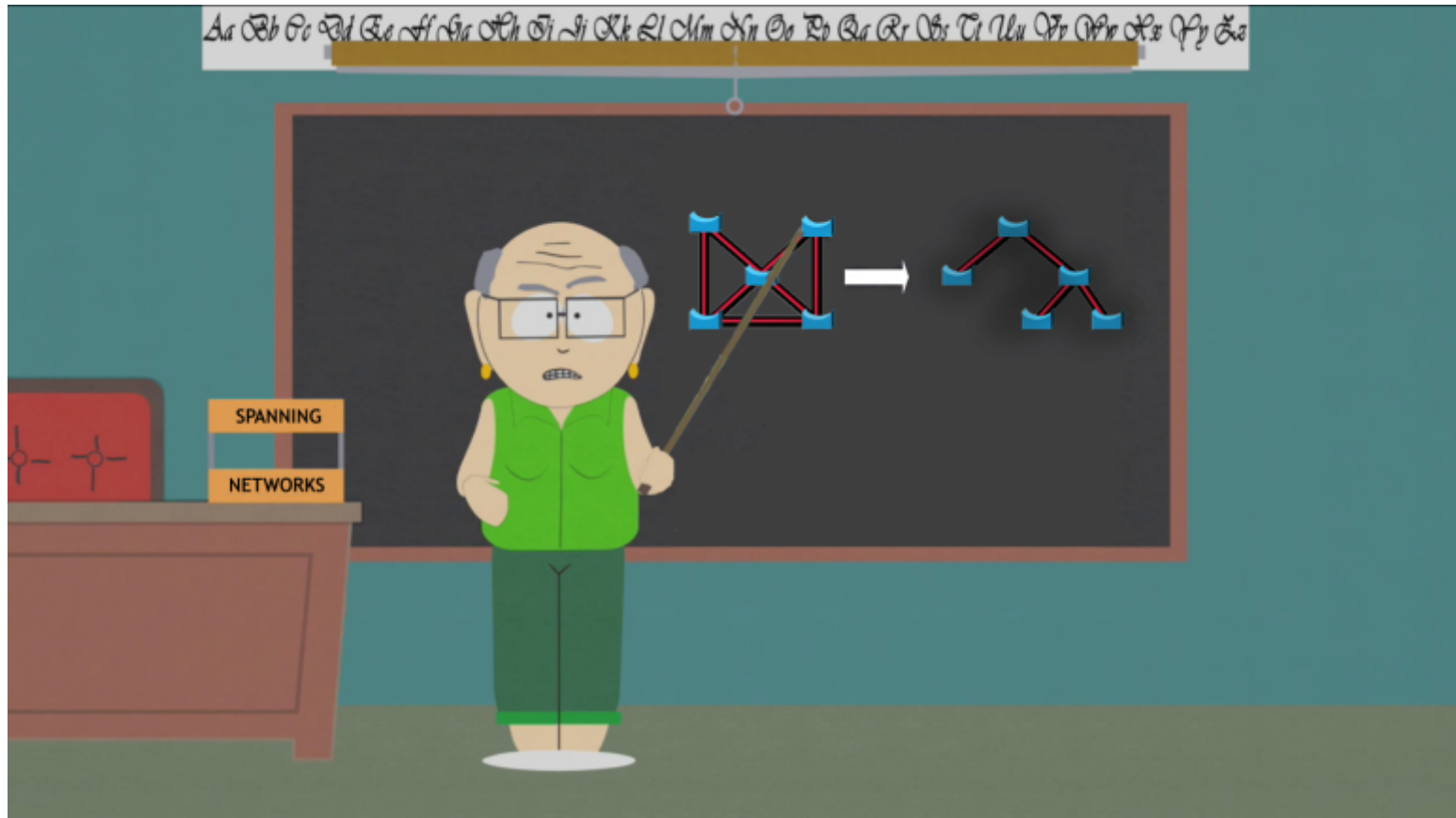


"Shortest path forwarding using OpenFlow"

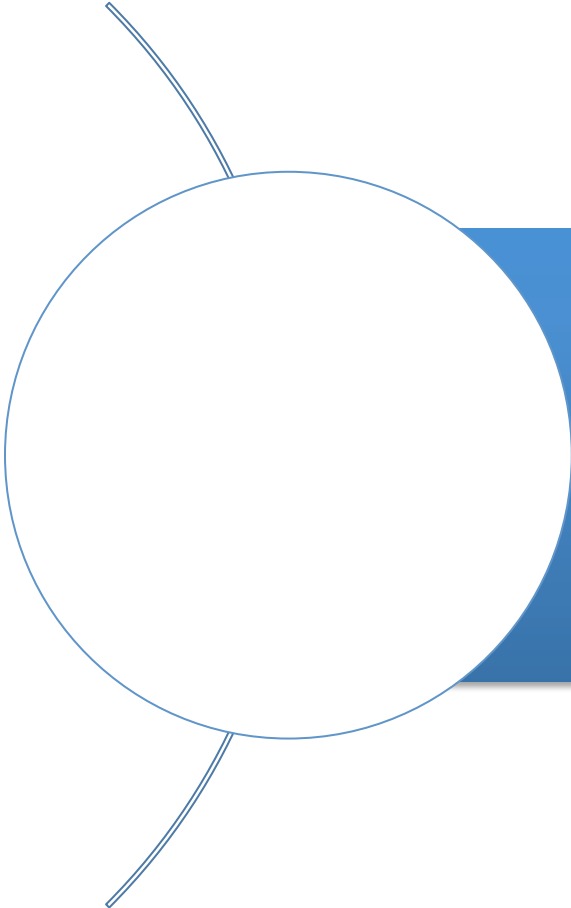
Iwan Hoogendoorn & Joris Soeurt

Supervisor: Ronald van der Pol

Wednesday 8 February 2012

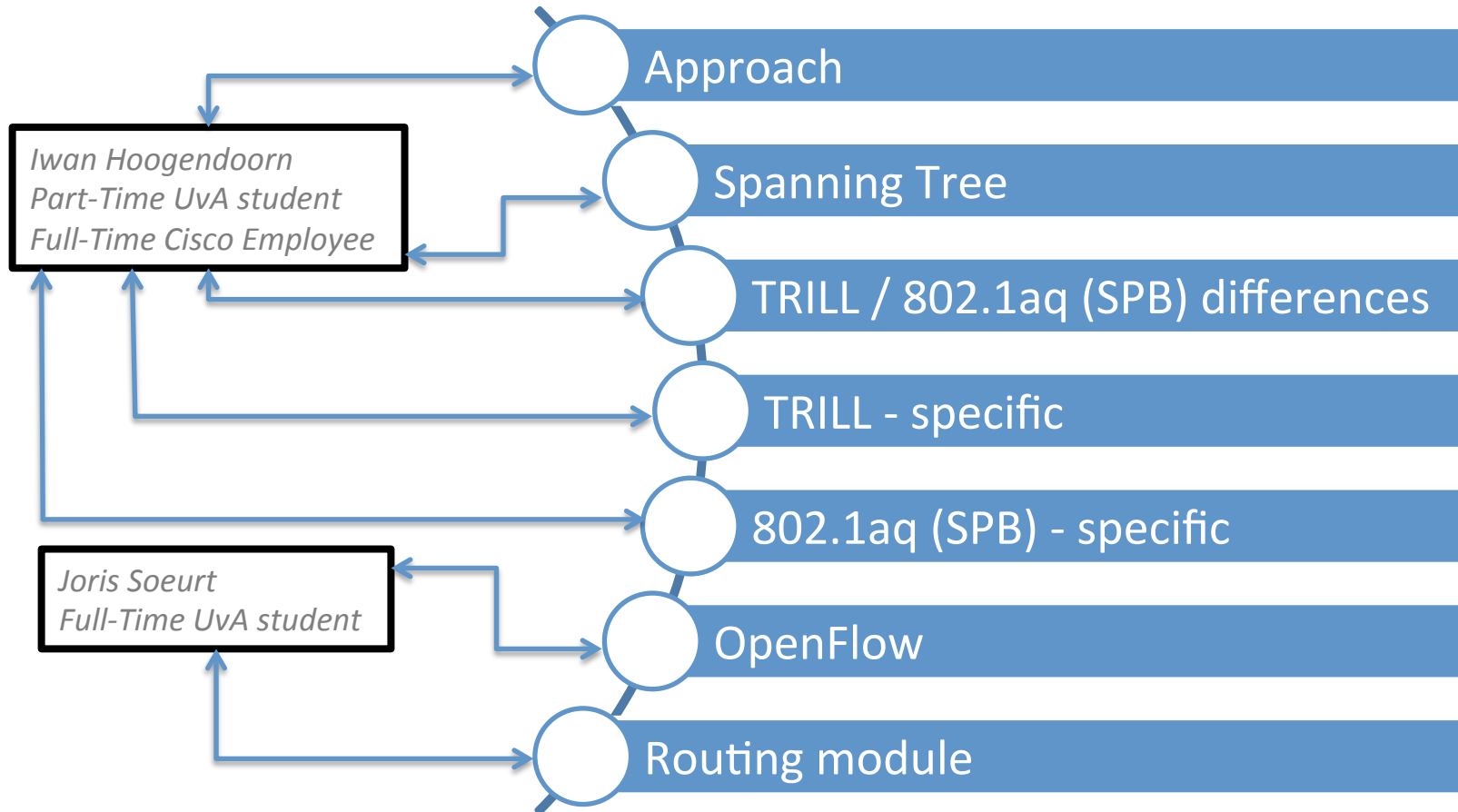


Research Question



In what way can shortest path bridging be implemented using OpenFlow?

Agenda



Approach

- Implement a (basic) shortest path bridging algorithm using OpenFlow
 - First see how TRILL / 802.1aq implement this
- Discovered existing routing module
 - No annotation in code (written in C)
 - Very little, outdated documentation
- Explored and tested this module
- Created improvement proposal

Spanning Tree – Why – What does it do?

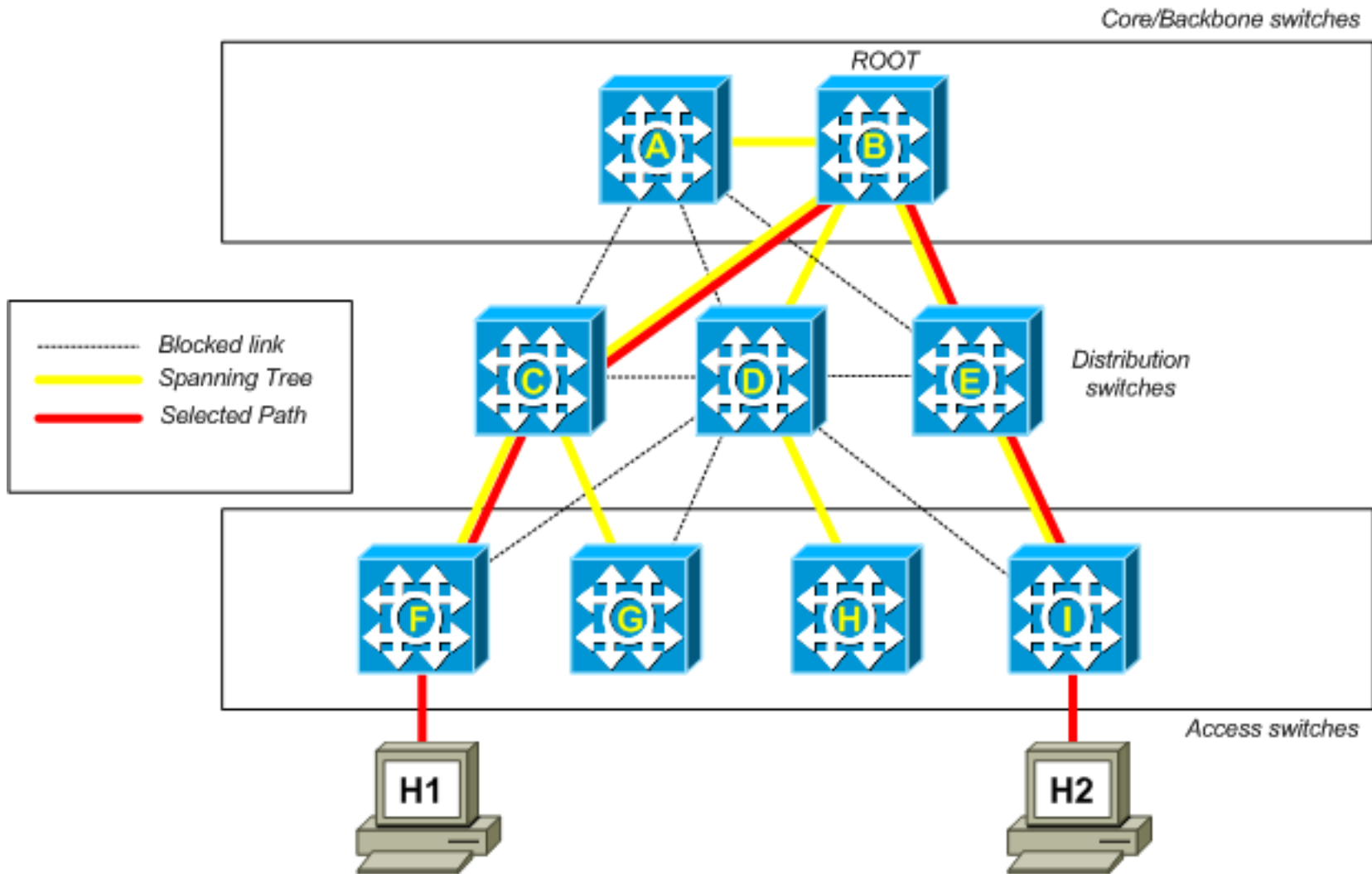
Radia Perlman



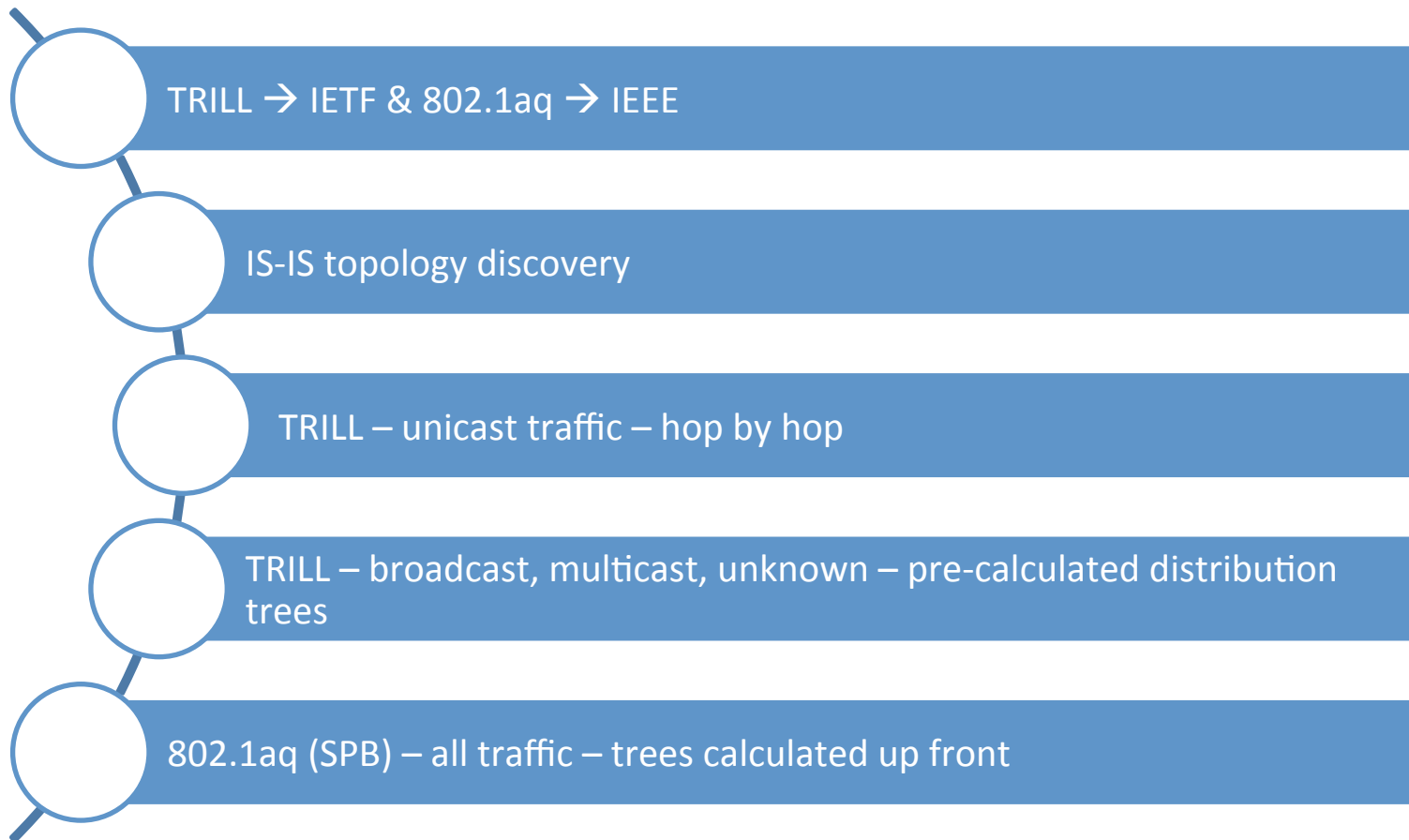
Wikipedia

“The Spanning Tree Protocol (STP) is a network protocol that ensures a loop-free topology for any bridged Ethernet local area network.”

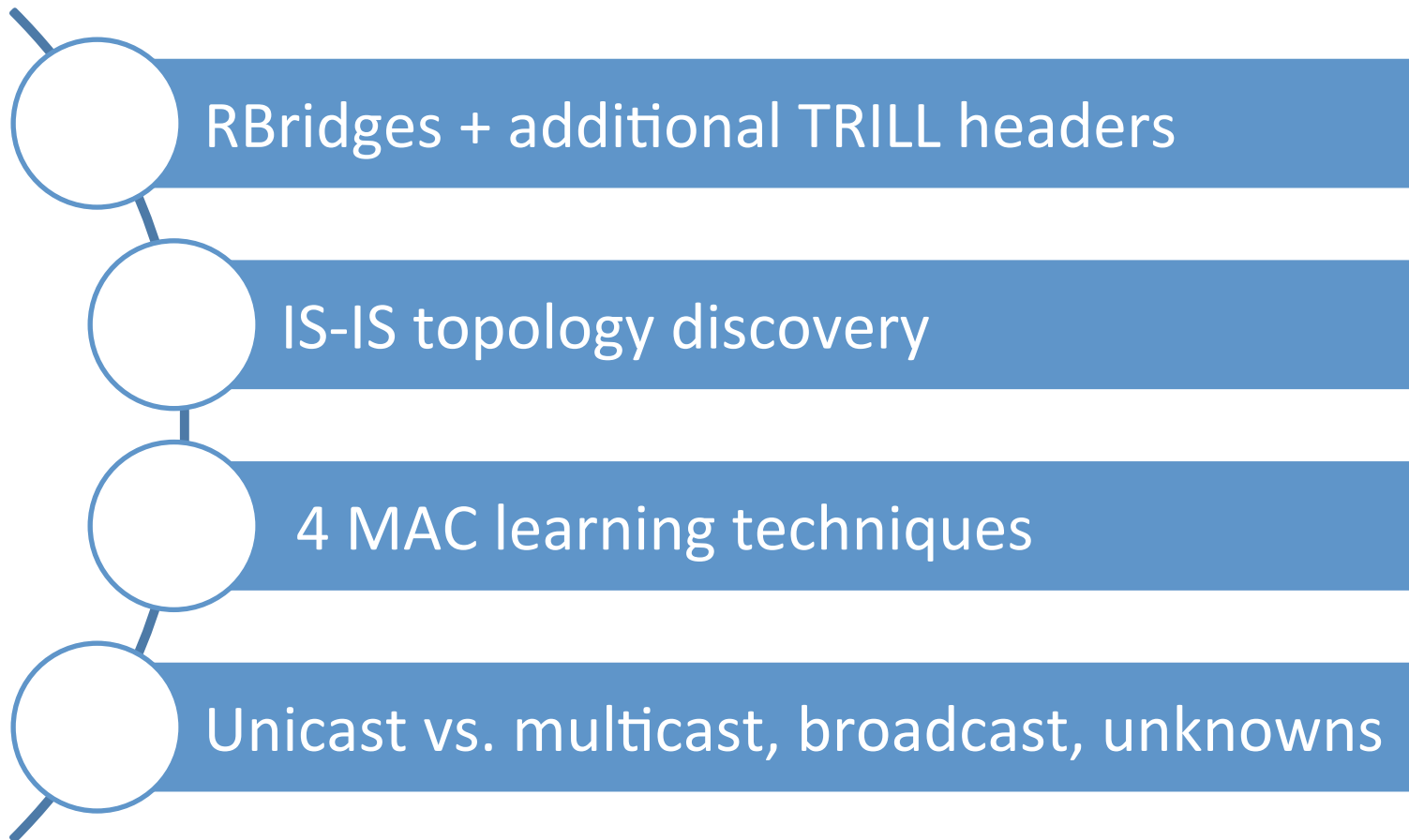
Spanning Tree



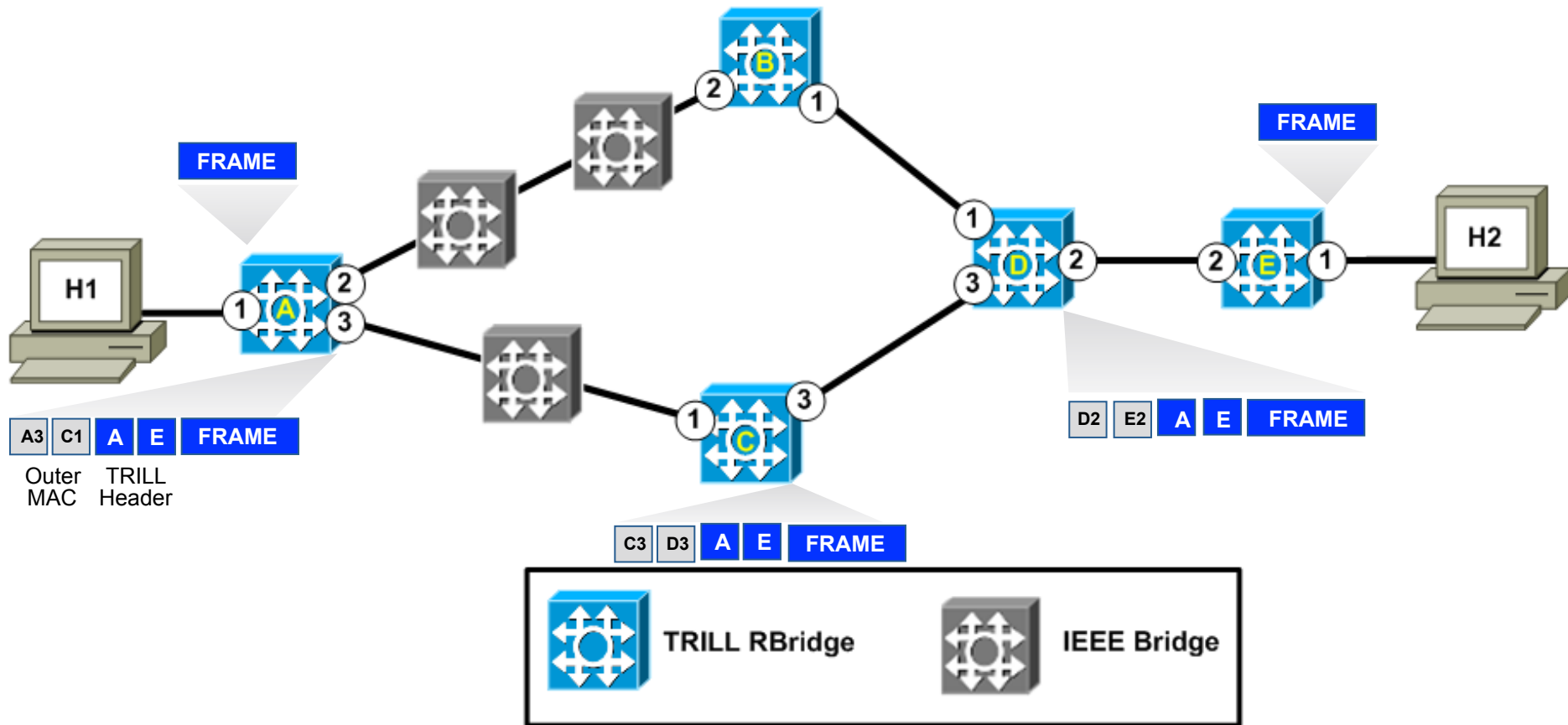
TRILL/802.1aq – Why better?



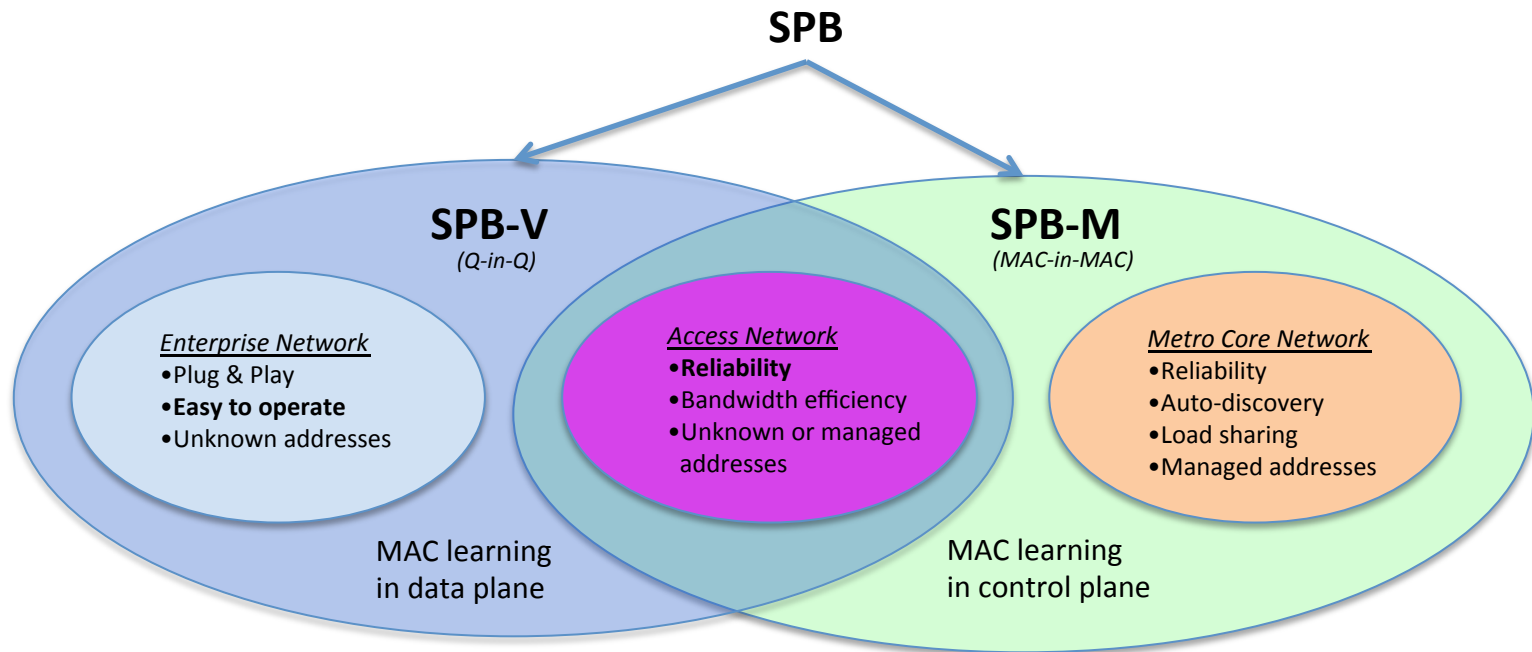
TRILL - concept



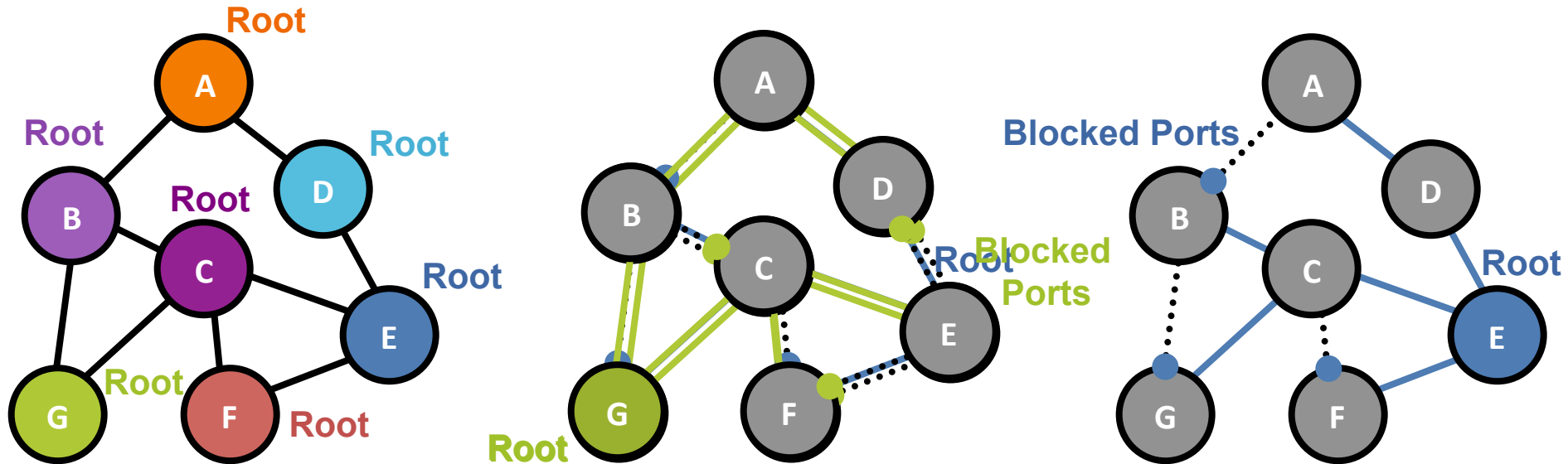
TRILL - diagram



802.1aq (SPB) - Types



802.1aq (SPB) - concept



- Each bridge is the “root” of a separate shortest path tree instance
- Bridge G is the root of the green tree
- Bridge E is the root of the blue tree
- Both trees are active AND symmetric at all times

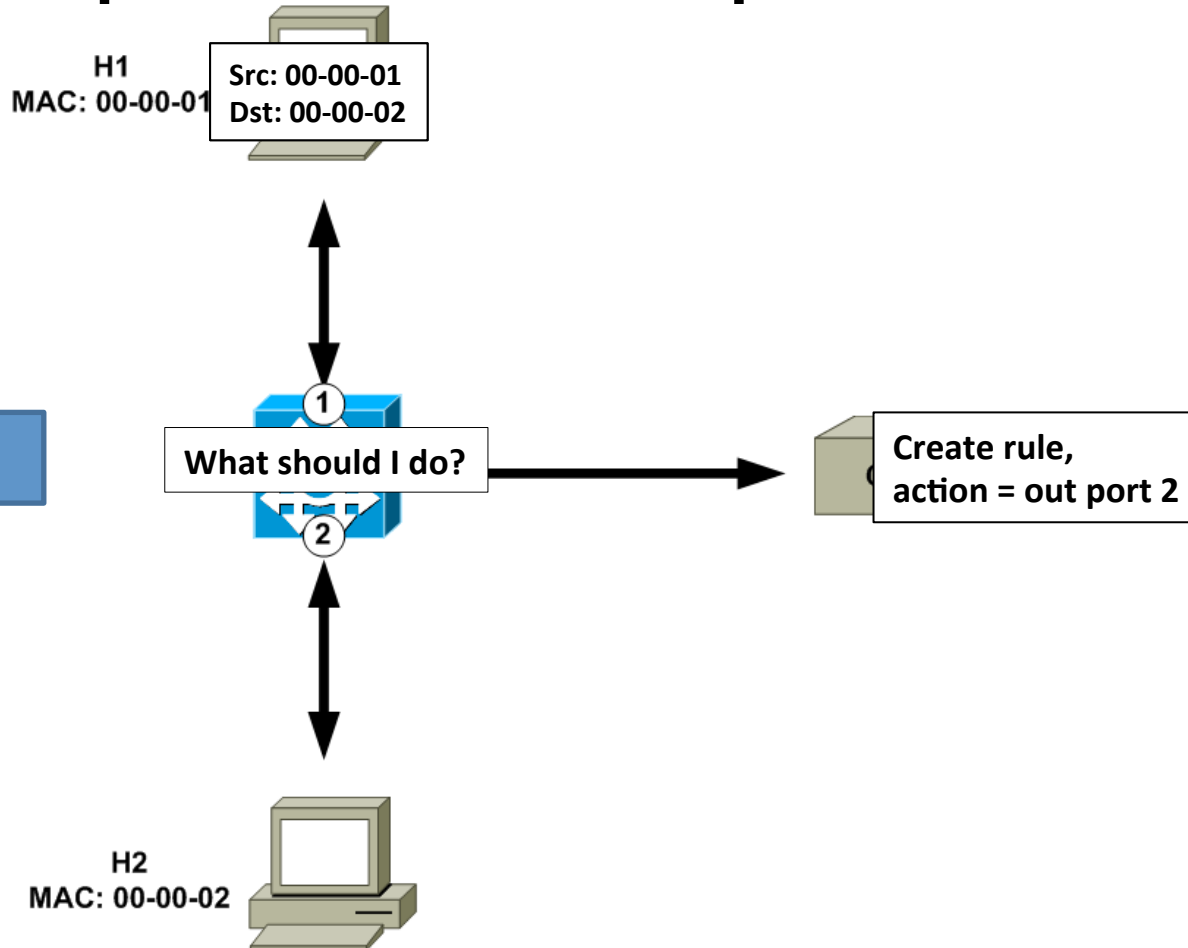
Introduction to OpenFlow

- Vendors generally don't like to make their firmware open to outsiders.
- No possibility to test new algorithms
- Stanford computer scientist Nick McKeown and colleagues developed a standard called OpenFlow

Introduction to OpenFlow

- Control and dataplane completely separated
- Control plane: Controller
 - NOX
 - Open source
 - Python programmable
- Data plane: OpenFlow aware switches
- Communication via standardized API.

Operation of OpenFlow



Flow table on switch

Src MAC	Dst MAC	Src IP	Dst IP	In port	Action
00-00-01	00-00-02			1	Out 2

Key features/advantages

- You are not limited by the functionality of the proprietary firmware of vendors.
- Computing power of a server
- Strength of a programming language
- You can implement any forwarding algorithm you want.

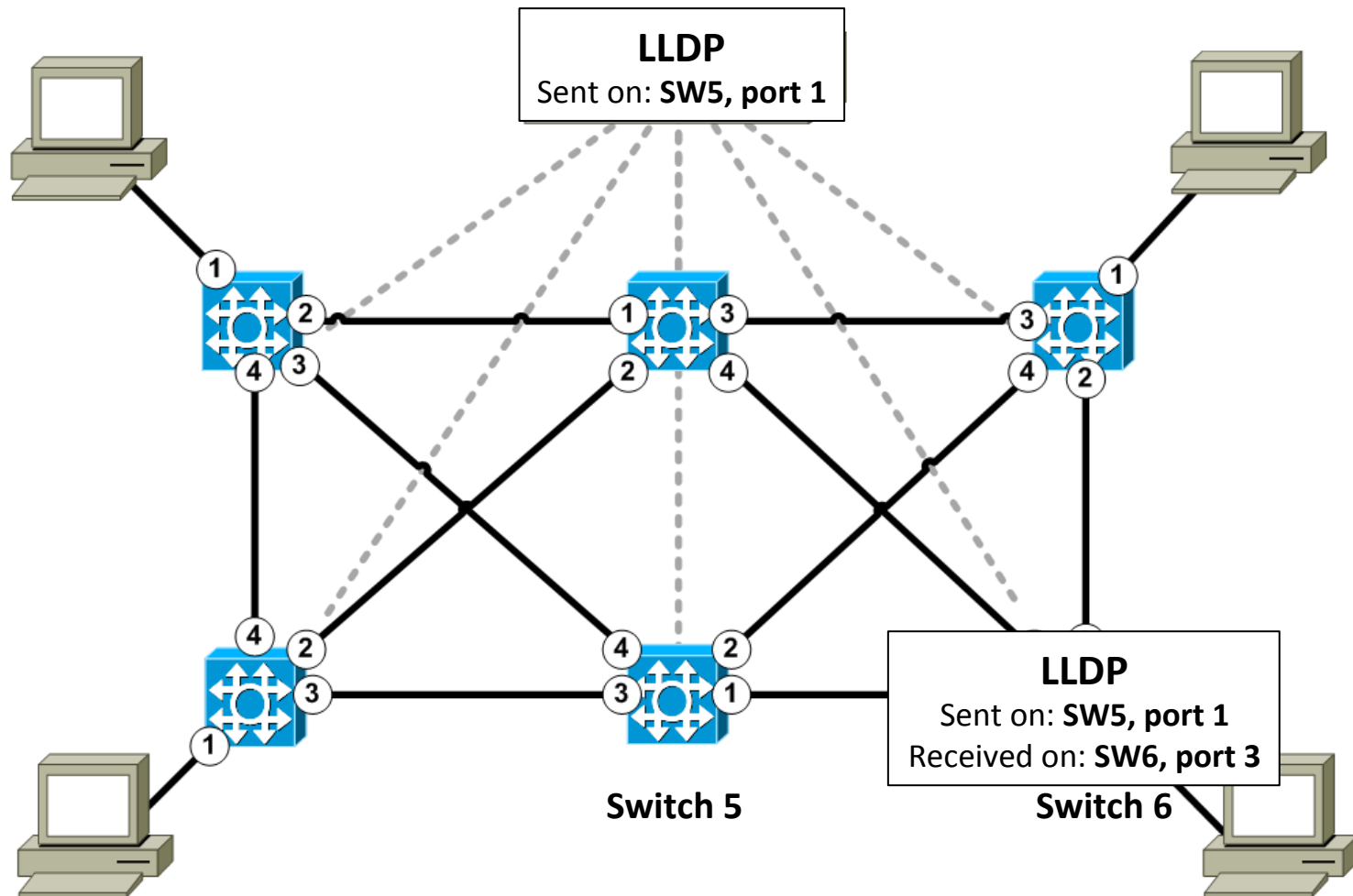
Testing...

- 6 OpenWrt switches with custom firmware
- Data gathered
 - Wireshark on controller
 - Tcpdump on hosts
 - NOX console output
- Information deduced
 - Communication
 - Innerworking
 - Link failover
 - Path determination

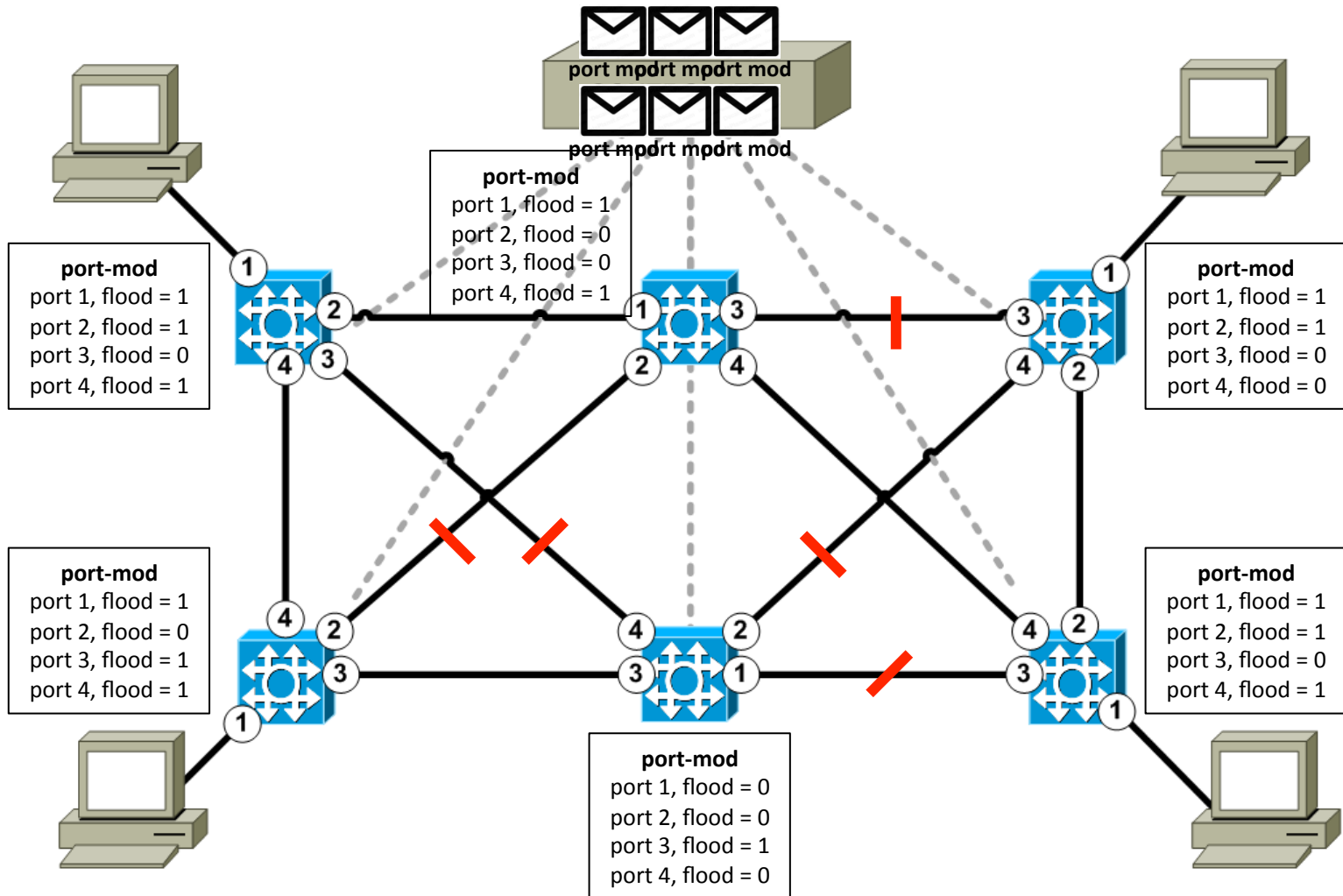
Parts of “routing” module

- Discovery module
 - Controller learns topology
- Spanning tree module
 - For broadcast/multicast/unknown unicast frames
 - Frames are flooded
- Shortest path module
 - For unicast frames
 - Frames are “routed”

Operation of discovery (1)



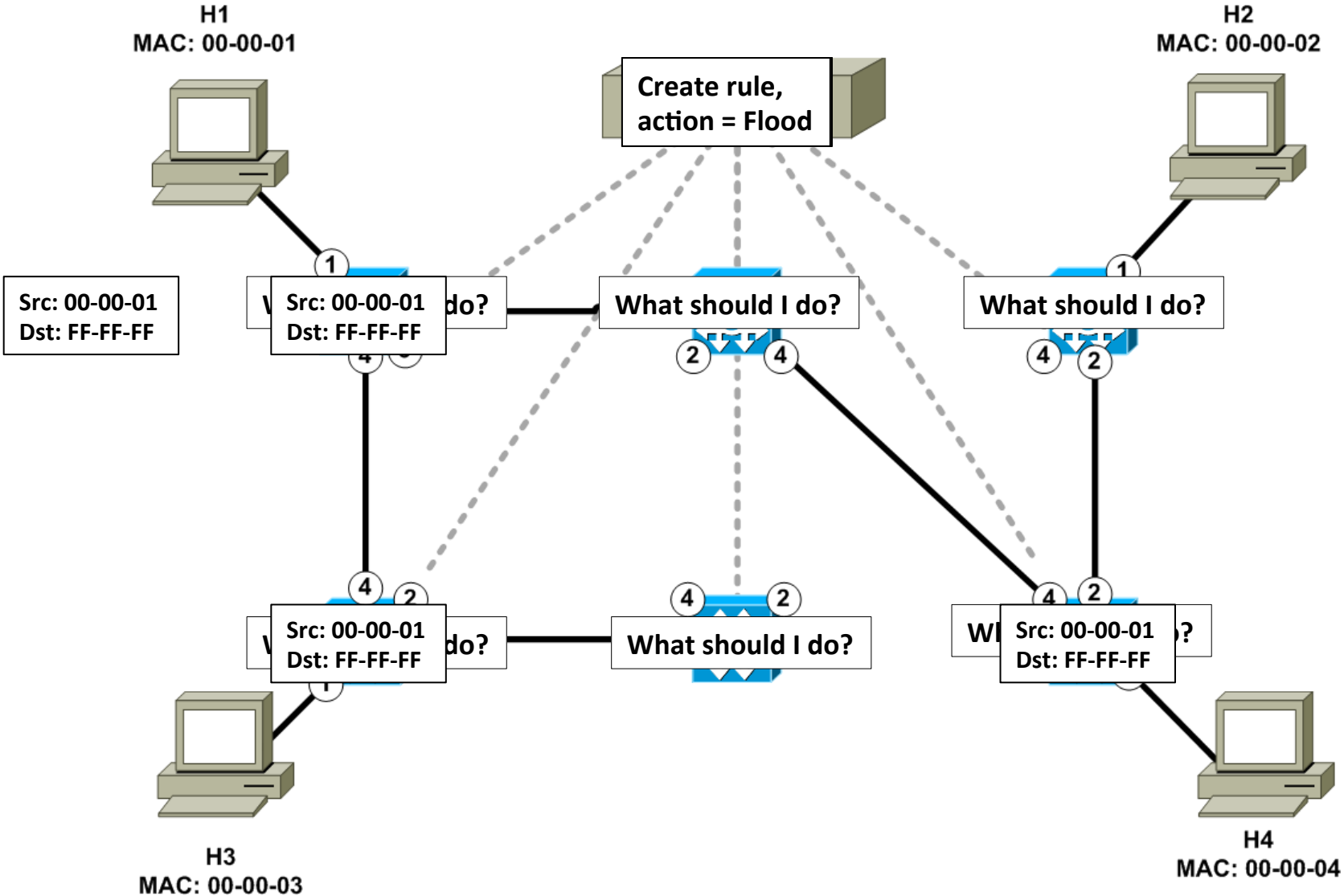
Spanning tree module



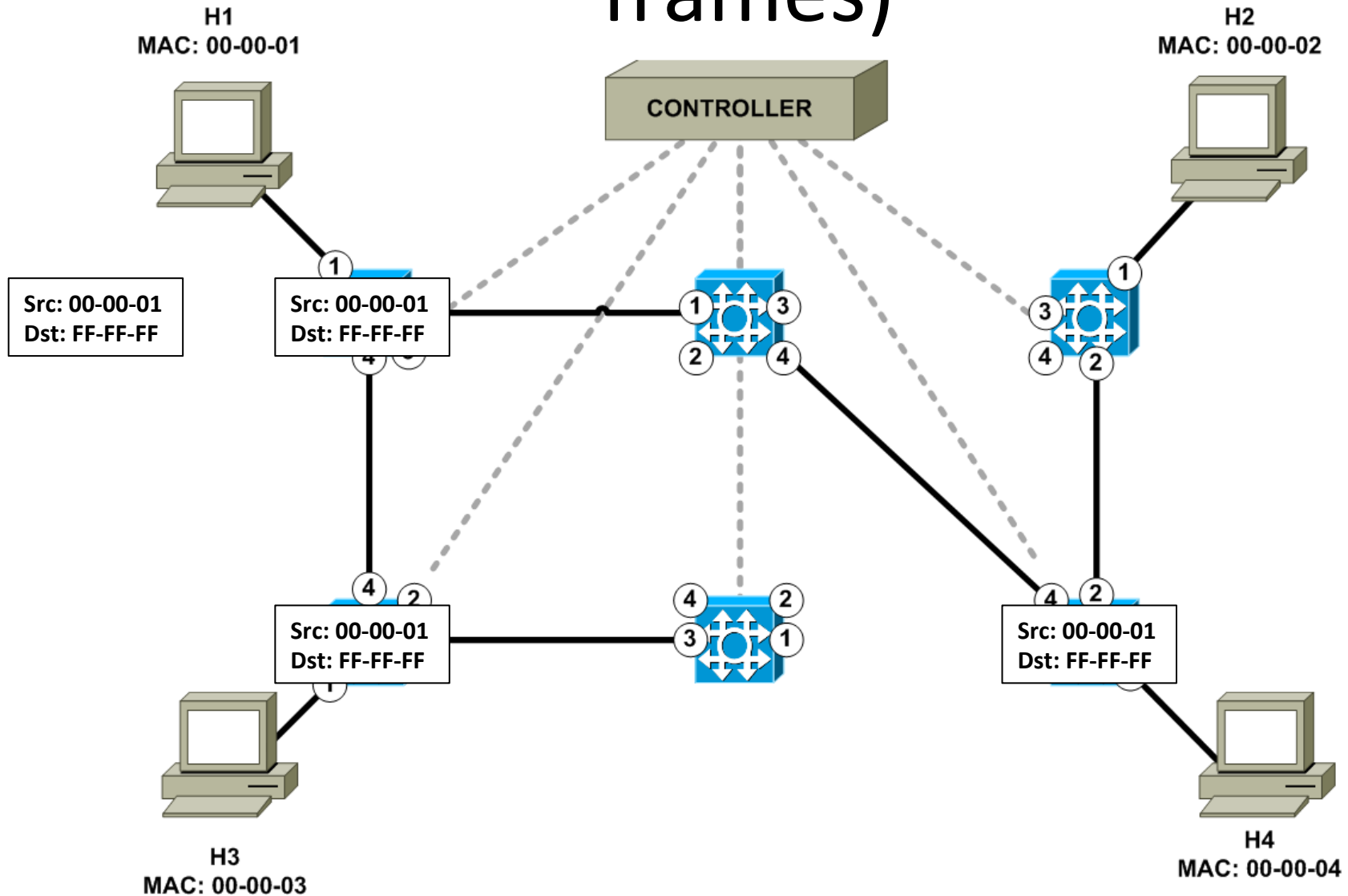
Broadcast / multicast / unknown

- Rule with action = flood
- Frames are flooded out all ports, except for....
 - Originating port
 - Port with flood flag set to disabled

Operation of flooding (first frame)



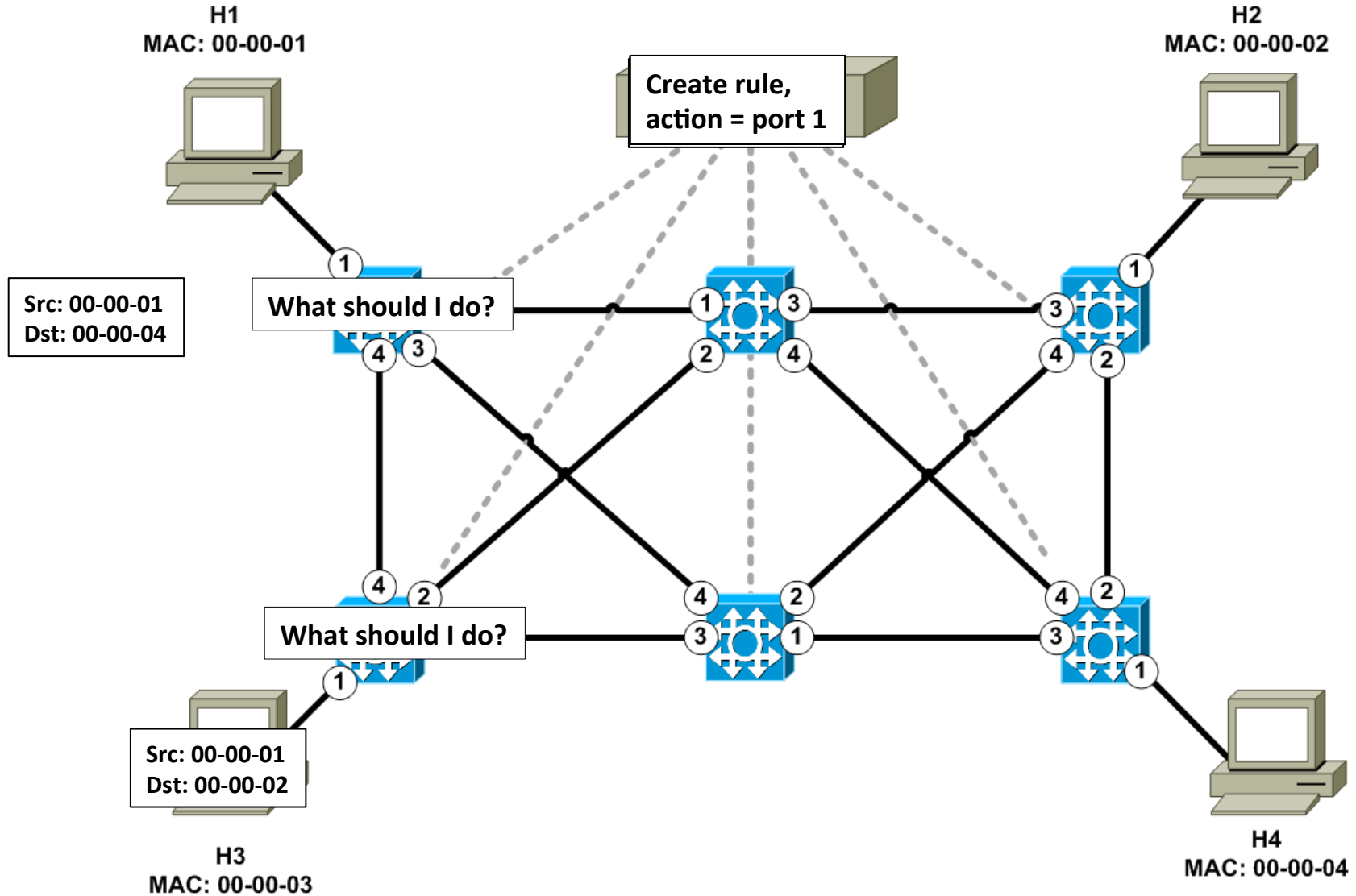
Operation of flooding (next frames)



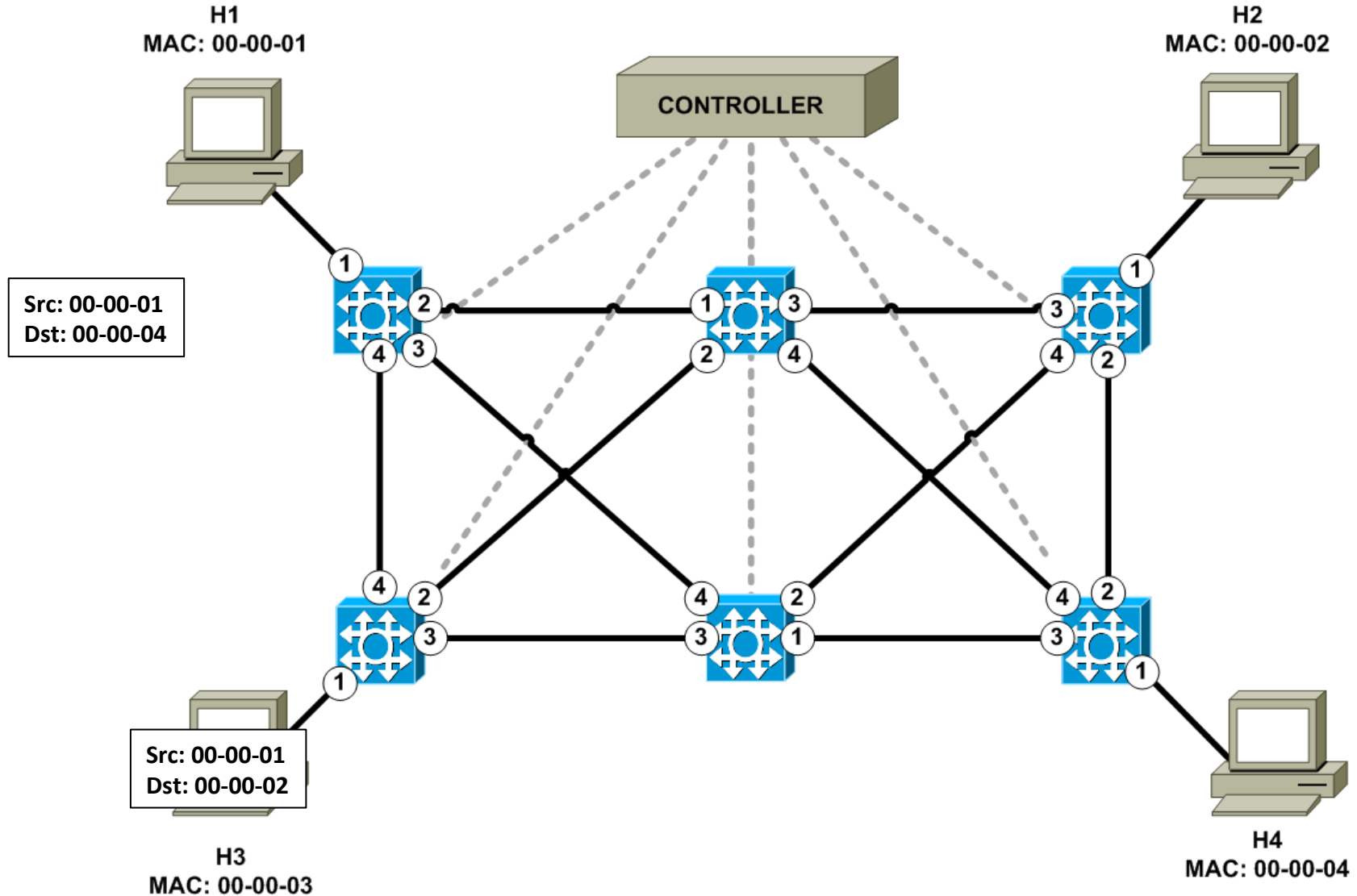
Unicast

- Controller knows location of hosts (registered by authenticator module)
- Can program complete path in advance

Operation of unicast (first frame)



Operation of unicast (next frames)



Shortcomings & improvement

- “Routing” module
 - Loadbalancing algorithm not optimal
 - Link failover not optimal
 - Instable (loops)
- Improvement proposal
 - Pseudocode

Comparison

- Centralized instead of distributed
 - No IS-IS needed
 - But, shortest path algorithm needed
- No standard (like IETF / IEEE)
- Spanning tree programmed to port property

Conclusion

- Powerful but not yet powerful enough?
 - Still in development (1.2 in March, 1.3 in April)
 - Version 1.2:
 - failover group
 - master/slave controller
- OpenFlow is generic, 802.1aq / TRILL are specific

Questions?