Network Automation

Jun Li

Contributions from my students, especially Xiaohe Hu

Increasing Network Complexity

- Expectations
  - User
- Scale
  - # of cores, VMs, and containers
  - # of RFCs, apps, and services
- Dynamics
  - Architecture & Topology
  - Workload

Nick & Jennifer: Network management has always been a worthwhile endeavor, but now it is mission critical.

Self-Driving Network

ACM SIGCOMM 2018 Workshop on Self-Driving Networks (SelfDN 2018)

(1) demonstrate the successful implementation of the different feedback control components so that, together, they can perform the tasks at hand in an automated way

(2) identify bottlenecks in existing technologies or methods that prevent the practical deployment of full-fledged self-driving networks

Calls for Network Automation

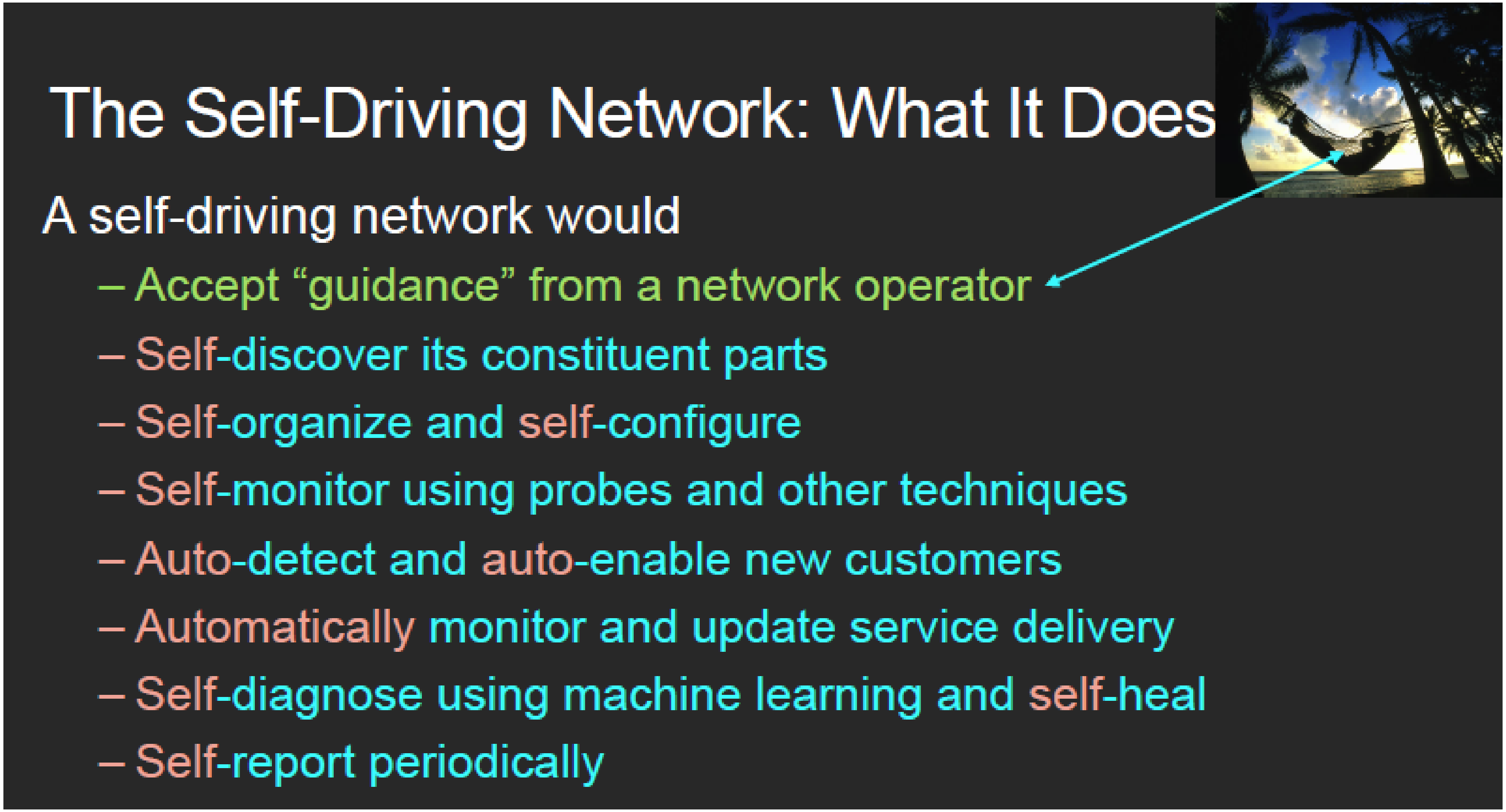
- Self-Driving Network – an automated, fully autonomous network, by Kireeti Kompella, CTO, Juniper Networks, Mar. 2016
- Zero Touch Networking, by Bikash Koley, Google, Apr. 12, 2017 (Designing Self-driving Networks Workshop)
- Why (and How) Networks Should Run Themselves, by Nick Feamster and Jennifer Rexford, Princeton University, Oct. 31, 2017

From Closed-form to Closed-loop

- Nick & Jennifer
  - – Current way: optimizations based on closed-form analysis of individual protocols
  - – New Approach: network operators need
    - Data-driven, machine-learning-based models of end-to-end
    - Application performance based on high-level policy goals
    - Holistic view of the underlying components

Self-Driving Network

- Kireeti Kompella: a grand challenge!

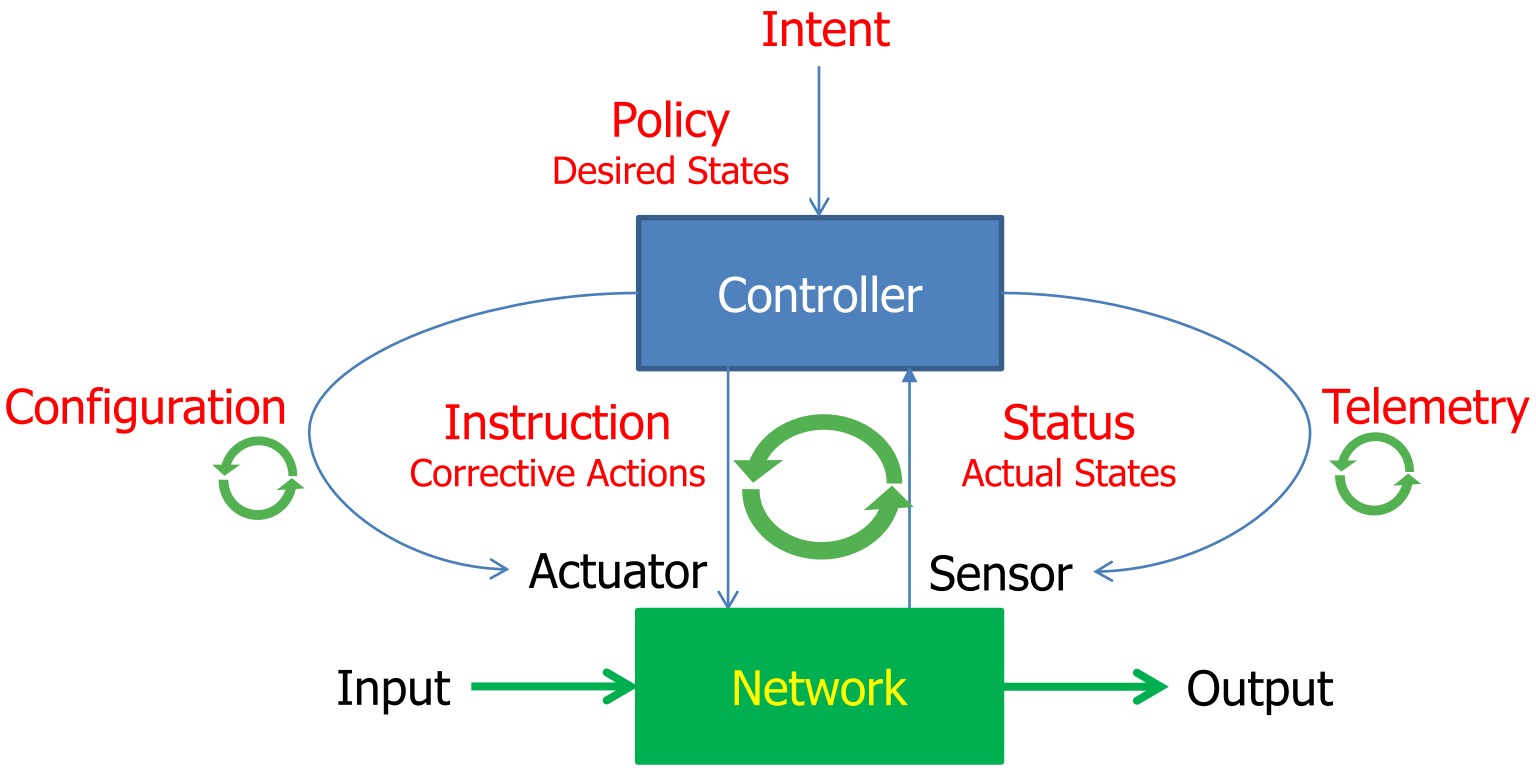


The Self-Driving Network: What It Does

A self-driving network would

- Accept “guidance” from a network operator
- Self-discover its constituent parts
- Self-organize and self-configure
- Self-monitor using probes and other techniques
- Auto-detect and auto-enable new customers
- Automatically monitor and update service delivery
- Self-diagnose using machine learning and self-heal
- Self-report periodically

AutoNet: Network Automation

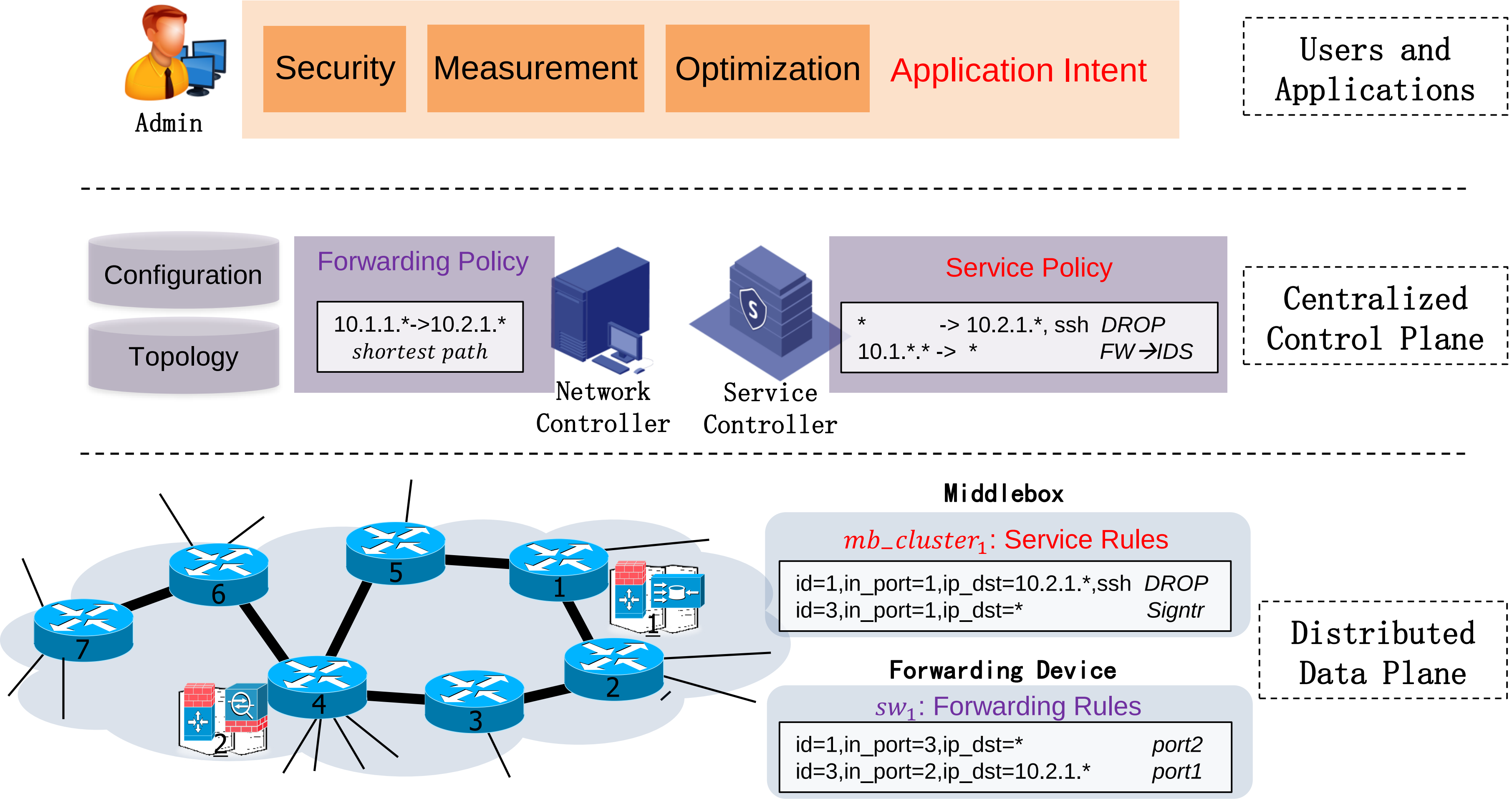


Nick & Jennifer：
The larger goal of relieving the operator’s burden as much as possible, and possibly altogether.

Back to Network Policies

- Nick & Jennifer:
  - Customer expectations → SLA
    - statistical guarantees on latency, jitter and DDoS response time
    - user quality-of-experience metrics, such as Mean Opinion Score (MOS) for VoIP traffic or page load time for web browsing
  - Network-wide goals for resource *optimization*
    - Maximizing link utilization
    - Minimizing congestion, such as video bitrates or rebuffering for QoE
  - Application functions and services (traffic transformations)
    - Header: NAT, ACL
    - Payload: transcoding, compression, and encrypt (MB, such as IDS)

Policy Management (I)



Forwarding Policy

10.1.1.*->10.2.1.*
shortest path

Service Policy

* -> 10.2.1.*, ssh *DROP*
10.1.*.* -> * *FW→IDS*

Middlebox

$mb\_cluster_1$: Service Rules

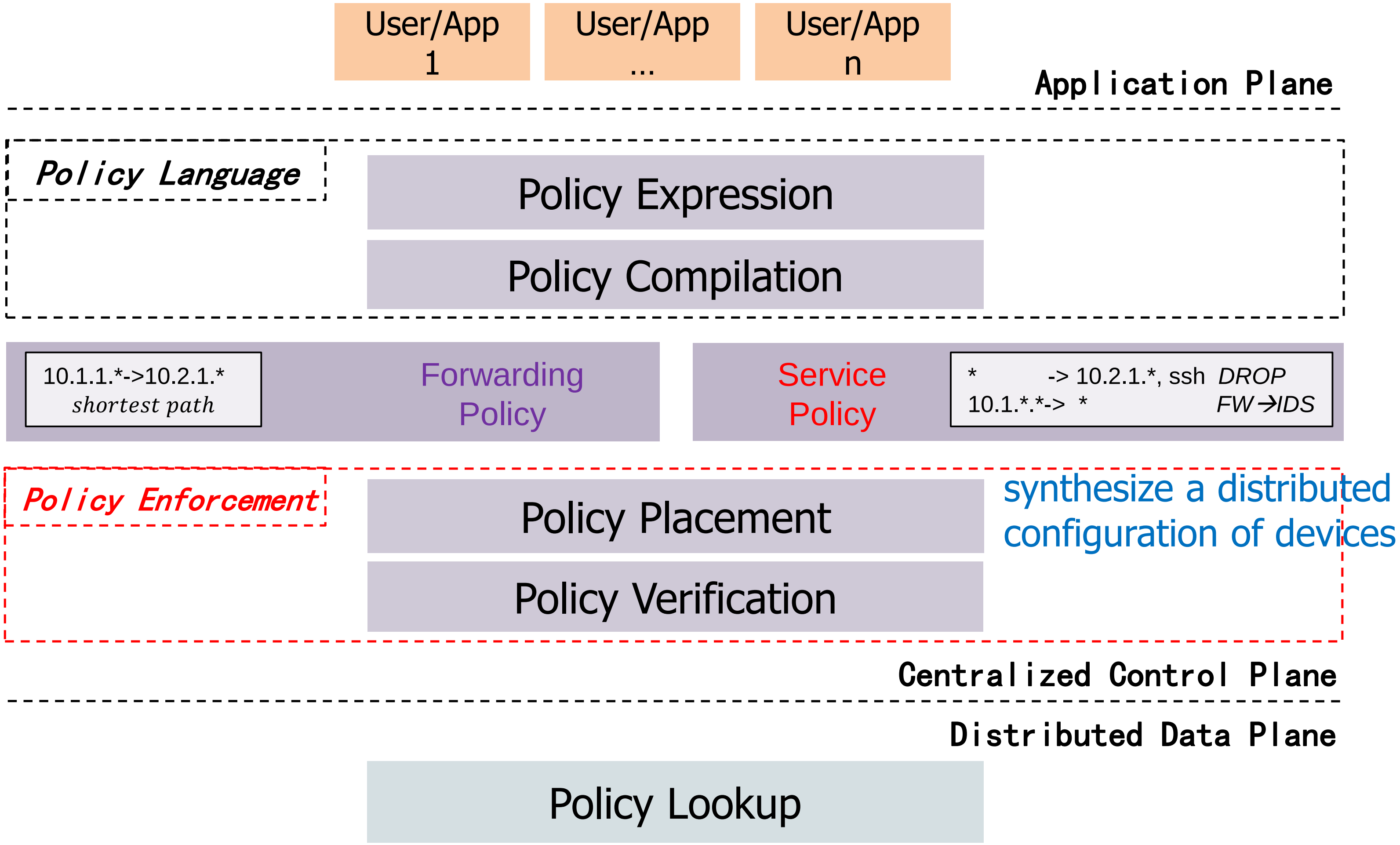
id=1,in_port=1,ip_dst=10.2.1.*,ssh *DROP*
id=3,in_port=1,ip_dst=* *Signtr*

Forwarding Device

sw_1: Forwarding Rules

id=1,in_port=3,ip_dst=* *port2*
id=3,in_port=2,ip_dst=10.2.1.* *port1*

Policy Management (II)



User/App 1

User/App …

User/App n

Application Plane

Policy Language

Policy Expression

Policy Compilation

10.1.1.*->10.2.1.*
shortest path

Forwarding Policy

Service Policy

* -> 10.2.1.*, ssh *DROP*
10.1.*.*-> * *FW→IDS*

Policy Enforcement

Policy Placement

Policy Verification

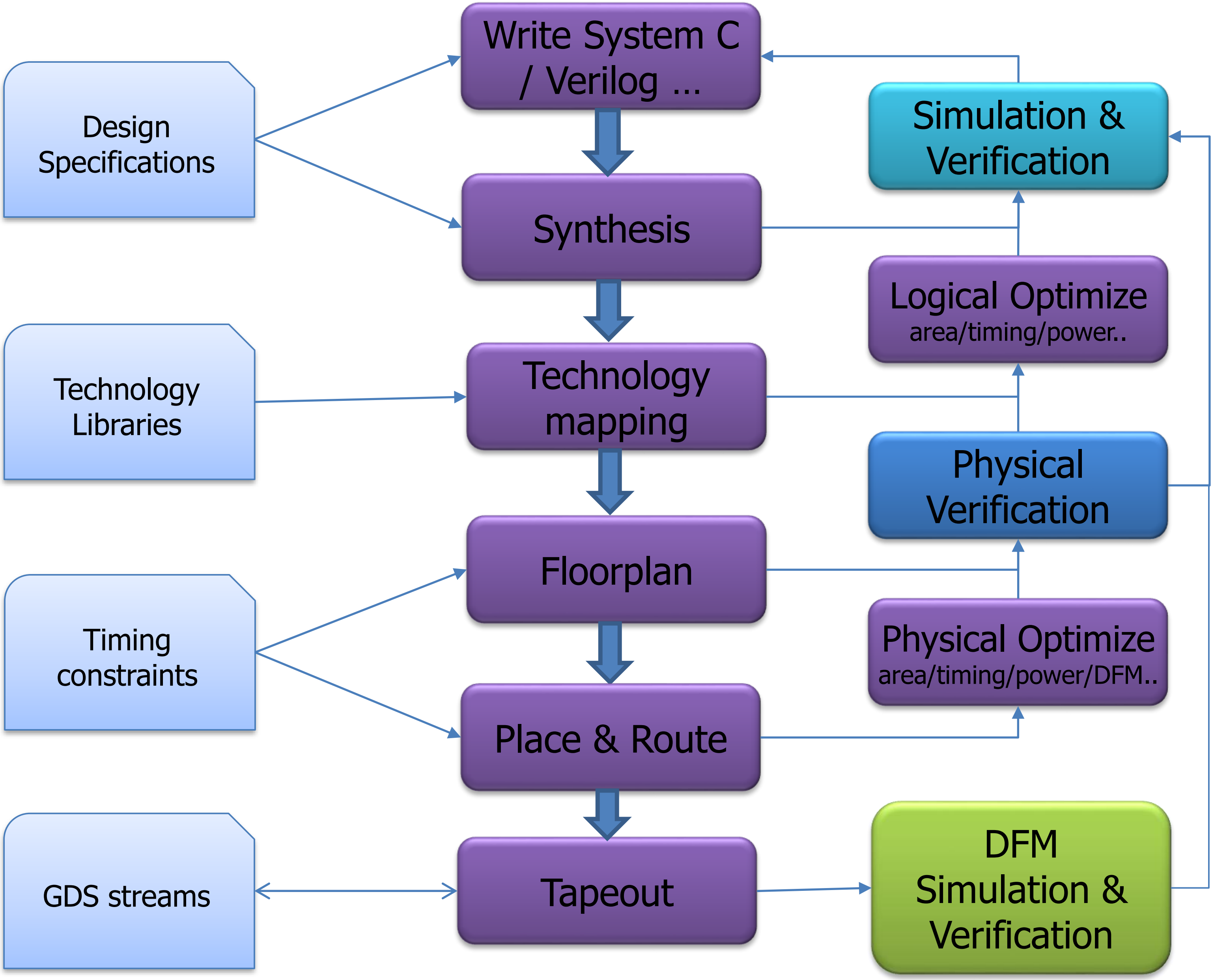
synthesize a distributed configuration of devices

Centralized Control Plane

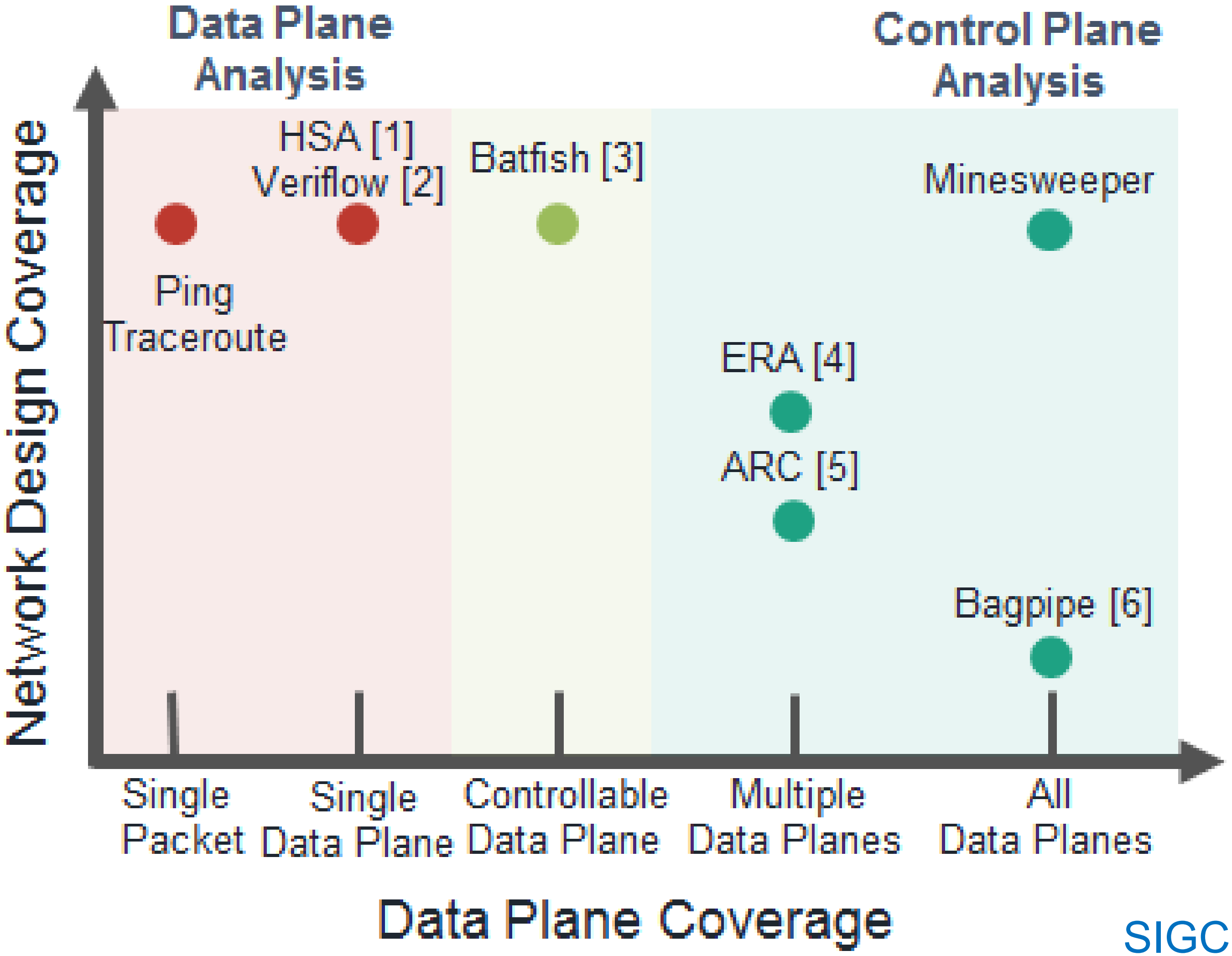
Distributed Data Plane

Policy Lookup

Electronic Design Automation Flow



Progress in Network Verification



SIGCOMM '17

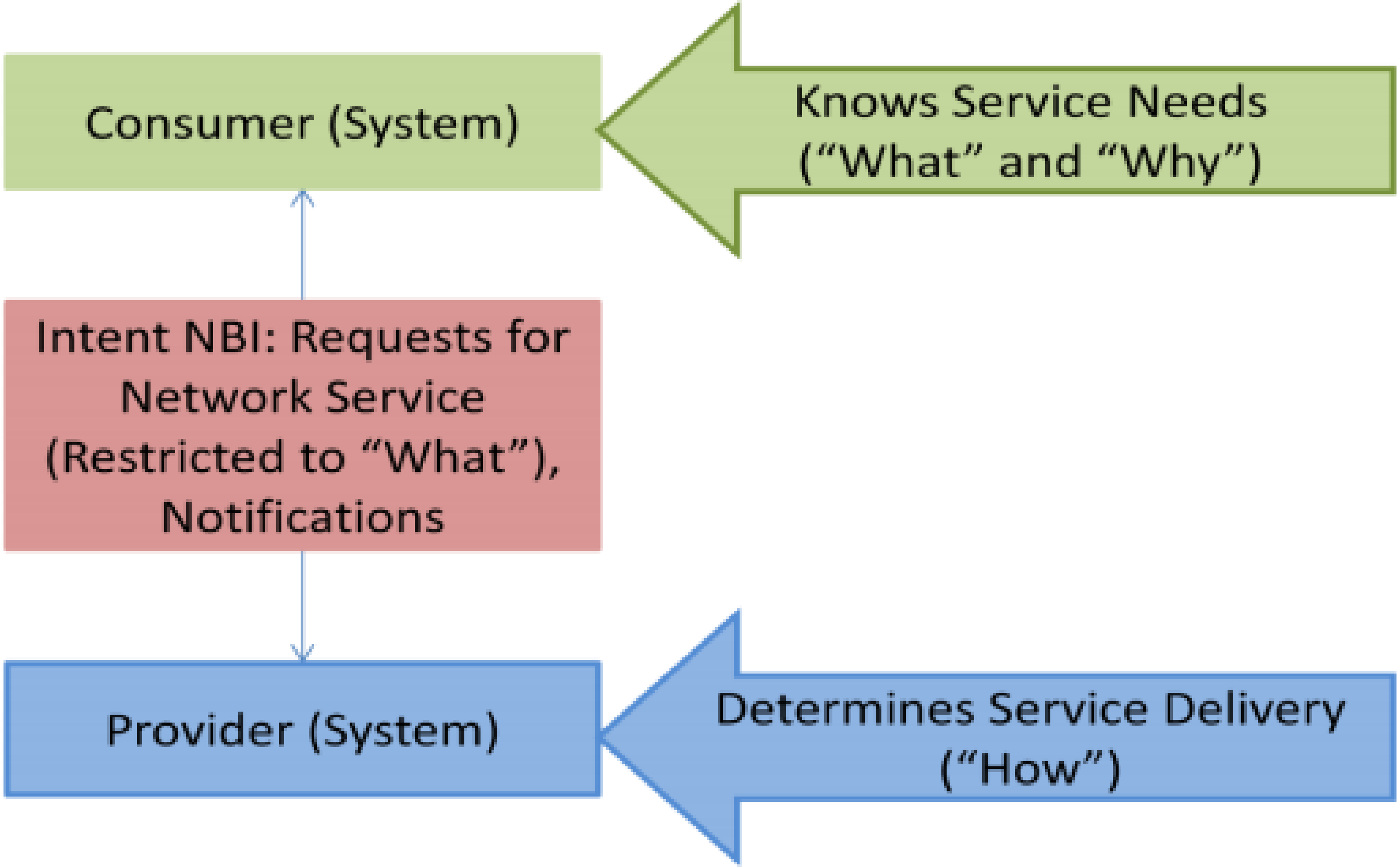
Intent Based Networking

ACM SIGCOMM 2018 Workshop on Network Meets AI & ML (NetAIM 2018) Sponsored by Huawei; Nicholas Zhang & David Meyer

Calls for Network Automation

- Intent NBI – Definition and Principles, by Christopher Janz, Huawei (Principal Author) Nigel Davis, Ciena David Hood, Ericsson Mathieu Lemay, Inocybe David Lenrow, Huawei Li Fengkai, Huawei Fabian Schneider, NEC John Strassner, Huawei Andrew Veitch, NEC-Netcracker, ONF, Oct. 2016
- Innovation Insight: Intent-Based Networking Systems, by Andrew Lerner, Joe Skorupa, Sanjit Ganguli, Gartner, Feb 7, 2017

Intent NBI (northbound interface)


Consumer (System)
Knows Service Needs ("What" and "Why")
Intent NBI: Requests for Network Service (Restricted to "What"), Notifications
Provider (System)
Determines Service Delivery ("How")

Intent-based Networking (I)

- Intent-based networking systems (IBNS) is a lifecycle management (design, implementation, operation and assurance) ~~middleware~~ product for networking infrastructure for network availability and agility.
- IBNS can reduce network infrastructure delivery times by 50% to 90%, while simultaneously reducing the number and duration of outages by at least 50%.
- By 2020, more than 1,000 large enterprises will use IBNS in production, up from less than 15 today.

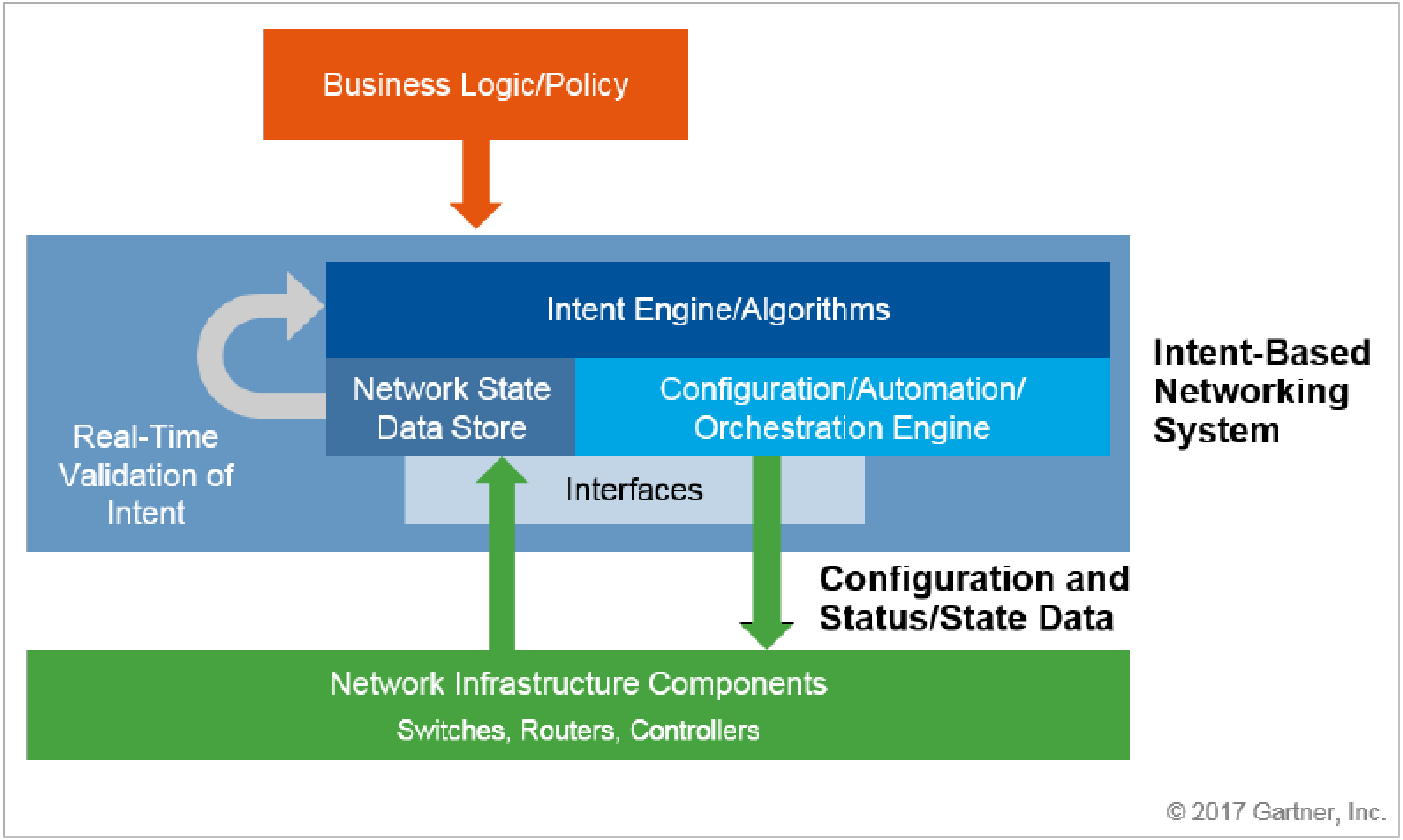
Intent-based Networking (II)

- Translation and Validation
  - The system takes a higher-level business policy (what) as input from end users and converts it to the necessary network configuration (how)， then generates and validates the resulting design and configuration for correctness.
- Automated Implementation
  - The system can configure the appropriate network changes (how) across existing network infrastructure. This is typically done via ~~network automation and/or~~ network orchestration.
  - IBNS could be an application that drives an SDN controller.

Intent-based Networking (III)

- Awareness of Network State
  - The system ingests real-time network status for systems under its administrative control, and is protocol- and transport-agnostic.
- Assurance and Dynamic Optimization/Remediation
  - The system continuously validates (in real time) that the original business intent of the system is being met, and can take corrective actions (such as blocking traffic, modifying network capacity or notifying) when desired intent is not met. There will be products that address some of these components, and other products that address all of them.

How IBNS Works


Business Logic/Policy
Intent Engine/Algorithms
Network State Data Store
Configuration/Automation/ Orchestration Engine
Interfaces
Real-Time Validation of Intent
Intent-Based Networking System
Configuration and Status/State Data
Network Infrastructure Components
Switches, Routers, Controllers

© 2017 Gartner, Inc.

IBNS Industrial Initiatives

- Startups
  - Apstra addresses several aspects of intent within multivendor data center networking environments, branded as Apstra Operating System (AOS).
  - Forward Networks has a platform that ingests data, builds a network state model and provides an assurance for business policy.
  - Veriflow addresses use cases that improve network uptime and network security, and can be a key component of an intent-based networking system.
  - ~~Waltz Networks addresses several aspects of intent, with a focus on WAN use cases.~~

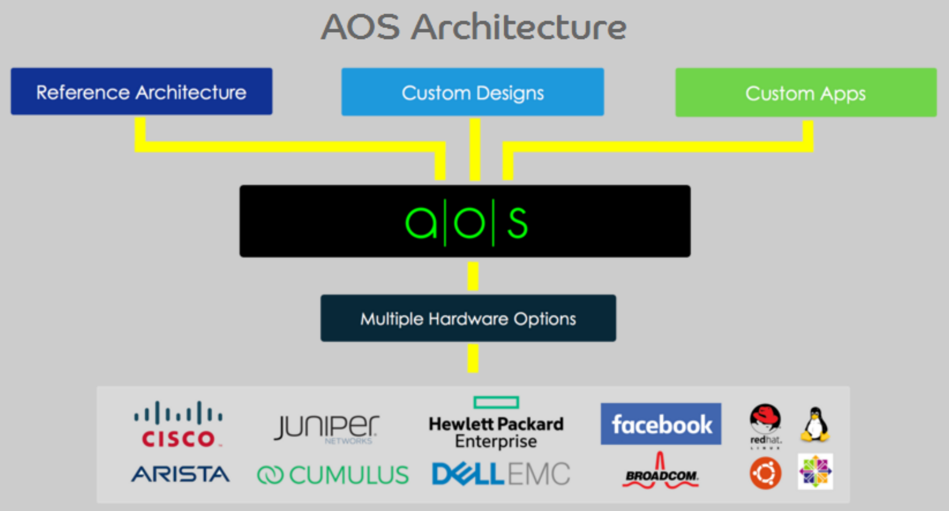
INTENTIONET

- Incumbents
  - Cisco will provide intent-based capability, based on a combination of products, including Cisco ACI, CloudCenter and Tetration.
  - Juniper Networks has described its vision to deliver Self-Driving Networks, based on its Contrail orchestration software, and we anticipate it will provide IBNS capability.

apstra

- Apstra pioneered Intent-Based Networking and delivered the first and only complete system which enables Cisco, Juniper, Arista, HP and white box customers to deploy an IBN system through the Apstra Operating System (AOS™).
- Motivations (Mansour Karam, Apstra CEO)
  - The key to simplifying operations is to run the network *as a system*, as opposed to box by box. Therefore, a *distributed systems approach* is required.
  - An *intent-driven* approach that focuses the network engineers on the services by an *integrated solution* that automates operations with closed-loop *continuous* validation at the core.
  - Network Engineers must maintain *choice and control* of the network equipment suppliers.

- Motivation (Sasha Ratkovic, Apstra CTO)
  - The hard problem is how to compose the complex infrastructure capabilities
    - State of infrastructure (compute, network, and storage)
    - State of business rules and policies
    - Frequency and complexity of constant changes
    - Composition, partition, and isolation
    - Design, build, deploy, validate

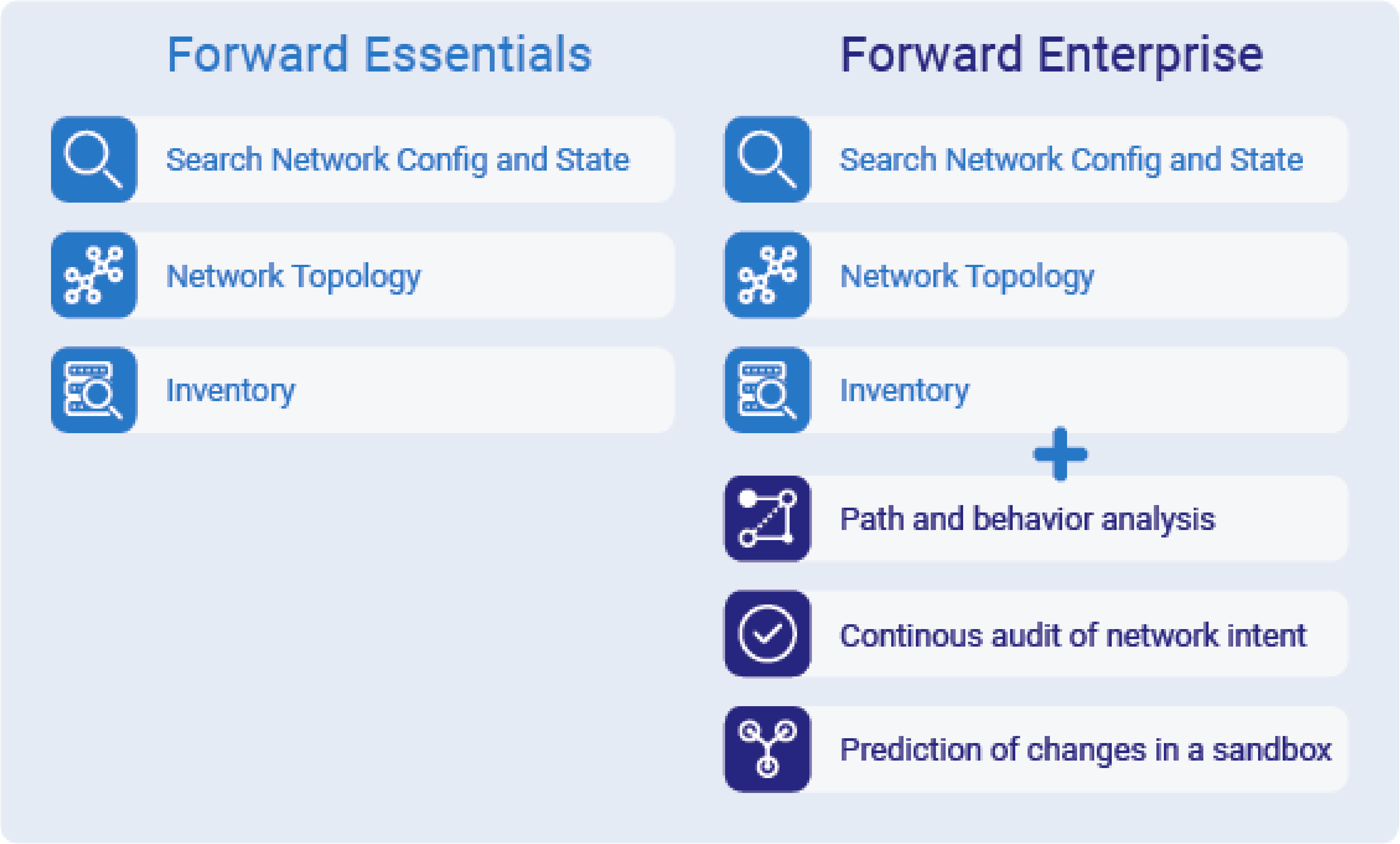


Scaling issues: decompose & update

The Search and Verification Engine for your IBN

- *Forward Essentials* Highlights
  - Organize network configuration and state information
  - Make it accessible and easily consumable by network teams
- *Forward Essentials* Benefits
  - Root-cause network issues
  - Maintain up-to-date network documentation (interactive topology + device inventory)

The Forward Platform



Forward Essentials

- Search Network Config and State
- Network Topology
- Inventory

Forward Enterprise

- Search Network Config and State
- Network Topology
- Inventory

+

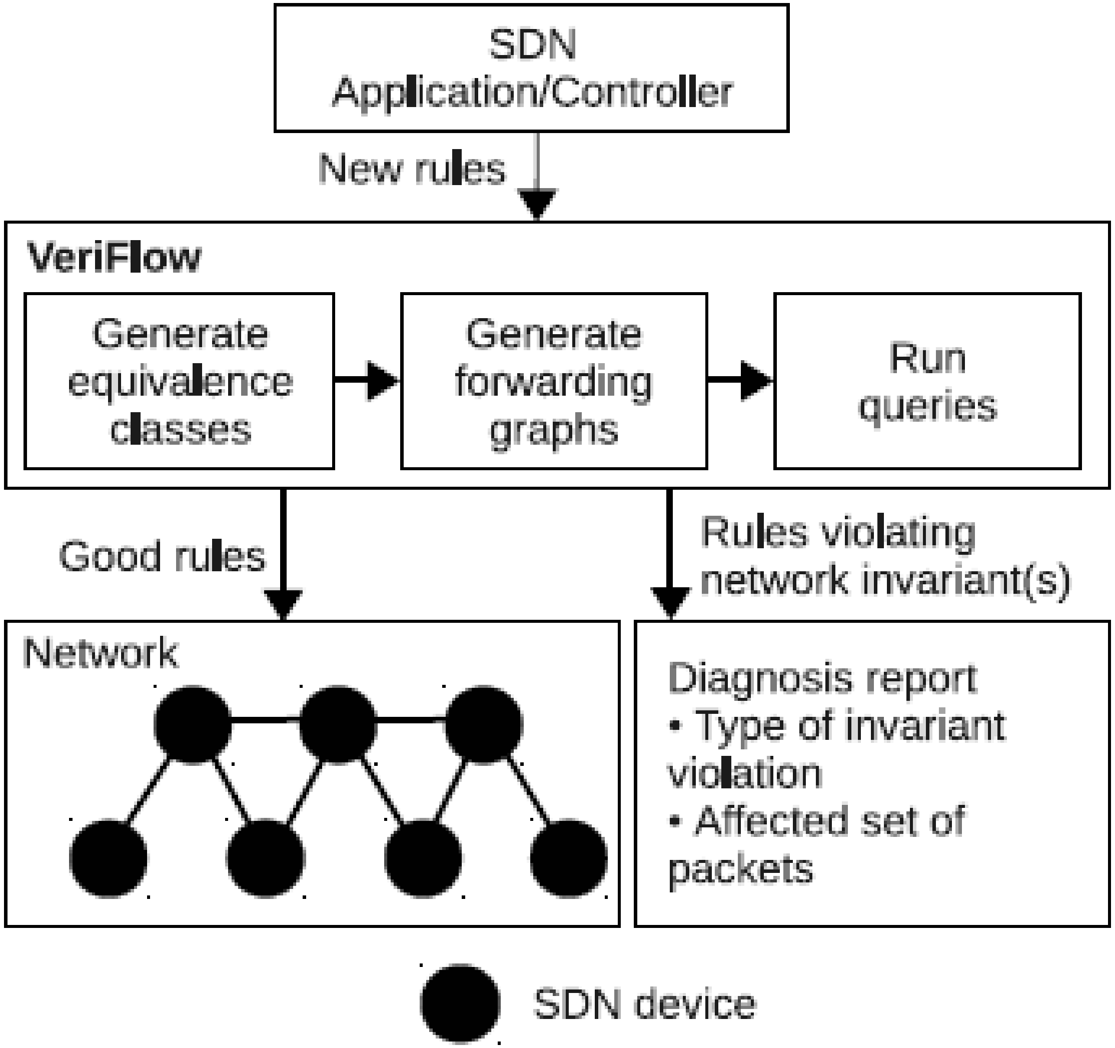
- Path and behavior analysis
- Continous audit of network intent
- Prediction of changes in a sandbox

Network visibility, policy verification, and change modeling

VERIFLOW

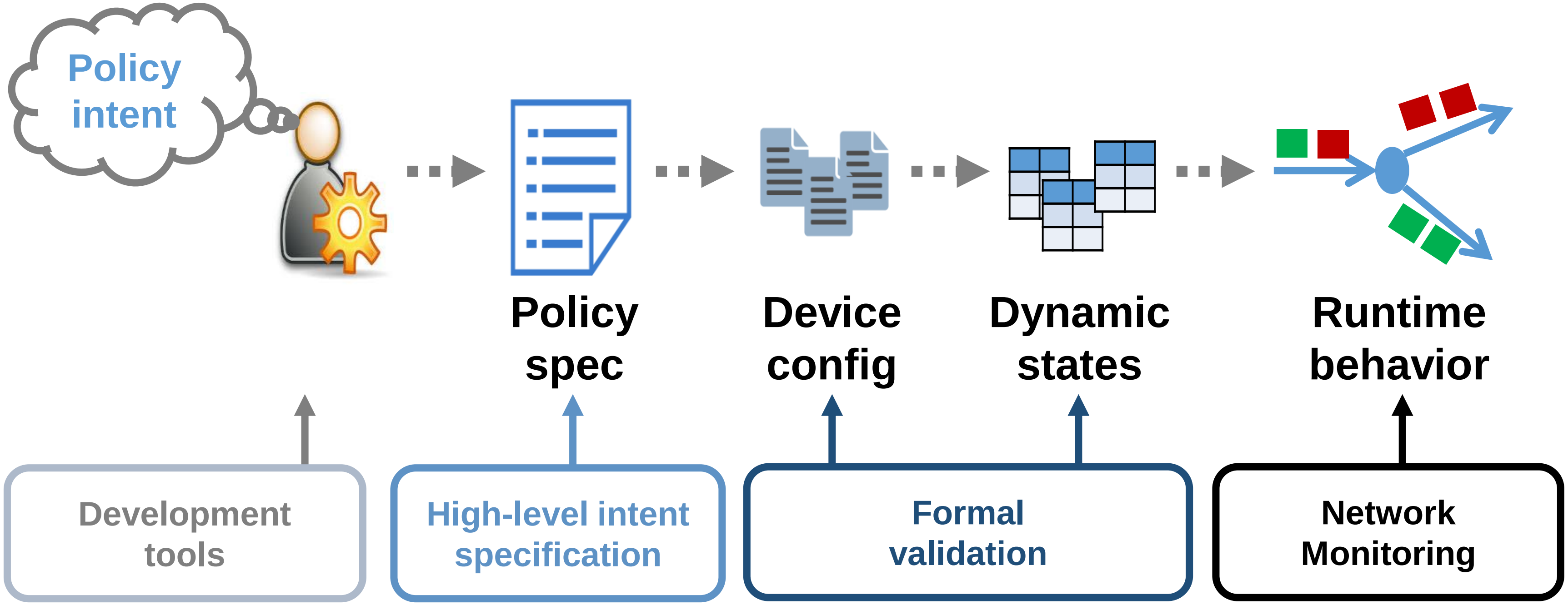
- Assuring network correctness, based on a mathematical verification of the entire network state using advanced algorithms and a predictive model of all possible network behavior.
- With continuous network verification, Veriflow predicts network outages and vulnerabilities before they occur.
- The result is a dramatic improvement in network design, implementation and operation, allowing for rigorous assurance of network resilience and protection.

Verifying Network-Wide Invariants in Real Time


SDN
Application/Controller
New rules
VeriFlow
Generate equivalence classes
Generate forwarding graphs
Run queries
Good rules
Rules violating network invariant(s)
Network
Diagnosis report
• Type of invariant violation
• Affected set of packets
● SDN device

Prescriptive (how) → Declarative (what)

INTENTIONET



Policy intent

Policy spec

Device config

Dynamic states

Runtime behavior

Development tools

High-level intent specification

Formal validation

Network Monitoring

Provide strong correctness *guaran*

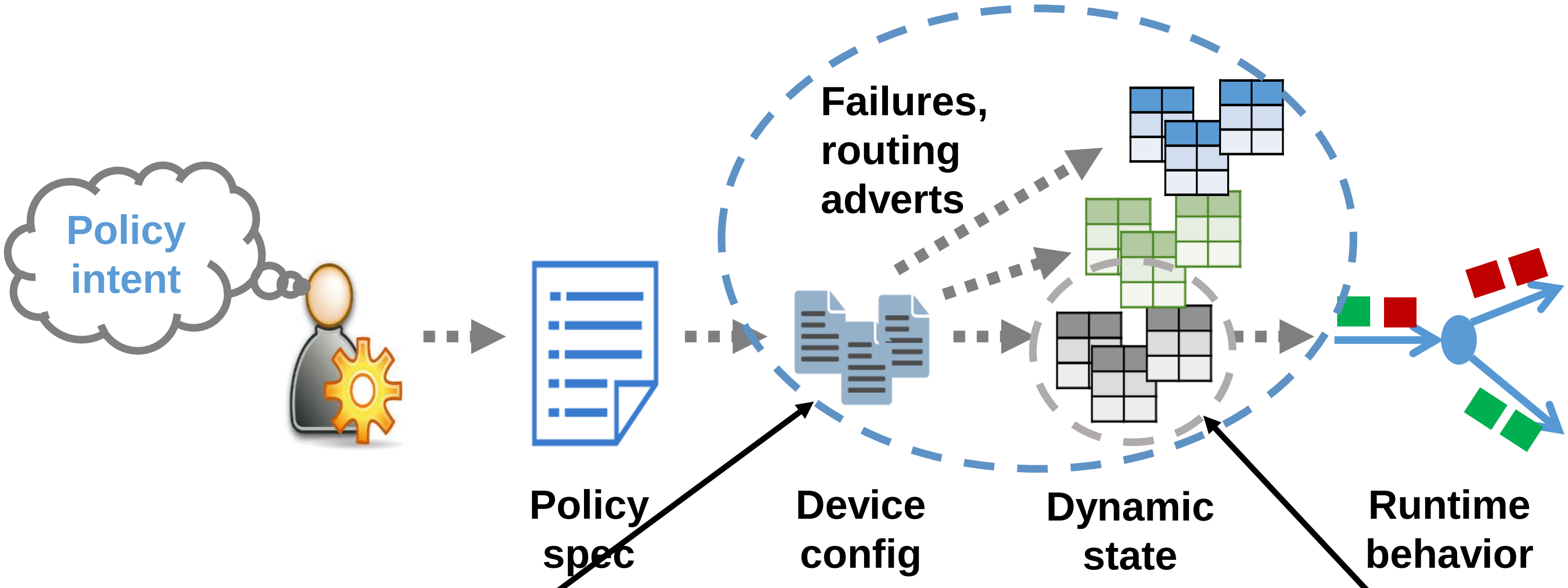
Catch bugs early in the pipeline

Simplify intent expression

Enable safe, rapid evolution

Possible today!

INTENTIONET



Policy intent

Failures, routing adverts

Policy spec

Device config

Dynamic state

Runtime behavior

Control plane validation

Analyzes configs, all possible network states

Batfish, ARC, Minesweeper, …

Data plane validation

Analyzes the current network state

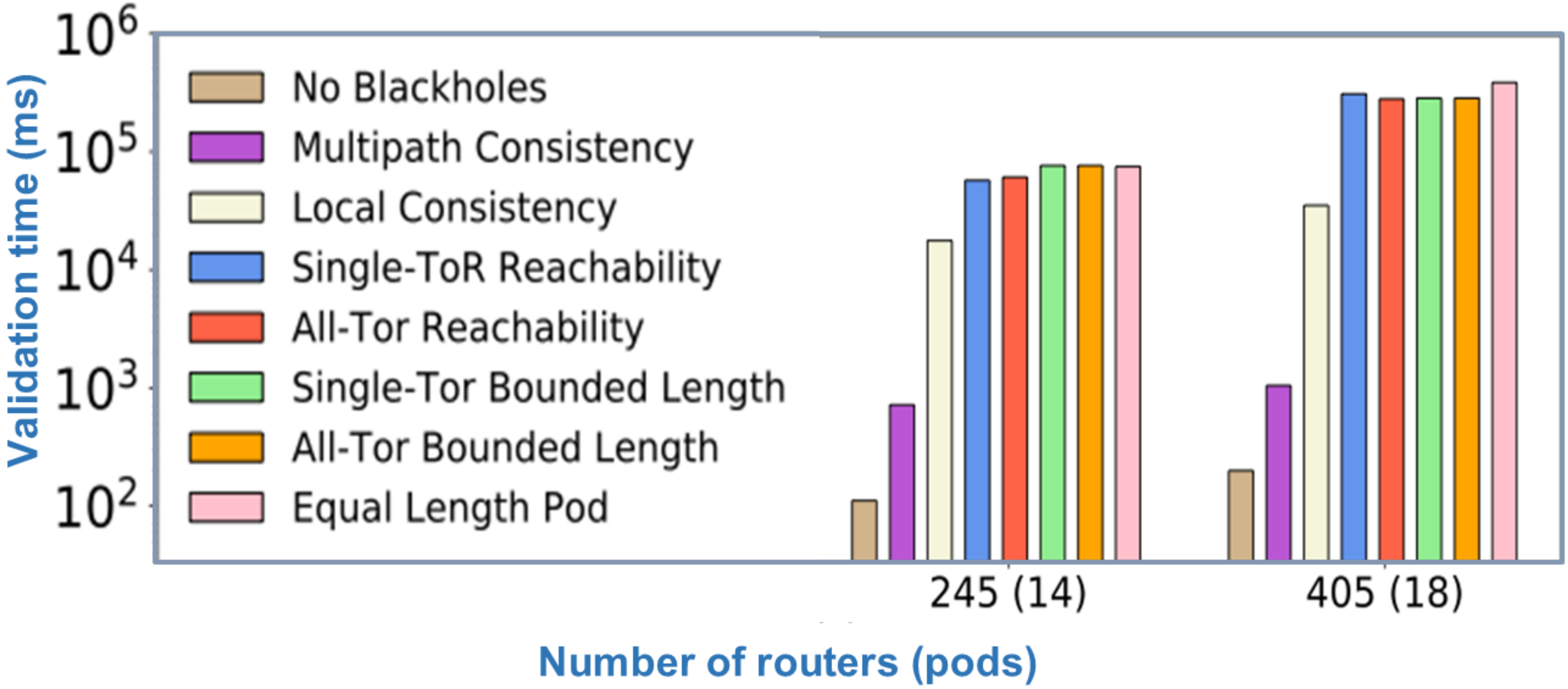
HSA, VeriFlow, Delta-Net, …

www.batfish.org

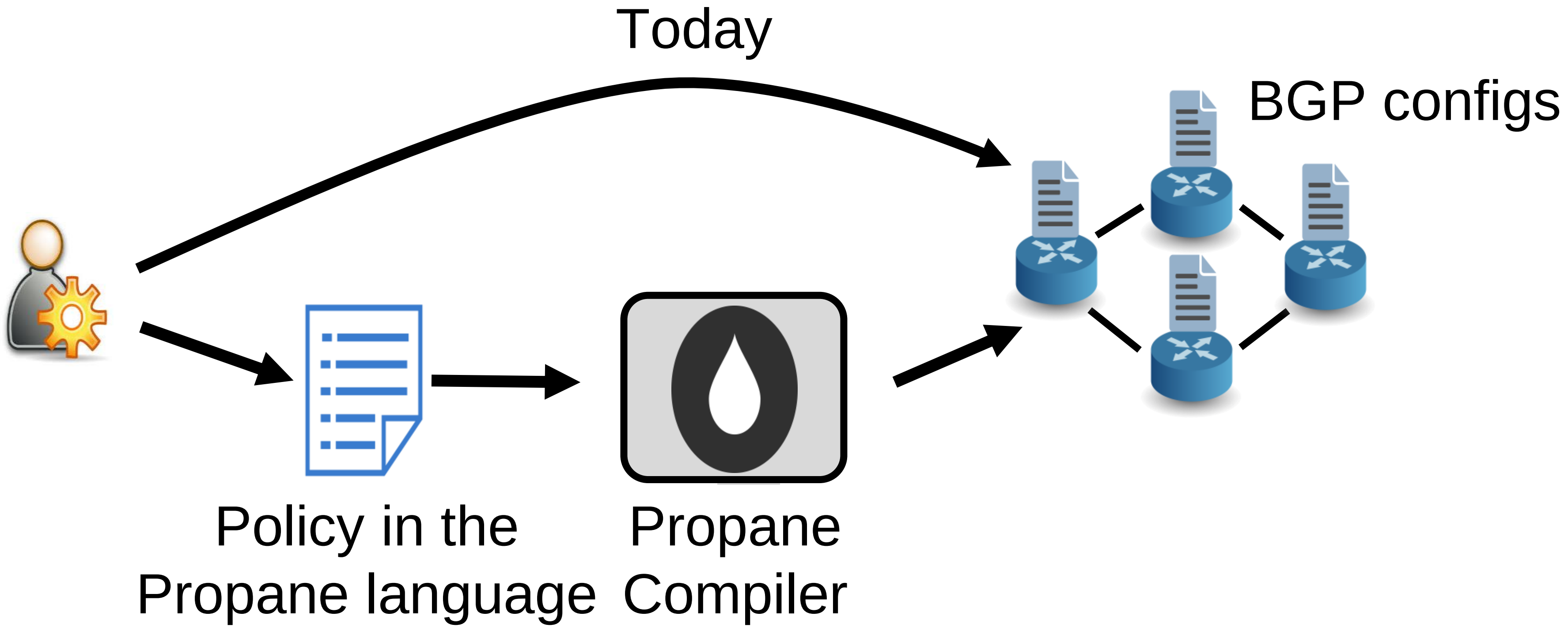
[A general approach to network configuration analysis, NSDI ‘15]

[A general approach to network configuration verification, SIGCOMM‘17]

Control plane validation for large networks



INTENTIONET



Today

BGP configs

Policy in the Propane language

Propane Compiler

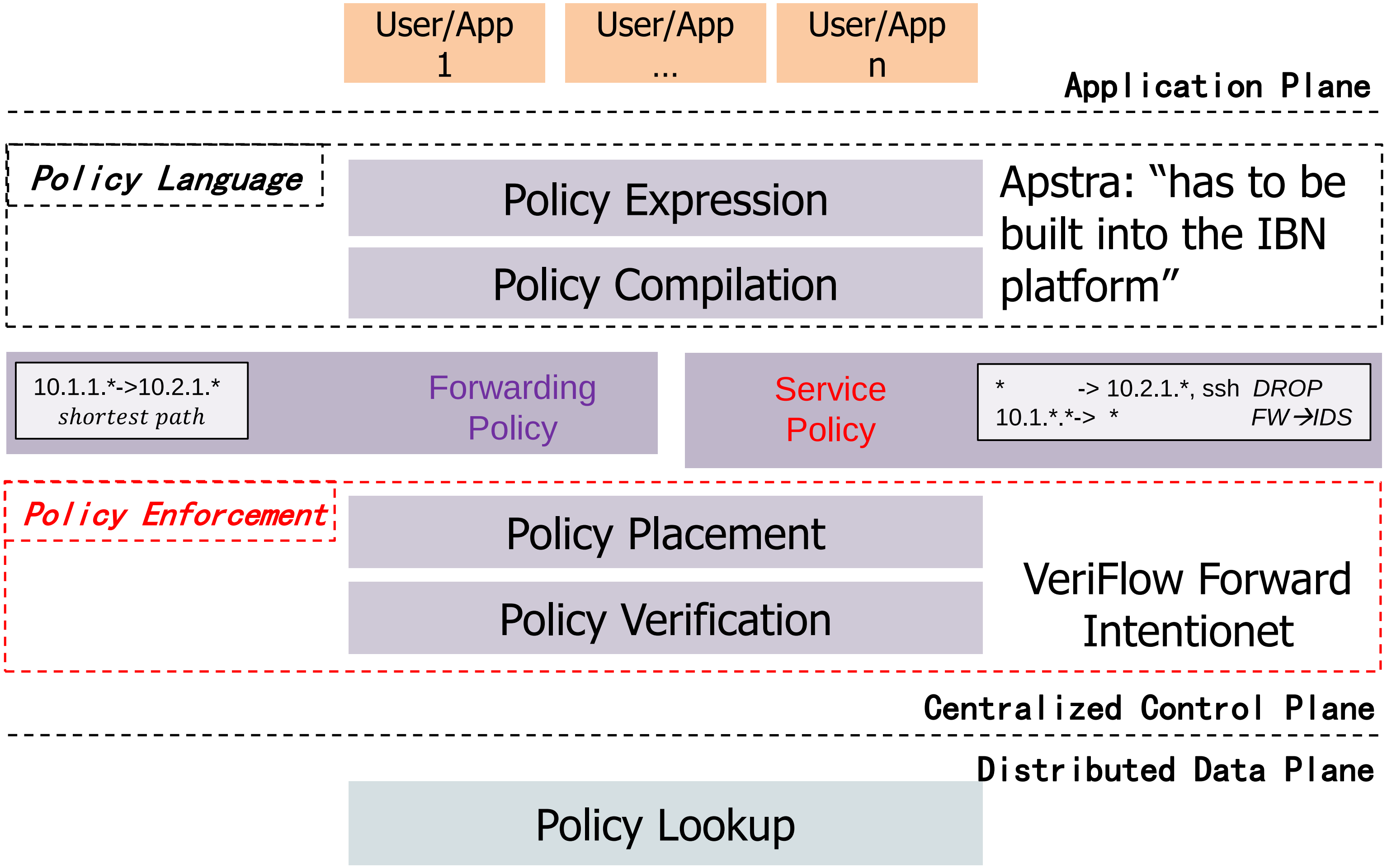
Data center	Backbone network
31 lines of Propane	43 lines of Propane

Human designed configurations were O(10K) lines!

www.propane-lang.org

[Don’t mind the gap: Bridging network-level objectives and device-level configurations, SIGCOMM ‘16]
[Network configuration synthesis with abstract topologies, PLDI ‘17]

Policy Management (review)



User/App 1

User/App …

User/App n

Application Plane

Policy Language

Policy Expression

Policy Compilation

Apstra: "has to be built into the IBN platform"

10.1.1.*->10.2.1.*
shortest path

Forwarding Policy

Service Policy

*   -> 10.2.1.*, ssh *DROP*
10.1.*.*-> *   *FW→IDS*

Policy Enforcement

Policy Placement

Policy Verification

VeriFlow Forward Intentionet

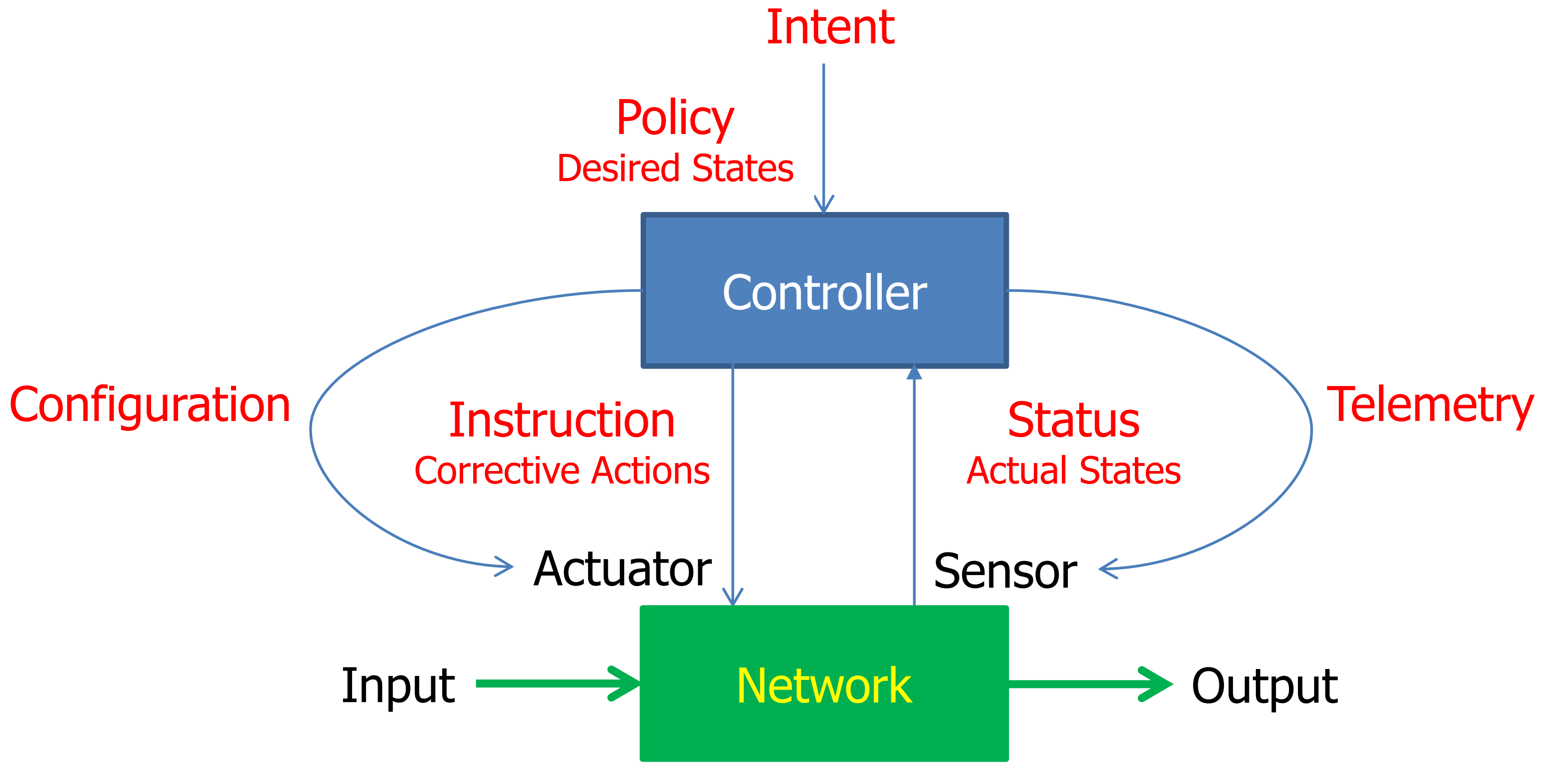
Centralized Control Plane

Distributed Data Plane

Policy Lookup

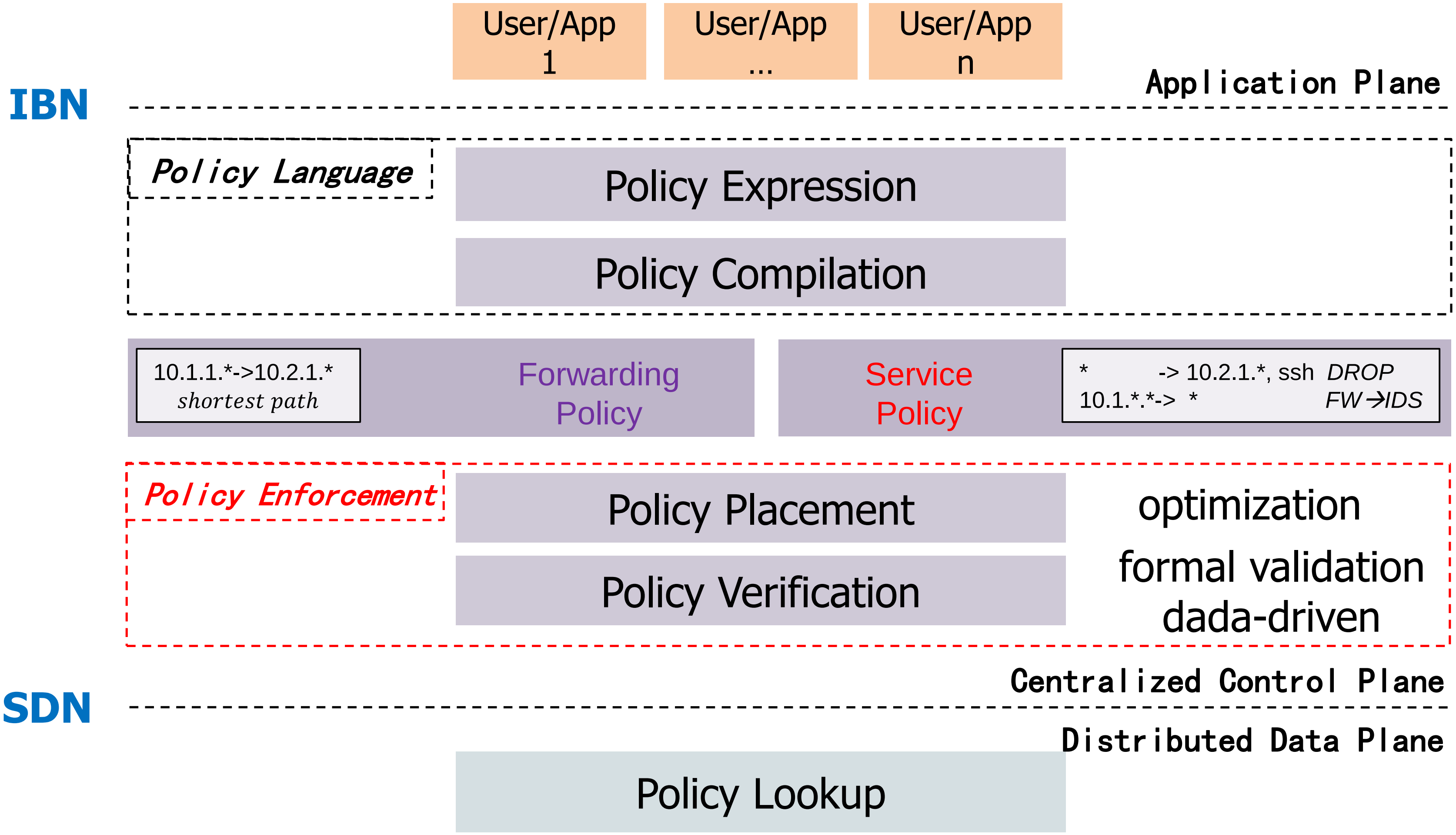
Research Directions

AutoNet: Network Automation (review)



VeriFlow
Intentionet
Forward

Policy Management (outlook)



User/App 1

User/App …

User/App n

Application Plane

IBN

Policy Language

Policy Expression

Policy Compilation

10.1.1.*->10.2.1.*
shortest path

Forwarding Policy

Service Policy

*	-> 10.2.1.*, ssh	*DROP*
10.1.*.*->	*	*FW→IDS*

Policy Enforcement

Policy Placement

Policy Verification

optimization

formal validation

dada-driven

Centralized Control Plane

SDN

Distributed Data Plane

Policy Lookup

Policy Language

- Policy Expression
  - DATALOG-based query
    - FML *[T. L. Hinrichs et al., WREN'09]*
  - Logical labels
    - PGA *[C. Prakash et al., SIGCOMM'15]*
- Policy Composition
  - High-level language for writing and composing modules
    - Frenetic *[N. Foster et al., SIGPLAN'11]*
    - Pyretic *[C. Monsanto et al., NSDI'13]*

Policy Enforcement (I)

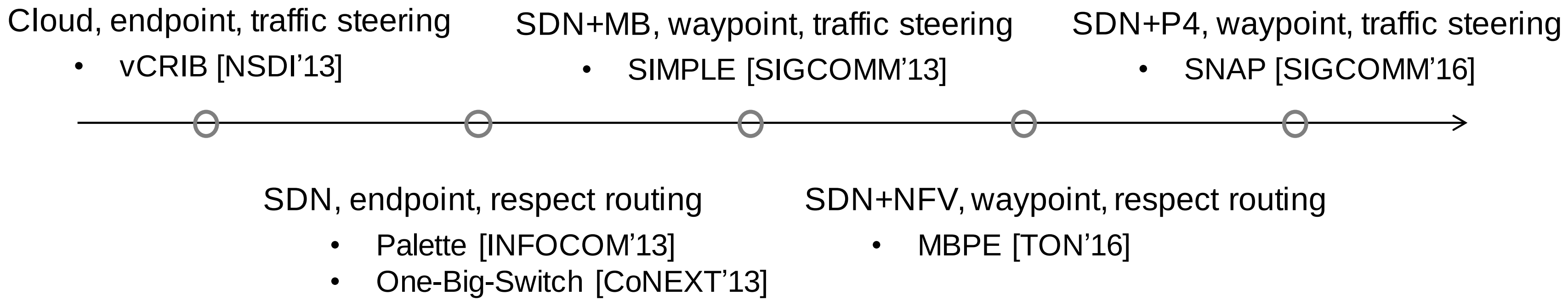
- Policy Placement
  - Distributed endpoint policies w/ forwarding changes
    - vCRIB *[M. Moshref et al., NSDI’13]*
  - Distributed endpoint policies w/o forwarding changes
    - Palette *[Y. Kanizo et al., INFOCOM’13]*
    - One-Big-Switch *[N. Kang et al., CoNEXT’13]*
  - Distributed waypoint policies w/ forwarding changes
    - SIMPLE *[Z. A. Qazi et al., SIGCOMM’13]*
    - *SNAP [M. T. Arashloo et al, SIGCOMM’16]*
  - Distributed waypoint policies w/o forwarding changes
    - MBPE *[X. Wang et al., TON’16]*

Policy Enforcement (II)

- Policy Verification
  - Telemetry
    - NetSight *[N. Handigol et al., NSDI’14]*
    - Everflow *[Y. Zhu et al., SIGCOMM’15]*
  - Data plane verification
    - HSA *[P. Kazemian et al., NSDI’12 & 13]*
    - Veriflow *[A. Khurshid et al., NSDI’13]*
  - Control plane verification
    - Batfish *[A. Fogel et al., NSDI’15]*
    - *Minesweeper [R.* Beckett et al, SIGCOMM’17*]*

Policy Placement

- Problem
  - Policy space partition and data plane location
  - Various application scenario, constraint and optimization objective



Cloud, endpoint, traffic steering

- vCRIB [NSDI’13]

SDN, endpoint, respect routing

- Palette [INFOCOM’13]
- One-Big-Switch [CoNEXT’13]

SDN+MB, waypoint, traffic steering

- SIMPLE [SIGCOMM’13]

SDN+NFV, waypoint, respect routing

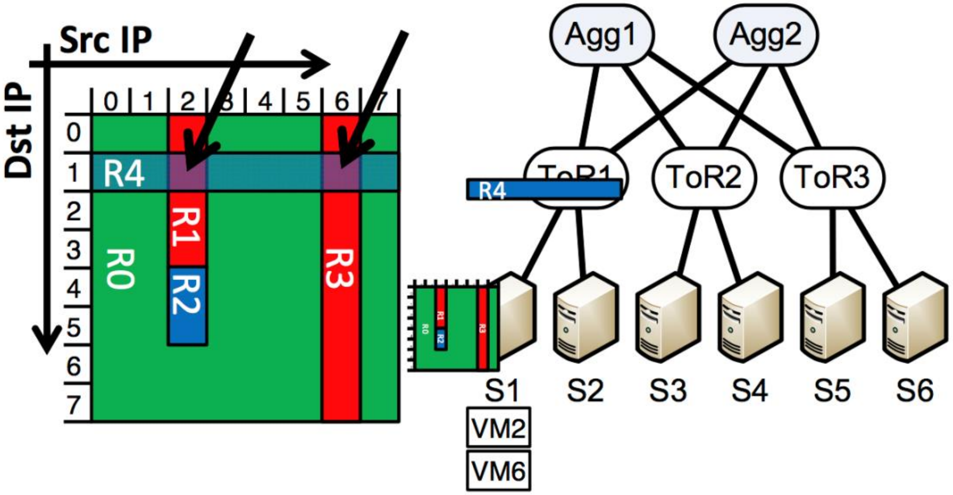
- MBPE [TON’16]

SDN+P4, waypoint, traffic steering

- SNAP [SIGCOMM’16]

vCRIB for Data Center

- Place rules at both hypervisors & switches
  - Per-source partition with rule replication
  - Partition assignment: minimize traffic subject to resource constraints (NP-hard)



S1. Resource-aware placement

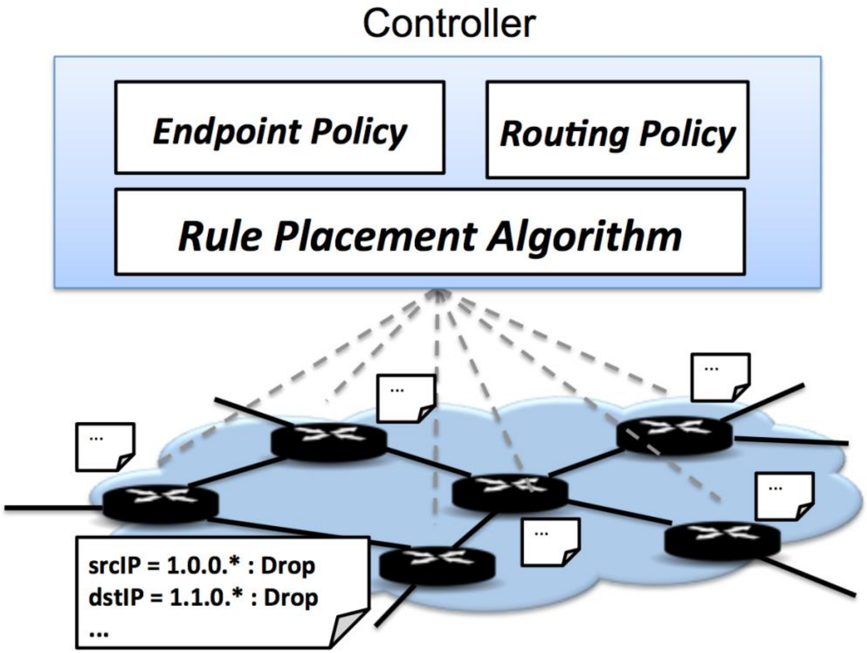
- Bin-packing problem to minimize placed device number

S2. Traffic-aware refinement

- Overhead-greedy algorithm to move partition to resource-sufficient and lowest traffic device

One Big Switch

- Place rules on capacity-limited switches
  - – Clean abstraction and respect routing



S1. Decompose network and policies based on paths

S2. Rule capacity allocation for each path by LP

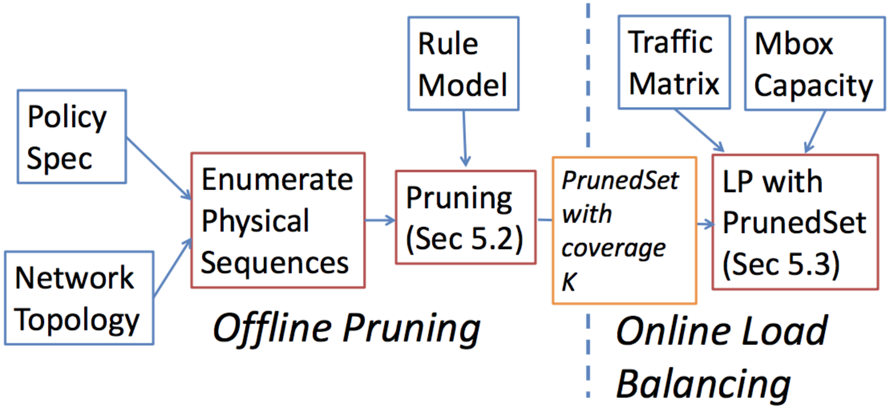
- If S3 fails, increase constraints and recompute

S3. Place rules along a path

- Greedy algorithm to choose rule space to max #internal / #overlap

SIMPLE for Legacy MBs

- Place waypoint policies
  - MB locations are fixed and steering traffic
  - Place rules on capacity-limited switches for MB LB
  - Similar to Online TE solution



Policy Spec

Network Topology

Enumerate Physical Sequences

Rule Model

Pruning (Sec 5.2)

PrunedSet with coverage K

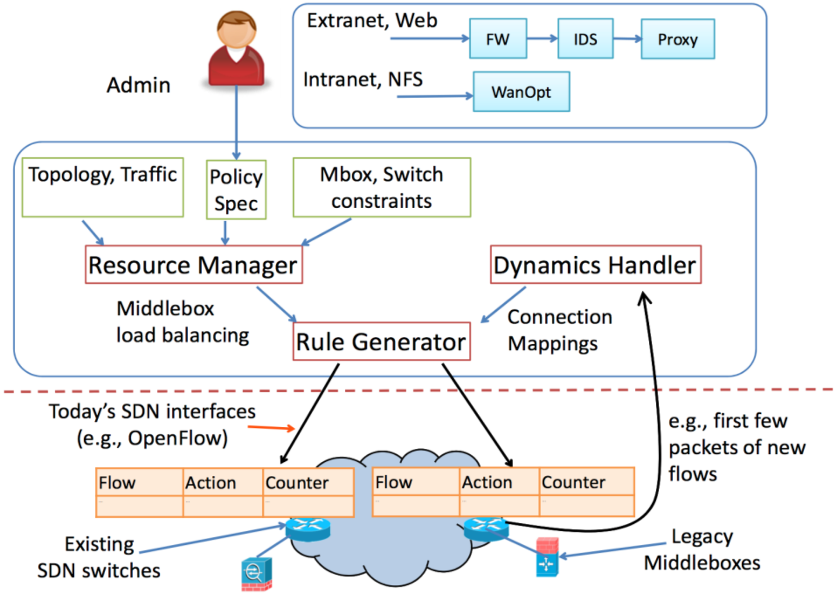
Traffic Matrix

Mbox Capacity

LP with PrunedSet (Sec 5.3)

Offline Pruning

Online Load Balancing



Admin

Extranet, Web

FW

IDS

Proxy

Intranet, NFS

WanOpt

Topology, Traffic

Policy Spec

Mbox, Switch constraints

Resource Manager

Dynamics Handler

Middlebox load balancing

Rule Generator

Connection Mappings

Today's SDN interfaces (e.g., OpenFlow)

e.g., first few packets of new flows

Flow	Action	Counter

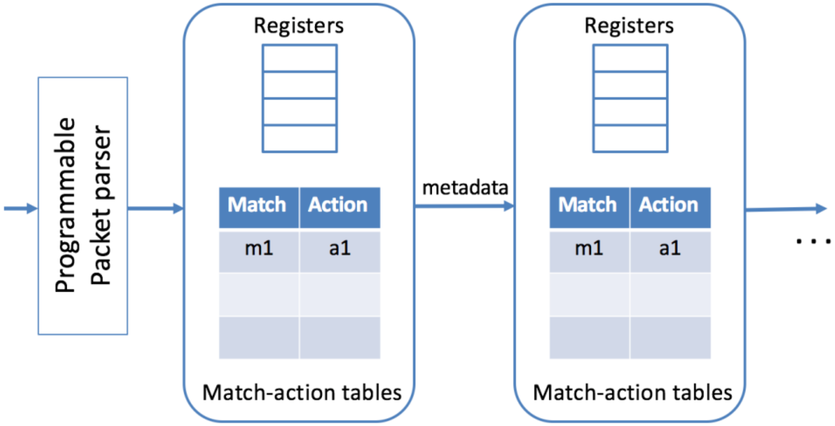
Flow	Action	Counter

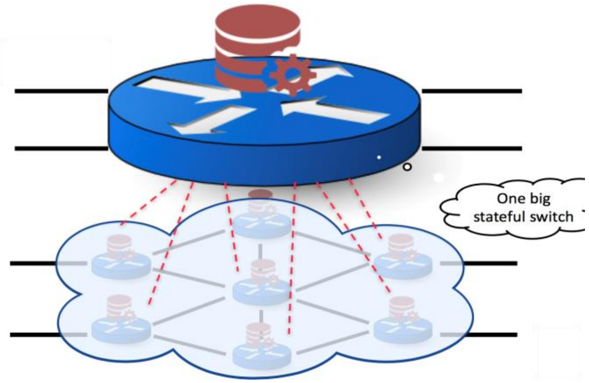
Existing SDN switches

Legacy Middleboxes

SNAP for P4

- Joint state placement and routing
  - Similar to offline TE solution
  - MILP to minimize link utilization with Link capacity constraints and traffic demands
  - Ensure that state isn't duplicated





Verify the Policy Stack

Telemetry

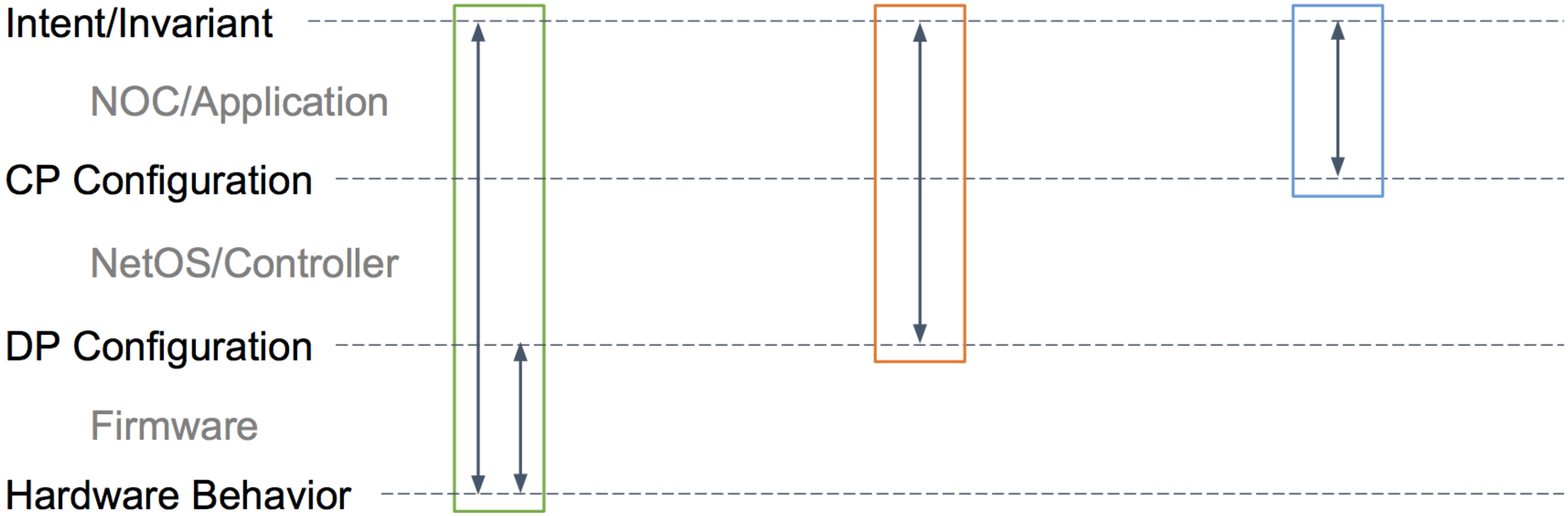
NDB, *HotSDN 12*
APTG, *CoNEXT 12*
NetSight, *NSDI 14*
Everflow, *SIGCOMM 15*
VeriDP, *CoNEXT 16*

DP Analysis

HSA, *NSDI 12*, NetPlumber, *NSDI 13*
VeriFlow, *NSDI 13*
NoD, *NSDI 15*
AP Verifier, *ICNP 13*, DeltaNet, *NSDI 17*
Libra, *NSDI 14*, Surgeries, *POPL 16*

CP Analysis

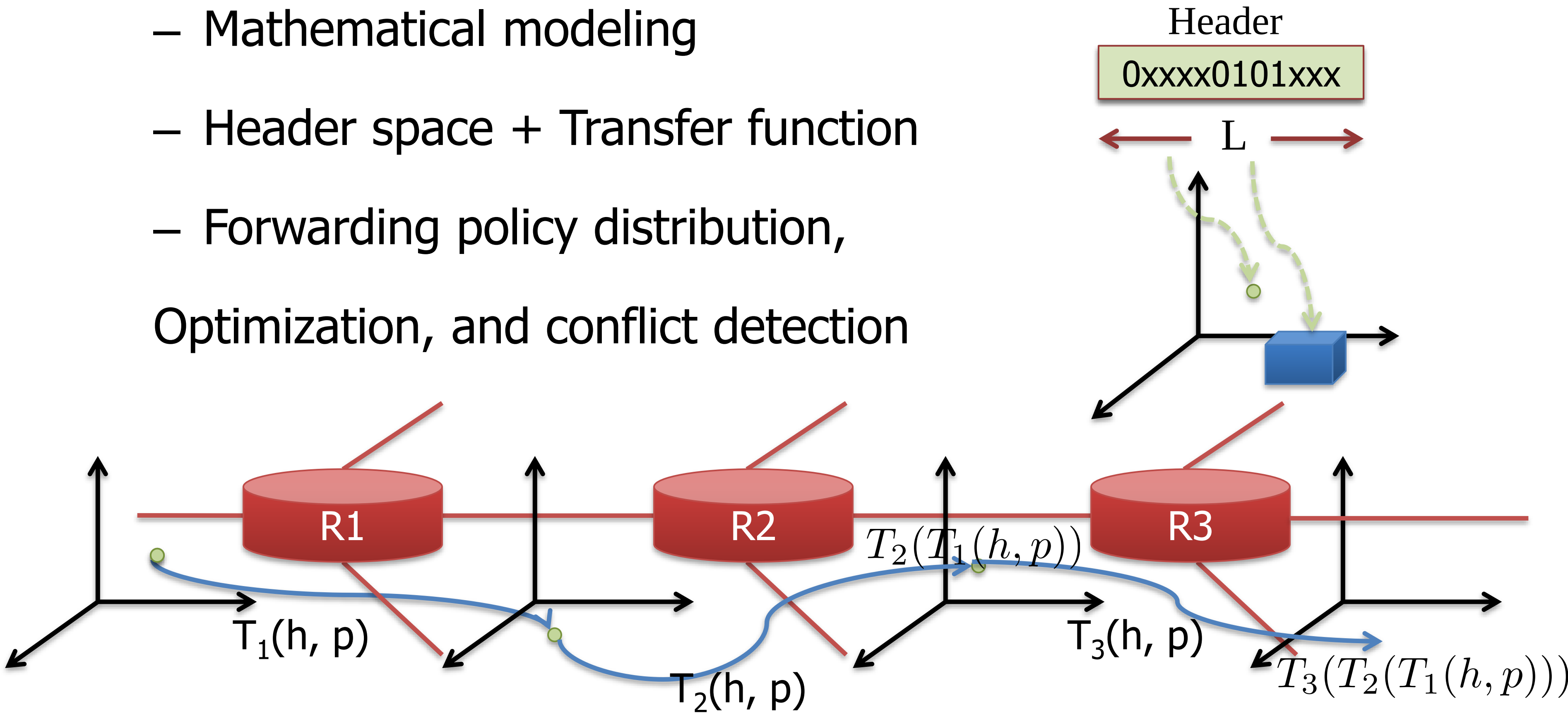
rcc, *NSDI 05,* NICE, *NSDI 12*
BatFish, *NSDI 15*
ERA, *OSDI 16*
ARC, *SIGCOMM 16*
Minesweeper, *SIGCOMM 17*



Intent/Invariant
NOC/Application
CP Configuration
NetOS/Controller
DP Configuration
Firmware
Hardware Behavior

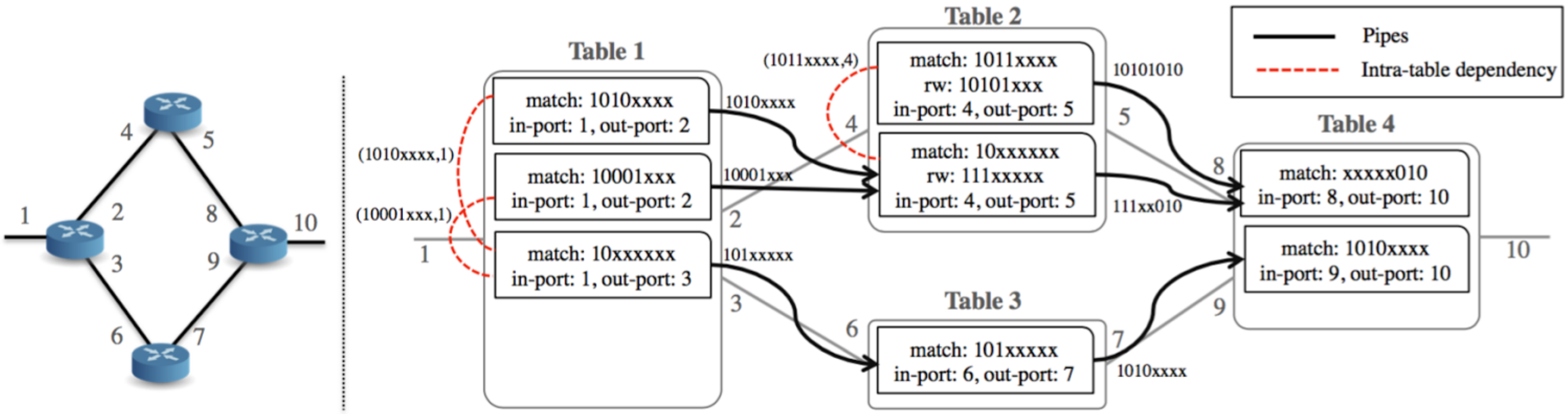
Header Space Analysis

- A simple abstraction to model all kinds of forwarding functionalities *[P. Kazemian et al., NSDI 2012 & 2013]*
  - Mathematical modeling
  - Header space + Transfer function
  - Forwarding policy distribution, Optimization, and conflict detection



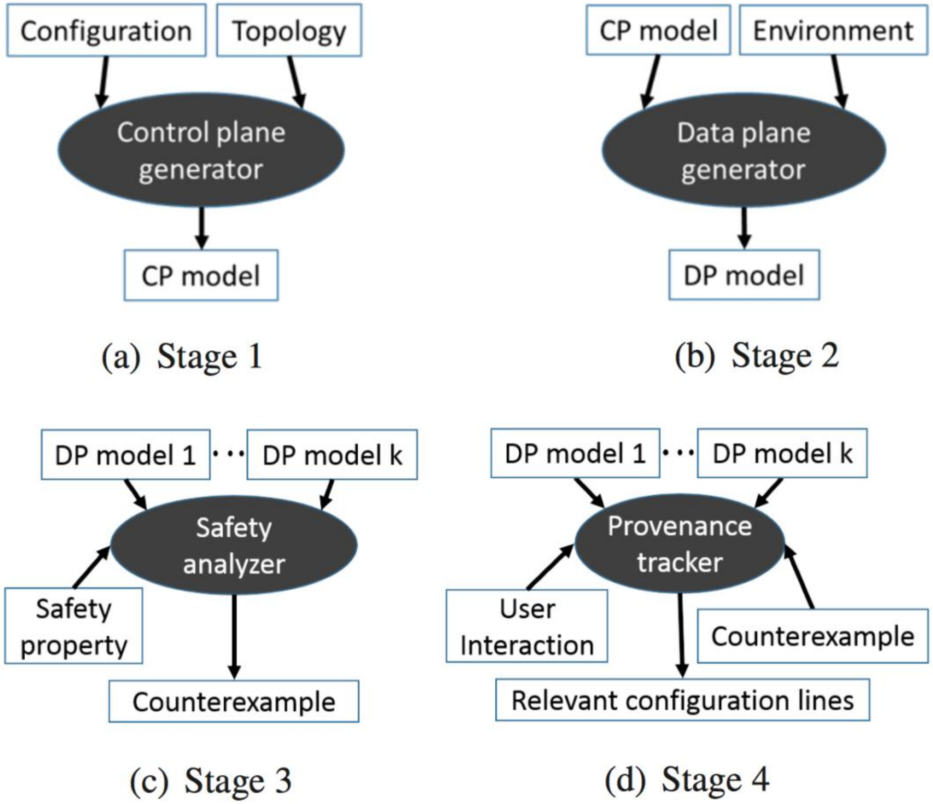
NetPlumber

- Verification in practice (real-time)
  - Check policy compliance and build rule set dependency graph
  - Check policy compliance on affected sub-graph if updates occur



Batfish

- Use CP model to generate DP configurations
- Use SMT to verify DP configurations



(a) Stage 1

(b) Stage 2

(c) Stage 3

(d) Stage 4

Environments

- link up/down
- Neighbor AS announcements

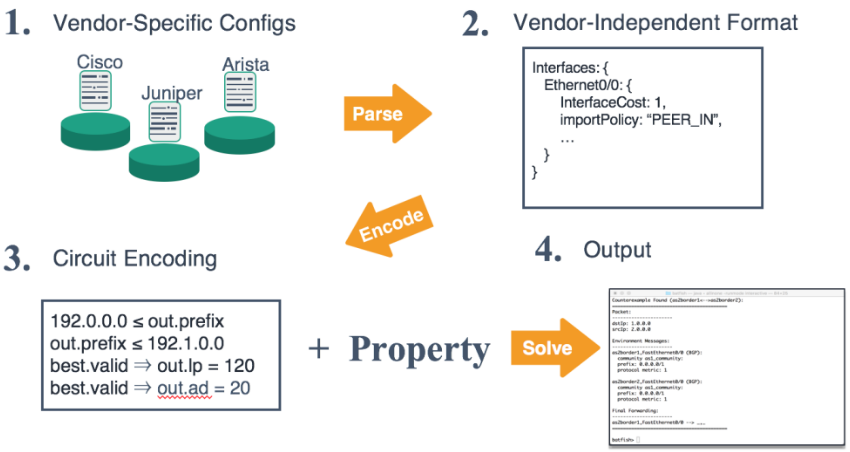
Implementation

- Vendor configuration parser
- Model RFC case by case

Minesweeper

INTENTIONET

- Directly verify CP model to cover all DPs
  - Network as a combinational circuit, inputs and outputs of devices with SMT constraints
  - Combinational search on satisfying assignments to logical formulas



```
Interfaces: {
  Ethernet0/0: {
     InterfaceCost: 1,
     importPolicy: "PEER_IN",
     ...
  }
}
```

```
192.0.0.0 ≤ out.prefix
out.prefix ≤ 192.1.0.0
best.valid ⇒ out.lp = 120
best.valid ⇒ out.ad = 20
```

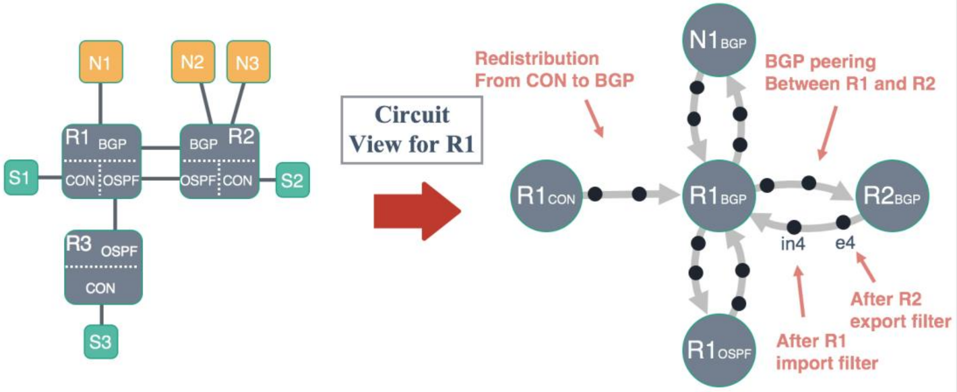
```
Counterexample Found (as2border1<-->as2border2):
==========================================
Packet:
----------------------
dstIp: 1.0.0.0
srcIp: 2.0.0.0

Environment Messages:
----------------------
as2border1,FastEthernet0/0 (BGP):
  community as1_community:
  prefix: 0.0.0.0/1
  protocol metric: 1

as2border2,FastEthernet0/0 (BGP):
  community as1_community:
  prefix: 0.0.0.0/1
  protocol metric: 1

Final Forwarding:
----------------------
as2border1,FastEthernet0/0 --> ...
==========================================

batfish> 
```



Policy Enforcement with Policy Space Analysis

Problems with HSA

- HSA is inefficient for service policy management
  - Space representation in indiscriminate bits
    - The number of HSA computing dimensions is 104 for the classic 5-tuple policy.
    - Service policies usually contain arbitrary range values.
  - Set operations in an overlapping manner
    - Header space *(xxxx)* minus point *(1010)* is *(xxx1)* union *(xx0x)* union *(x1xx)* union *(0xxx)*, resulting in more computing tasks and duplicated sub-spaces while doing set operations.
  - Lack of efficient indexing data structures
    - Set operations are conducted in a linear manner.

Policy Space Analysis

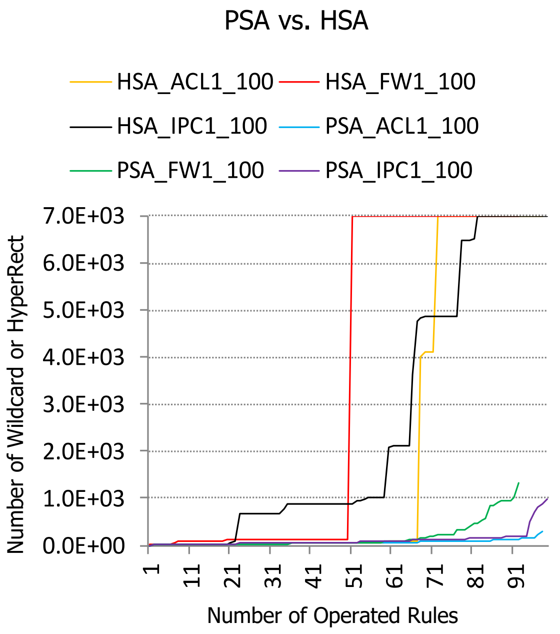
PSA

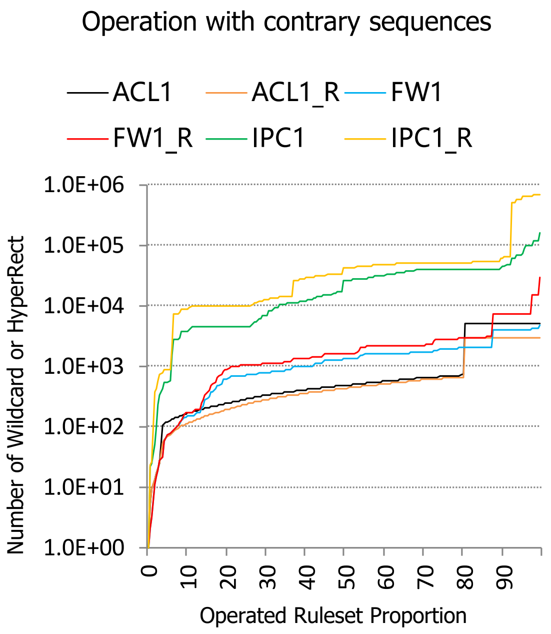
[X. Wang et al., TON 2016]

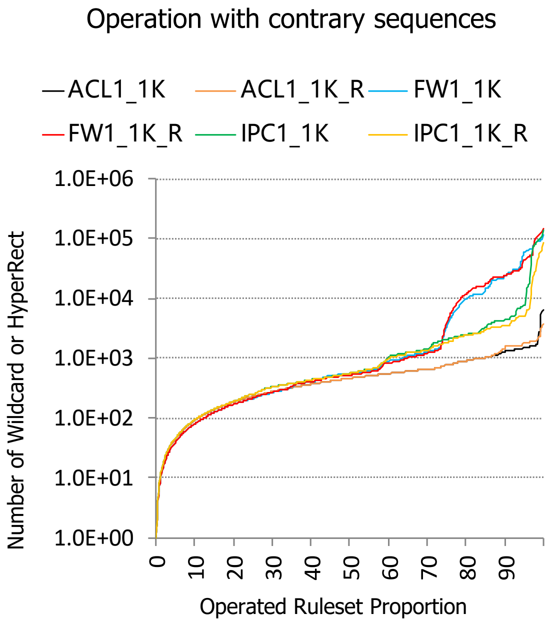
- Computational geometry view
  - Multi-dimensional space
- Expression
  - *HyperRect*: a D-field rule is viewed as a range-based D-dimension hyper-rectangle
  - *PolicySpace*: a set of multiple non-overlapping *HyperRects*
- *Boolean and Set Operations*
  - Leveraged the clipping approach in computer graphics
  - Supported by both *HyperRect* and *PolicySpace*
  - *Boolean Operations*
    - *is_equal*, *is_subset*, and *is_intersected*
  - *Set Operations*
    - *intersect*, *subtract*, and *union*

PSA Evaluation

- Spatial performance
  - Iterated *subtraction* of high-priority rules from low-priority rules to construct a non-overlapping ruleset


PSA vs. HSA


Operation with contrary sequences


Operation with contrary sequences

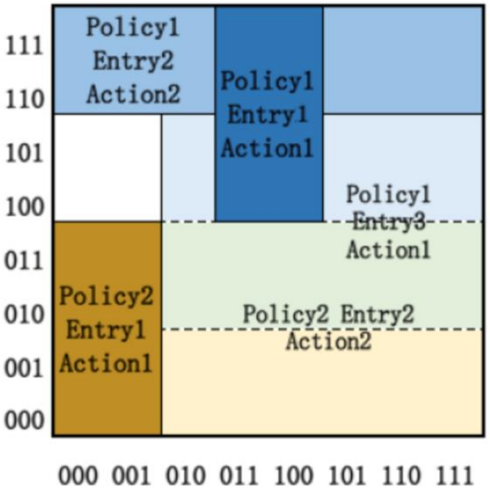
Atomic HyperRect Index

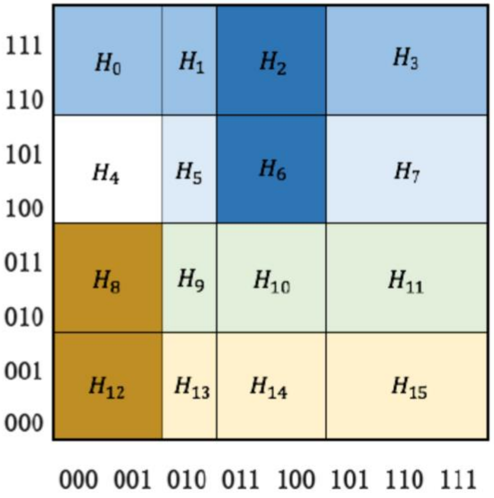
AHI

- Accelerate set operations

[D. Li et al., ICC 2017]

  - Spatial projection and universal space atomization
  - Policy entries as bitmaps of the atomic spaces
  - Convert the set operations between policies into set operations between bitmaps





Policy1 Action1:

$H_2 \cup H_5 \cup H_6 \cup H_7 \cup H_9 \cup H_{10} \cup H_{11}$

0010 0111 0111 0000

Policy2 Action2:

$H_9 \cup H_{10} \cup H_{11} \cup H_{13} \cup H_{14} \cup H_{15}$

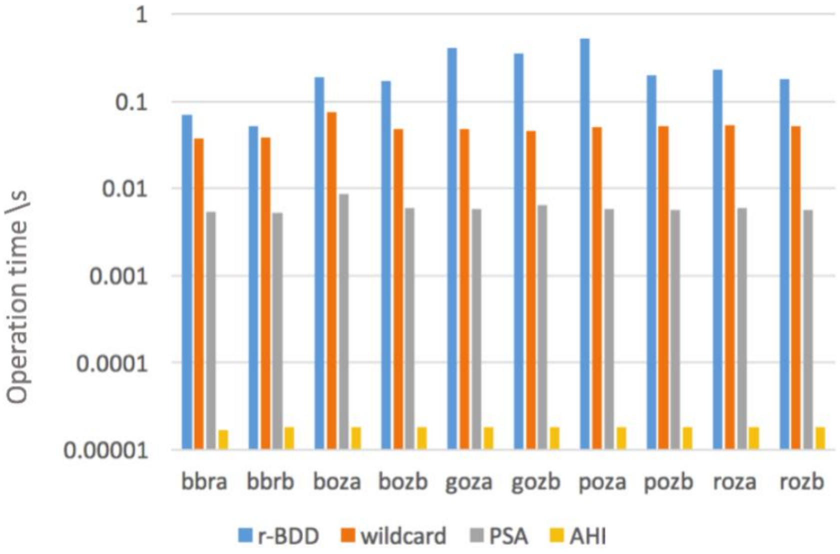
0000 0000 0111 0111

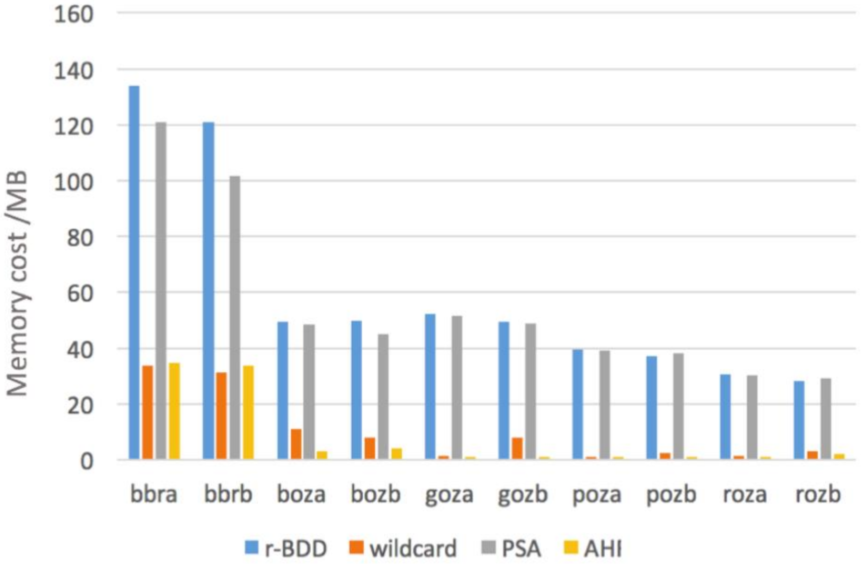
Policy1 Action1 ∩ Policy2 Action2:

0000 0000 0111 0111

AHI Evaluation

- Benchmark test on Stanford network
  - Choose policy randomly to do set operations





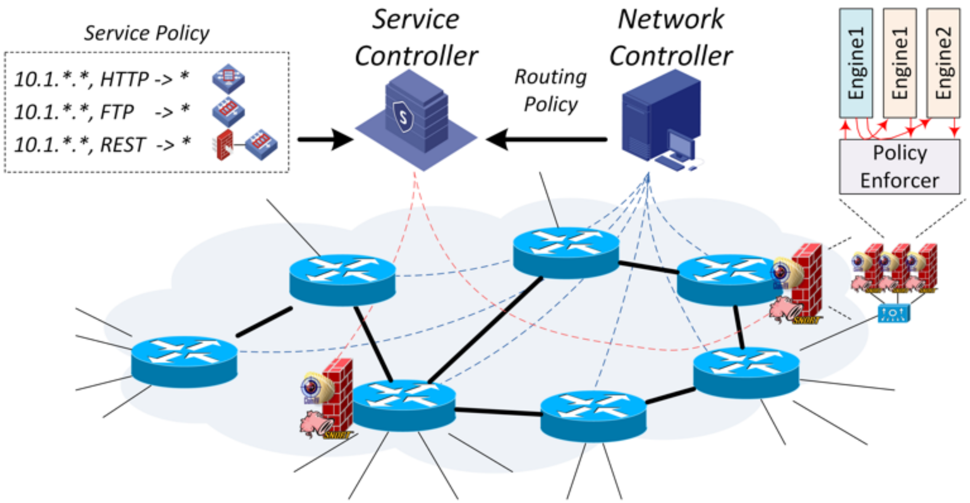
Policy Placement for NFV

- Place waypoint polices to minimize NFV cluster node
  - From path-wise to network-wise to reduce wildcard rules replication
  - On-path processing to reduce bandwidth cost introduced by traffic steering
  - Each policy or policy partition is processed once and only once

MBPE

[X. Wang et al., TON 2016]

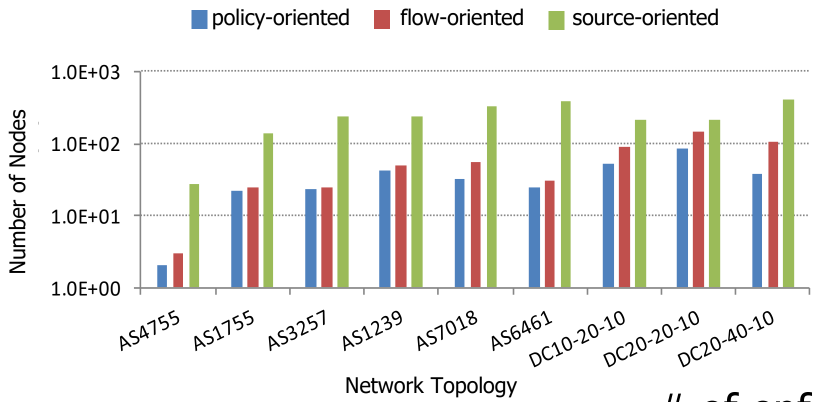
- Modeled as Set Cover problem
  - Use node routing header space to cover (split) waypoint space
  - Greedily choose node with the largest covered space



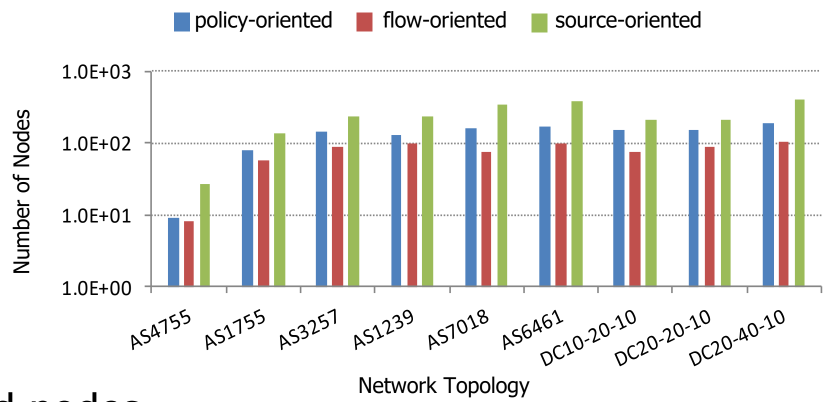
Evaluation

- Enforced nodes and rules

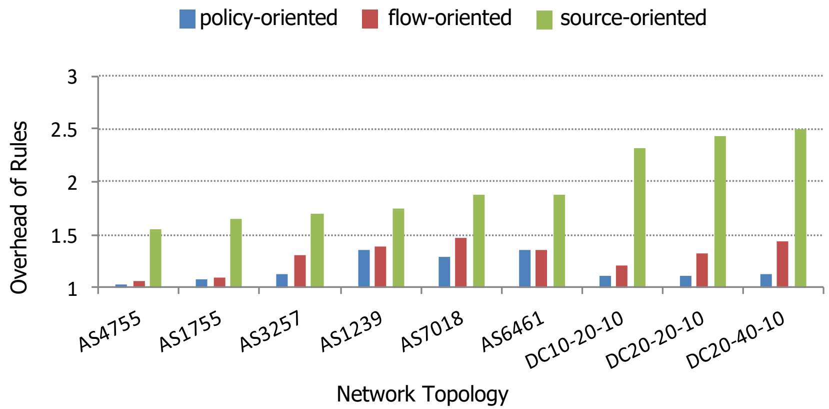
exact policies

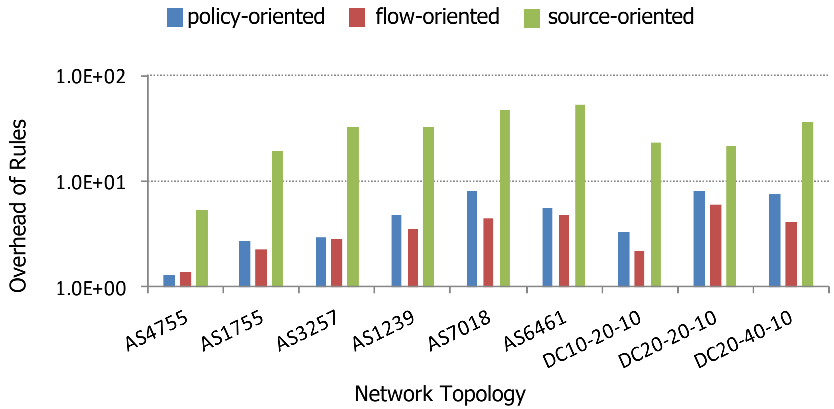


wildcard policies



# of enforced nodes



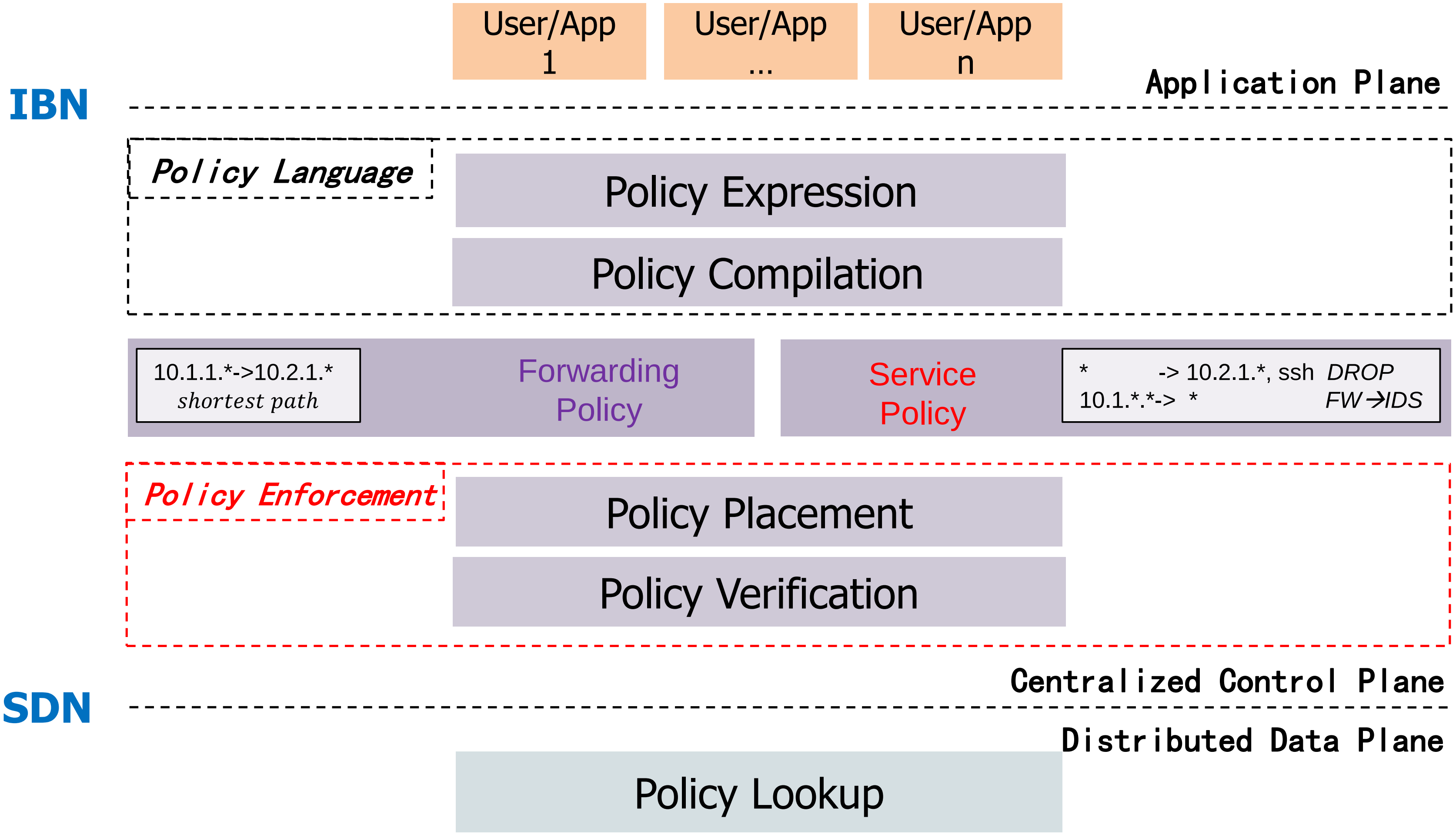


# of all enforced rules / # of the original policy rules

Ongoing Work

- Extended MBPE (policy placement)
  - Practical constraints to Set Cover Problem
- Practical DP analysis (policy verification)
  - Tradeoff between performance and functionality
  - Packet modification and fault localization
- Scalable software classification (policy lookup)
  - Algorithm replacement with decision tree
  - Classification-friendly configuration by policy hypervisor
- Scalable software rate limiting (policy lookup)
  - Lock-free design on multi-core platform
  - HTB decomposition with mathematical formula

Policy Management (summary)



User/App 1

User/App …

User/App n

Application Plane

IBN

Policy Language

Policy Expression

Policy Compilation

10.1.1.*->10.2.1.*
shortest path

Forwarding Policy

Service Policy

* -> 10.2.1.*, ssh	*DROP*
10.1.*.*-> *	*FW→IDS*

Policy Enforcement

Policy Placement

Policy Verification

Centralized Control Plane

SDN

Distributed Data Plane

Policy Lookup

Thanks