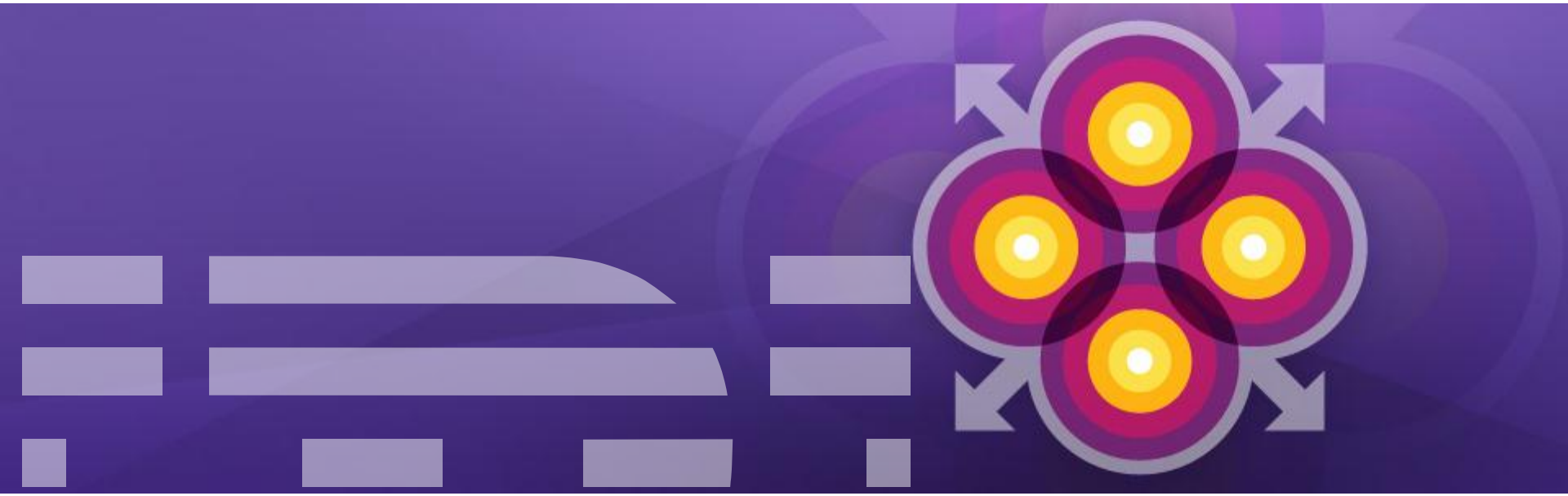
Anees Shaikh
IBM TJ Watson Research Center

IBM RESEARCH

SDN-enhanced Services in Enterprises and Data Centers

Collaboration with Mohammad Banikazemi, Salman Baset, Jack Kouloheris, David Olshefski, John Tracey, Guohui Wang



IBM RESEARCH

What this talk is about

- State of the API / network models in SDN controllers
  - ultimately crucial to realize the full promise of SDN
- Unexplored use cases for SDN in the data center
  - is this considered “SDN Research” ? maybe not, but it should be …

Does anyone care about network APIs?

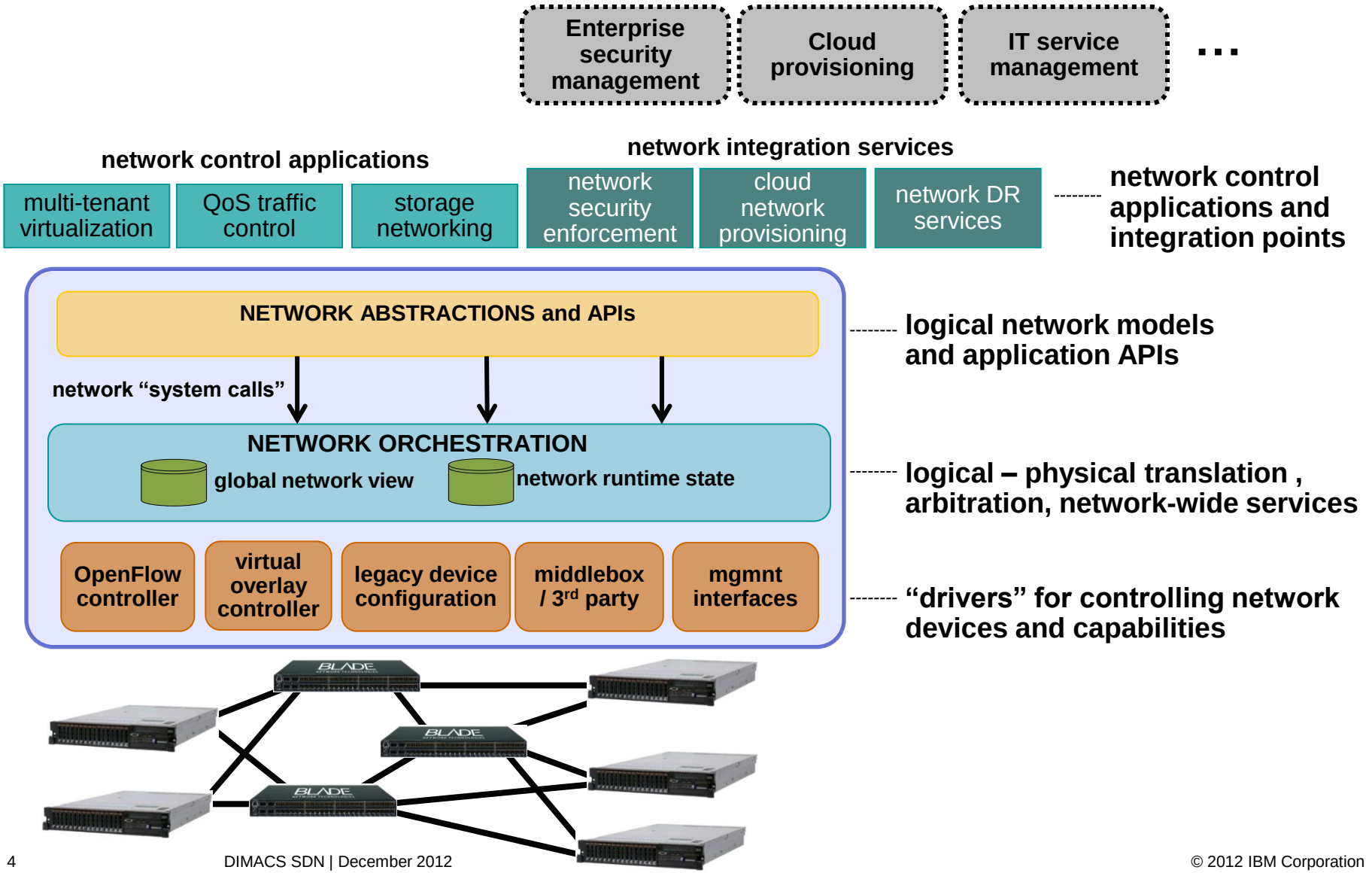
Network engineer view

- *"Vendors offering APIs to network engineers to help them solve their issues is akin to a an auto manufacturer handing a pile of metal and a welding torch to someone who needs a new car"*
- *"Some vendors believe the best approach is to hand off APIs, under the assumption that the network engineers know what they need and (apparently) have time to learn a programming language. The reality is that network engineers don't have that kind of time"* -- "Why are network engineers sick of hearing about SDN?," Nov 2012, packetpushers.net

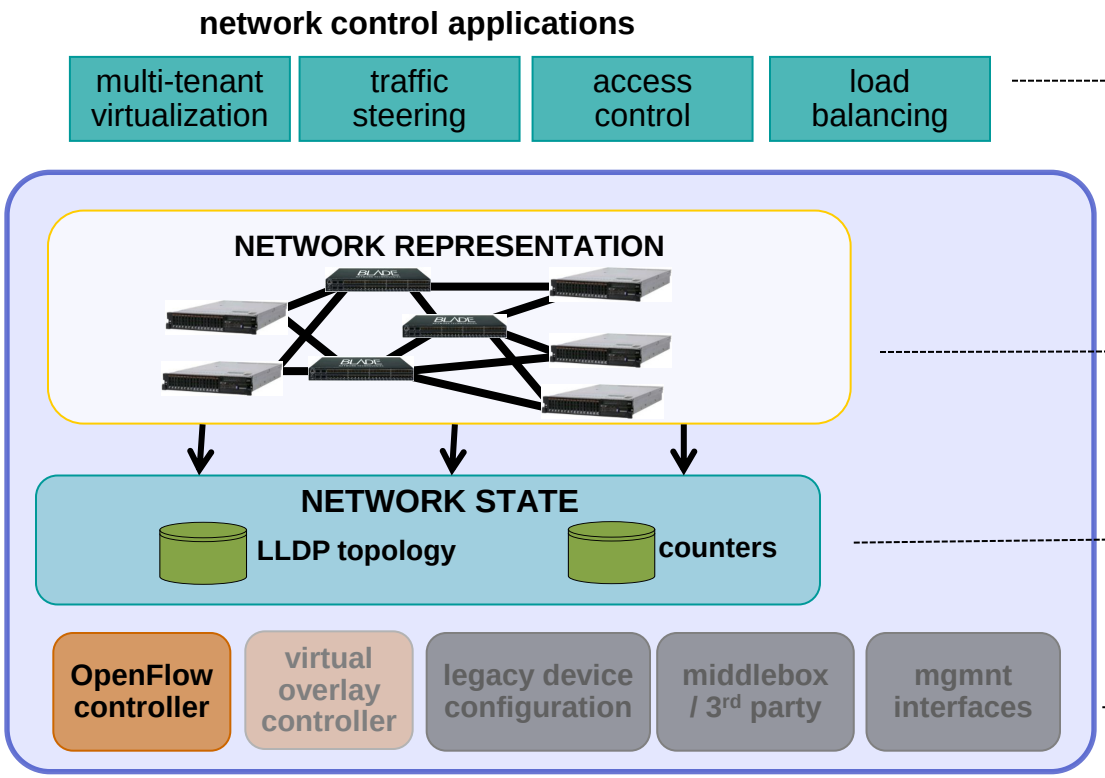
Application developer view

- Application developers are primarily just *users* of the network
- Little interest or ability to understand details of network operation
  - rely on the "network guys"
- DevOps model requires more direct ability to influence the operational network
  - need for network APIs and tools for developers
- (no good blog quotes yet)

A reference software-defined networking controller platform



Where we are

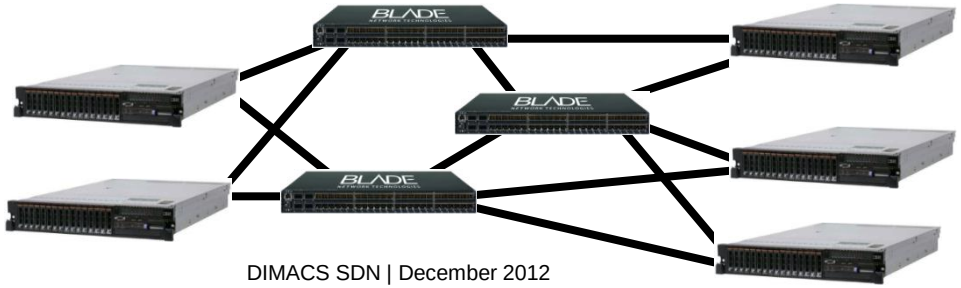


network-level applications with legacy equivalents

representation of the physical network

limited network state, limited orchestration

OpenFlow as the primary driver (fowarding plane)



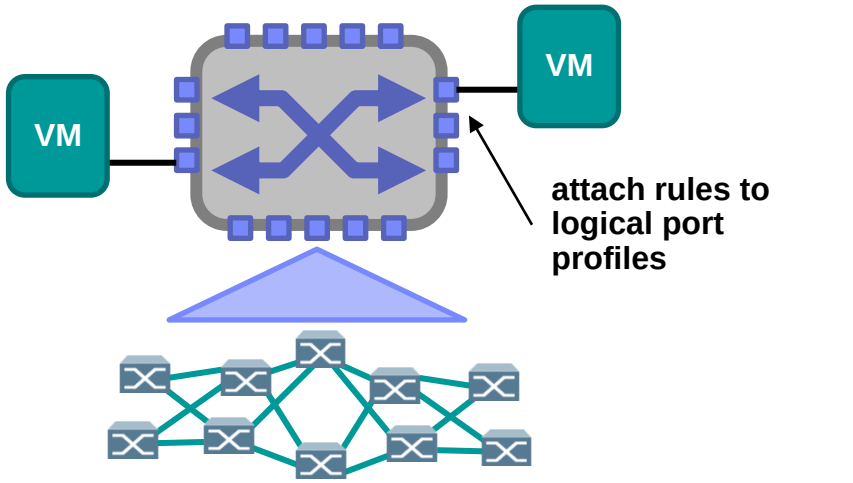
Current SDN controller APIs

- Floodlight – mostly OF protocol wrappers, with some abstractions
  - query for switch properties, read per-switch and global flow counters
  - insert / delete flow entries
    - `{"switch": "00:00:00:00:00:00:00:01", "name":"flow-mod-1", "cookie":"0", "priority":"32768", "ingress-port":"1","active":"true", "actions":"output=2"}`
  - ACL rules
    - `{"src-ip": "10.0.0.4/32", "nw-proto":"UDP", "tp-src":"5010", "action":"DENY" }`
  - Attach to virtual Quantum network
    - `{"attachment": {"id": "NetworkId1", "mac": "00:00:00:00:00:08"}}`
  - Trema, NOX/POX, Ryu … similar (or fewer) abstractions
- Frenetic – programming simplification on top of a standard OF protocol driver
  - SQL-like queries for collecting network data
  - simplified composition and optimization of application rules
  - abstracted topology views (!)

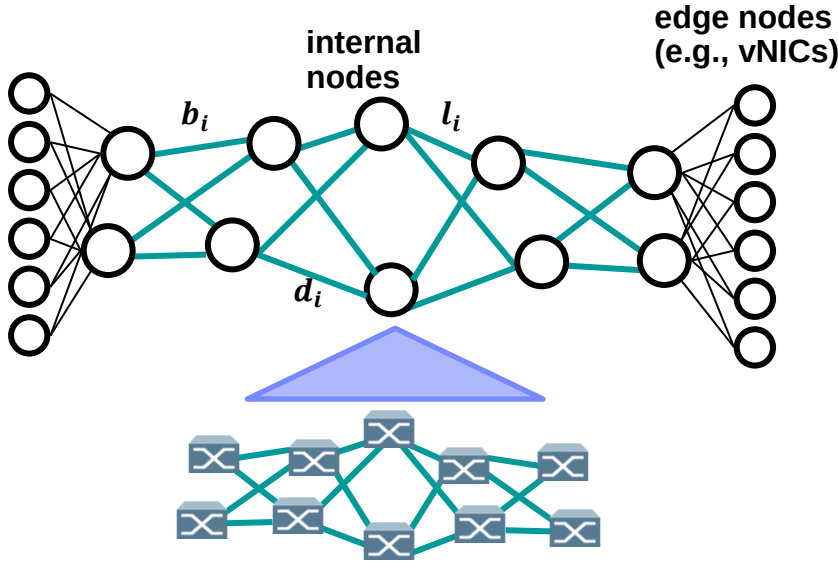
Need APIs that provide alternative models of the network for applications and services

Example abstract network models for SDN applications

- **Single virtual switch** – policy-based connectivity between physical or virtual machines
  - use switch-level concepts to describe connectivity and policies – ports, VLANs, profiles, etc.
  - switch per tenant view or single large switch for global policies
  - attach high-level rules or policies to logical "port profiles" (ACLs, in/out firewall, traffic marking/policing, etc.)

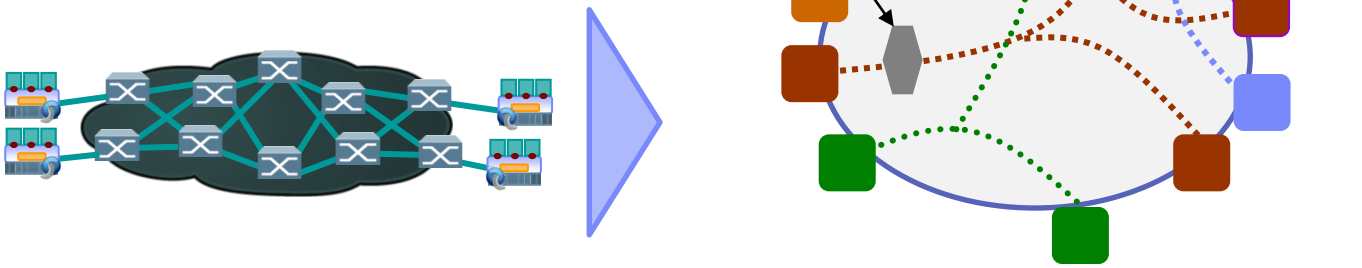


- **Annotated network graph** – suitable for running variety of graph algorithms
  - example edge annotations: utilization / avail bw, reserved bw, buffer occupancy stats, pkt drop stats
  - represent full or subset of actual topology (e.g., rack-level topology)
  - couple with management tooling to collect link or node metrics

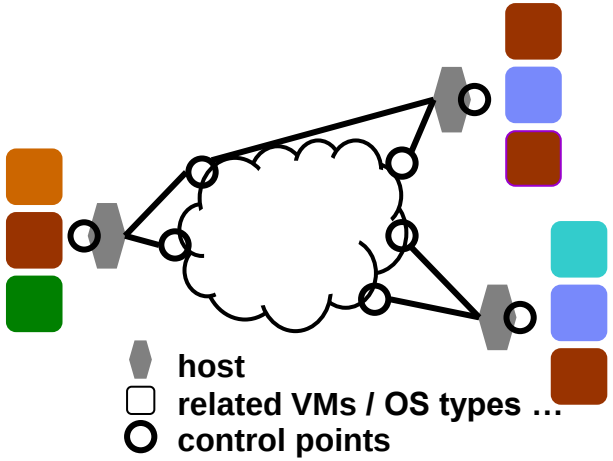


Example abstract network models for SDN applications

- **Tenant connectivity model** – multi-tenant virtual network service
  - service model represents tenant VMs, virtual network segments, middleboxes, connectivity policies
  - requires labeling VMs with tenant ids, access user-defined policies, etc.



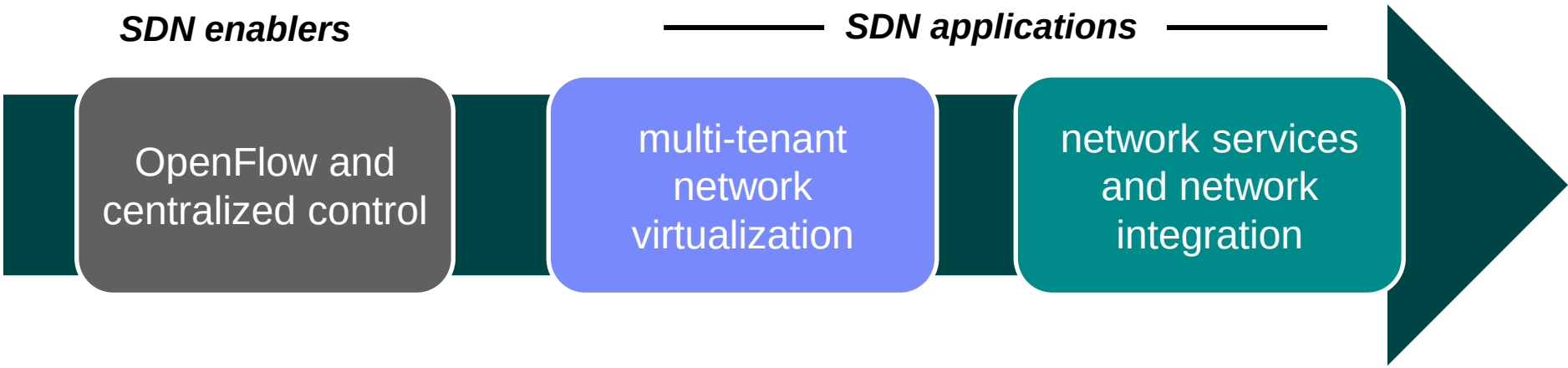
- **Security enforcement model** – provides control points for enforcing security actions
  - model shows applications/OS, VMs, hosts, and control points (not full topology)
  - integrates varying levels of application-level visibility (e.g., from provisioning engine)



What this talk is about

- State of the API / network models in SDN controllers
  - ultimately crucial to realize the full promise of SDN
- Services use cases for SDN in the enterprise data center
  - IT service management processes
  - application connectivity services

SDN progression in the data center

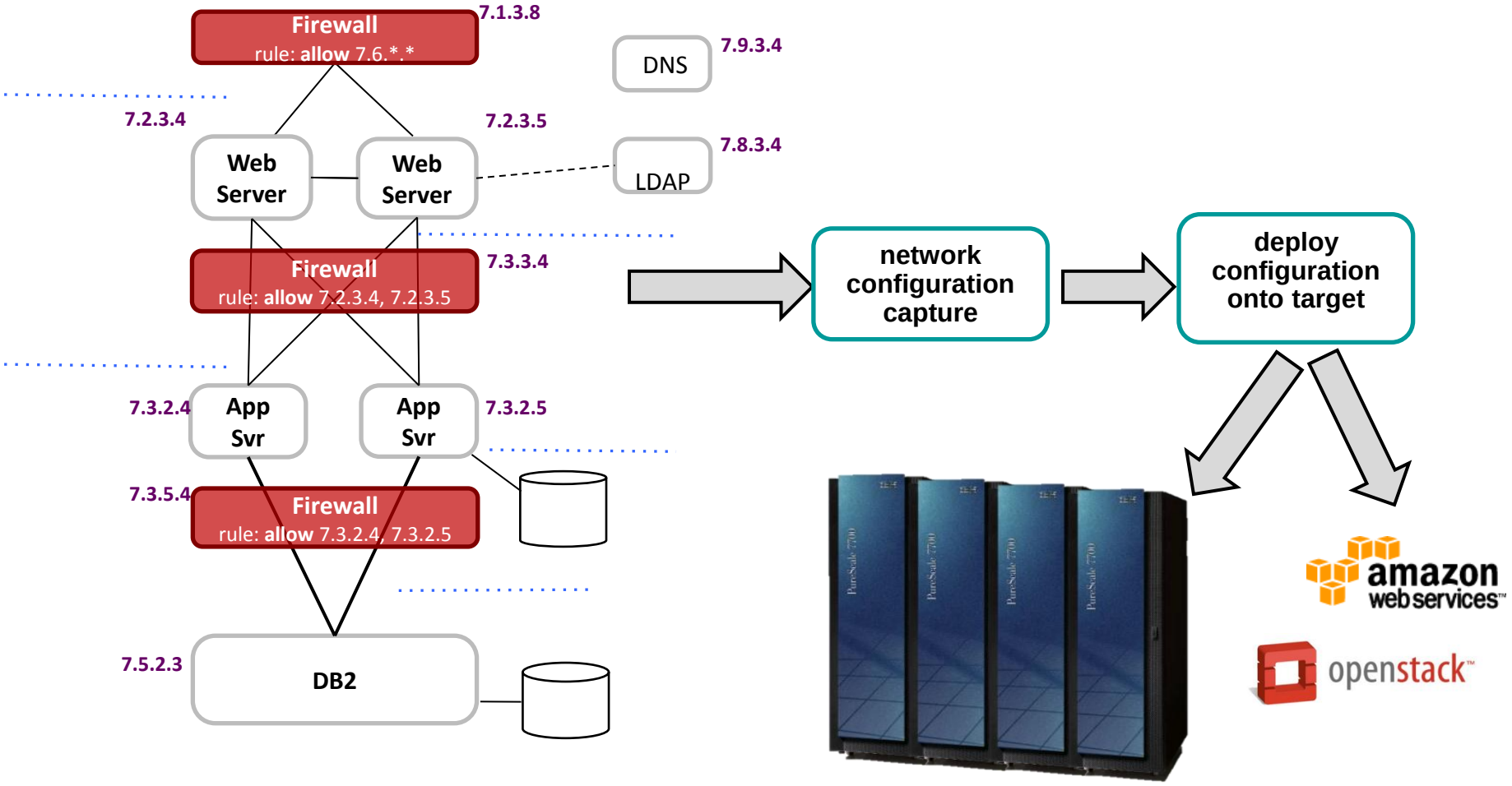


- industry standard protocol for SDN
- hardware and software enablement
- quickly becoming a standard feature on switches
- more OF controller options

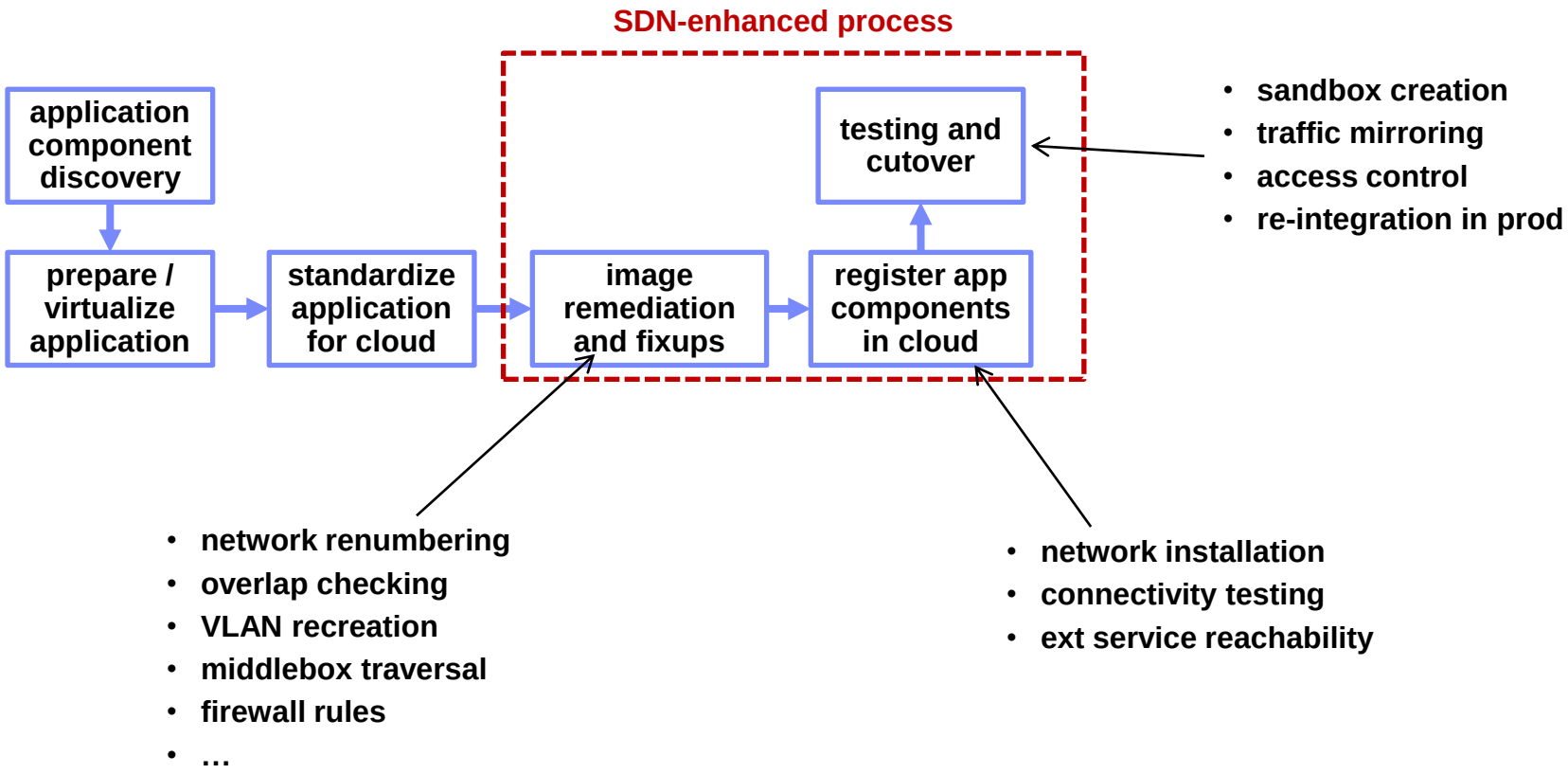
- first production use case for SDN
- solutions with and without OpenFlow
- ultimately, a universal feature

- network-level automation
- application-level network services
- services on converged networks
- integration with security, provisioning, disaster recovery, etc.

Migration and workload consolidation

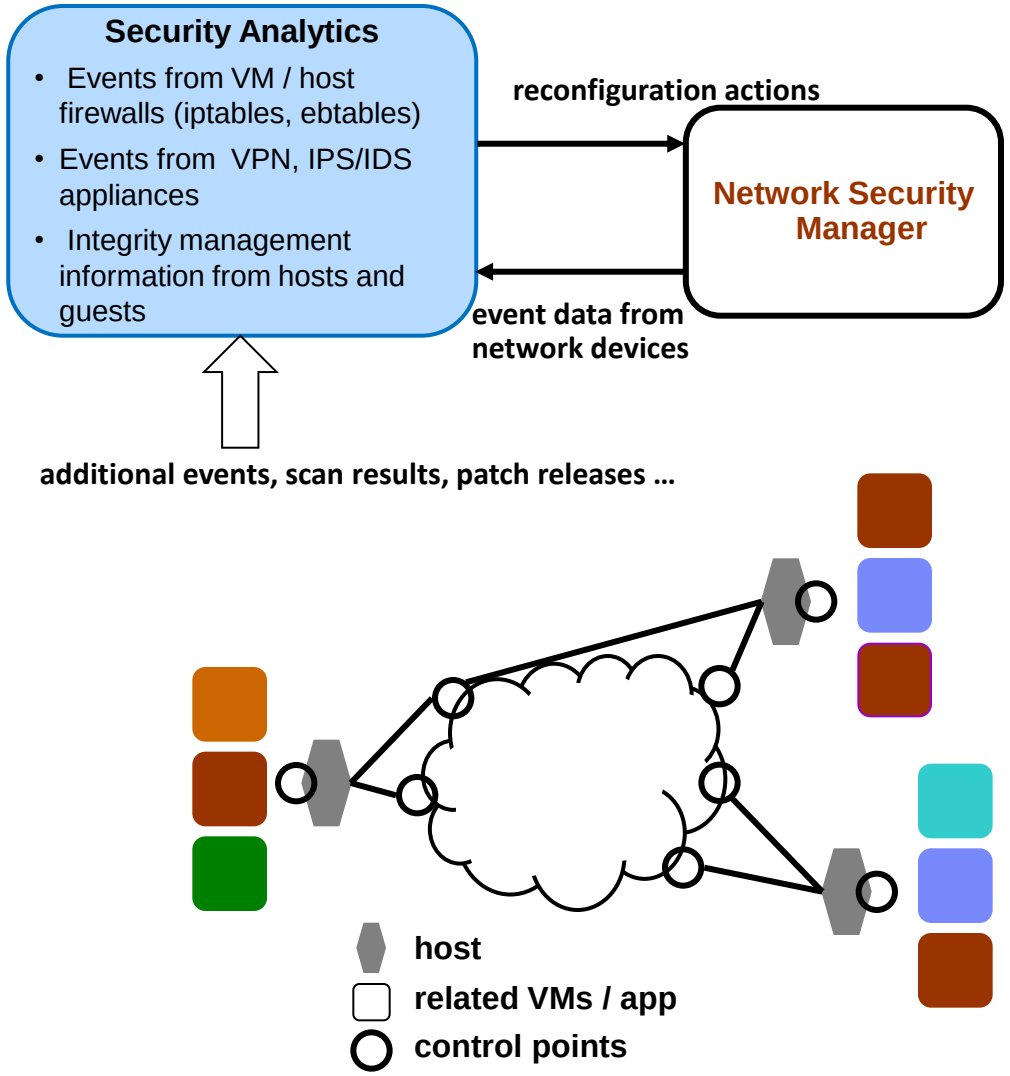


Migration and consolidation – IT process view (*service transition*)

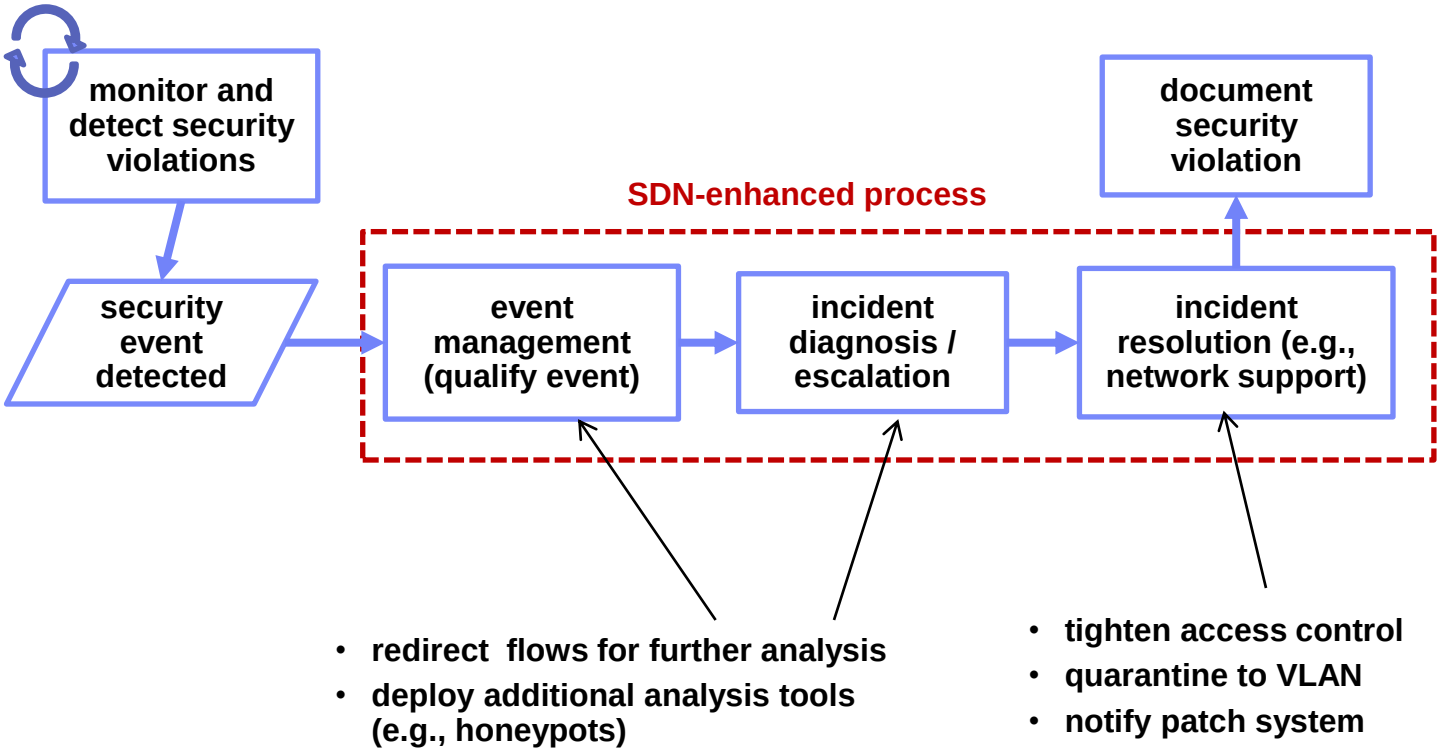


Security: Closed-loop network reconfiguration

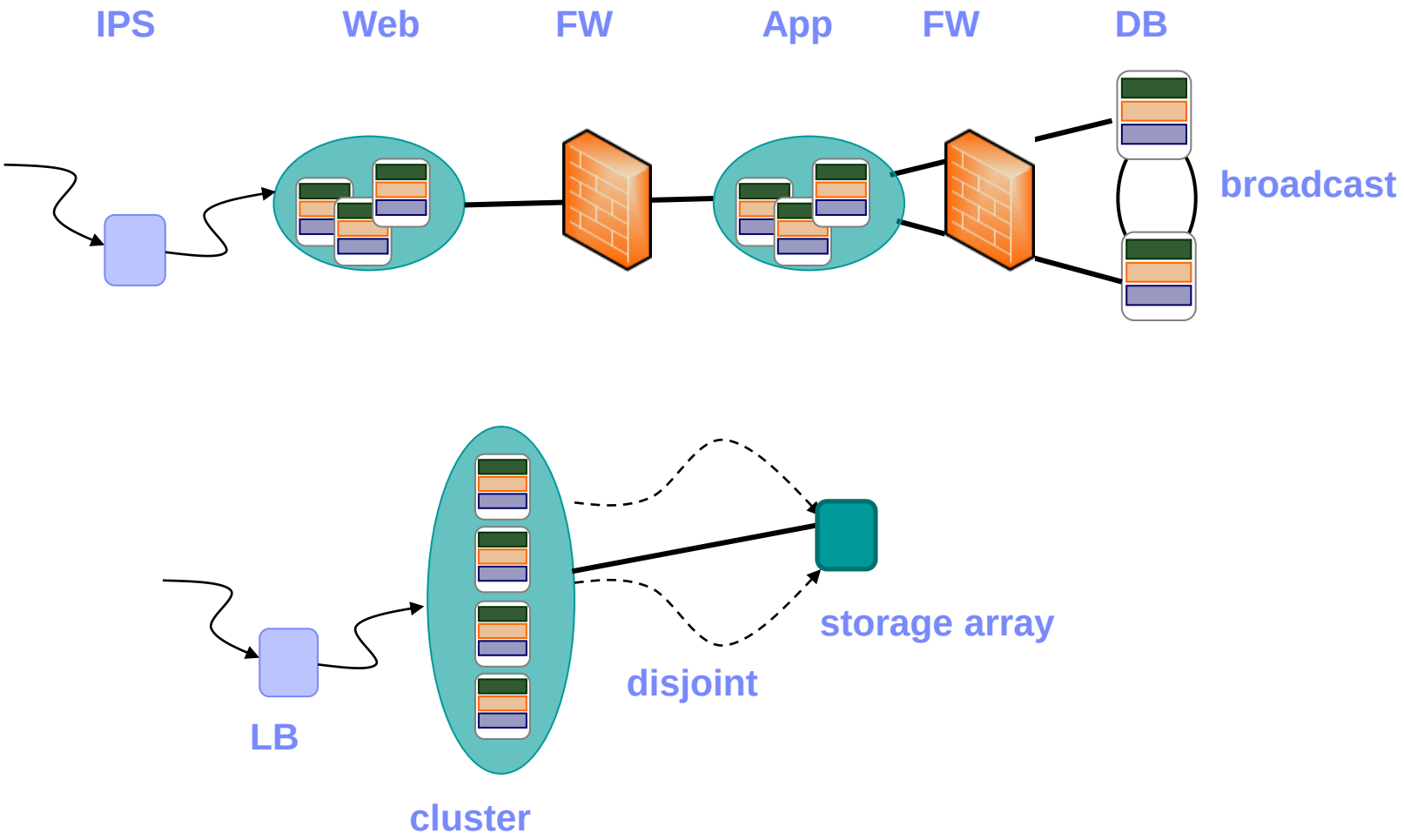
- Determine optimal security policies in response to anomalies detected or workload observed
- abstract model shows applications/OS, VMs, hosts, and control points (not full topology)
- *collaboration with IBM CRL, Security Research Dept. at Watson*



Security management – IT process view (*service operations*)



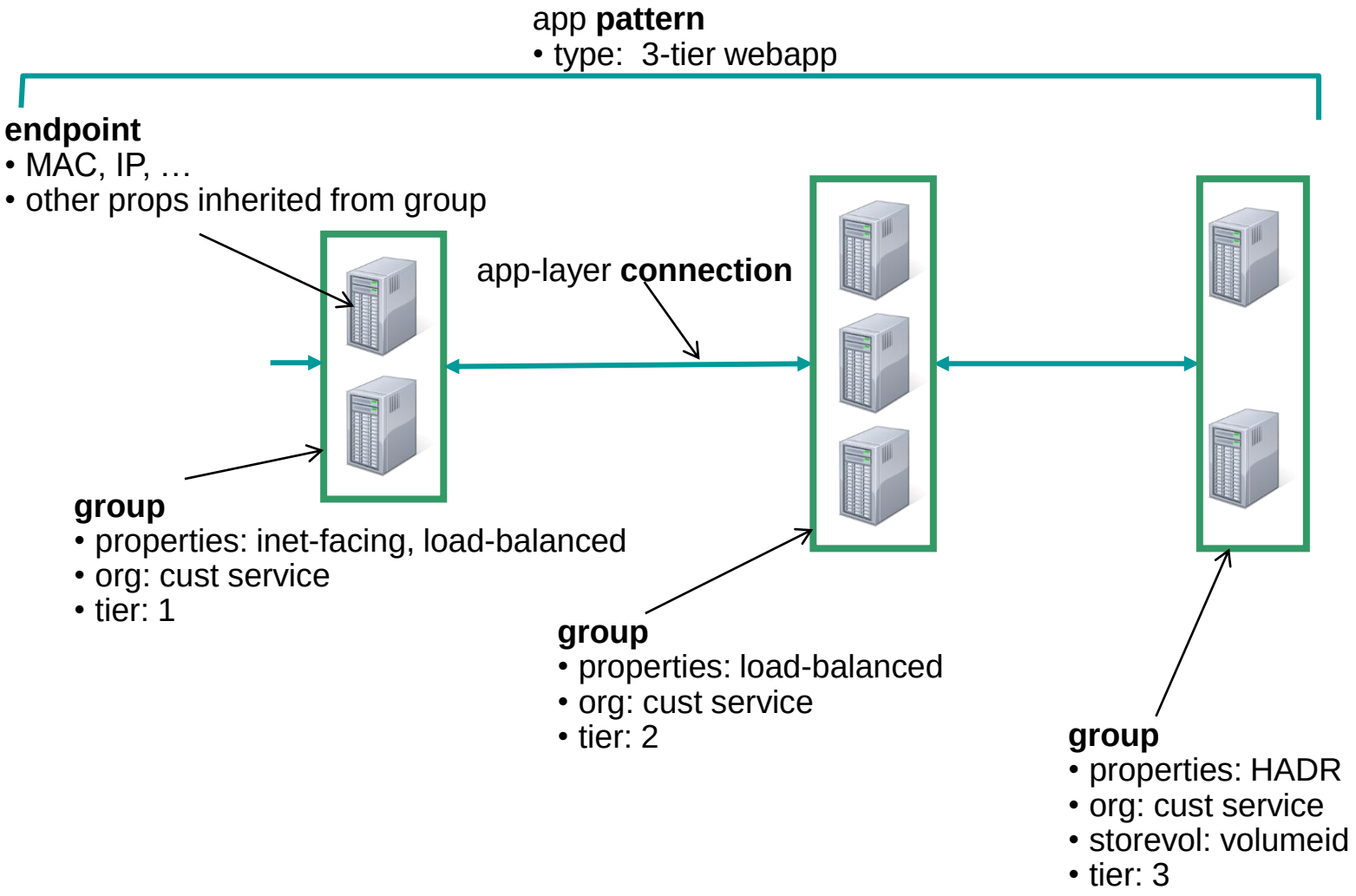
Application connectivity patterns



Connectivity services for enterprises

- Enterprises often have organizational silos between application and network teams
  - information / requirements exchanged through tickets, email, etc.
  - iterative process for complex applications
- Traditional application teams have little understanding of connectivity, security, or network performance requirements for their applications
  - SDN-based connectivity service should not place burden of specifying network details on application deployers
  - contrast to self-service cloud model
- SDN connectivity service interface should mimic the application ←→ network team information exchange
  - SDN controller and apps assist network team through automation and consistent operations
- SDN connectivity application must be populated with semantics and policy information to perform correct configurations
  - Q: What is the interface for populating this knowledge? Who provides it?

Example: enterprise connectivity service abstractions



Summary: SDN consumption models

- SDN promises more seamless integration of the network with IT processes …
- but, SDN APIs and abstractions are primarily geared for low-level network control
  - more work needed to develop the application interface of the SDN stack
  - discussed some examples in various states of experimentation
- SDN use cases that matter to non-network experts
  - maybe more important than solving problems of correctness, reliability, scalability in SDN protocols and devices
  - needs research (this isn't a marketing problem)

IBM RESEARCH

Thank you