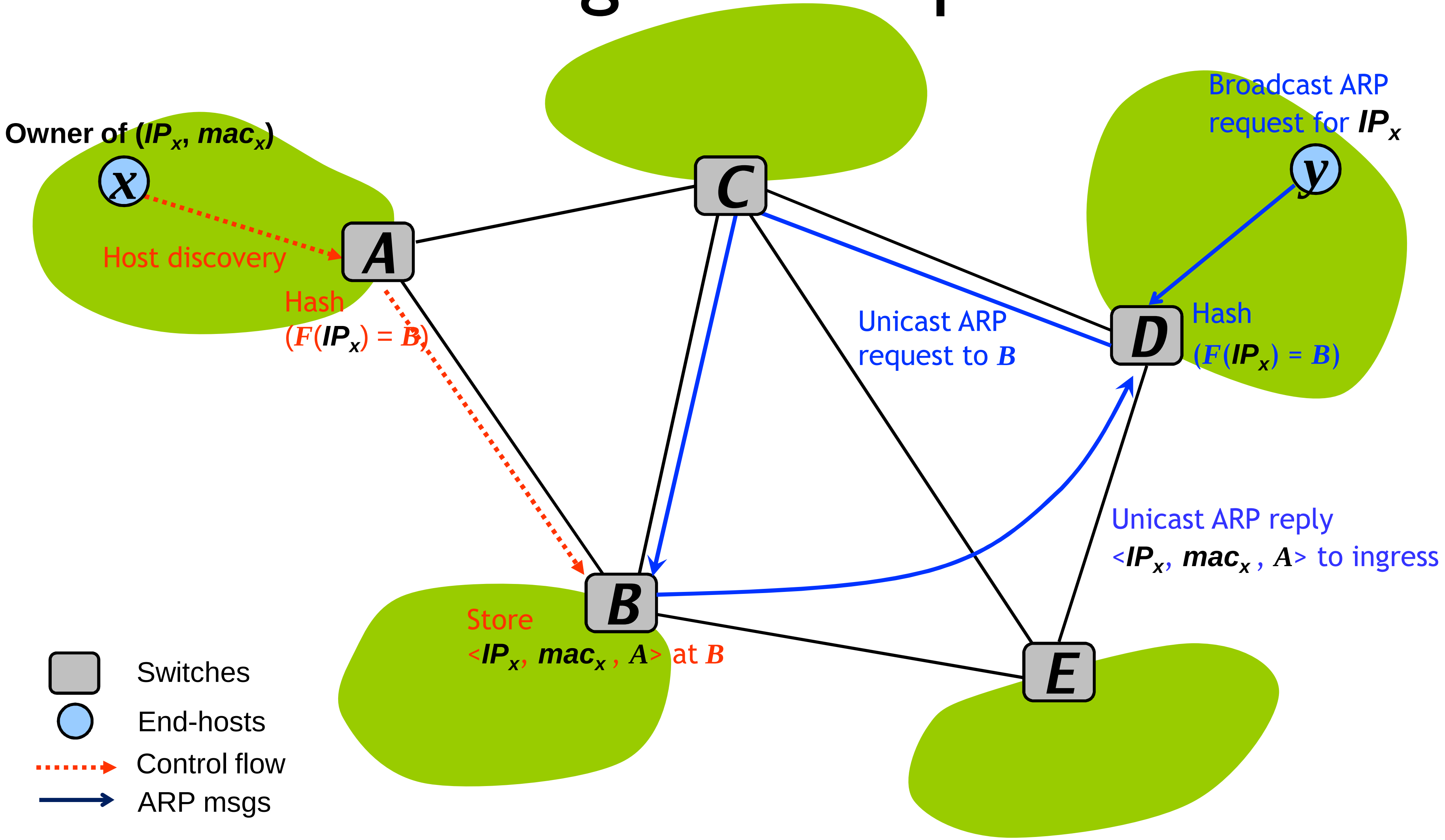
Handling host dynamics

- Events NOT modifying the set of live switches
  - E.g., most link failure/recovery
  - LS routing
- Events modifying the set of live switches
  - E.g., switch failure/recovery
  - ***F*** works differently after a change
  - Two simple operations
    - If $\boldsymbol{F}_{new}(k)$!= $\boldsymbol{F}_{old}(k)$, owner re-publishes to $\boldsymbol{F}_{new}(k)$
    - Remove any <k, v> published by non-existing owners

Handling host dynamics



Handling ARP requests



Enhancements

- **Goal:** Dealing with switch-level heterogeneity
- **Sol:** Virtual switches

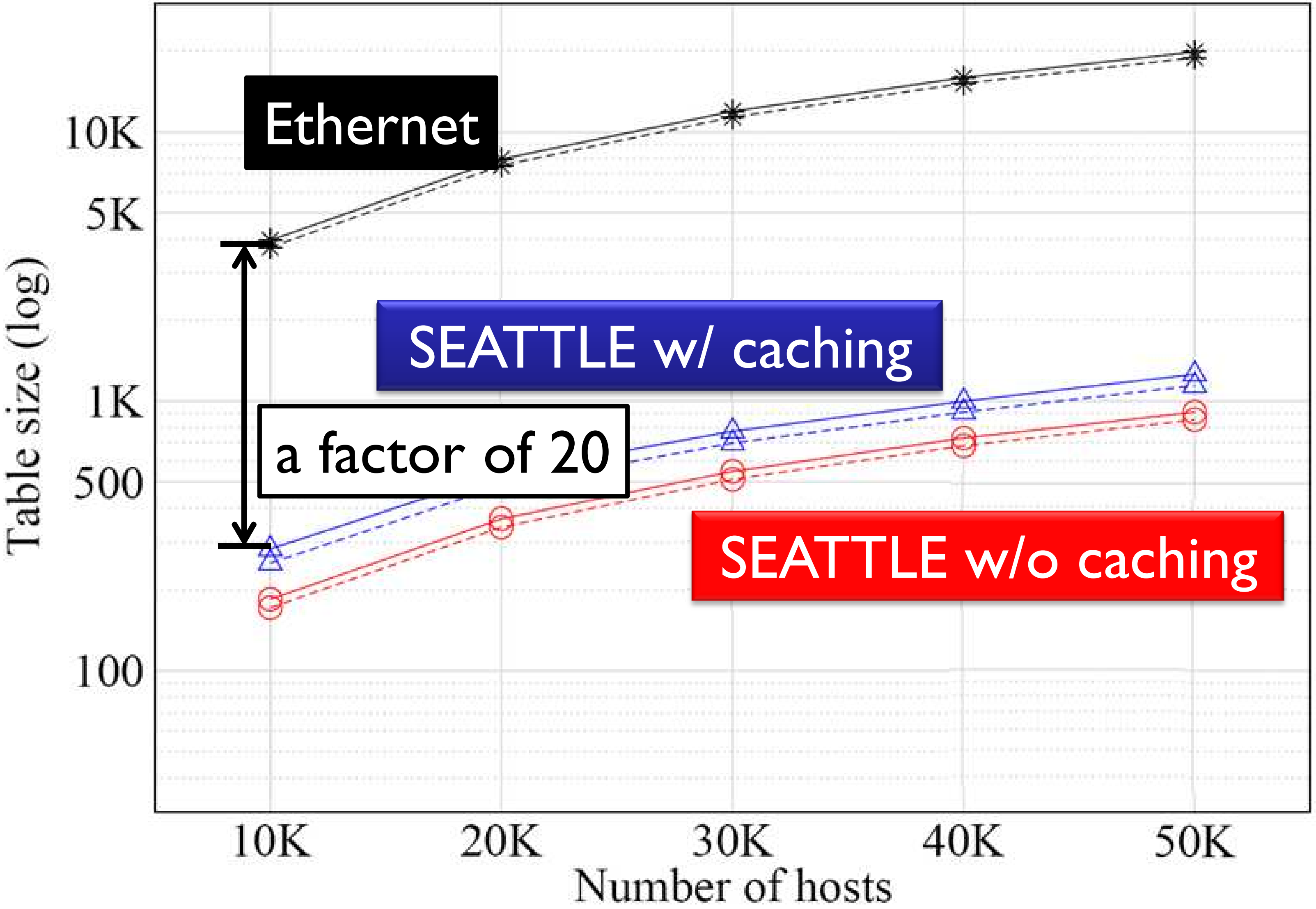
- **Goal:** Attaining very high availability of resolution
- **Sol:** Replication via multiple hash functions

- **Goal: Dividing administrative control to sub-units**
- **Sol:** Multi-level one-hop DHT

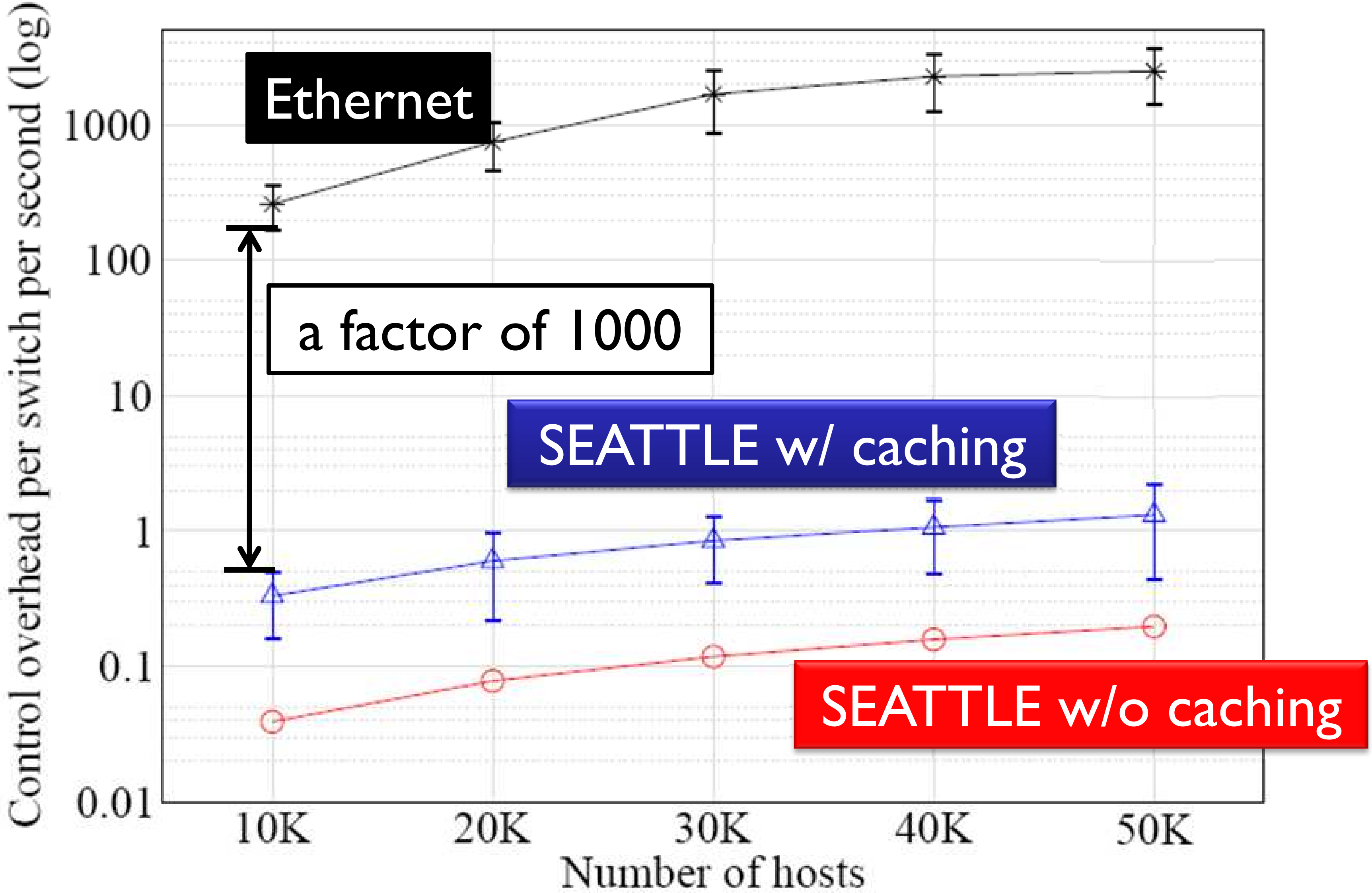
Performance Evaluation

- Packet-level Simulation
  - Synthetic traffic based on real traces from LBNL
    - Inflated the trace while preserving original properties
  - Real topologies from campus, data centers, and ISPs
- Emulation with prototype switches
  - Click/XORP implementation

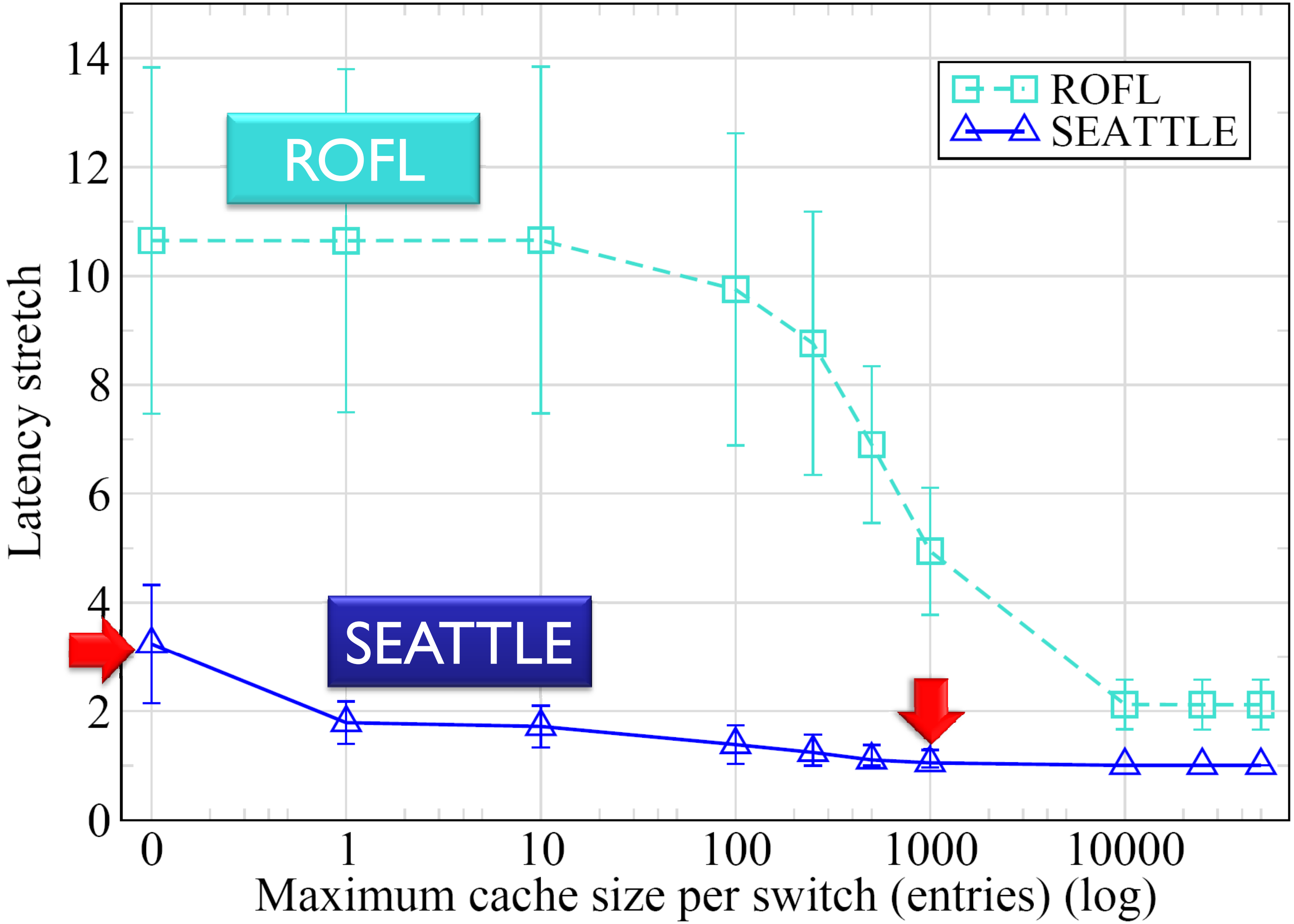
Amount of Routing State



Control Overhead



Latency Stretch



Conclusion

- SEATTLE is a plug-and-playable network architecture ensuring both scalability and efficiency
- One-hop DHT coupled with LS routing
- Reactive location resolution and caching
- Shortest-path forwarding
- Discussion
  - Higher delay stretch
  - Limited switch-level scale
  - Failure