Software Defined Networking

Dr. Nick Feamster
Professor

In this course, you will learn about software defined networking and how it is changing the way communications networks are managed, maintained, and secured.

This Module: Verification

- **Motivation:** How do you know the network is doing the right thing?
- Verification techniques
  - Configuration Verification: rcc (pre-SDN)
  - Control Plane Verification: Kinetic
  - Data Plane Verification
    - Header Space Analysis
    - Veriflow

Simple Questions are Hard

- What are all the packet headers from A that can reach B?
- What will happen if I remove an entry from a firewall?
- Is Group X provably isolated from Group Y?
- Are there any loops in the network?
- Why is my network slow?

Configuration Defines Behavior

Provides flexibility for realizing operational goals

- How traffic enters and leaves the network
  - Load balance
  - Traffic engineering
  - Primary/backup paths
- Which neighboring networks can send traffic
  - Defines business relationships and contracts
- How routers within the network learn routes
  - Scaling and performance

Flexibility ⟶ **Complexity**

Most Important Goal: Correctness

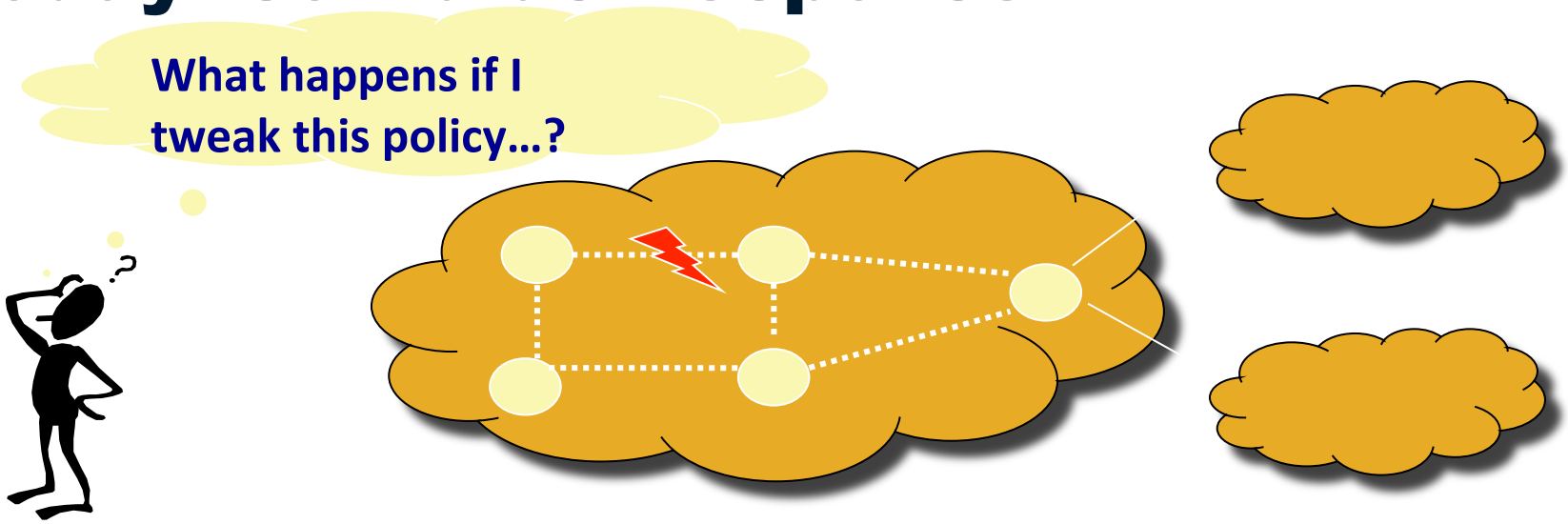
Unfortunately…

Mistakes happen!

Why?

- Configuration is difficult. Operators make mistakes.
  - Complex policies
  - Configuration is distributed across routers
- Each network independently configured
  - Unintended policy interactions

Today: Stimulus-Response

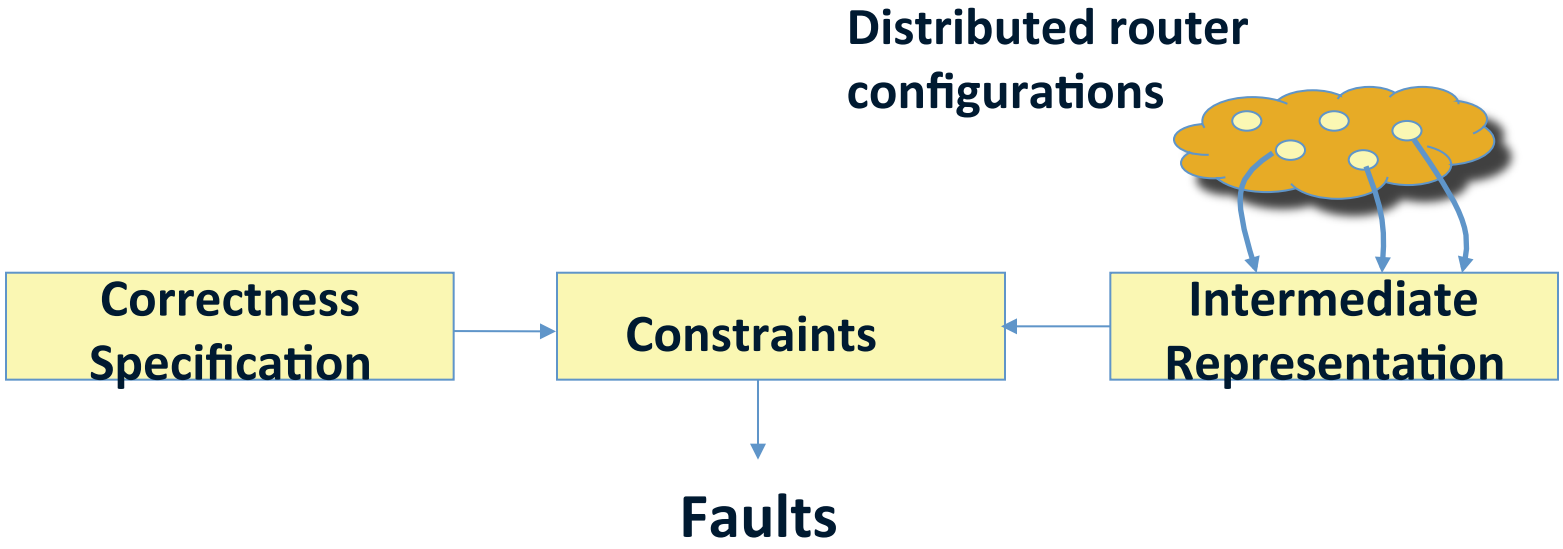


- Unacceptable programming environment
- Mistakes cause downtime
- Mistakes often not immediately apparent

Checking Configuration

- Correctness specification and constraints for global Internet routing
- rcc ("router configuration checker")
  - Static configuration analysis tool for fault detection
  - Used by network operators (including large backbone networks)
- Analysis of real-world network configurations from 17 autonomous systems

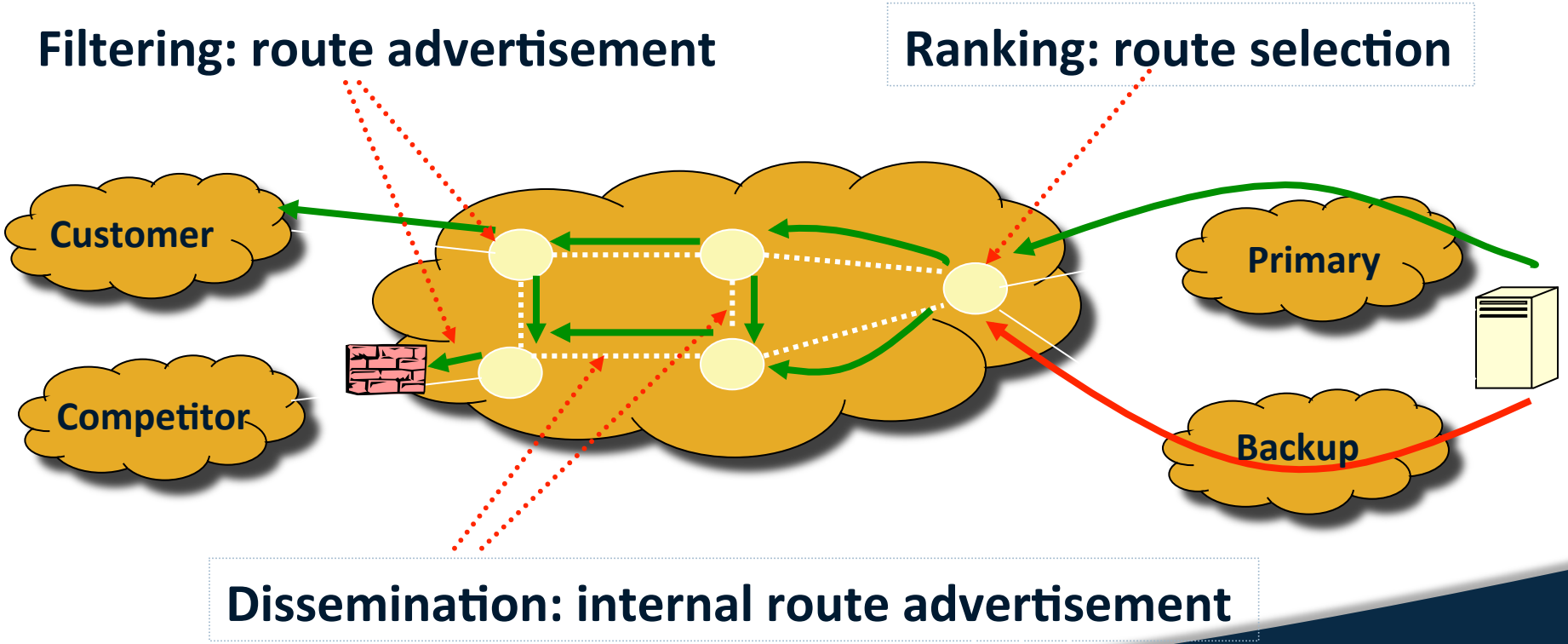
rcc Design


Distributed router configurations
Correctness Specification
Constraints
Intermediate Representation
Faults

Challenges

- Defining a correctness specification
- Deriving verifiable constraints from specification
- Analyzing complex, distributed configuration

Factoring Routing Configuration



Path Visibility

If every router learns a route for every usable path, then path visibility is satisfied.

A *usable path*:

- Reaches the destination
- Corresponds to the path that packets take when using that route
- Conforms to the policies of the routers on that path

Possible path visibility faults

Dissemination

- **Partition in session-level graph that disseminates routes**

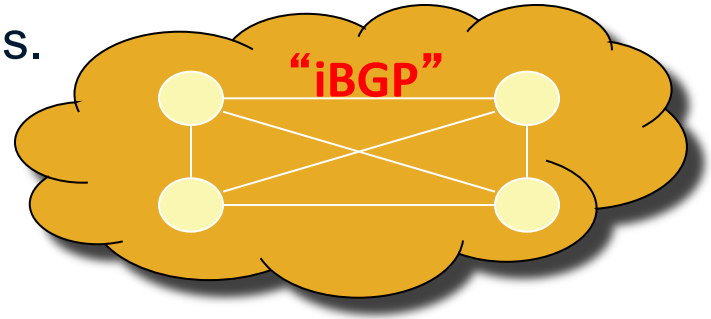
Filtering

- **Filtering routes for prefixes for usable paths**

Path Visibility: Internal BGP (iBGP)

- **Default:** dont re-advertise iBGP-learned routes.

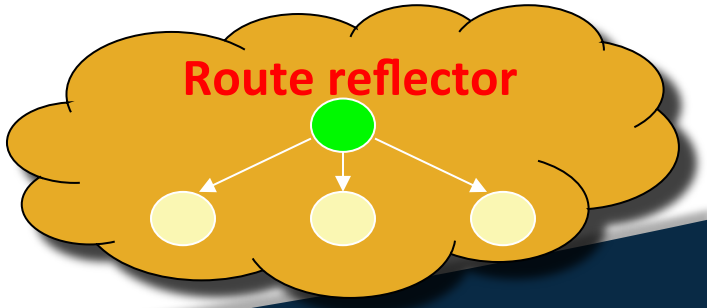
  Complete propagation requires “full mesh” iBGP. *Doesn’t scale.*



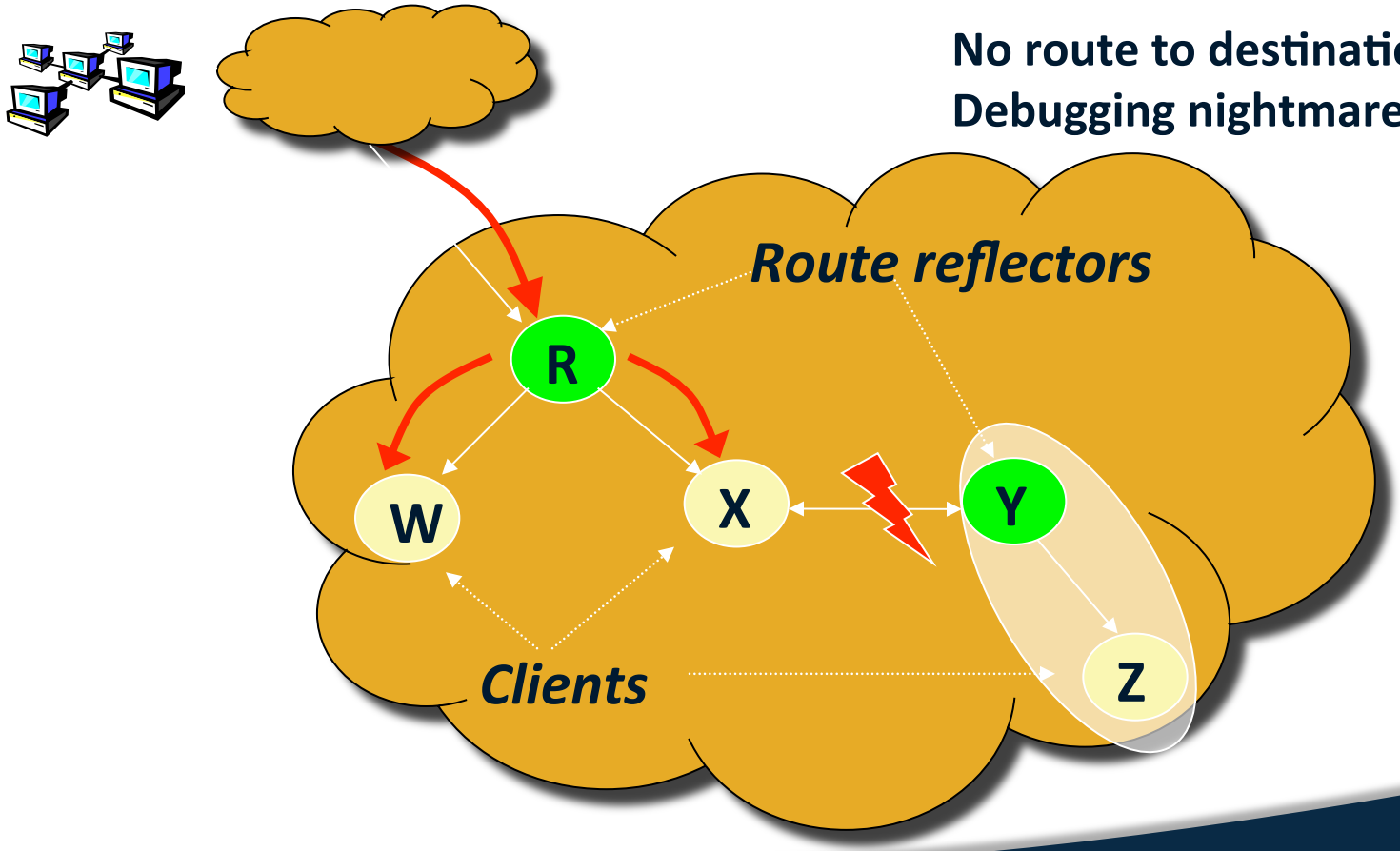
- “Route reflection” improves scaling.

  Client: re-advertise as usual.

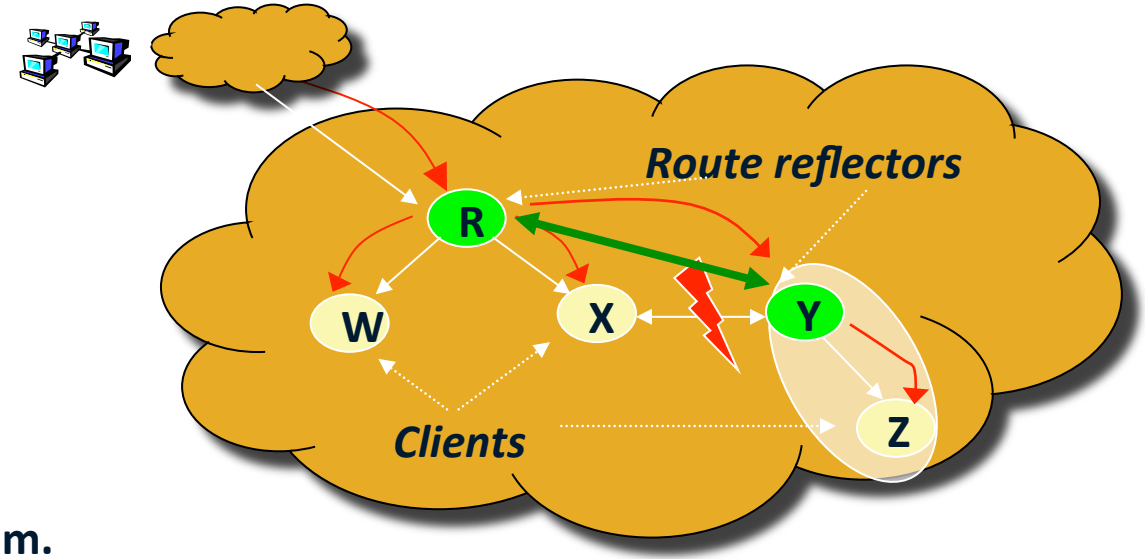
  Route reflector: reflect non-client routes to all clients, client routes to non-clients and other clients.



Path Visibility: iBGP Signaling



Path Visibility: iBGP Signaling



Theorem.
Suppose the iBGP reflector-client relationship graph contains no cycles. Then, path visibility is safisfied if, and only if, *the set of routers that are not route reflector clients forms a full mesh*.

Condition is easy to check with static analysis.

Route Validity

If every route that a router learns corresponds to a usable path, then route validity is satisfied.

A *usable path*:

- Reaches the destination
- Corresponds to the path that packets take when using that route
- Conforms to the policies of the routers on that

Possible route validity faults

Filtering

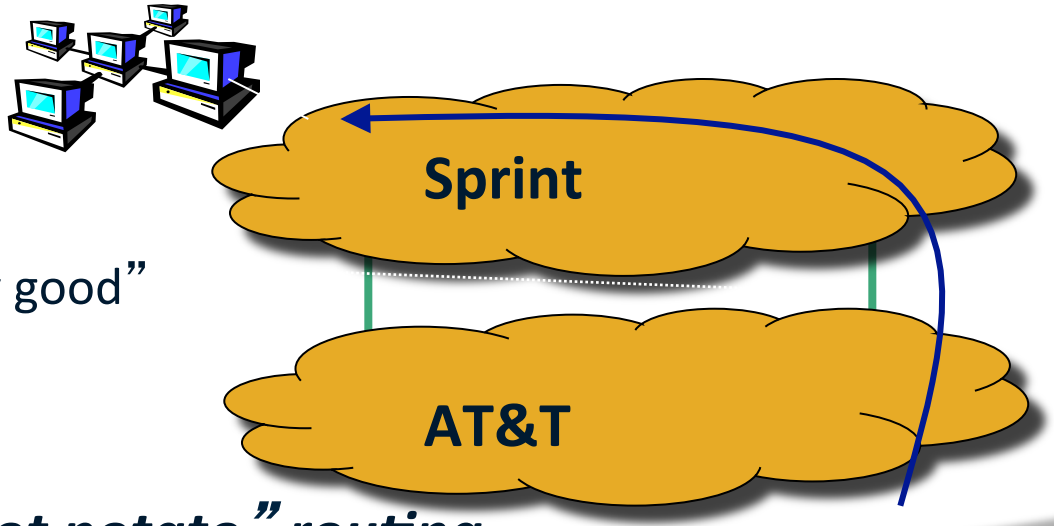
- Unintentionally providing transit service
- Advertising routes that violate higher-level policy
- Originating routes for private (or unowned) address space

Dissemination

- Loops and “deflections

Route Validity: Consistent Export

- Rules of *settlement-free peering*:
  - Advertise routes at all peering points
  - Advertised routes must have equal "AS path length"



Enables "hot potato" routing.

Summary: Verifying Config is Hard

- Distributed configuration is a bad idea.
- SDN can allow us to treat verification as a distributed program, applying concepts from software engineering, testing, formal verification…

SDN to the rescue…