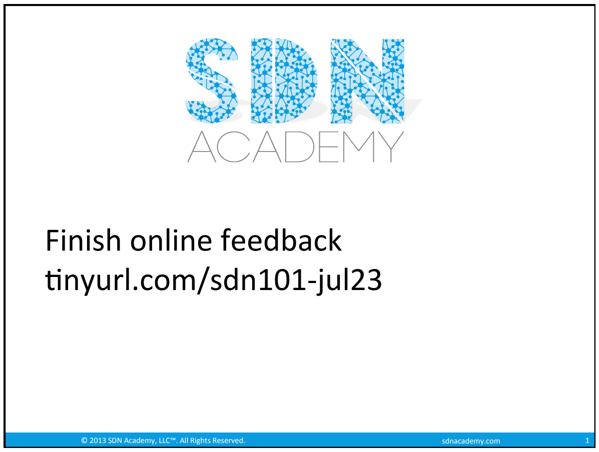
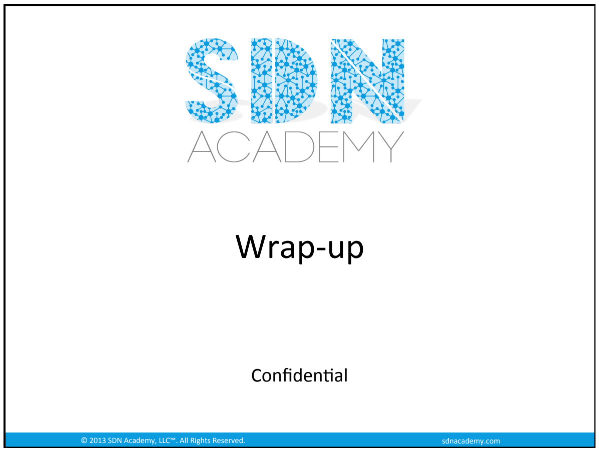


Finish online feedback
tinyurl.com/sdn101-jul23

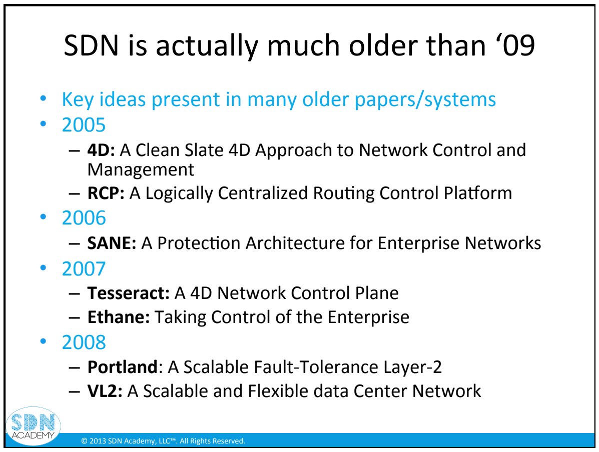
© 2013 SDN Academy, LLC™. All Rights Reserved. sdnacademy.com 1



Wrap-up

Confidential

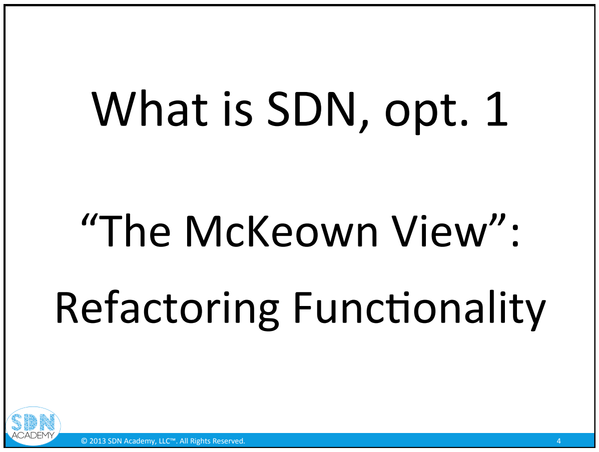
© 2013 SDN Academy, LLC™. All Rights Reserved. sdnacademy.com



SDN is actually much older than '09

- Key ideas present in many older papers/systems
- 2005
  - **4D:** A Clean Slate 4D Approach to Network Control and Management
  - **RCP:** A Logically Centralized Routing Control Platform
- 2006
  - **SANE:** A Protection Architecture for Enterprise Networks
- 2007
  - **Tesseract:** A 4D Network Control Plane
  - **Ethane:** Taking Control of the Enterprise
- 2008
  - **Portland**: A Scalable Fault-Tolerance Layer-2
  - **VL2:** A Scalable and Flexible data Center Network

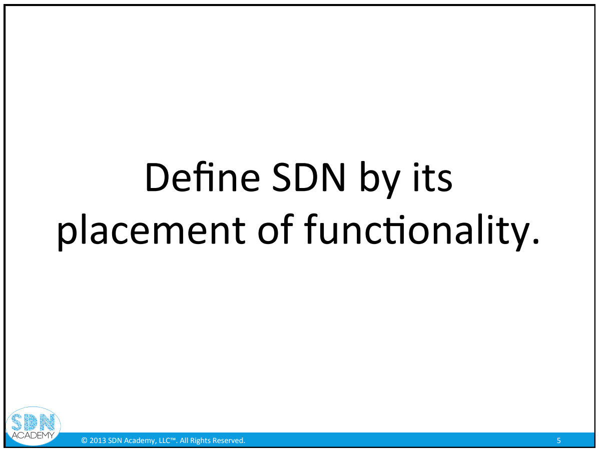
© 2013 SDN Academy, LLC™. All Rights Reserved.



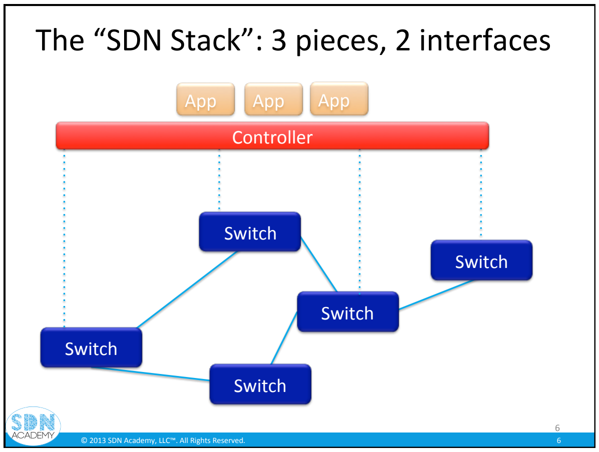
What is SDN, opt. 1

"The McKeown View": Refactoring Functionality

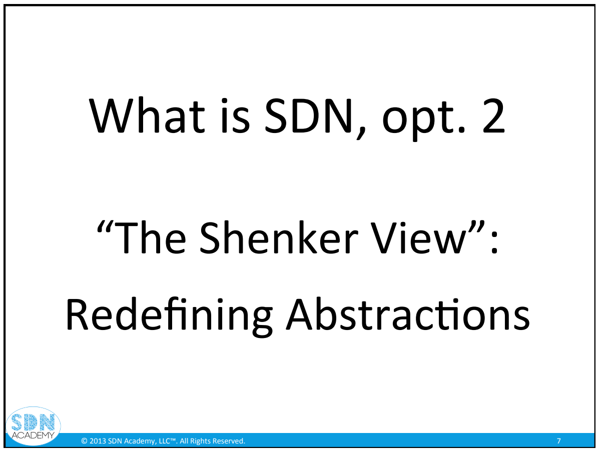
© 2013 SDN Academy, LLC™. All Rights Reserved. 4



Define SDN by its placement of functionality.



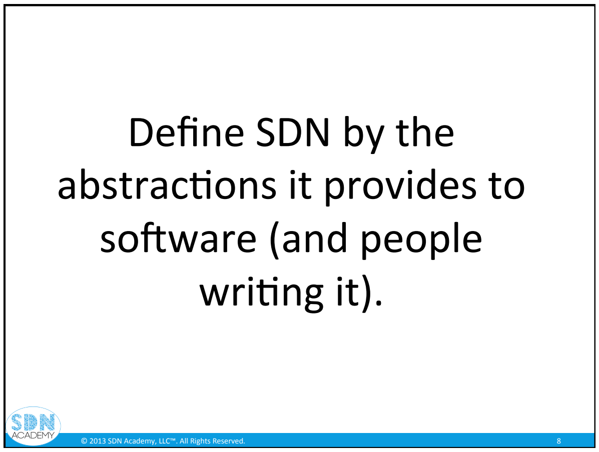
The "SDN Stack": 3 pieces, 2 interfaces



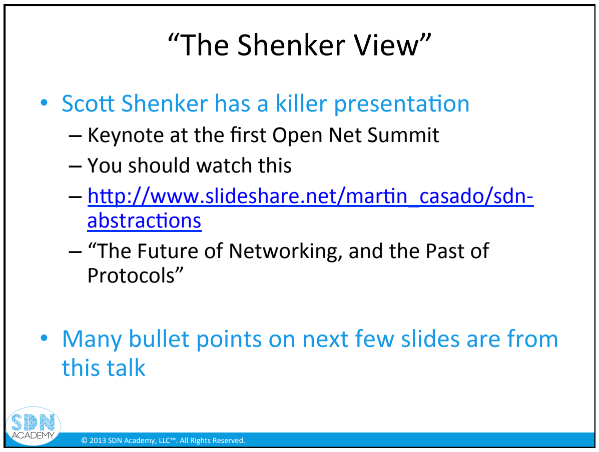
What is SDN, opt. 2

"The Shenker View":

Redefining Abstractions

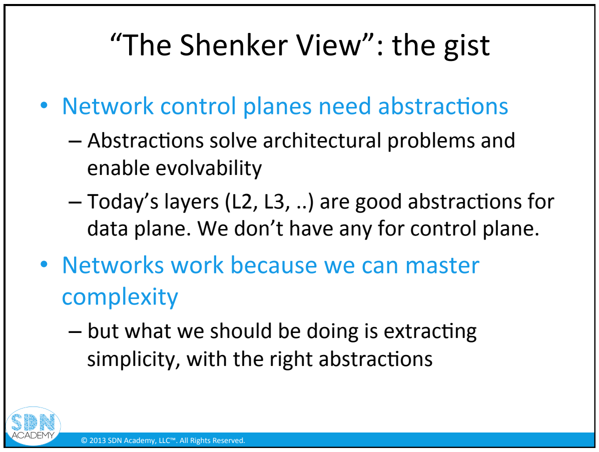


Define SDN by the abstractions it provides to software (and people writing it).



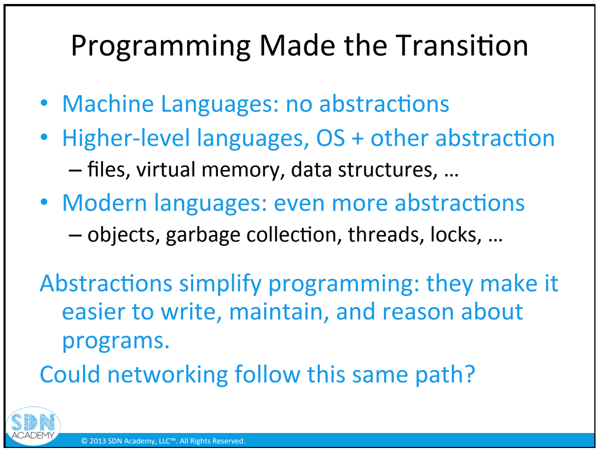
“The Shenker View”

- Scott Shenker has a killer presentation
  - Keynote at the first Open Net Summit
  - You should watch this
  - http://www.slideshare.net/martin_casado/sdn-abstractions
  - “The Future of Networking, and the Past of Protocols”
- Many bullet points on next few slides are from this talk



“The Shenker View”: the gist

- Network control planes need abstractions
  - Abstractions solve architectural problems and enable evolvability
  - Today’s layers (L2, L3, ..) are good abstractions for data plane. We don’t have any for control plane.
- Networks work because we can master complexity
  - but what we should be doing is extracting simplicity, with the right abstractions

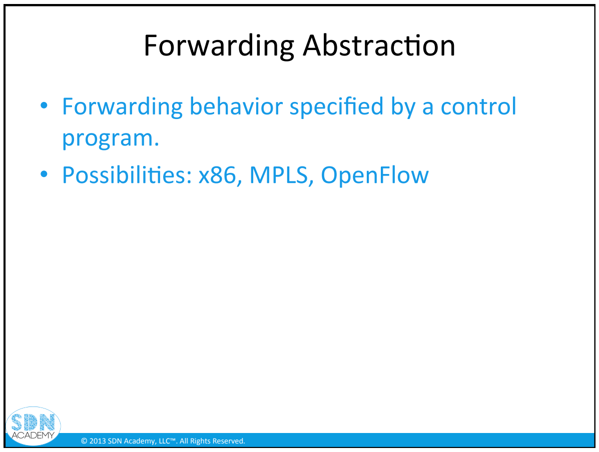


Programming Made the Transition

- Machine Languages: no abstractions
- Higher-level languages, OS + other abstraction
  - files, virtual memory, data structures, …
- Modern languages: even more abstractions
  - objects, garbage collection, threads, locks, …

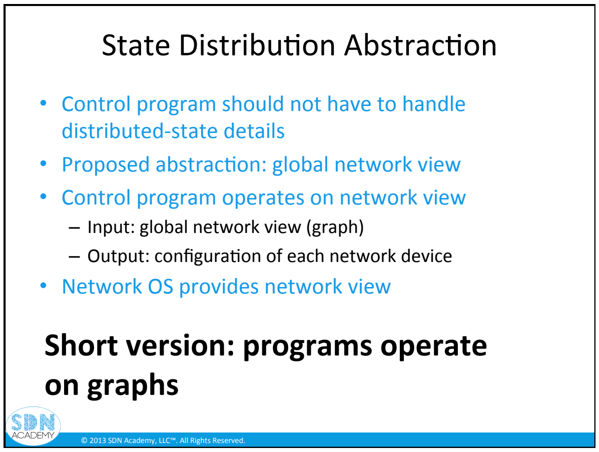
Abstractions simplify programming: they make it easier to write, maintain, and reason about programs.

Could networking follow this same path?



Forwarding Abstraction

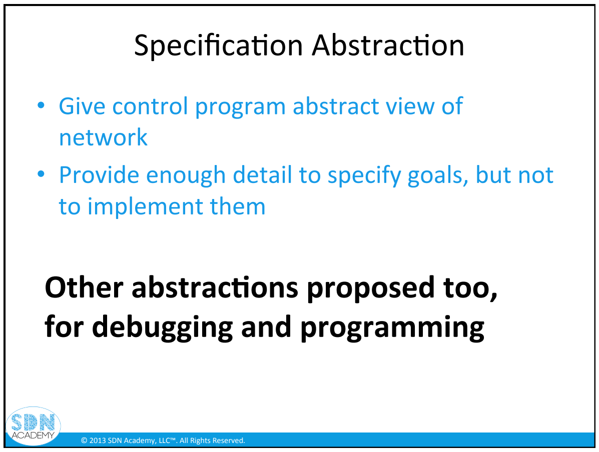
- Forwarding behavior specified by a control program.
- Possibilities: x86, MPLS, OpenFlow



State Distribution Abstraction

- Control program should not have to handle distributed-state details
- Proposed abstraction: global network view
- Control program operates on network view
  - Input: global network view (graph)
  - Output: configuration of each network device
- Network OS provides network view

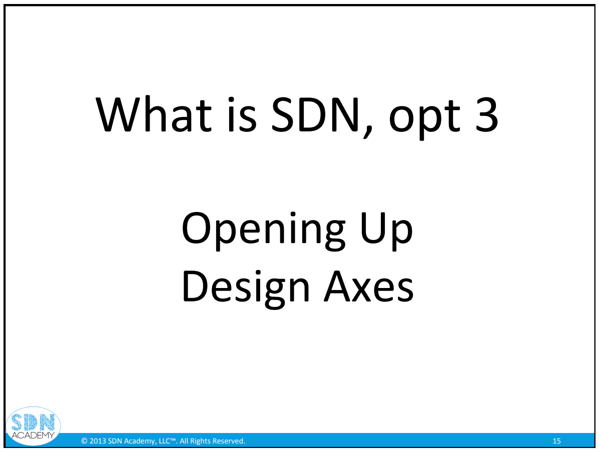
Short version: programs operate on graphs



Specification Abstraction

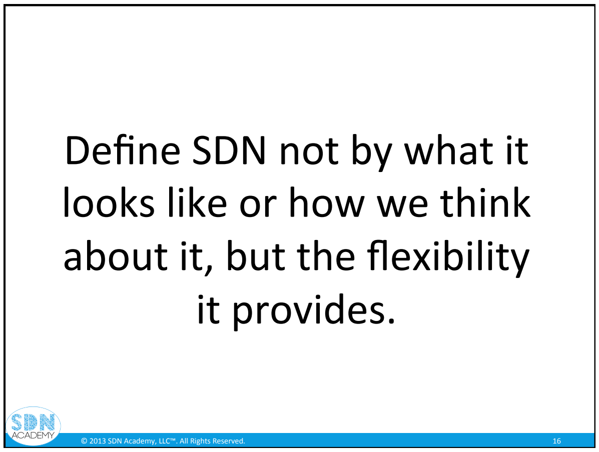
- Give control program abstract view of network
- Provide enough detail to specify goals, but not to implement them

Other abstractions proposed too, for debugging and programming

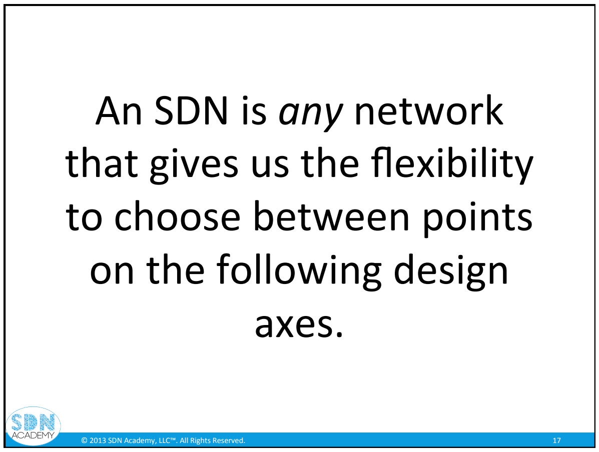


What is SDN, opt 3

Opening Up
Design Axes

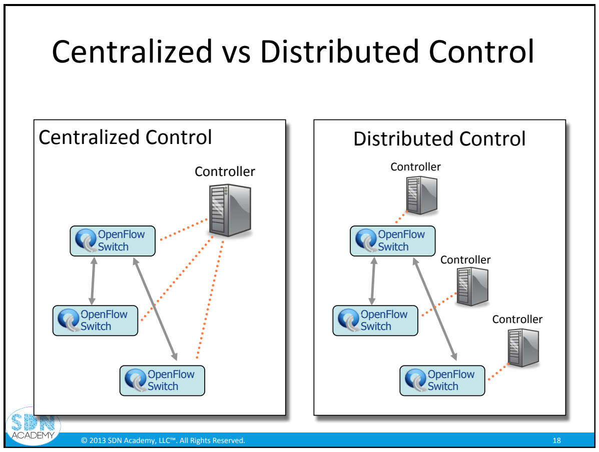


Define SDN not by what it looks like or how we think about it, but the flexibility it provides.



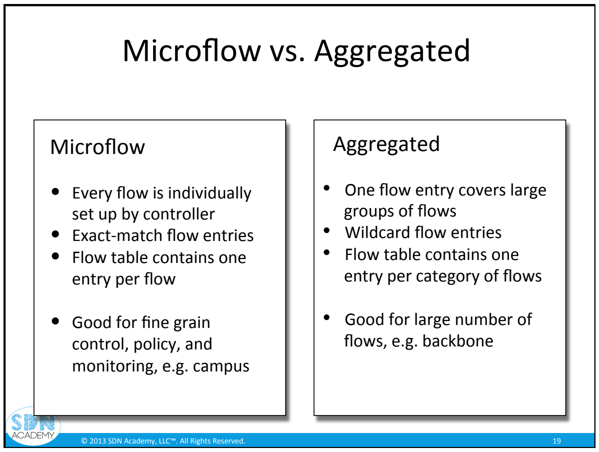
An SDN is *any* network that gives us the flexibility to choose between points on the following design axes.

17



Centralized vs Distributed Control

18



Microflow vs. Aggregated

Microflow

- Every flow is individually set up by controller
- Exact-match flow entries
- Flow table contains one entry per flow

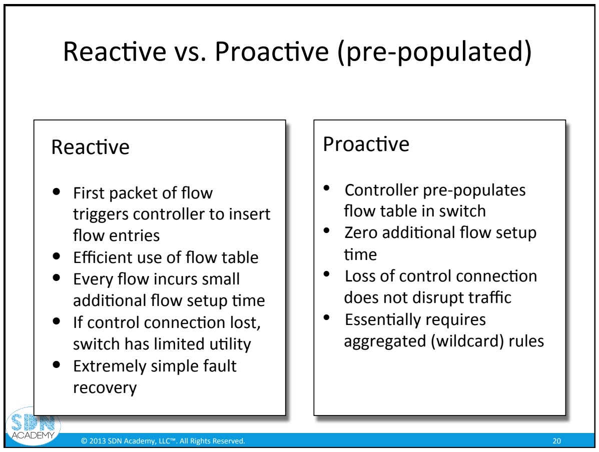
- Good for fine grain control, policy, and monitoring, e.g. campus

Aggregated

- One flow entry covers large groups of flows
- Wildcard flow entries
- Flow table contains one entry per category of flows

- Good for large number of flows, e.g. backbone

19



Reactive vs. Proactive (pre-populated)

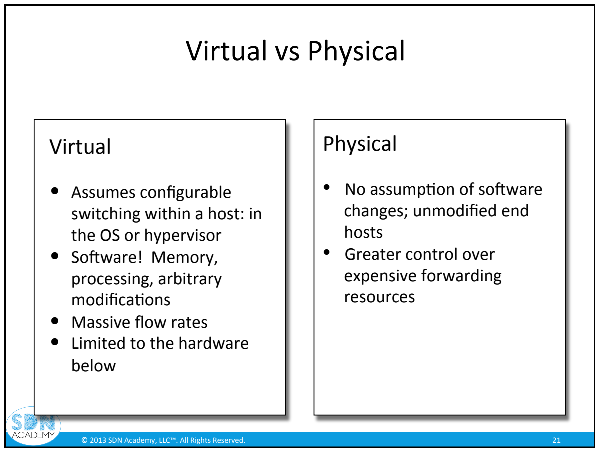
Reactive

- First packet of flow triggers controller to insert flow entries
- Efficient use of flow table
- Every flow incurs small additional flow setup time
- If control connection lost, switch has limited utility
- Extremely simple fault recovery

Proactive

- Controller pre-populates flow table in switch
- Zero additional flow setup time
- Loss of control connection does not disrupt traffic
- Essentially requires aggregated (wildcard) rules

20



Virtual vs Physical

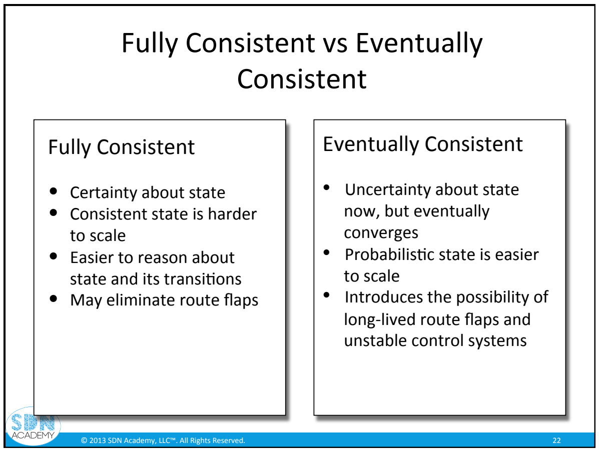
Virtual

- Assumes configurable switching within a host: in the OS or hypervisor
- Software! Memory, processing, arbitrary modifications
- Massive flow rates
- Limited to the hardware below

Physical

- No assumption of software changes; unmodified end hosts
- Greater control over expensive forwarding resources

21



Fully Consistent vs Eventually Consistent

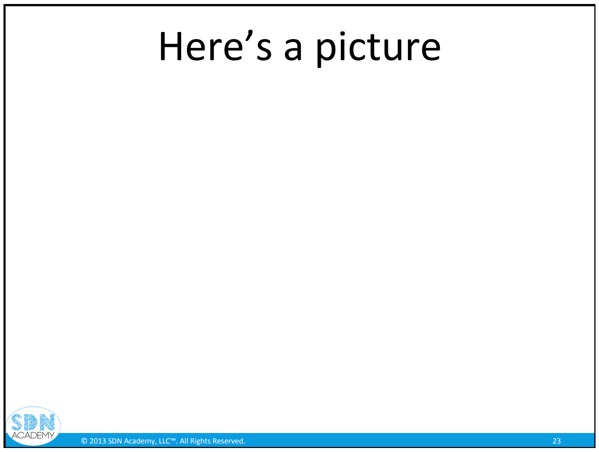
Fully Consistent

- Certainty about state
- Consistent state is harder to scale
- Easier to reason about state and its transitions
- May eliminate route flaps

Eventually Consistent

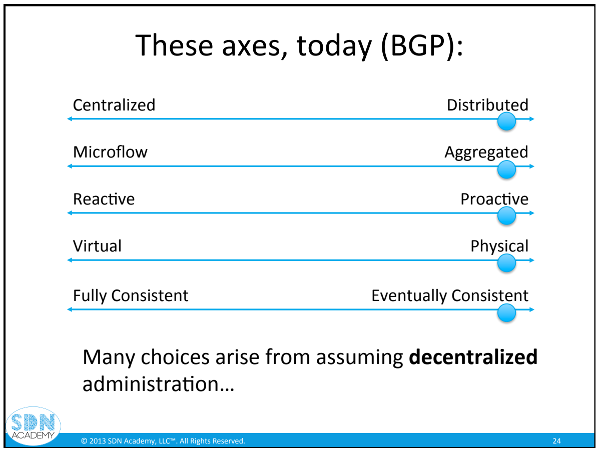
- Uncertainty about state now, but eventually converges
- Probabilistic state is easier to scale
- Introduces the possibility of long-lived route flaps and unstable control systems

22



Here's a picture

23



These axes, today (BGP):

Centralized — Distributed

Microflow — Aggregated

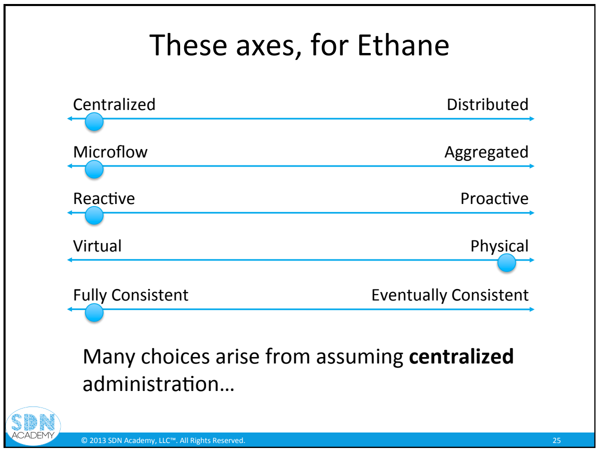
Reactive — Proactive

Virtual — Physical

Fully Consistent — Eventually Consistent

Many choices arise from assuming **decentralized** administration…

24

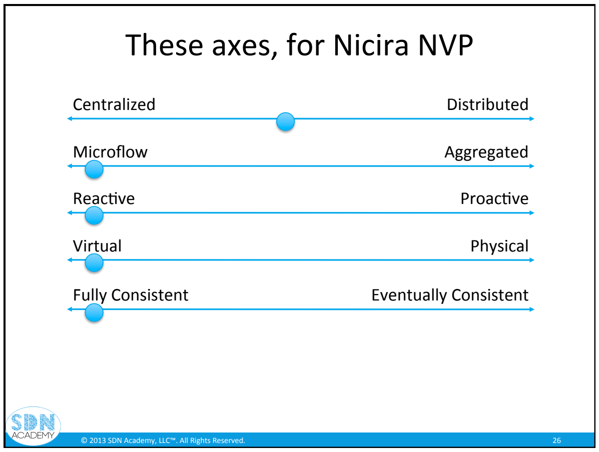


These axes, for Ethane

Centralized	Distributed
Microflow	Aggregated
Reactive	Proactive
Virtual	Physical
Fully Consistent	Eventually Consistent

Many choices arise from assuming **centralized** administration…

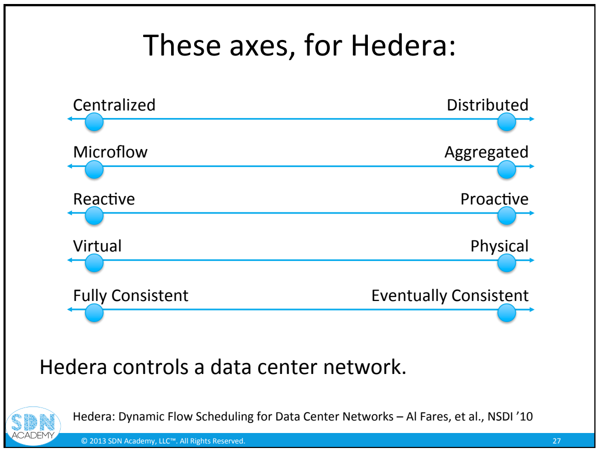
25



These axes, for Nicira NVP

Centralized	Distributed
Microflow	Aggregated
Reactive	Proactive
Virtual	Physical
Fully Consistent	Eventually Consistent

26



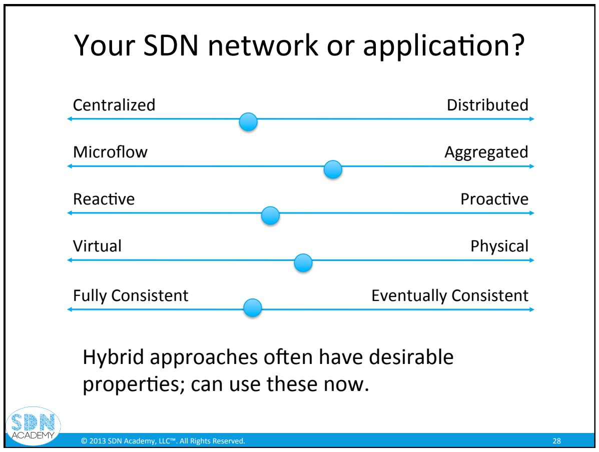
These axes, for Hedera:

Centralized	Distributed
Microflow	Aggregated
Reactive	Proactive
Virtual	Physical
Fully Consistent	Eventually Consistent

Hedera controls a data center network.

Hedera: Dynamic Flow Scheduling for Data Center Networks – Al Fares, et al., NSDI '10

27

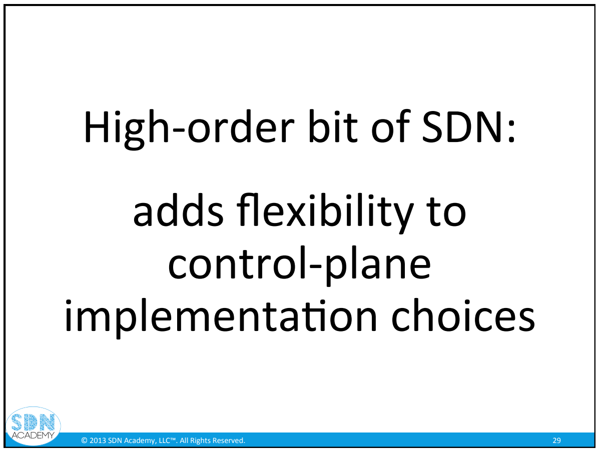


Your SDN network or application?

Centralized	Distributed
Microflow	Aggregated
Reactive	Proactive
Virtual	Physical
Fully Consistent	Eventually Consistent

Hybrid approaches often have desirable properties; can use these now.

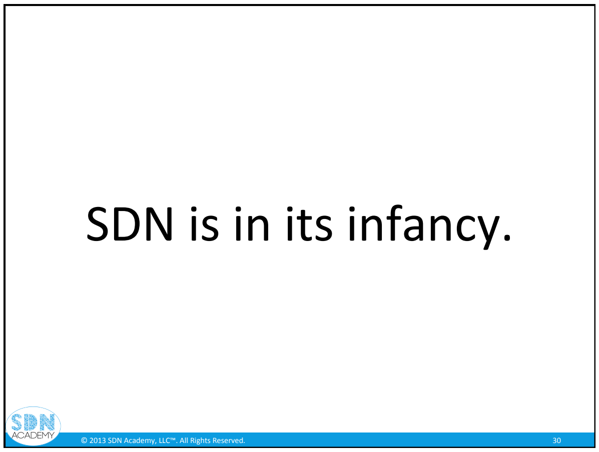
28



High-order bit of SDN:

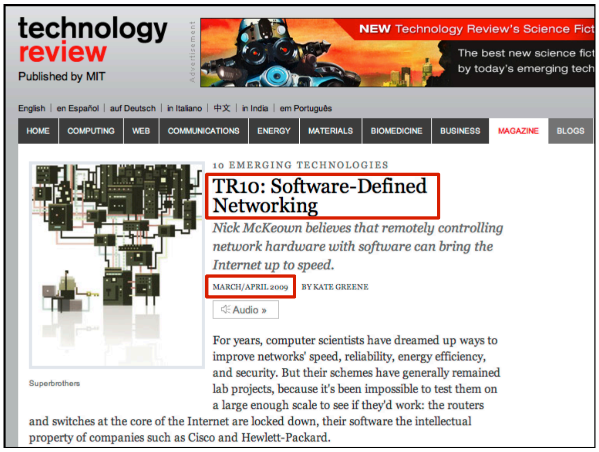
adds flexibility to control-plane implementation choices

29



SDN is in its infancy.

30



10 EMERGING TECHNOLOGIES

TR10: Software-Defined Networking

Nick McKeown believes that remotely controlling network hardware with software can bring the Internet up to speed.

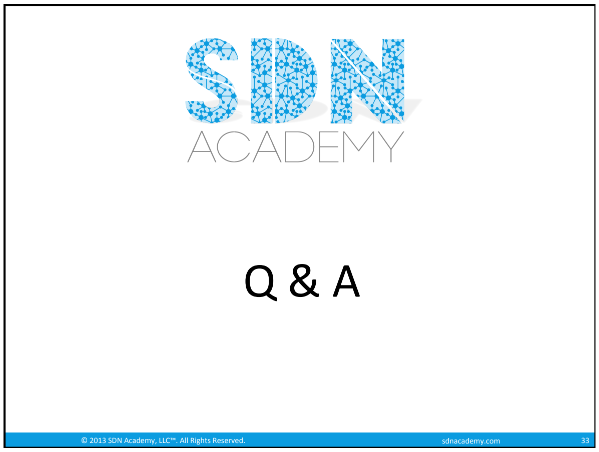
MARCH/APRIL 2009 BY KATE GREENE

Superbrothers

For years, computer scientists have dreamed up ways to improve networks' speed, reliability, energy efficiency, and security. But their schemes have generally remained lab projects, because it's been impossible to test them on a large enough scale to see if they'd work: the routers and switches at the core of the Internet are locked down, their software the intellectual property of companies such as Cisco and Hewlett-Packard.

Opportunity:
Every piece of the SD stack can be improved.

32



SDN ACADEMY

Q & A

© 2013 SDN Academy, LLC™. All Rights Reserved. sdnacademy.com 33

SDN ACADEMY

Grab a certificate of completion on your way out.

Thanks for attending!

© 2013 SDN Academy, LLC™. All Rights Reserved. sdnacademy.com 34