CS4450

Computer Networks: Architecture and Protocols

Lecture 17

BGP recap

Packet header as an interface

Rachit Agarwal

What do we know so far [1] …

- **Network performance metrics**
    - Transmission delay, propagation delay, queueing delay, bandwidth
- **Sharing networks**
    - Circuit switching, packet switching, and associated tradeoffs
    - **Why** is Internet packet switched?
- **Architectural principles and design goals**
    - Layering principle, End-to-end principle, Fate sharing principle
    - Many important design goals from David Clark's paper
        - And many important missing goals
- **Addressing**
    - **Link layer MAC names,** and scalability challenges at the Internet
    - **Network layer IP addresses**: three requirements, aggregation, CIDR

What do we know so far [2] …

- **Link Layer**
    - Sharing a Broadcast medium, associated challenges, CSMA/CD
    - Link layer addressing: MAC names
    - **Why** Frames? **Why** Switched Ethernet?
    - The Spanning Tree Protocol (STP)

- **Network Layer**
    - **Why Network Layer? Why not just use STP across the Internet?**
    - **IP Addressing**
    - **Routing Tables:** A collection of spanning trees, one per destination
    - **Generating Valid Routing tables (within a domain):**
        - Global view (Link-State Protocol), and limitations
        - Local view (Distance-vector Protocol)
    - **Generating Valid Routing tables (across domains):**
        - **Border Gateway Protocol, Internet structure, routing policies**

Goals for Today's Lecture

- Recap IP addressing and BGP quickly

- Understand IP (the Internet Protocol)
    - Packet Header as a network "interface"

Network Layer

- THE functionality: **delivering the data**

- **THE protocol: Internet Protocol (IP)**

- **Achieves its functionality (delivering the data), using three ideas:**
    - **Addressing** (IP addressing)
    - **Routing** (using a variety of protocols)
    - **Packet header as an interface** (Encapsulating data into packets)

Recap: Three requirements for addressing

- **Scalable routing**
    - How must state must be stored to forward packets?
    - How much state needs to be updated upon host arrival/departure?

- **Efficient forwarding**
    - How quickly can one locate items in routing table?

- **Host must be able to recognize packet is for them**

Recap: L2 addressing does not enable scalable routing

- **Scalable routing**
    - How much state to forward packets?
        - One entry per host per switch
    - How much state updated for each arrival/departure?
        - One entry per host per switch

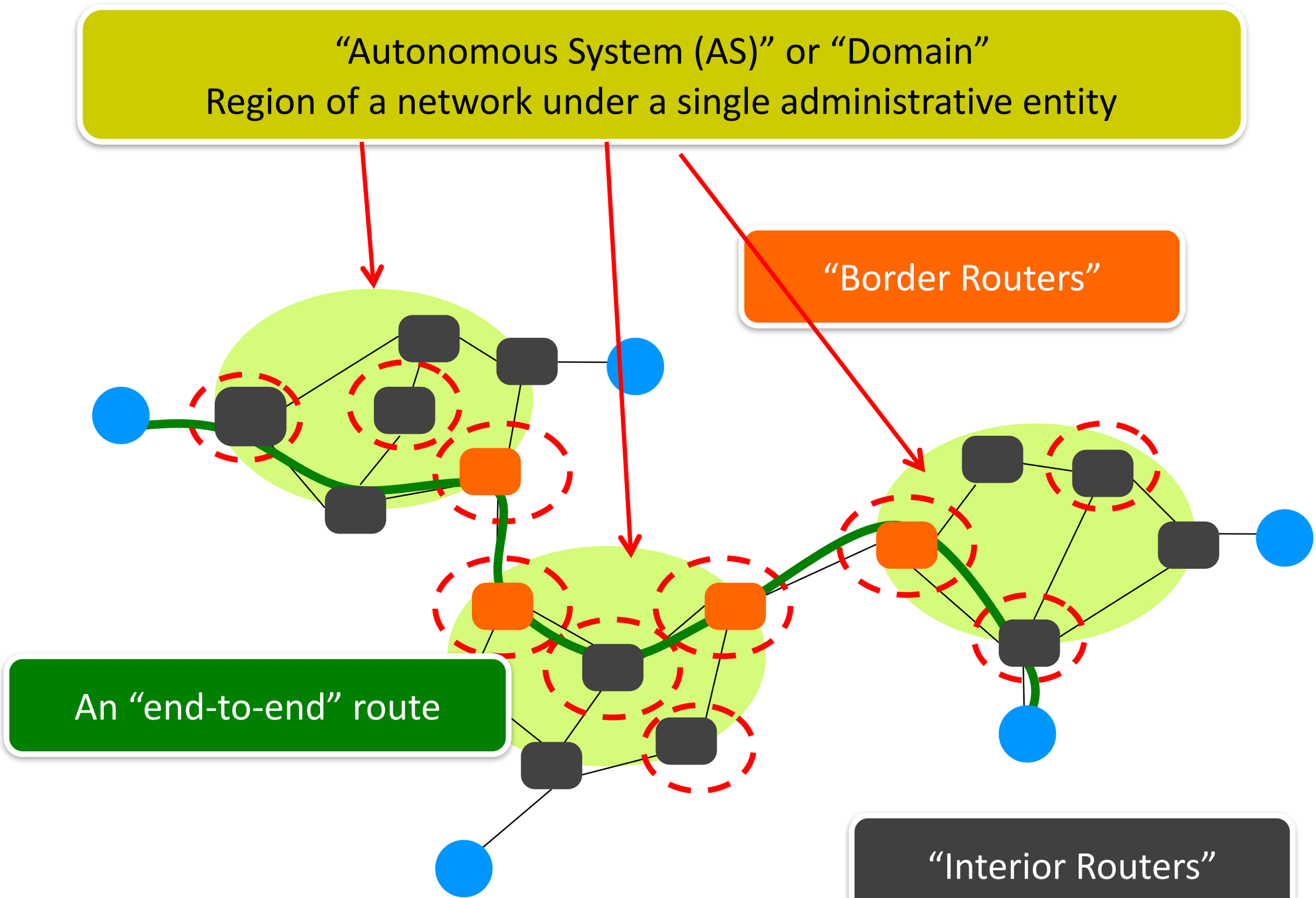
- **Efficient forwarding**
    - Exact match lookup on MAC addresses (exact match is easy!)

- **Host must be able to recognize the packet is for them**
    - MAC address does this perfectly

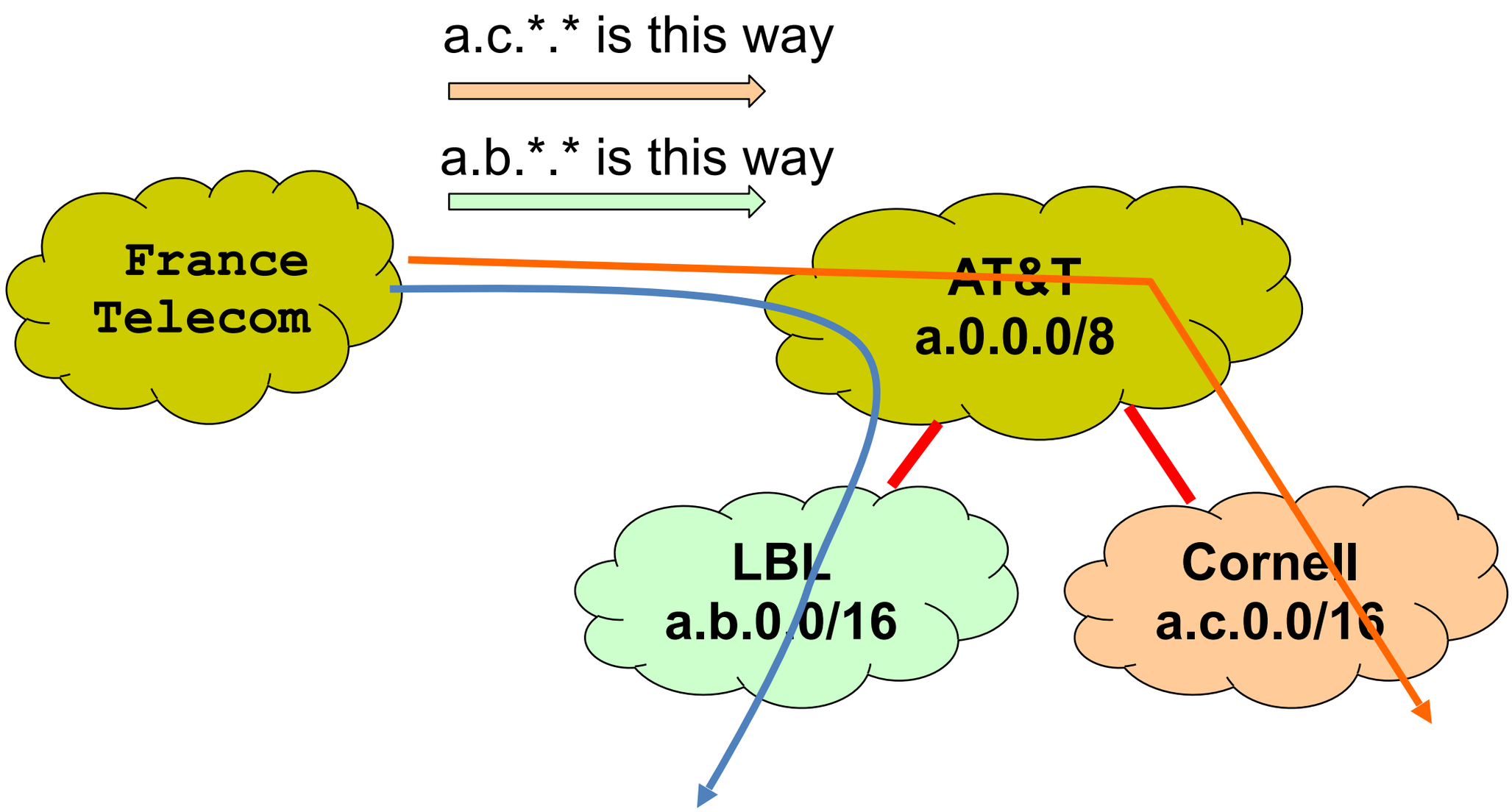
Recap: Today's Internet Addressing: CIDR

- **Classless Inter-domain Routing**
- Idea: Flexible division between network and host addresses
- Prefix is **network address**
- Suffix is **host address**
- **Example:**
    - **128.84.139.5/23 is a 23 bit prefix with:**
    - First 23 bits for network address
    - Next 9 bits for host addresses: maximum 2^9 hosts
- **Terminology: "Slash 23"**

Recap: What does a computer network look like?

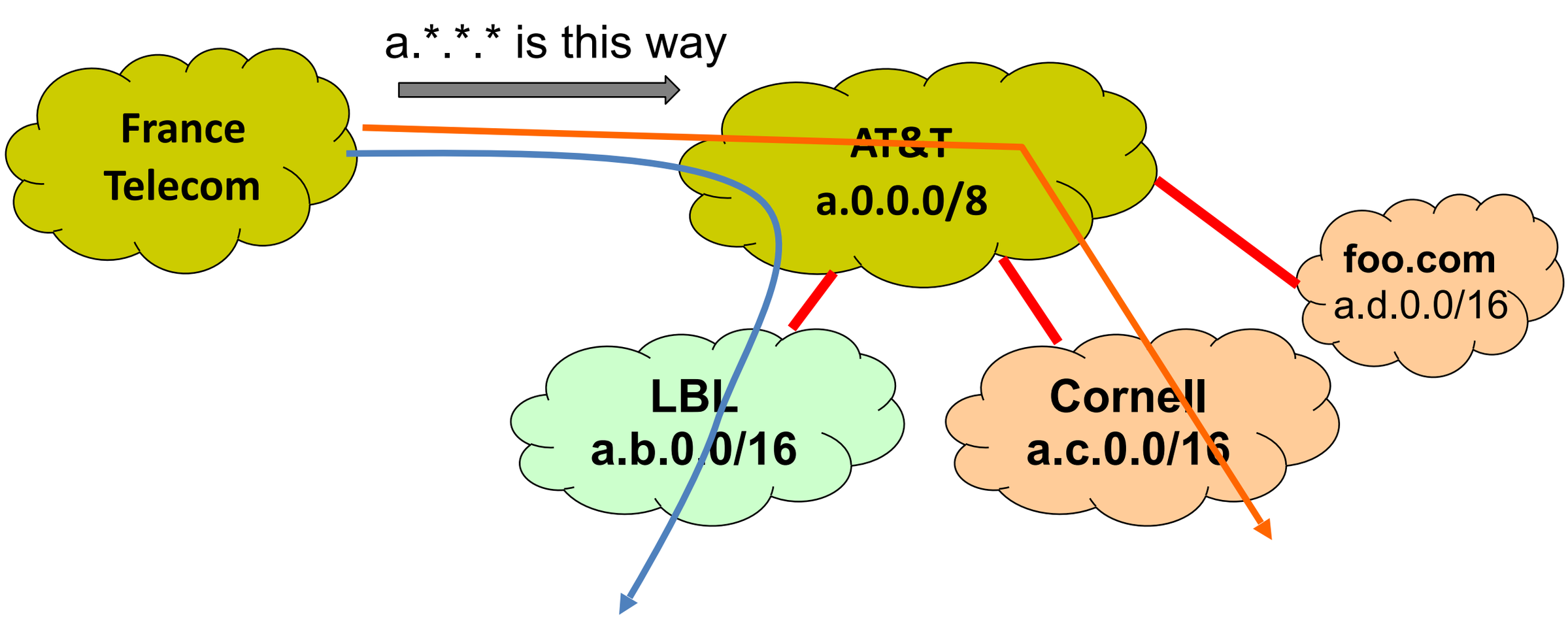


Recap: IP addressing -> Scalable Routing?



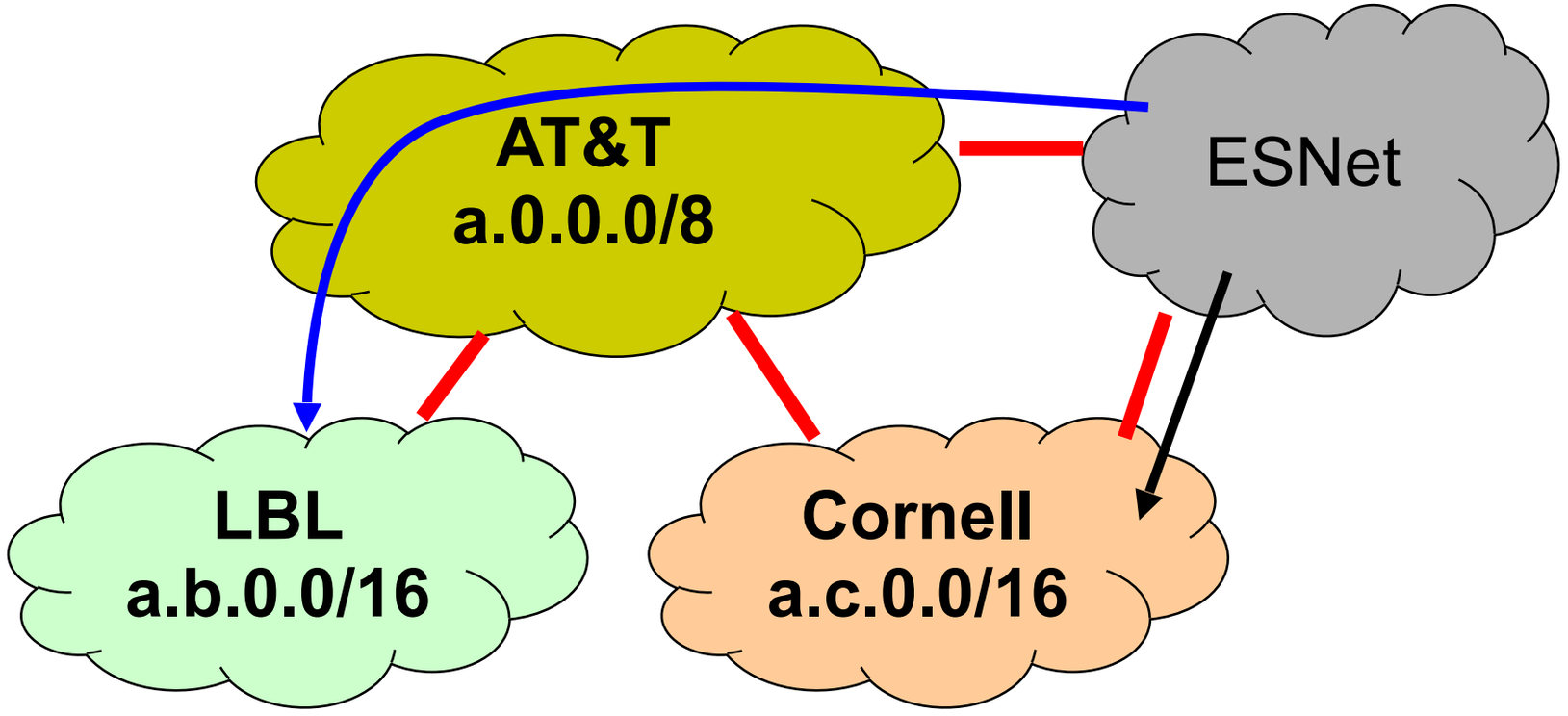
Recap: IP addressing -> Scalable Routing?

Can add new hosts/networks without updating the routing entries at France Telecom



Recap: IP addressing -> Scalable Routing?

ESNet must maintain routing entries for both a.*.*.* and a.c.*.*



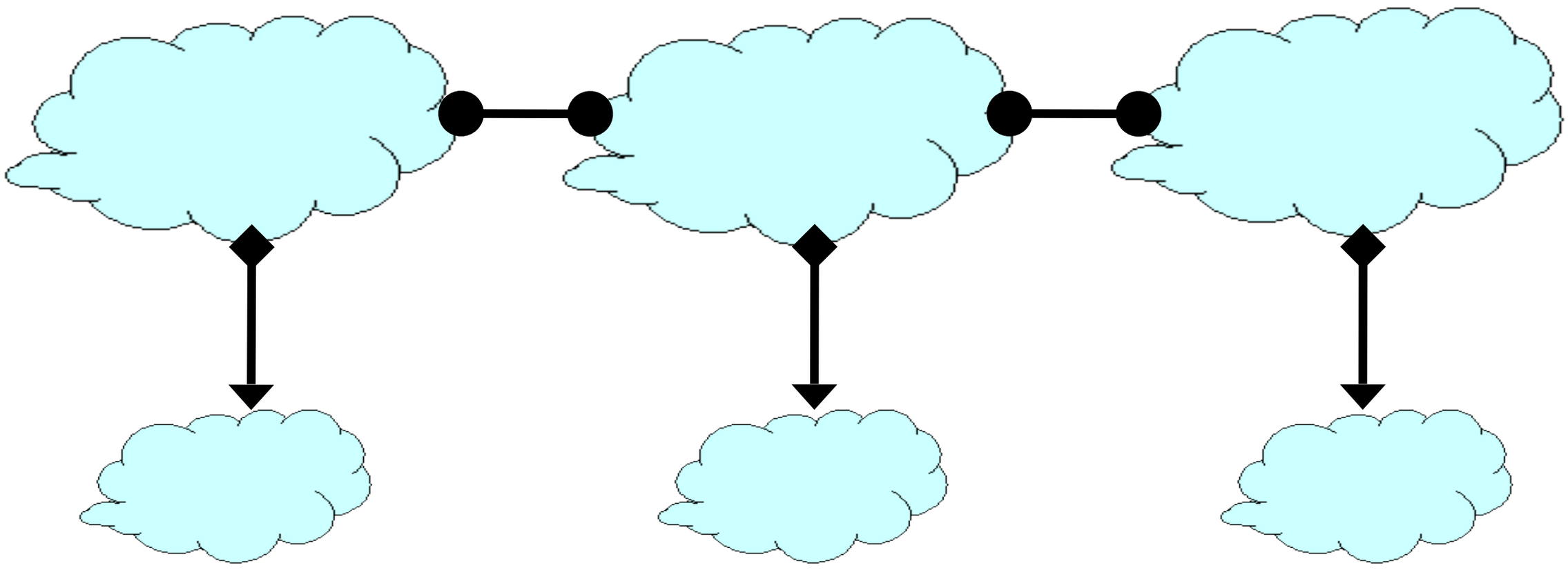
Recap: Administrative Structure Shapes Inter-domain Routing

- ASes want freedom to pick routes based on policy
  - *"My traffic can't be carried over my competitor's network!"*
  - *"I don't want to carry A's traffic through my network!"*
  - Cannot be expressed as Internet-wide "least cost"

- ASes want autonomy
  - Want to choose their own internal routing protocol
  - Want to choose their own policy

- ASes want privacy
  - Choice of network topology, routing policies, etc.

Recap: Business Relationships



Relations between ASes

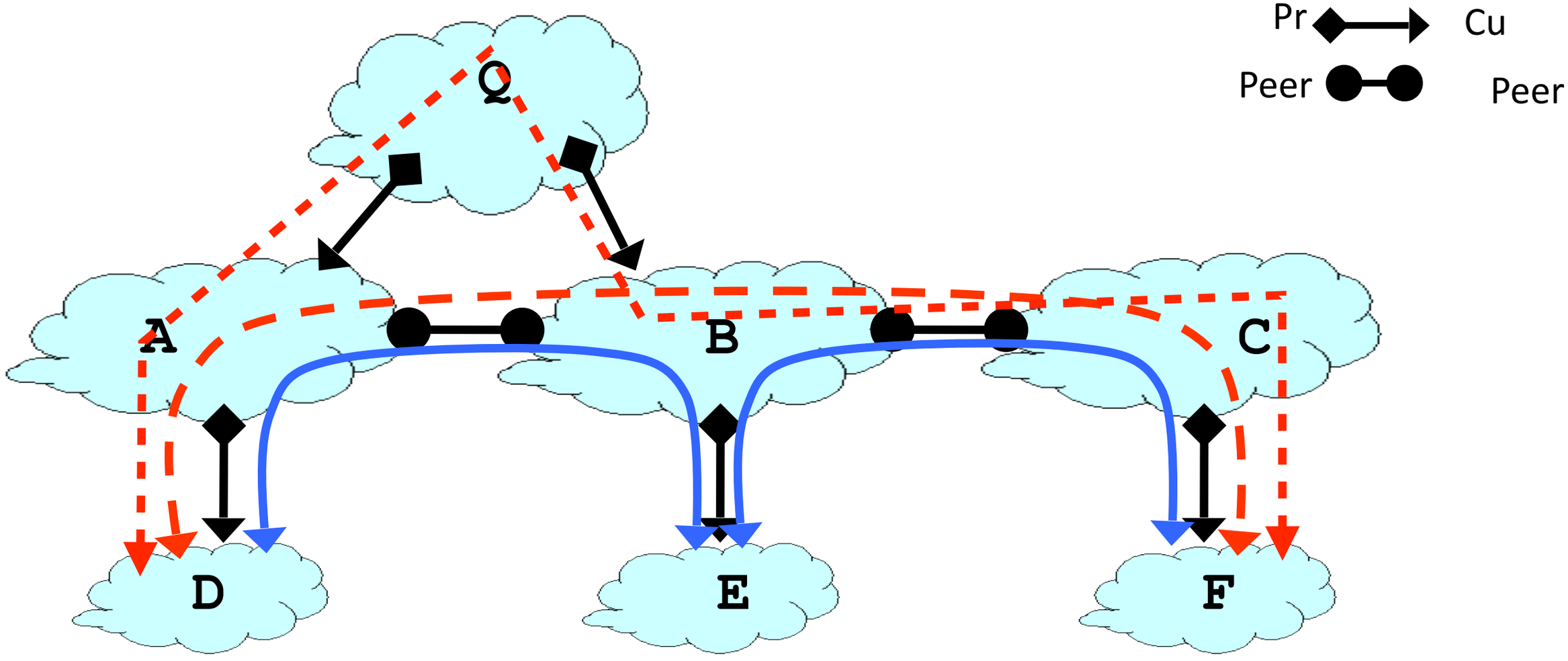
provider ◆——▶ **customer**

peer ●—— **peer**

Business Implications

- **Customers pay provider**
- **Peers don’t pay each other**

Recap: Inter-domain Routing Follows the Money



- ASes provide “transit” between their customers
- Peers do not provide transit between other peers

Recap: BGP Inspired by Distance Vector

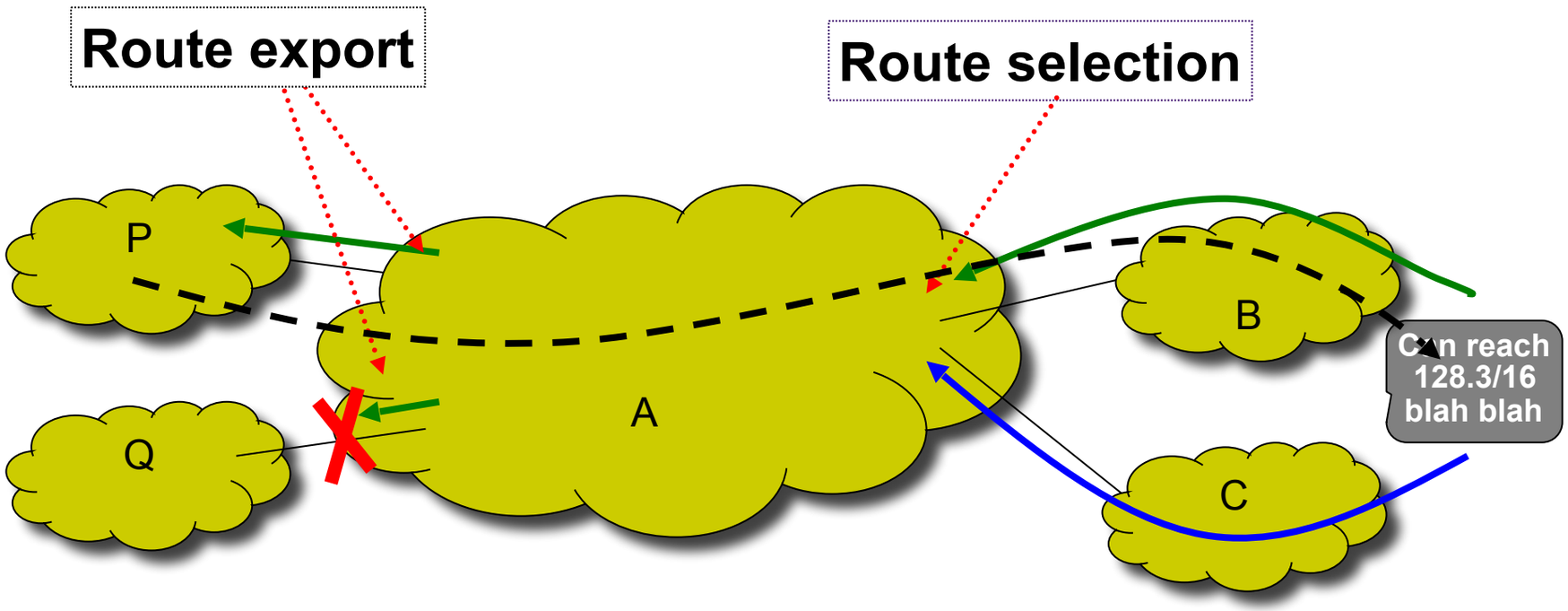
- Per-destination route advertisements
- No global sharing of network topology
- Iterative and distributed convergence on paths
- But, four key differences
  - BGP does not pick shortest paths
  - Path-vector rather than distance vector
    - Each announcement contains the path for each destination
  - Selective route advertisement
    - If I select a path, I don't *have to* advertise it to others
  - Route aggregation
    - Rather than storing a.b.*.*/16 and a.c.*.*/16, store a.*.*.*/8

Recap: BGP Outline

- BGP Policy
  - Typical policies and implementation
- BGP protocol details
- Issues with BGP

Recap: Policy:

Imposed in how routes are selected and exported



- **Selection**: Which path to use
  - Controls whether / how traffic leaves the network
- **Export**: Which path to advertise
  - Controls whether / how traffic enters the network

Recap: Typical Export Policy

Destination prefix advertised by…	Export route to…
Customer	Everyone (providers, peers, other customers)
Peer	Customers
Provider	Customers

Known as the “Gao-Rexford” rules
Capture common (but not required!) practice

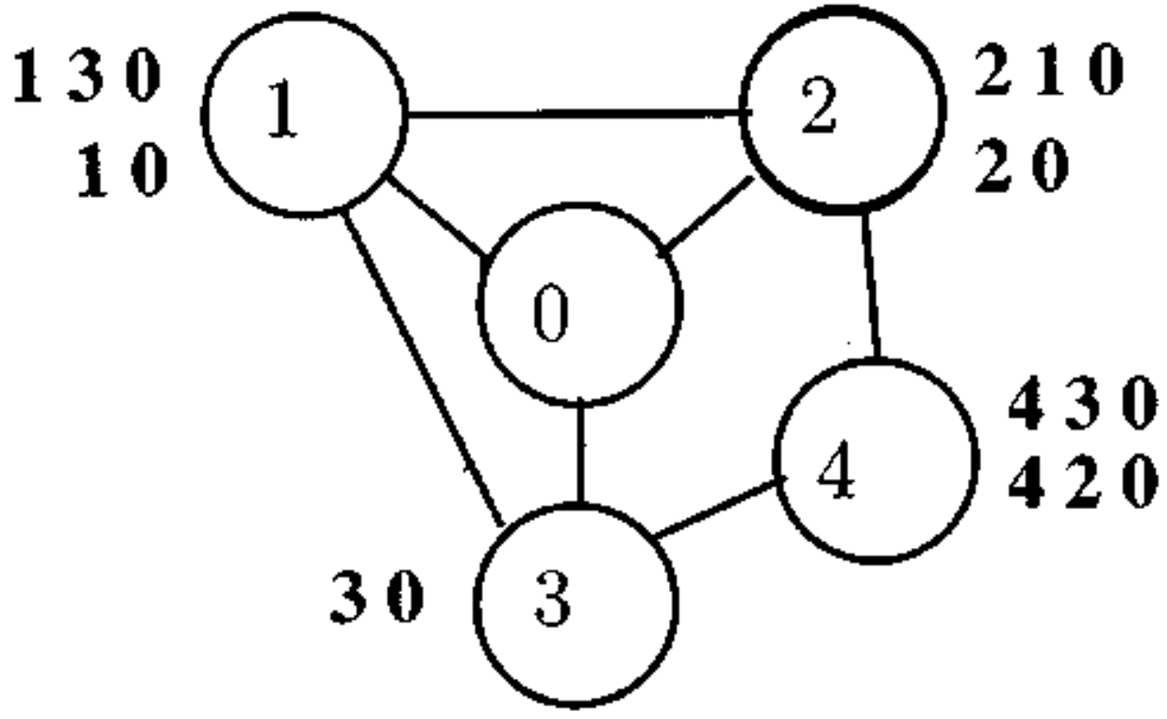
Recap: Typical Selection Policy

- In decreasing order of priority:
  1. Make or save money (send to customer > peer > provider)
  2. Maximize performance (smallest AS path length)
  3. Minimize use of my network bandwidth (“hot potato”)
  4. …

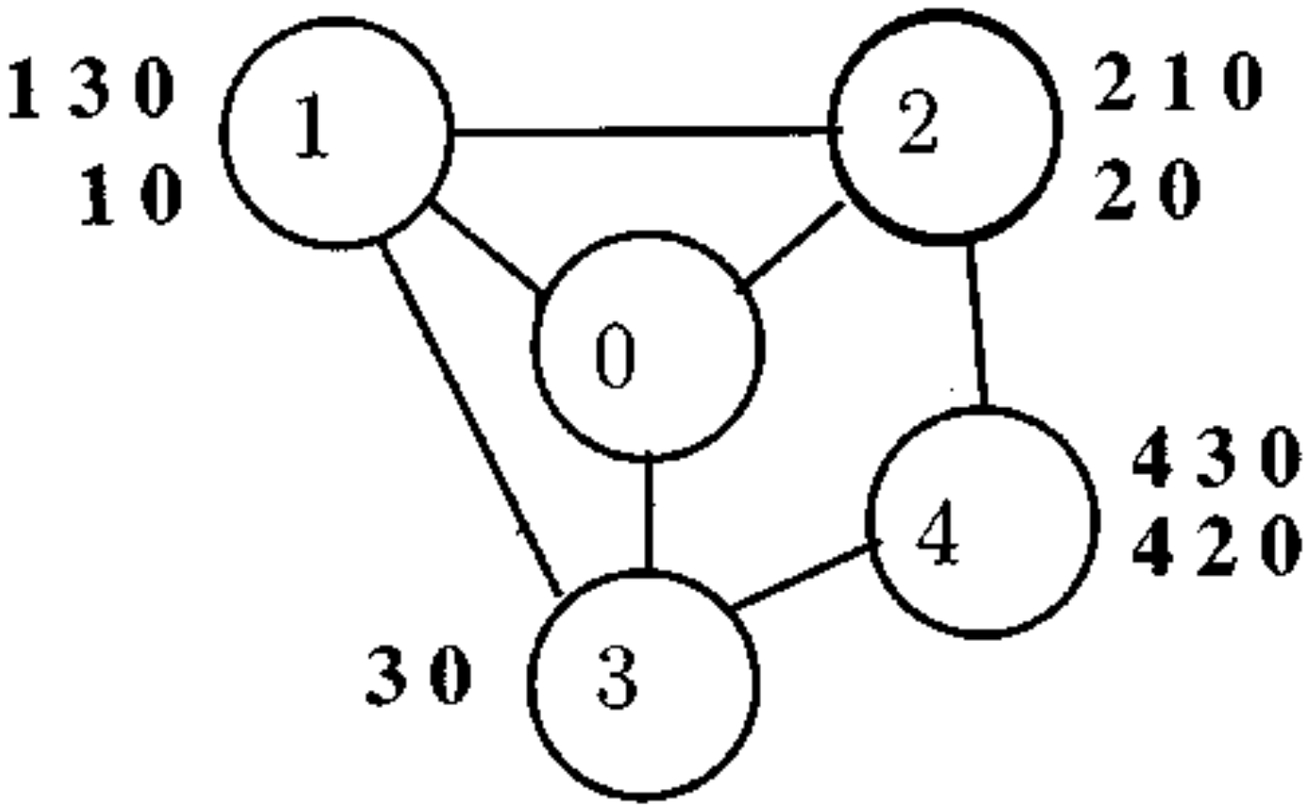
Recap: BGP implicit decision making

- Export policy
  - Gives a set of “possible” paths for each AS

- Selection policy
  - Gives a ranking over all the possible paths



BGP Example (All good)



GOOD GADGET

	1	2	3	4
R1	10	20	**30**	-
R2	**130**	20	30	**430**

Assumes a particular advertisement ordering

May look different with different ordering

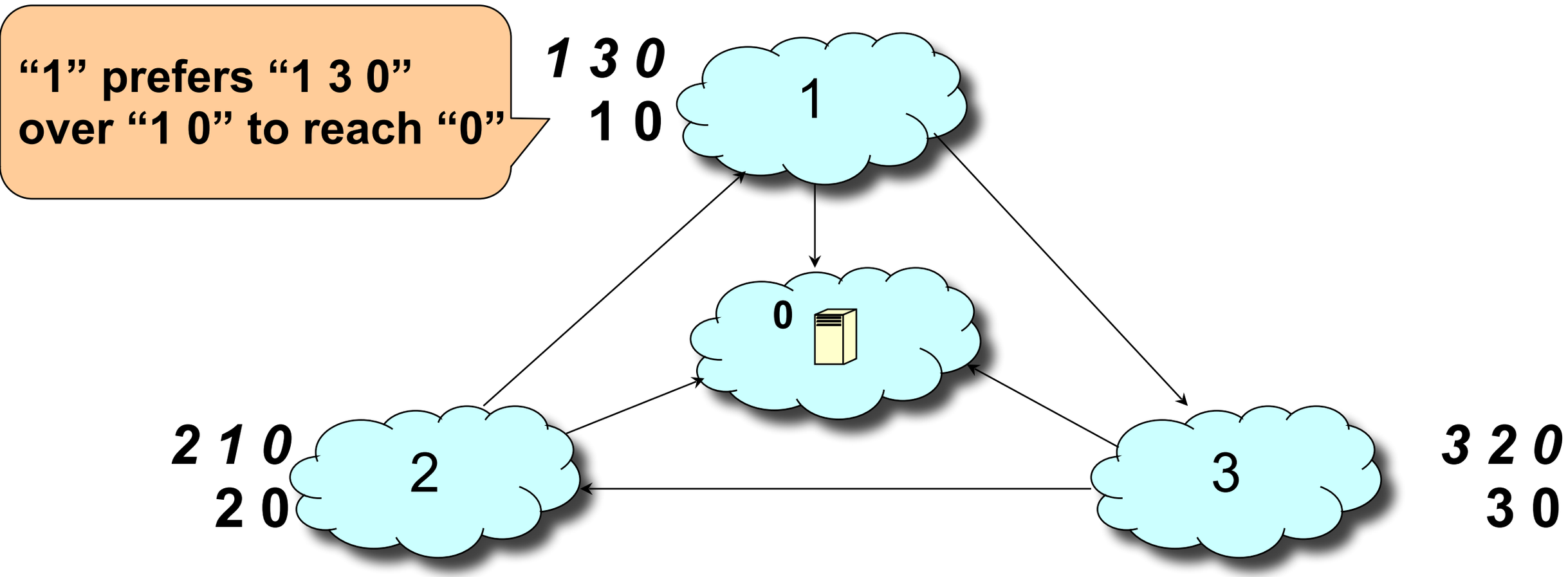
In real-world: ordering depends on latency, processing delay, etc.

BGP: Issues

- Reachability
- Security
- Convergence
- Performance
- Anomalies

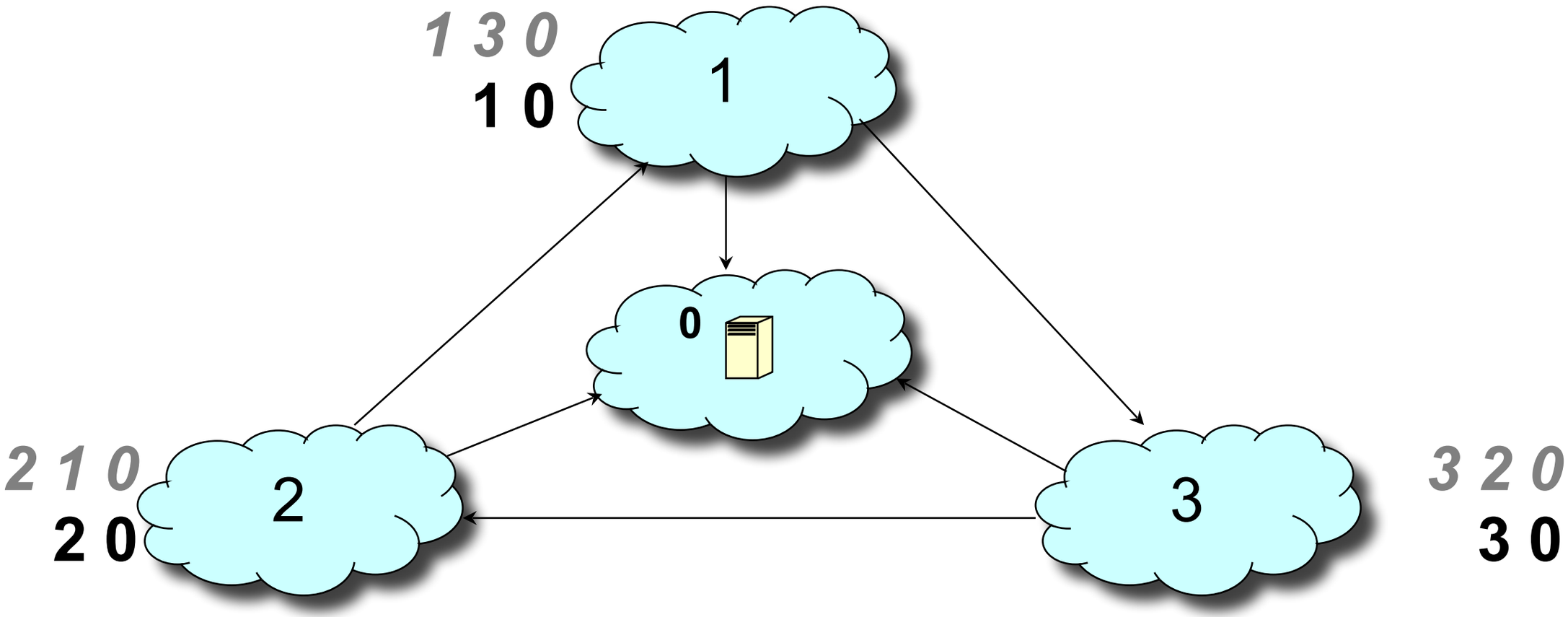
Example of Policy Oscillation

"1" prefers "1 3 0"
over "1 0" to reach "0"



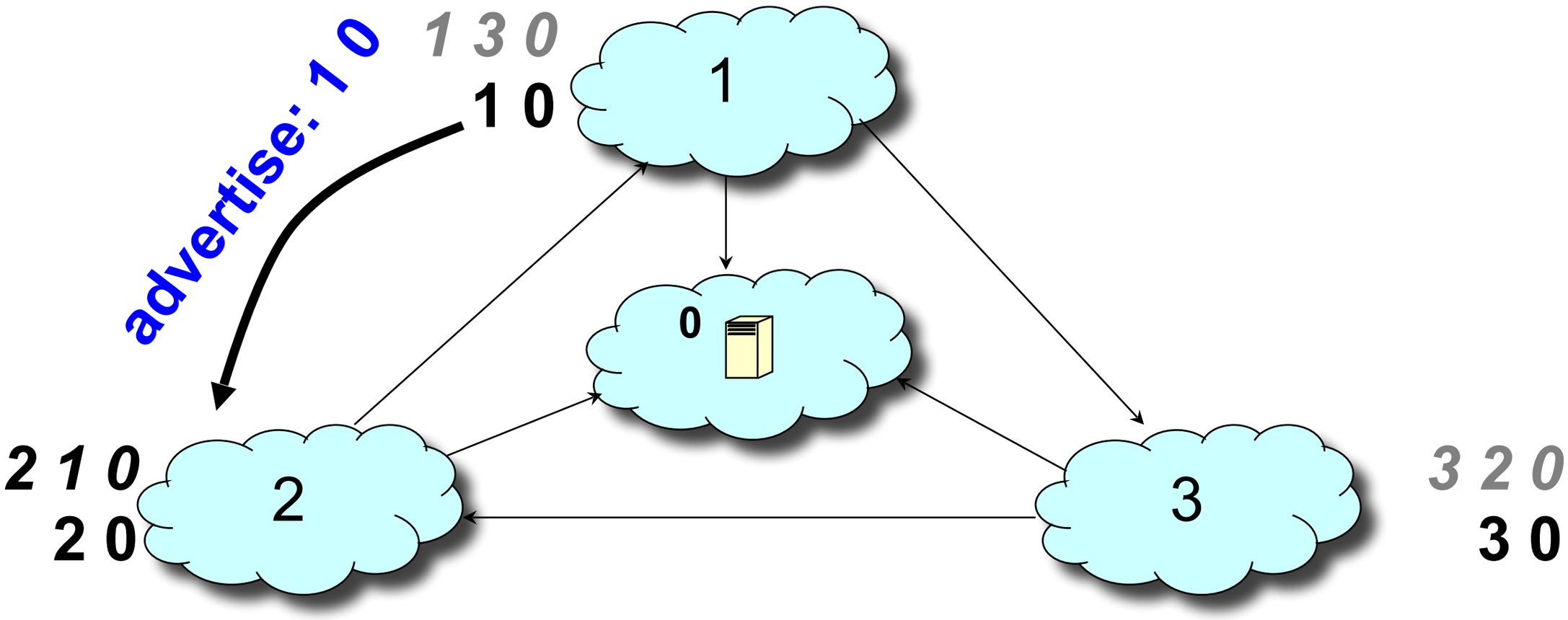
Step-by-step Policy Oscillation

Initially: nodes 1, 2, 3 know only shortest path to 0

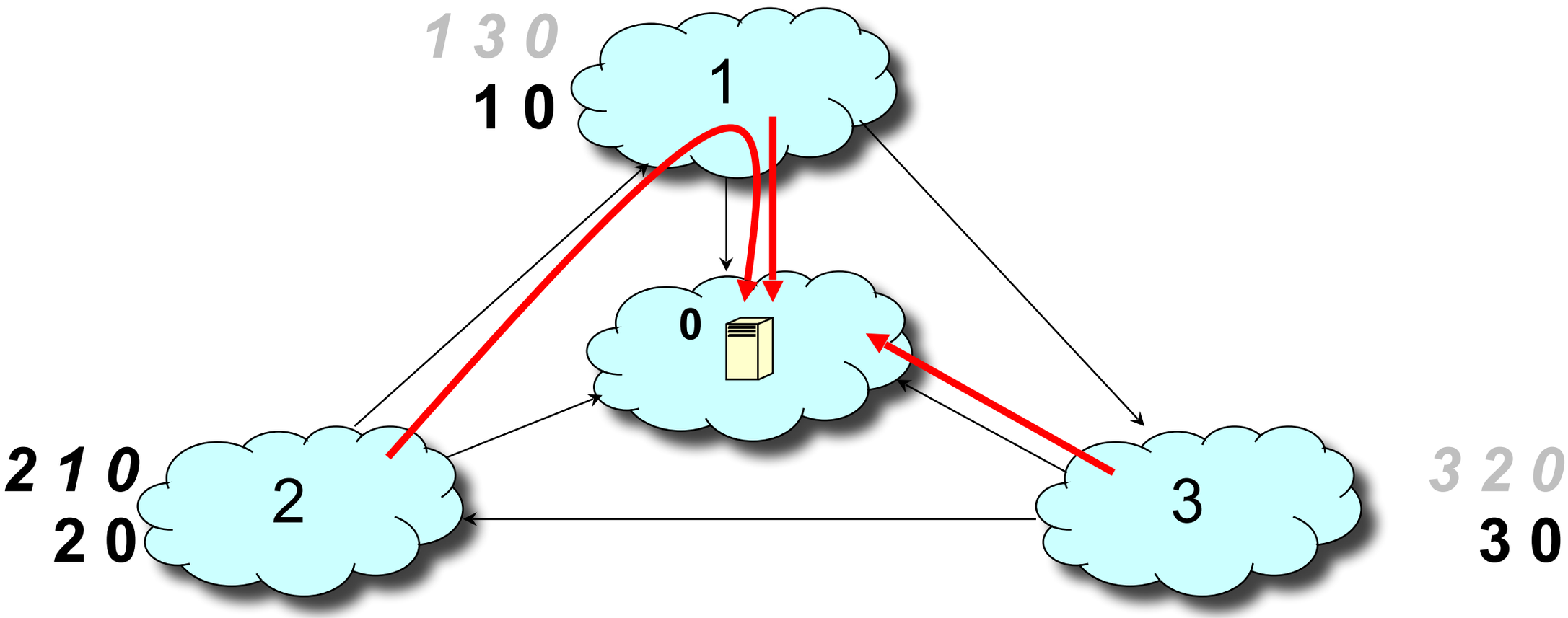


Step-by-step Policy Oscillation

1 **advertises** its path 1 0 to 2

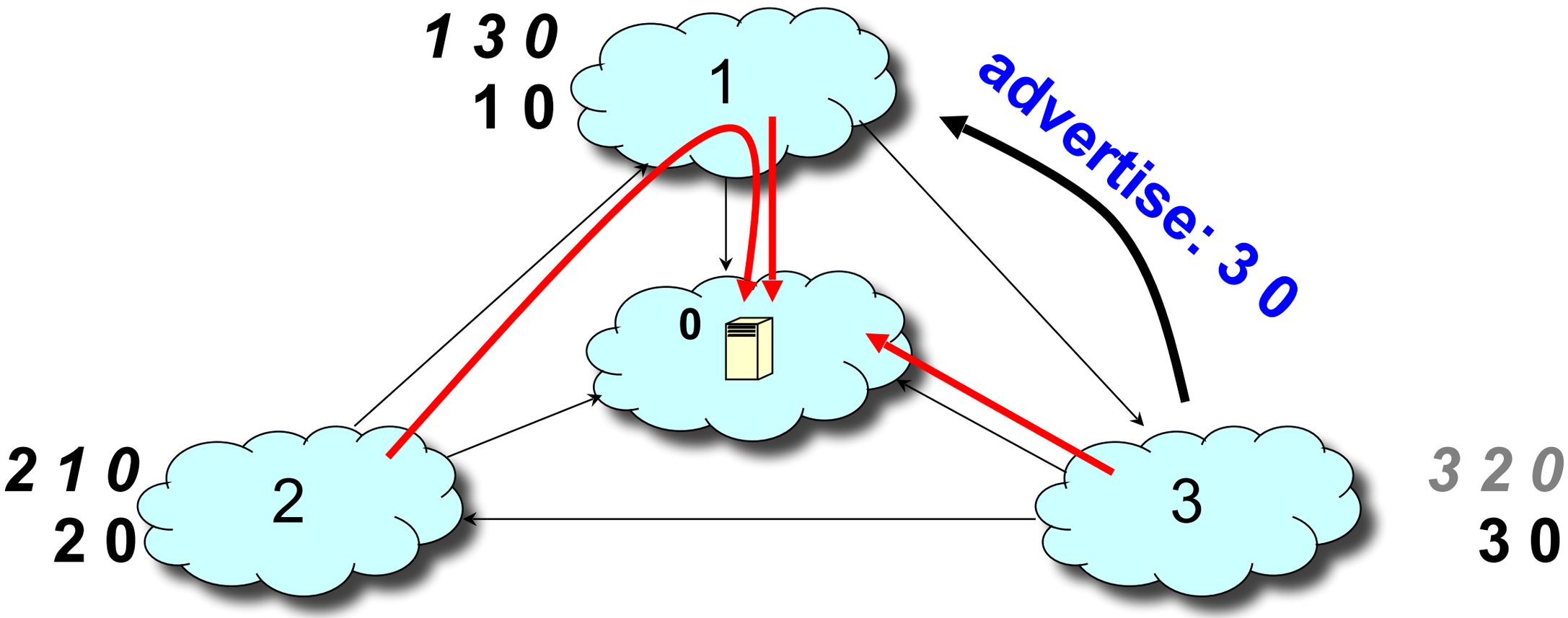


Step-by-step Policy Oscillation

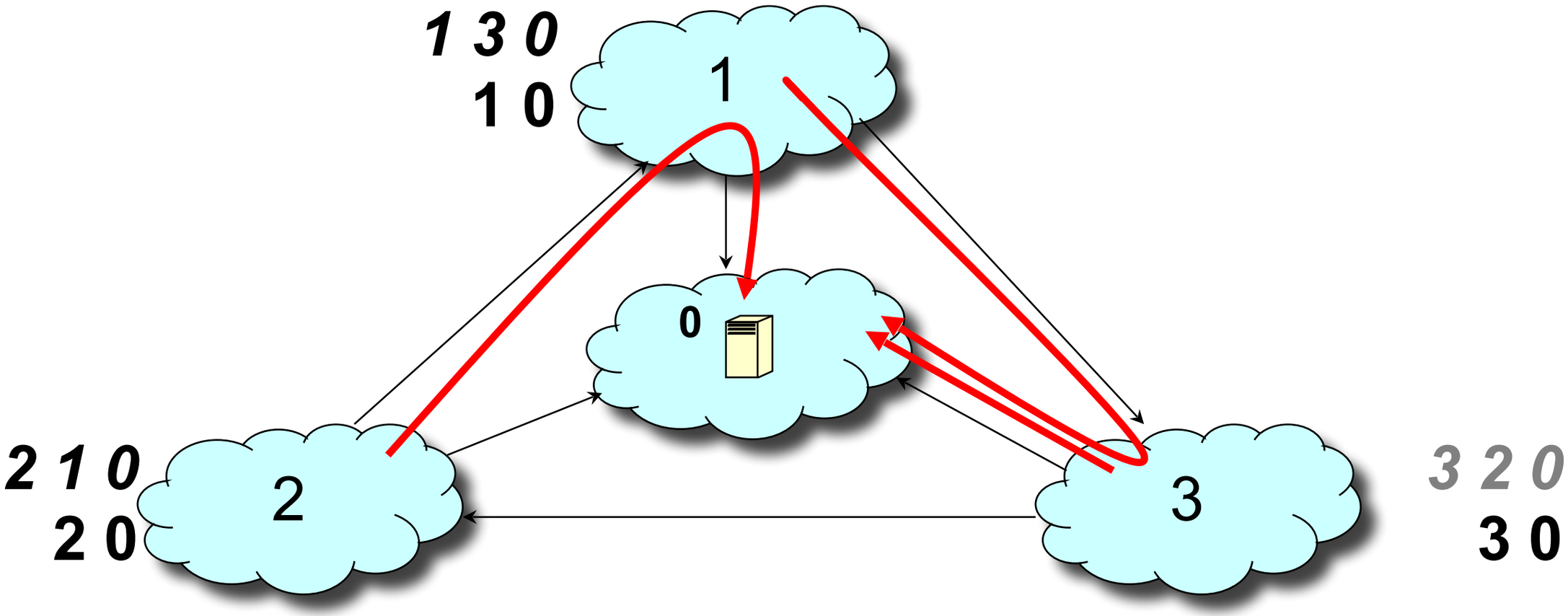


Step-by-step Policy Oscillation

3 **advertises** its path 3 0 to 1

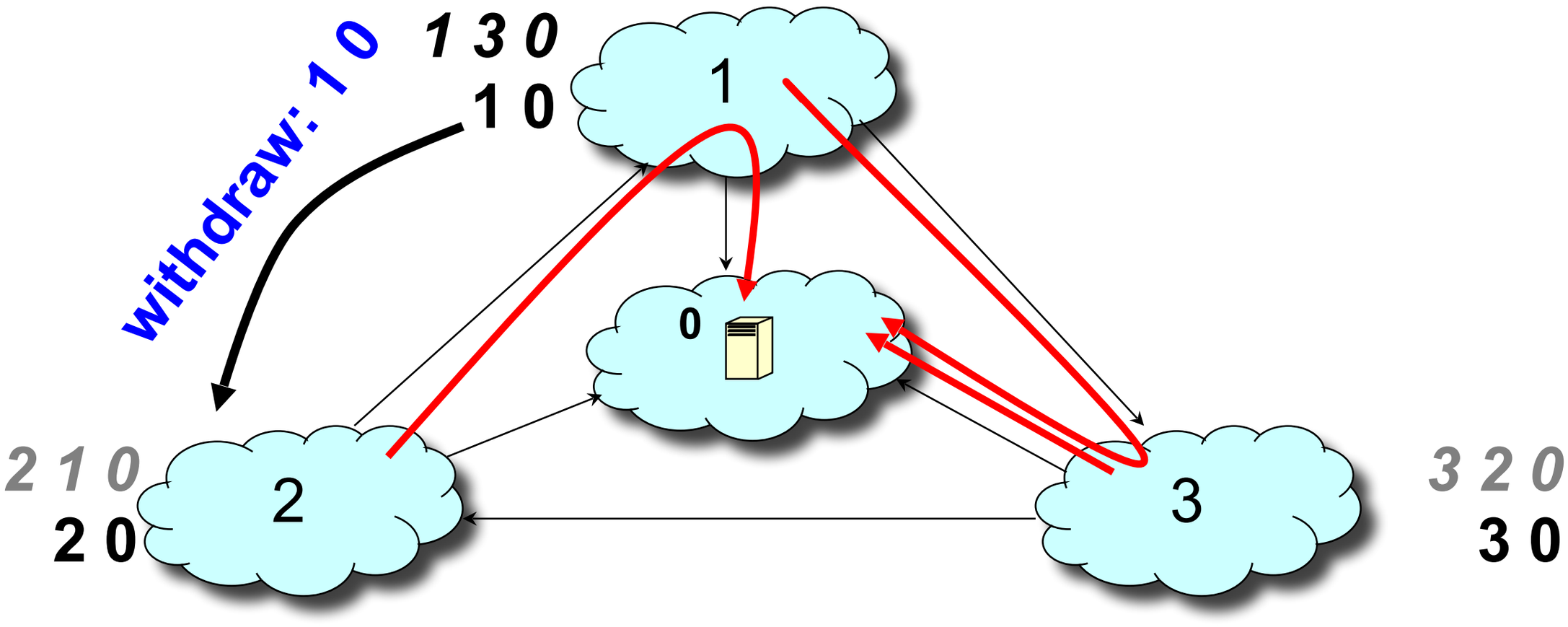


Step-by-step Policy Oscillation

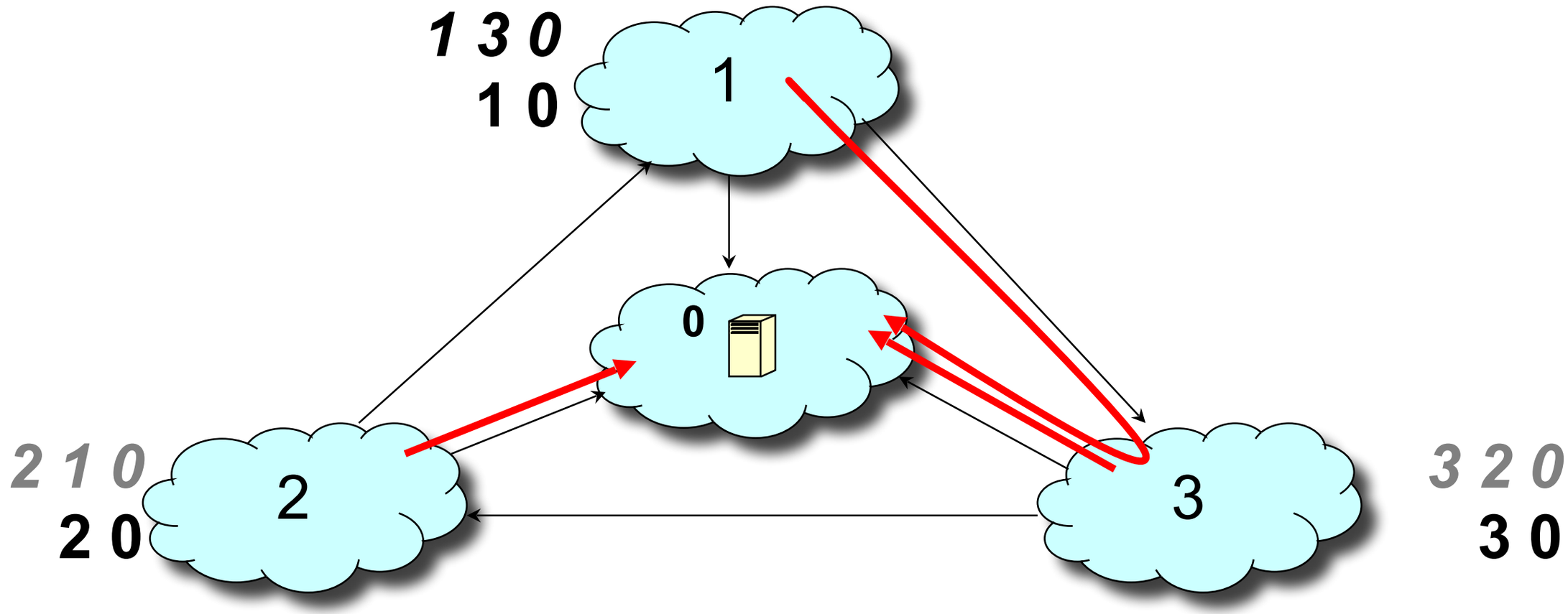


Step-by-step Policy Oscillation

1 **withdraws** its path 1 0 from 2

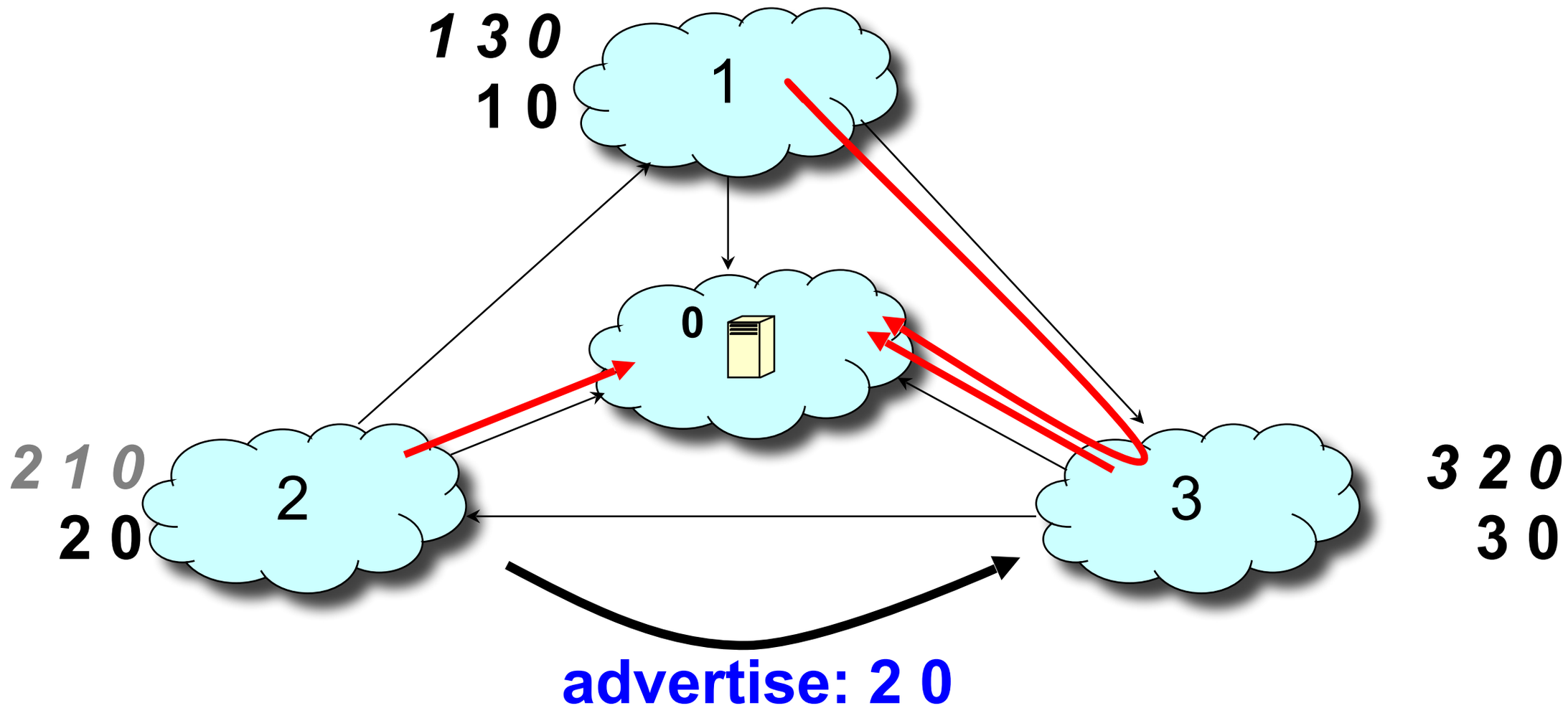


Step-by-step Policy Oscillation

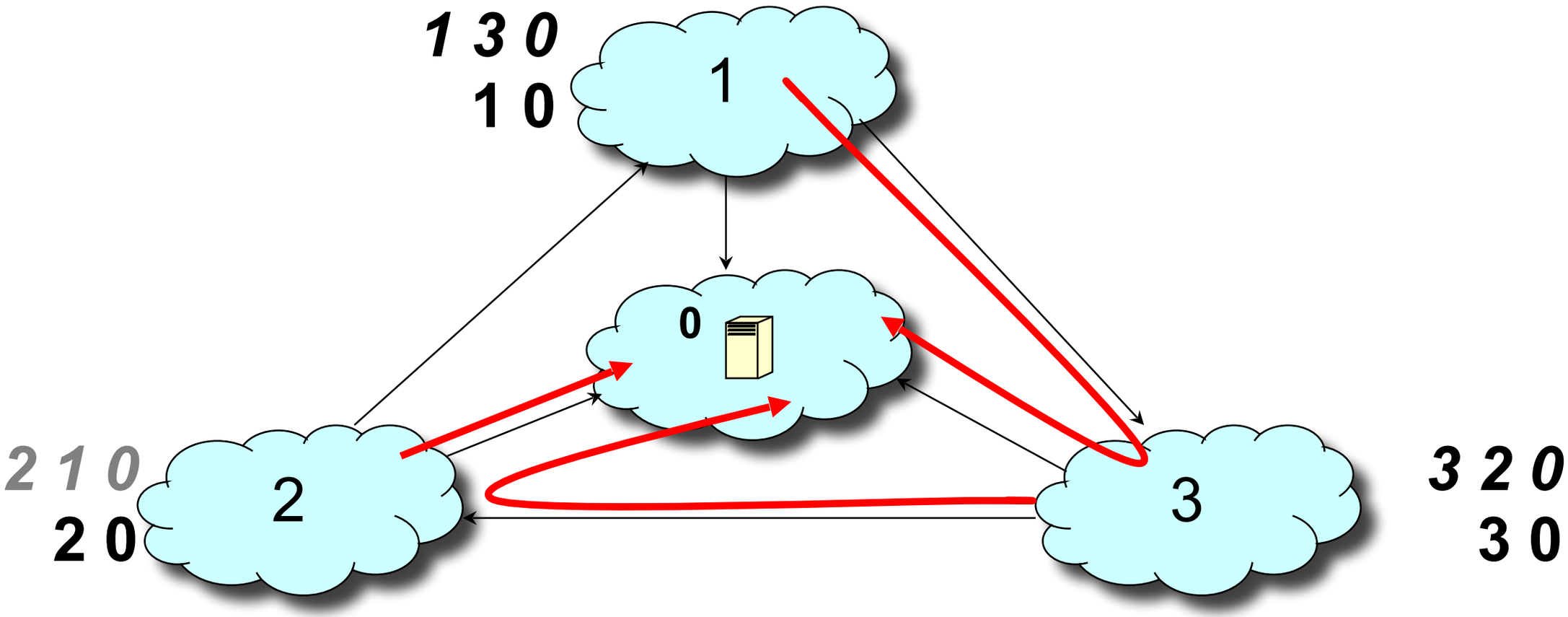


Step-by-step Policy Oscillation

2 **advertises** its path 2 0 to 3

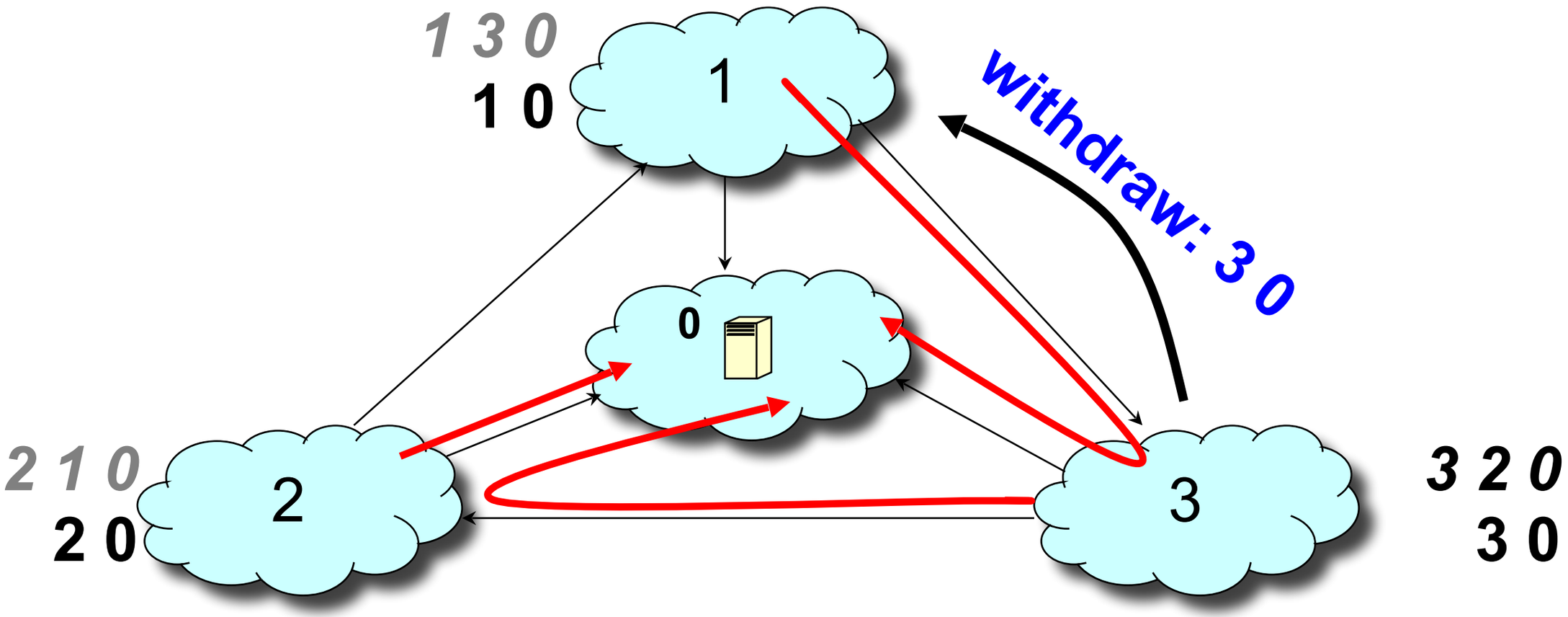


Step-by-step Policy Oscillation

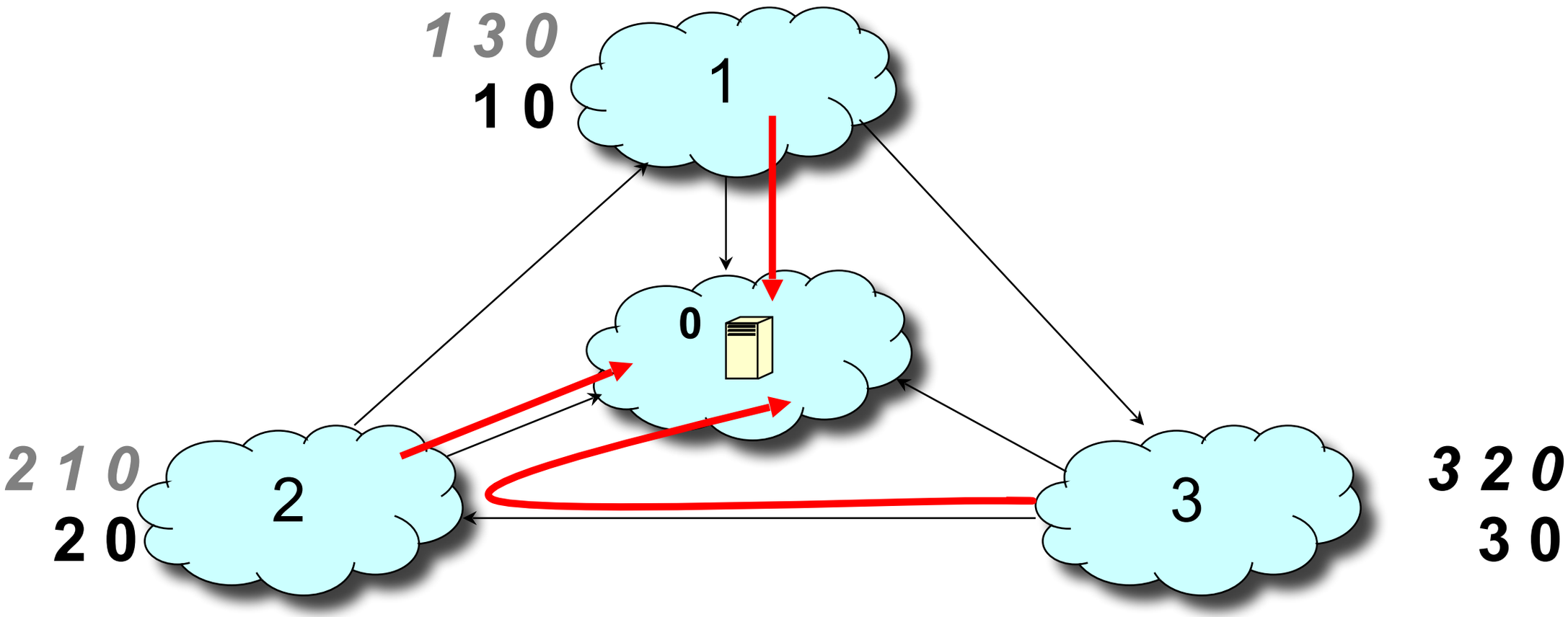


Step-by-step Policy Oscillation

3 **withdraws** its path 3 0 from 1

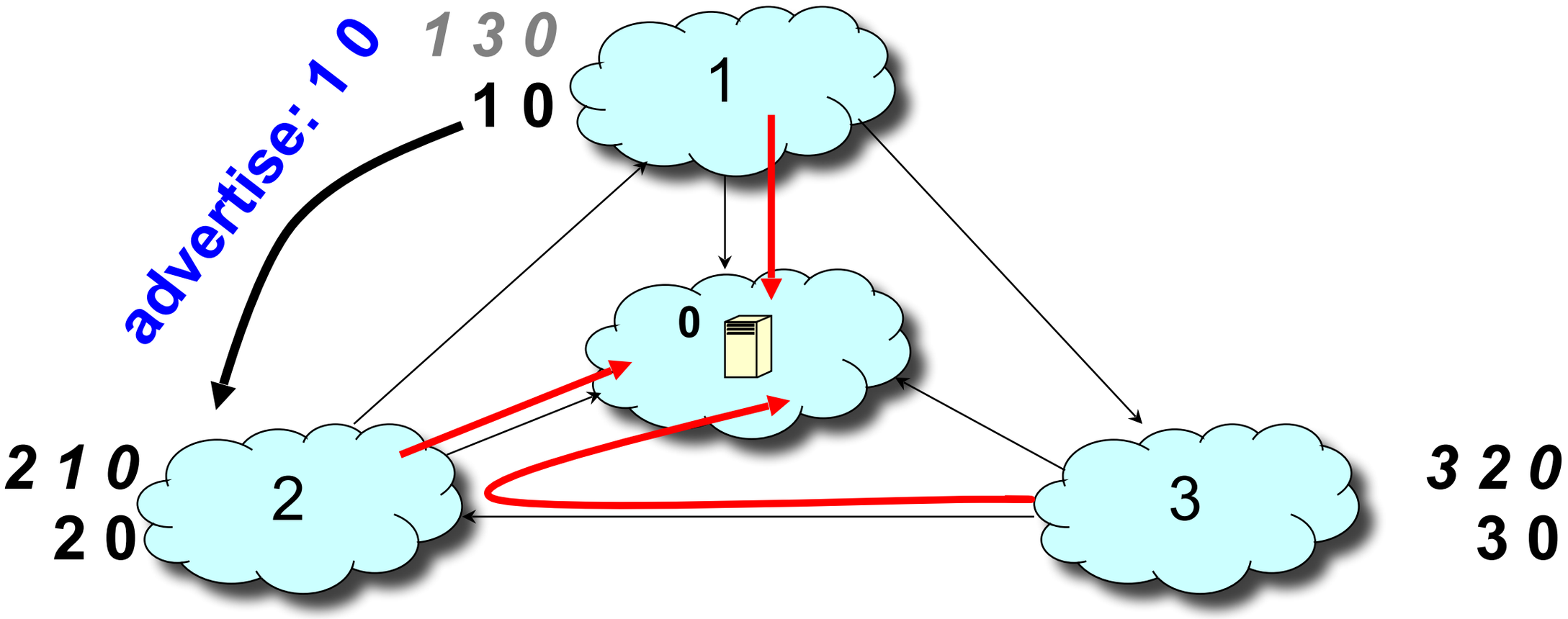


Step-by-step Policy Oscillation

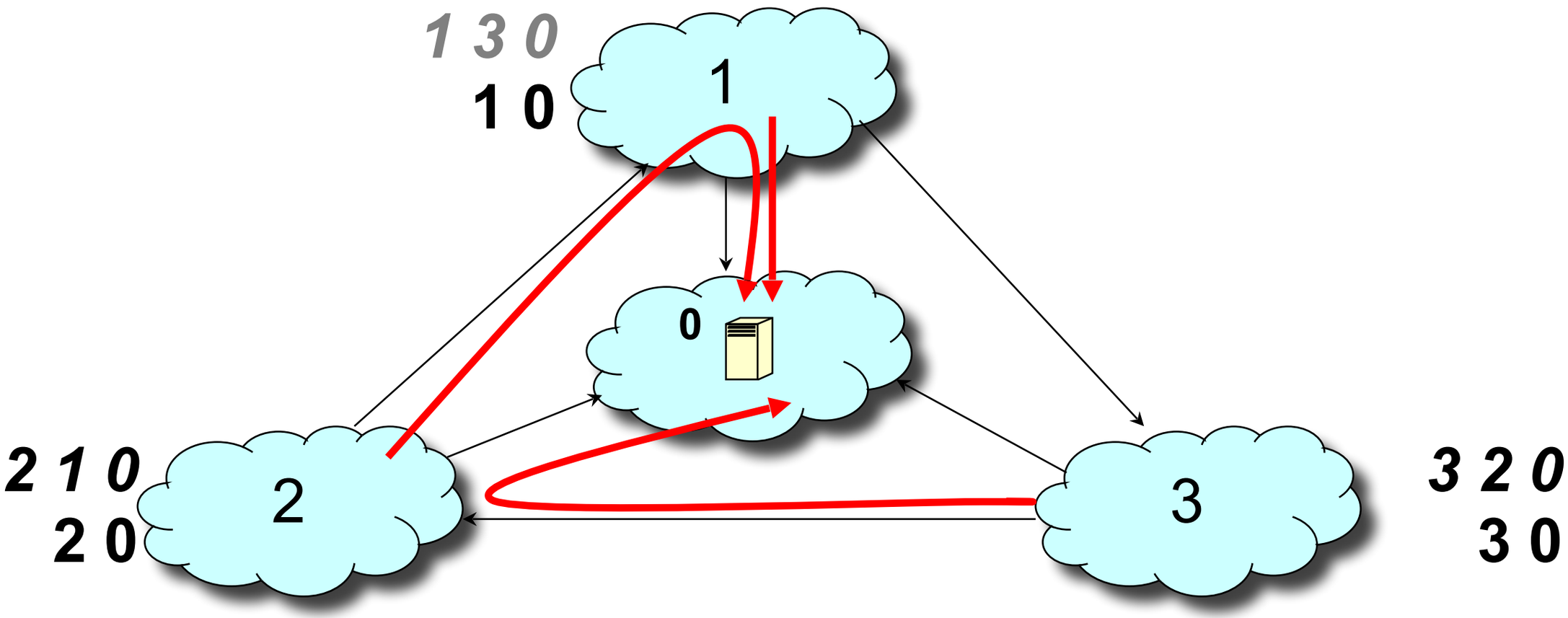


Step-by-step Policy Oscillation

1 **advertises** its path 1 0 to 2

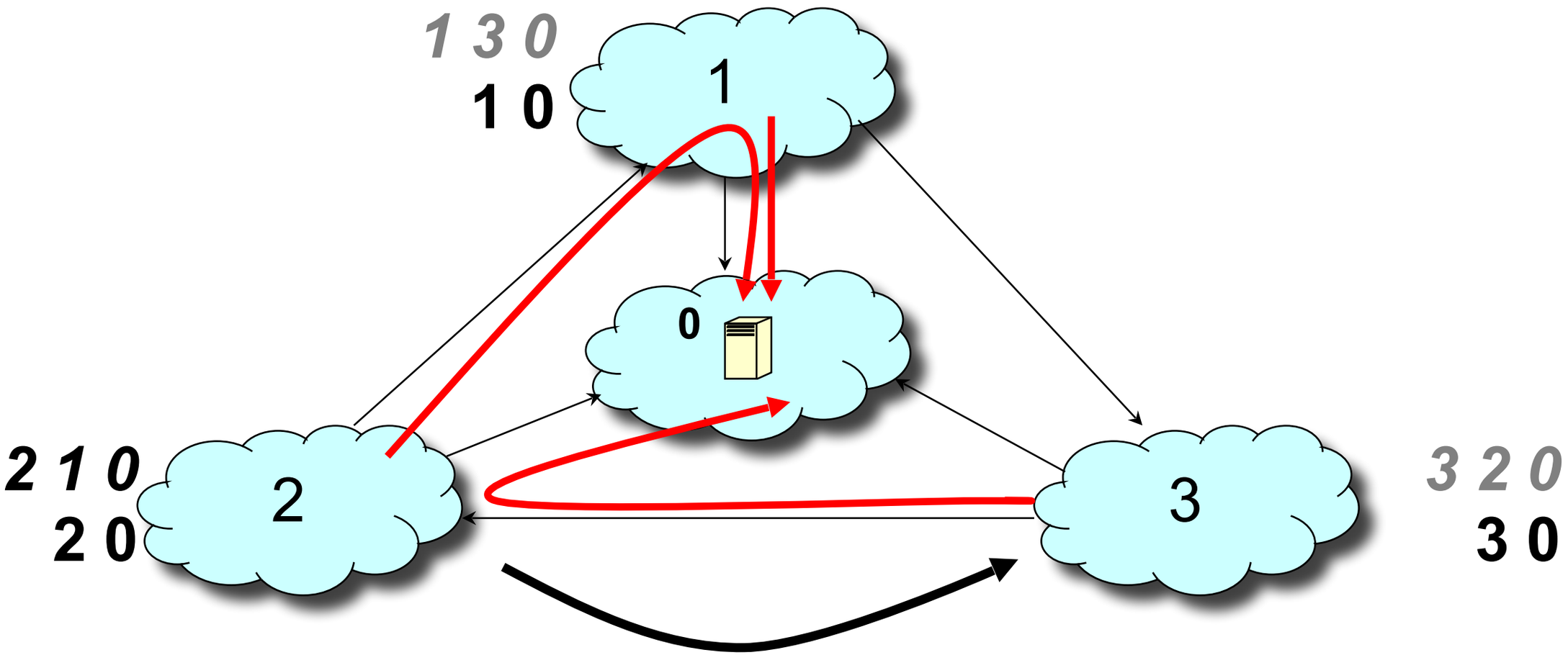


Step-by-step Policy Oscillation

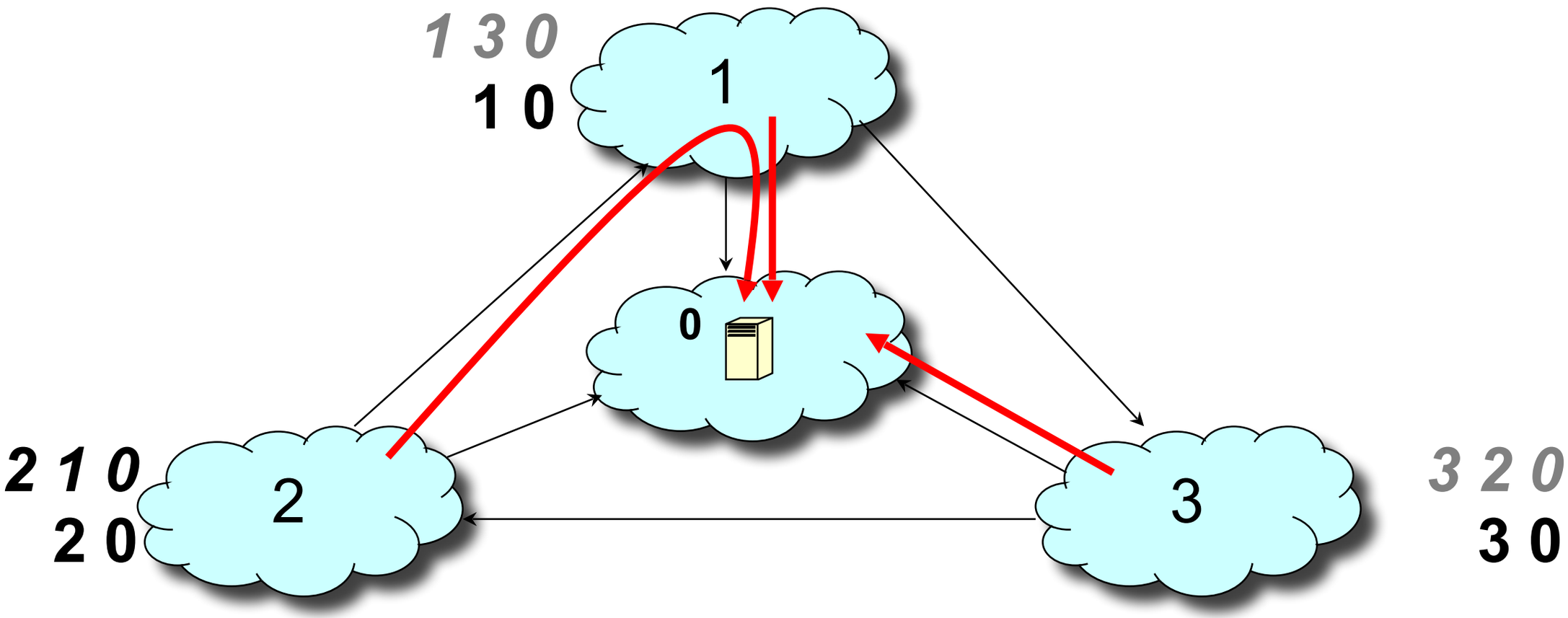


Step-by-step Policy Oscillation

2 **withdraws** its path 2 0 from 3



Step-by-step Policy Oscillation



We are back to where we started!

Network Layer

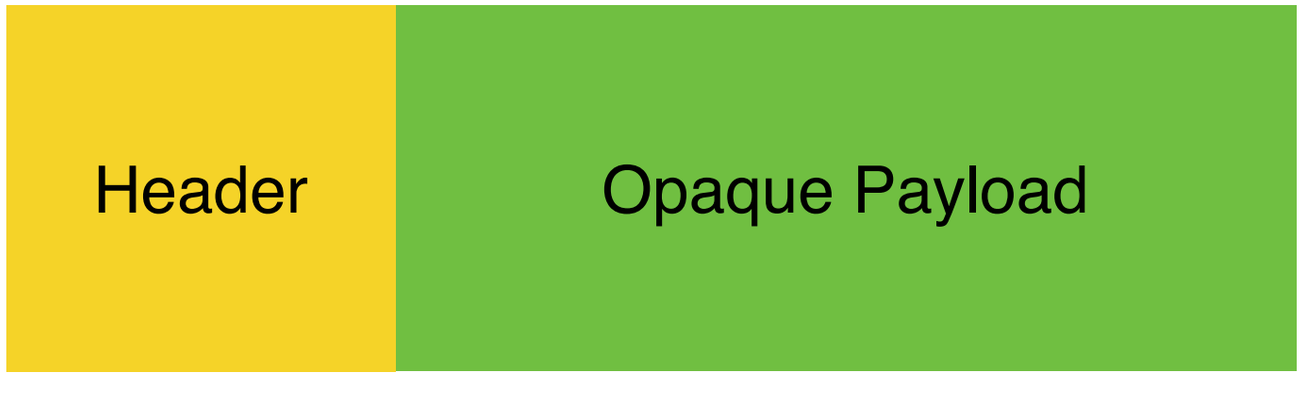
- THE functionality: **delivering the data**

- **THE protocol: Internet Protocol (IP)**

- **Achieves its functionality (delivering the data), using three ideas:**
  - **Addressing** (IP addressing)
  - **Routing** (using a variety of protocols)
  - **Packet header as an interface** (Encapsulating data into packets)

What is Designing IP?

- Syntax: format of packet
    - Nontrivial part: packet “header”
    - Rest is opaque payload (**why opaque?**)



- Semantics: meaning of header fields
    - Required processing

Packet Header as Interface

- Think of packet header as interface
    - Only way of passing information from packet to switch
- Designing interfaces:
    - What task are you trying to perform?
    - What information do you need to accomplish it?
- Header reflects information needed for basic tasks

What Tasks Do We Need to Do?

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with the packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Reading Packet Correctly

- Where does the header end?
- Where the the packet end?
- What protocol are we using?
    - Why is this so important?

Getting to the Destination

- Provide destination address
- Should this be location or identifier (name)?
    - And what's the difference?
- If a host moves should its address change?
    - If not, how can you build scalable Internet?
    - If so, then what good is an address for identification?

Getting Response Back to Source

- Source address
- Necessary for routers to respond to source
    - When would they need to respond back?
        - Failures!
    - Do they really need to respond back?
        - How would the source know if the packet has reached the destination?

Carry Data

- Payload!

Questions?

List of Tasks

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Telling Destination How to Process Packet

- Indicate which protocols should handle packet
- What layers should this protocol be in?
- What are some options for this today?
- How does the source know what to enter here?

Special Handling

- Type of service, priority, etc.
- Options: discuss later

Dealing With Problems

- Is packet caught in loop?
    - TTL
- Header corrupted:
    - Detect with Checksum
    - What about payload checksum?
- Packet too large?
    - Deal with fragmentation
    - Split packet apart
    - Keep track of how to put together

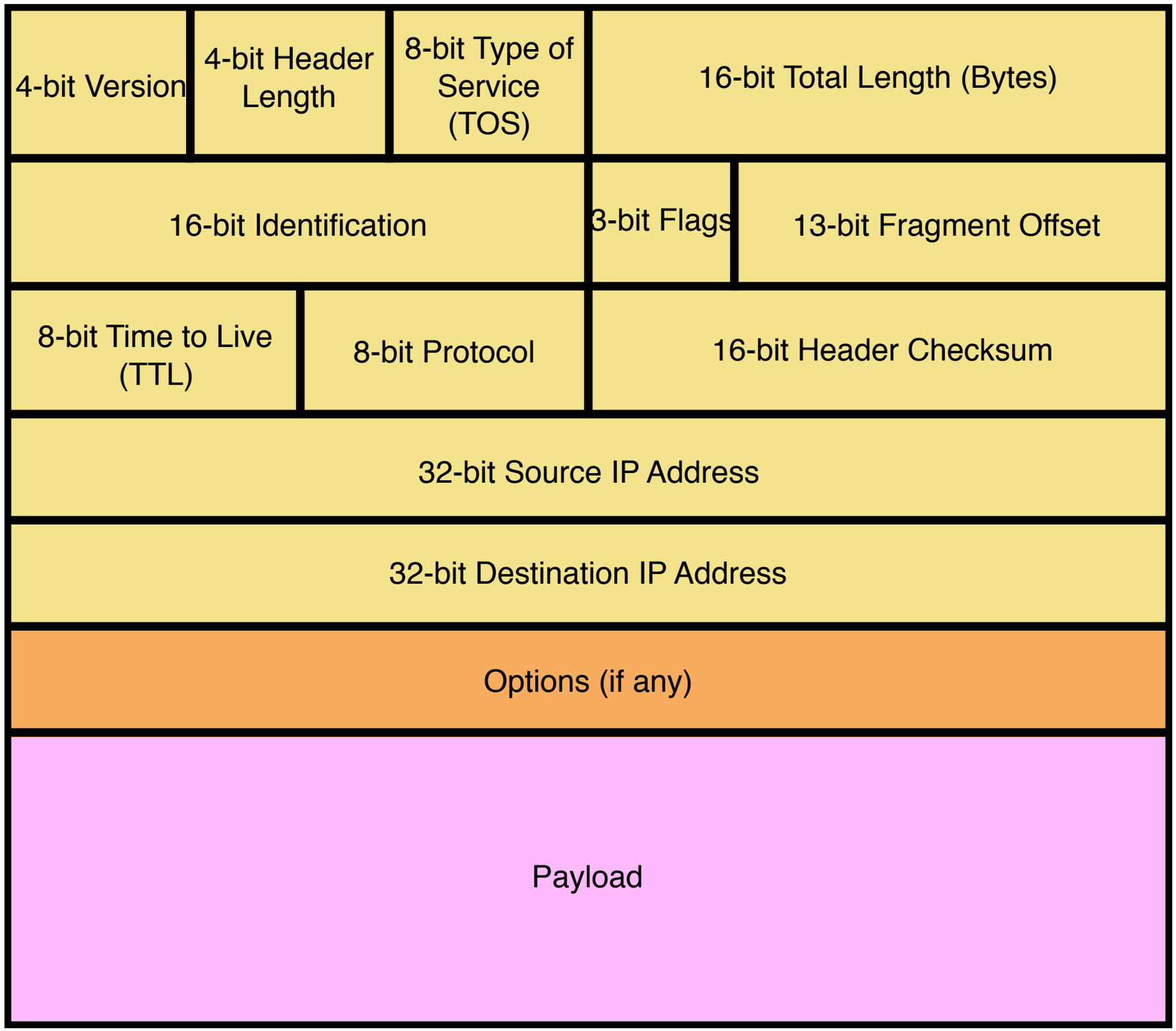
Are We Missing Anything?

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

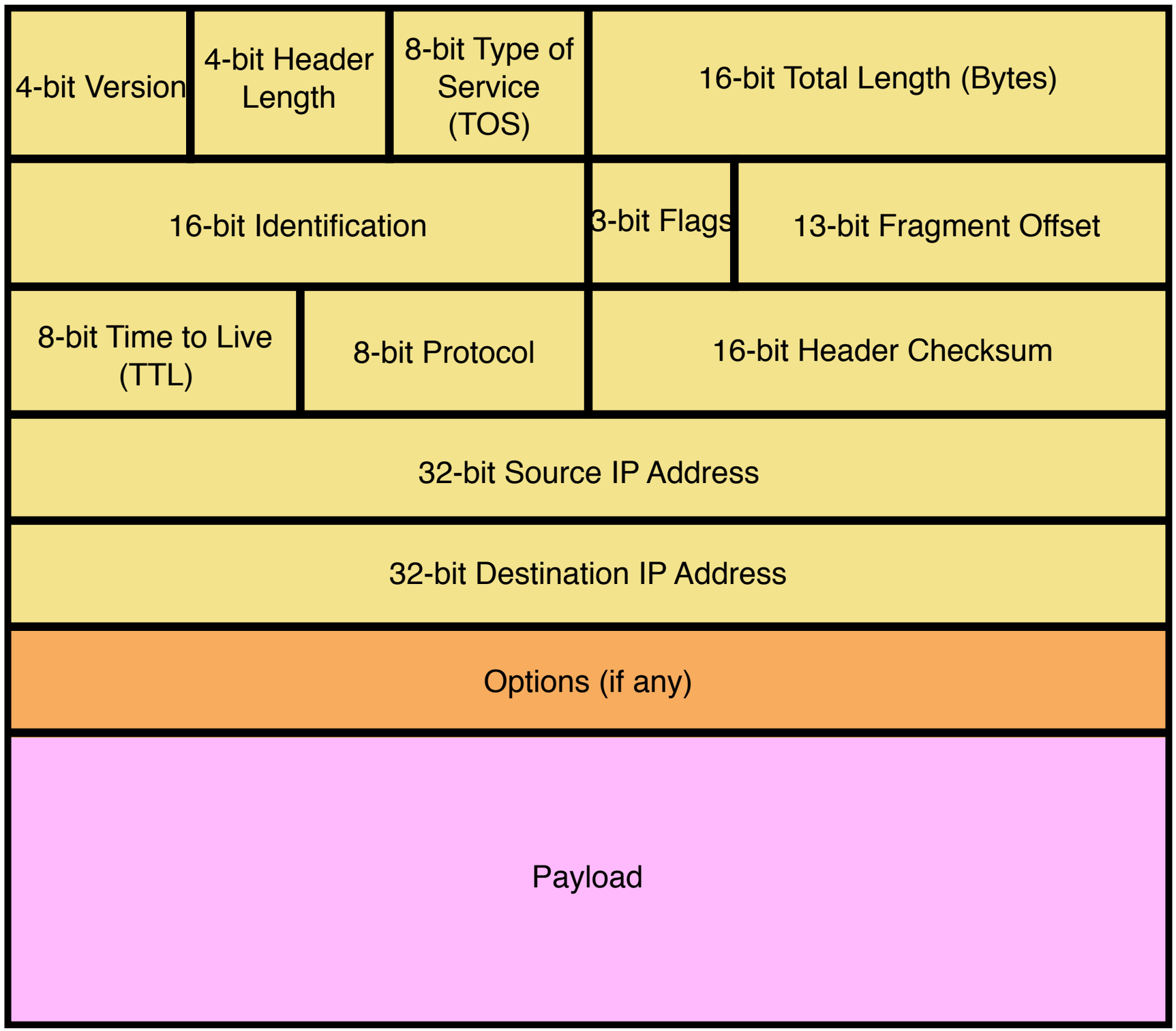
From Semantics to Syntax

- The past few slides discussed the information the header must provide
- Will now show the syntax (layout) of IPv4 header, and discuss the semantics in more detail

IP Packet Structure



20 Bytes of Standard Header, then Options



Next Set of Slides

- Mapping between tasks and header fields
- Each of these fields is devoted to a task
- Let’s find out which ones and why…

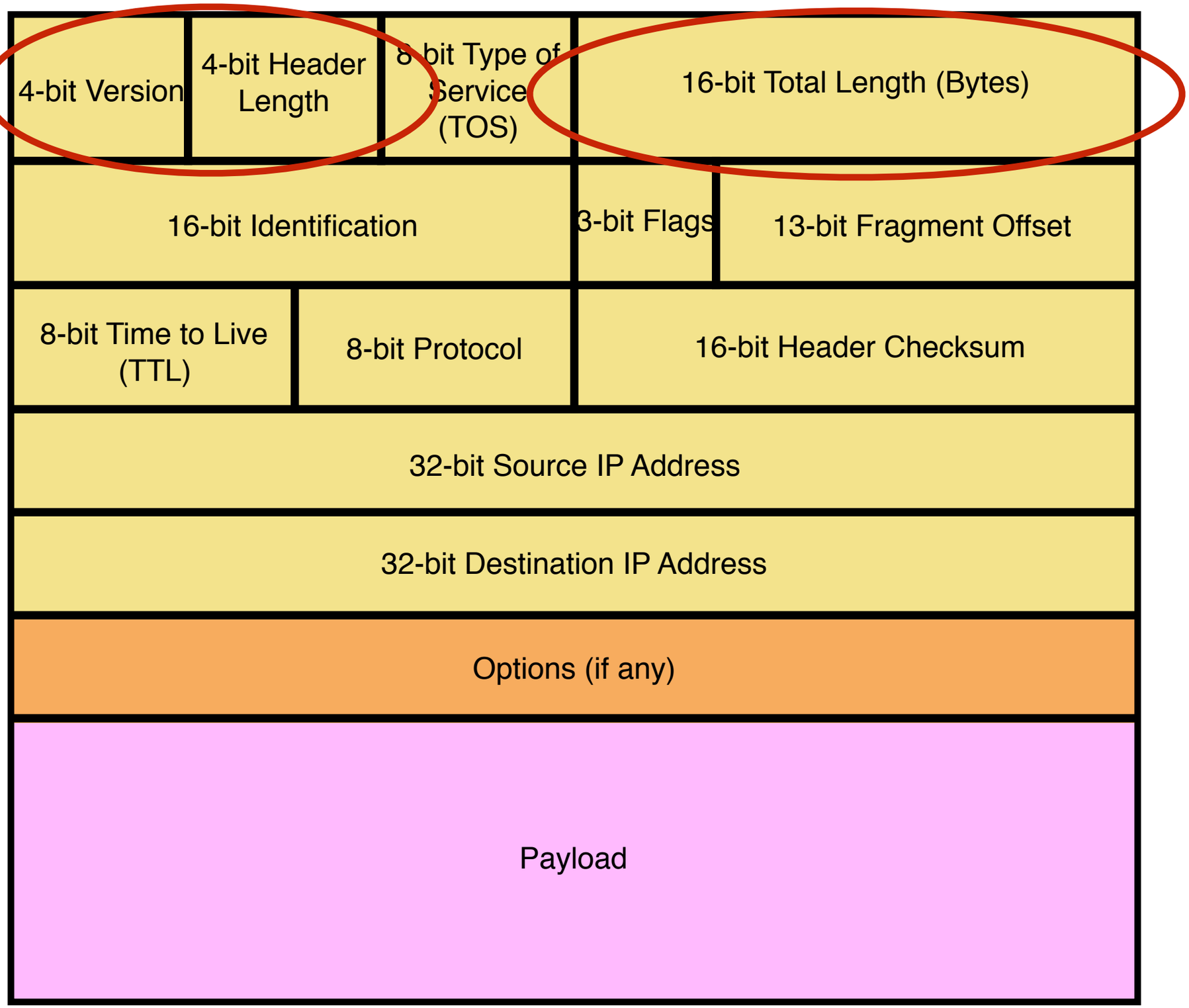
Go Through Tasks One-by-One

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Read Packet Correctly

- **Version number** (4 bits)
    - Indicates the version of the IP protocol
    - Necessary to know what other fields to expect
    - Typically “4” (for IPv4), and sometimes “6” (for IPv6)
- **Header length** (4 bits)
    - Number of 32-bit words in the header
    - Typically “5” (for a 20-byte IPv4 header)
    - Can be more when IP options are used
- **Total length** (16 bits)
    - Number of bytes in the packet
    - Maximum size is 65,535 bytes (2^16 -1)
    - … though underlying links may impose smaller limits

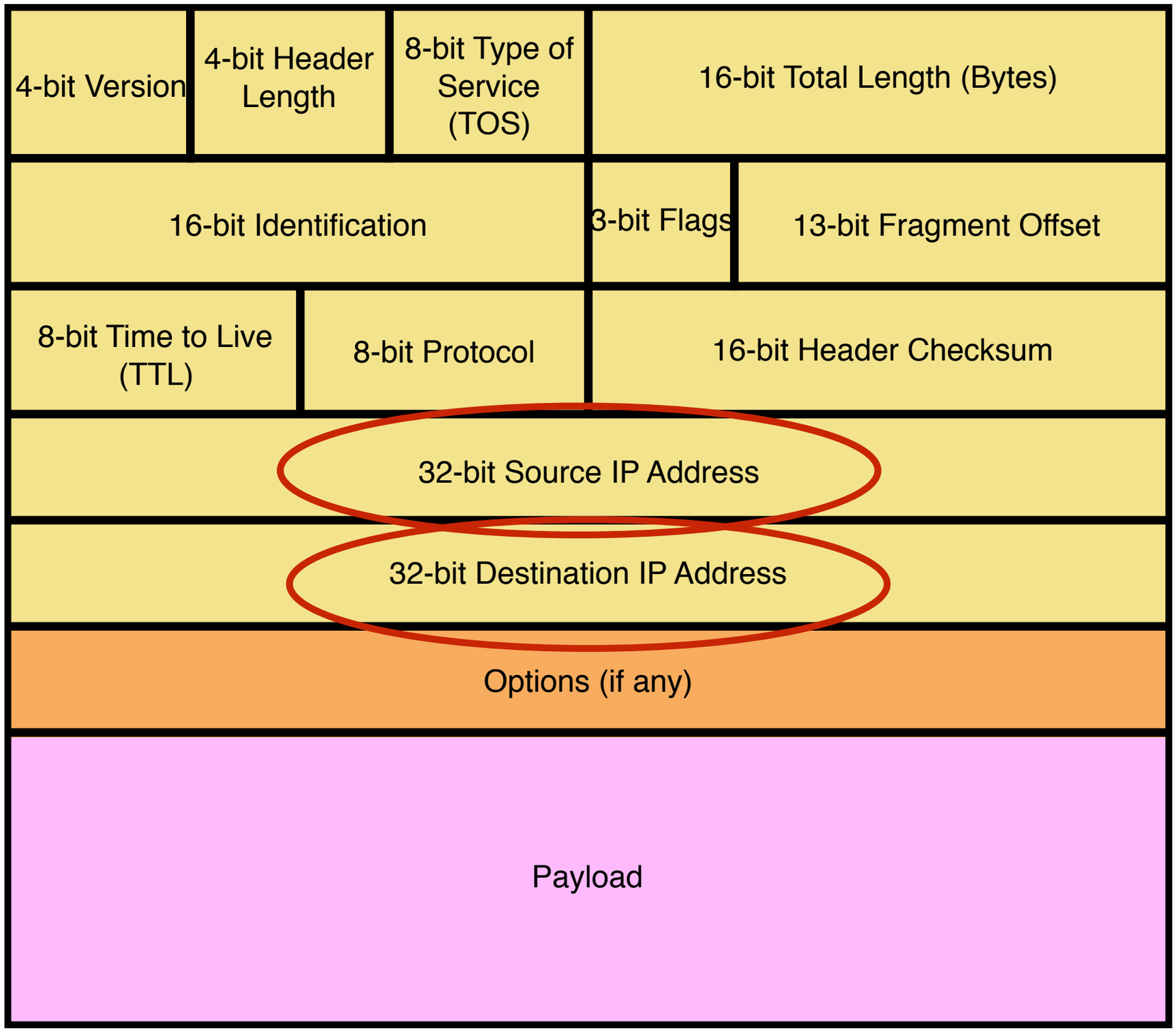
Fields for Reading Packet Correctly



Getting Packet to Destination and Back

- **Two IP addresses**
    - Source IP address (32 bits)
    - Destination IP address (32 bits)
- **Destination Address**
    - Unique locator for the receiving host
    - Allows each node to make forwarding decisions
- **Source Address**
    - Unique locator for the sending host
    - Recipient can decide whether to accept packet
    - Enables recipient to send a reply back to the source

Fields for Reading Packet Correctly



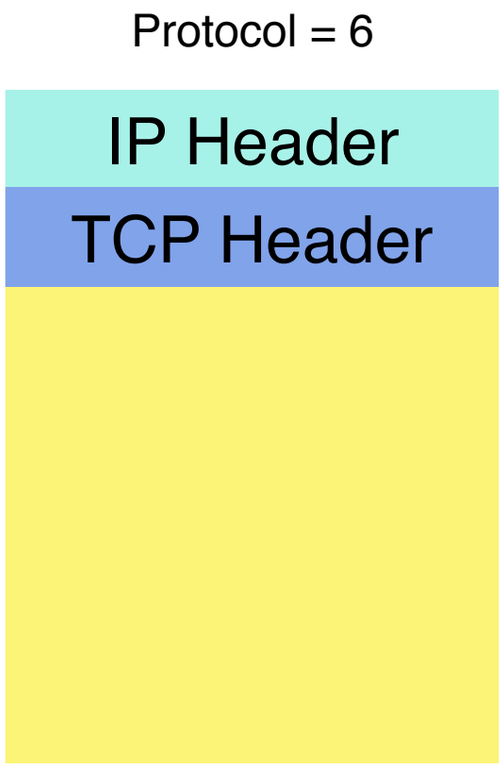
Questions?

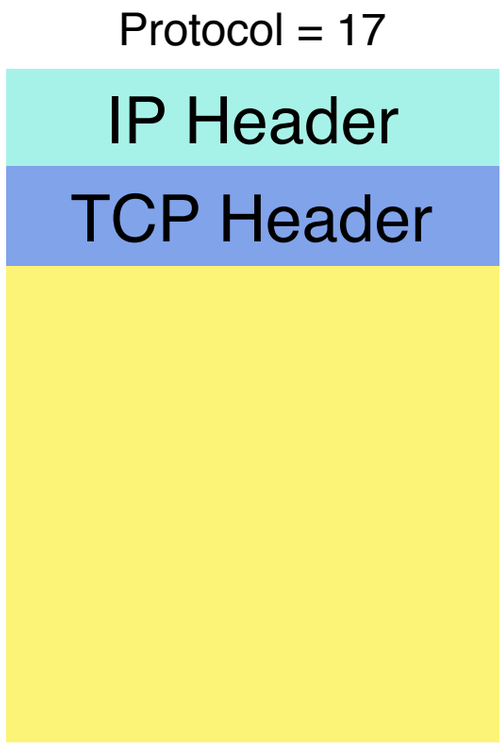
List of Tasks

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- **Tell host what to do with packet once arrived**
- Specify any special network handling of the packet
- Deal with problems that arise along the path

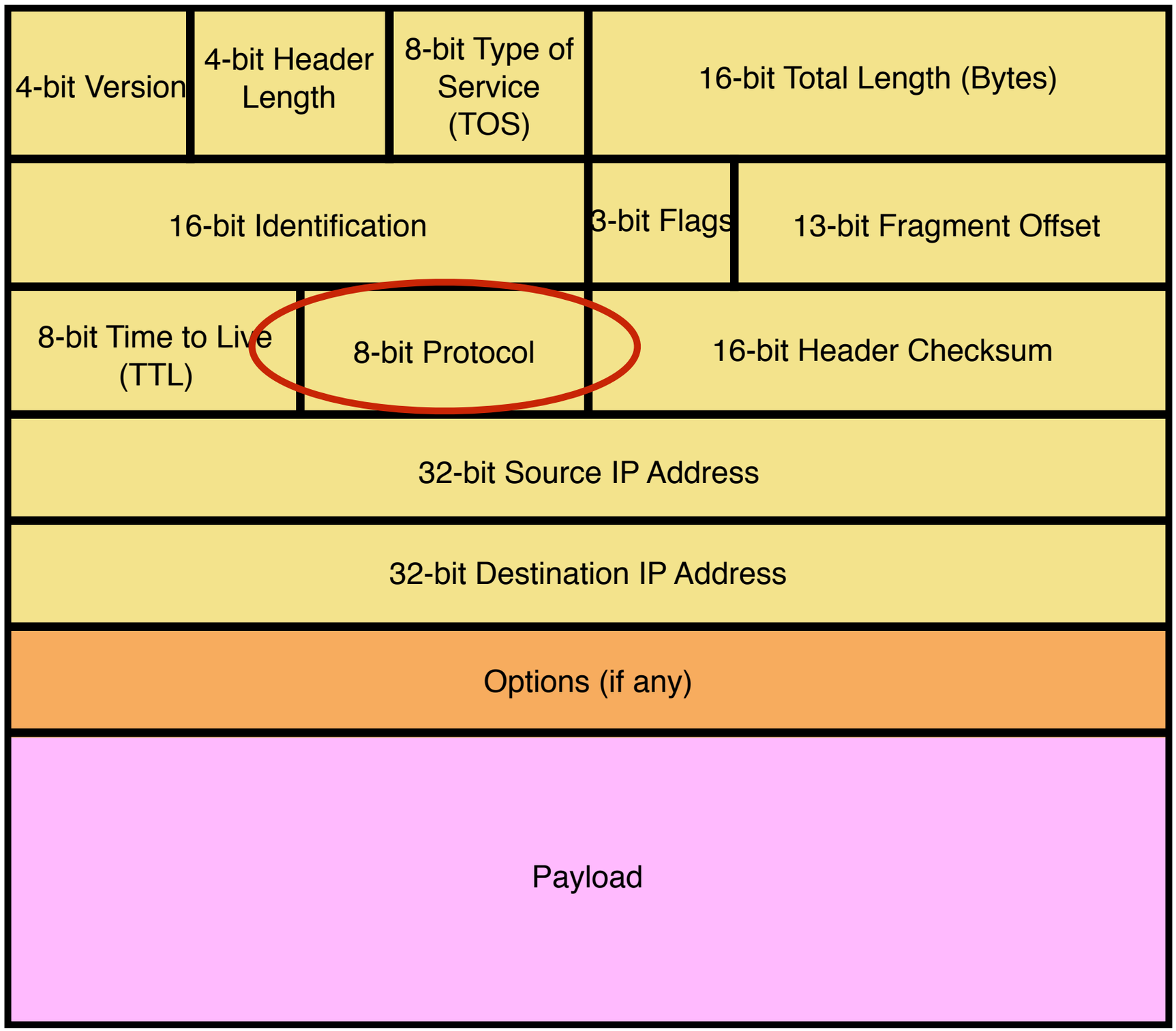
Telling Host How to Handle Packet

- **Protocol (8 bits)**
  - Identifies the higher level protocol
  - Important for demultiplexing at receiving host
- **Most common examples**
  - E.g., “6” for the Transmission Control Protocol (TCP)
  - E.g., “17” for the User Datagram Protocol





Fields for Reading Packet Correctly



Special Handling

- **Type-of-Service (8-bits)**
  - Allow packets to be treated differently based on needs
  - E.g., low delay for audio, high bandwidth for bulk transfer
  - Has been redefined several times, no general use
- **Options**
  - Ability to specify other functionality
  - Extensible format

Examples of Options

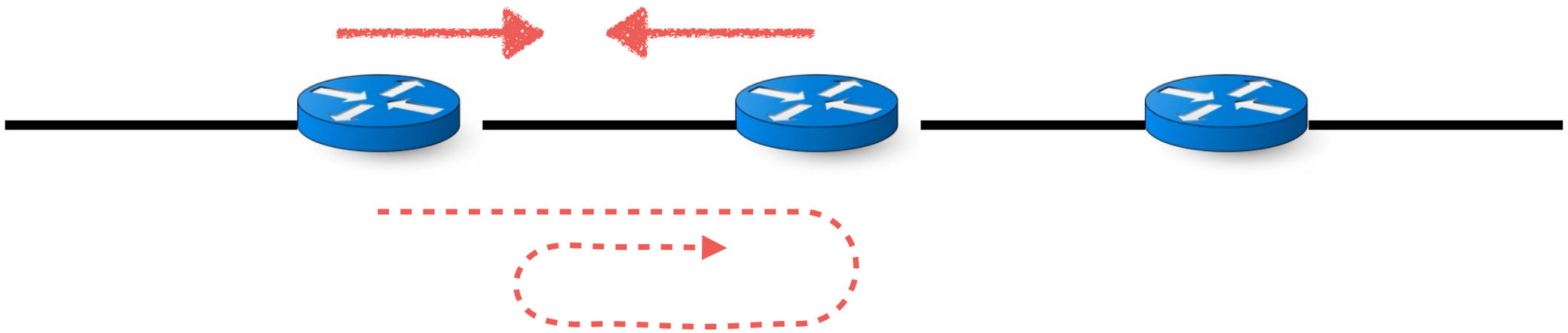
- Record Route
- Strict Source Route
- Loose Source Route
- Timestamp
- Traceroute
- Router Alert
- ...

Potential Problems

- Header Corrupted: **Checksum**
- Loop: **TTL**
- Packet too large: **Fragmentation**

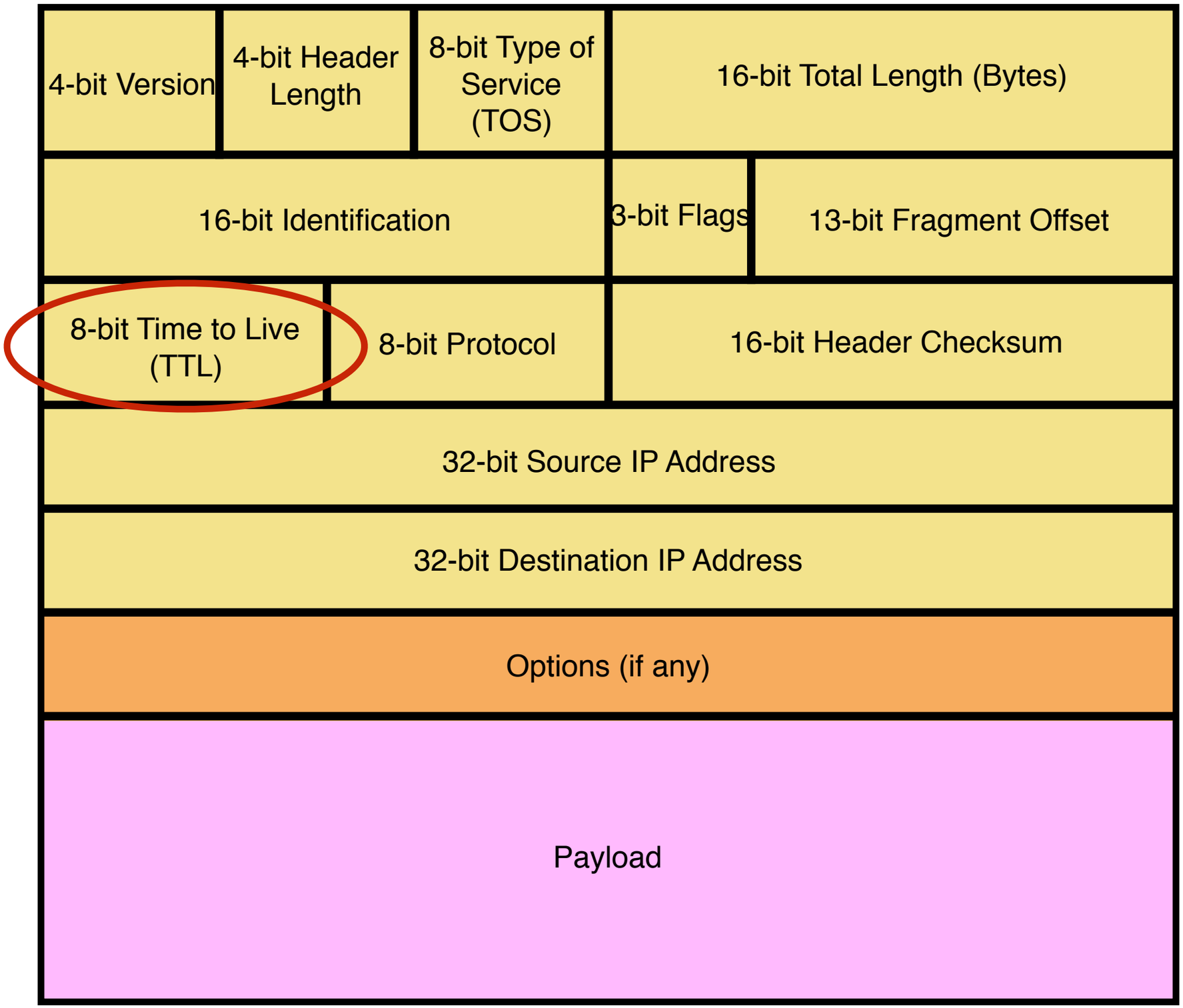
Preventing Loops

- Forwarding loops cause packets to cycle forever
    - As these accumulate, eventually consume all capacity



- Time-to-live (TTL) Field (8-bits)
    - Decremented at each hop, packet discarded if reaches 0
    - … and “time exceeded” message is sent to the source
        - Using “ICMP” control message; basis for traceroute

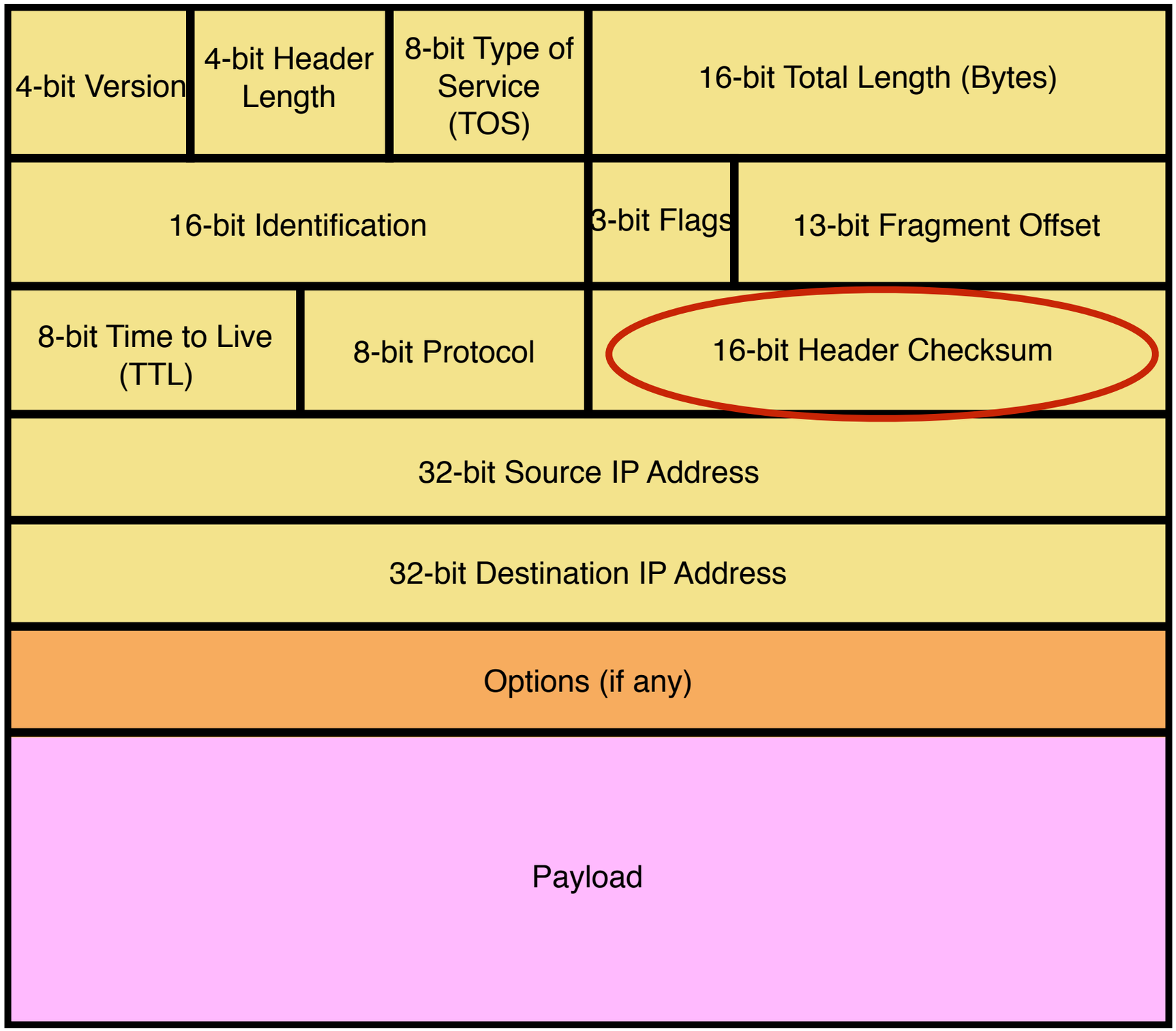
TTL Field


4-bit Version
4-bit Header Length
8-bit Type of Service (TOS)
16-bit Total Length (Bytes)
16-bit Identification
3-bit Flags
13-bit Fragment Offset
8-bit Time to Live (TTL)
8-bit Protocol
16-bit Header Checksum
32-bit Source IP Address
32-bit Destination IP Address
Options (if any)
Payload

Header Corruption

- Checksum (16 bits)
    - Particular form of checksum over packet header
- If not correct, router discards packets
    - So it doesn’t act in bogus information
- Checksum recalculated at every router
    - Why?
    - Why include TTL?
    - Why only header?

Checksum Field



Packet Header as an interface

- **Useless to learn the header format by heart**
  - If you remember the tasks that need to be performed …
  - Understanding **why** header format is what it is …
  - In general: if you understand the problem, solution is easy
  - As the problem evolves, you will know where to look for a solution
- **Transition from IPv4 to IPv6**
  - Gradually happening …
  - If you want to learn a bit, see backup slides

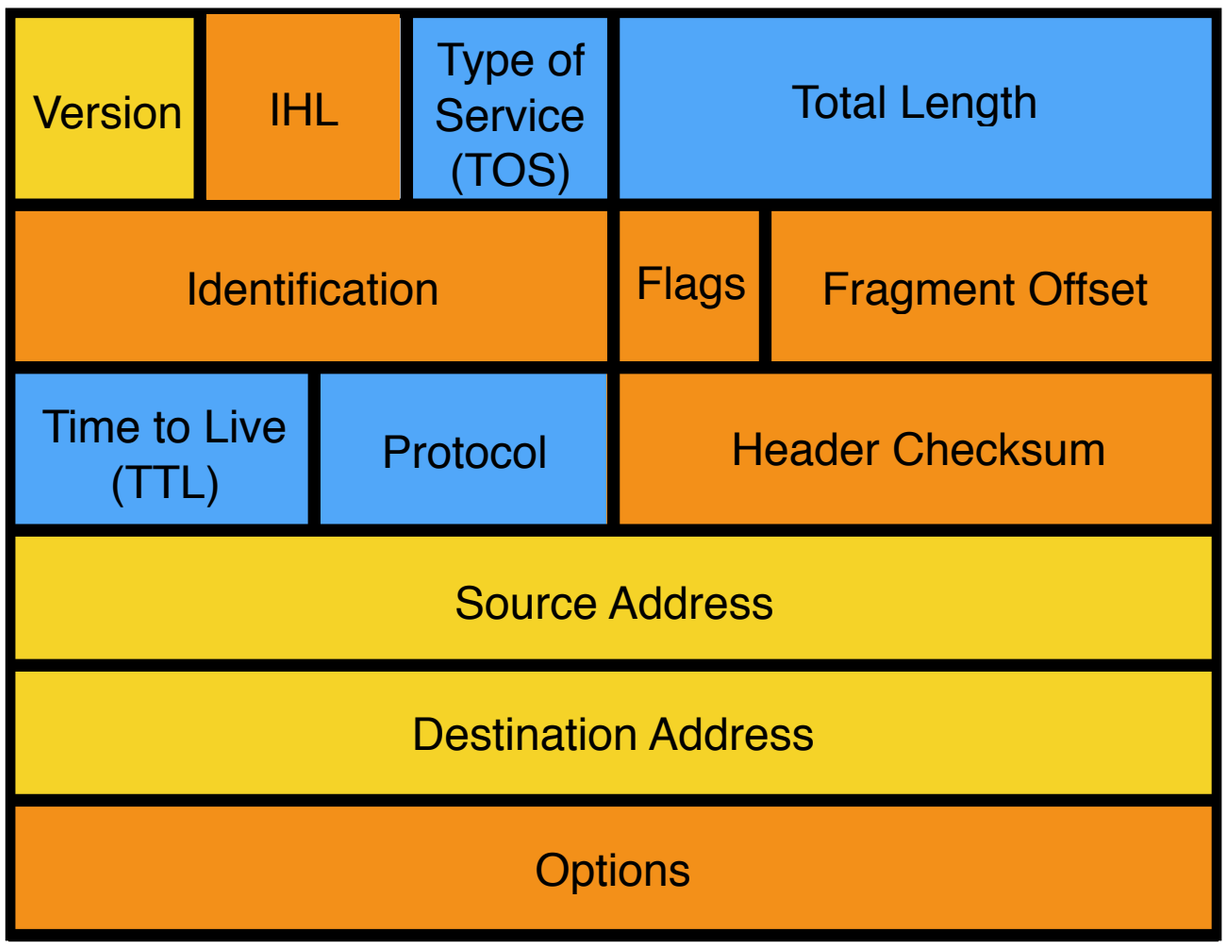
This is it for today!

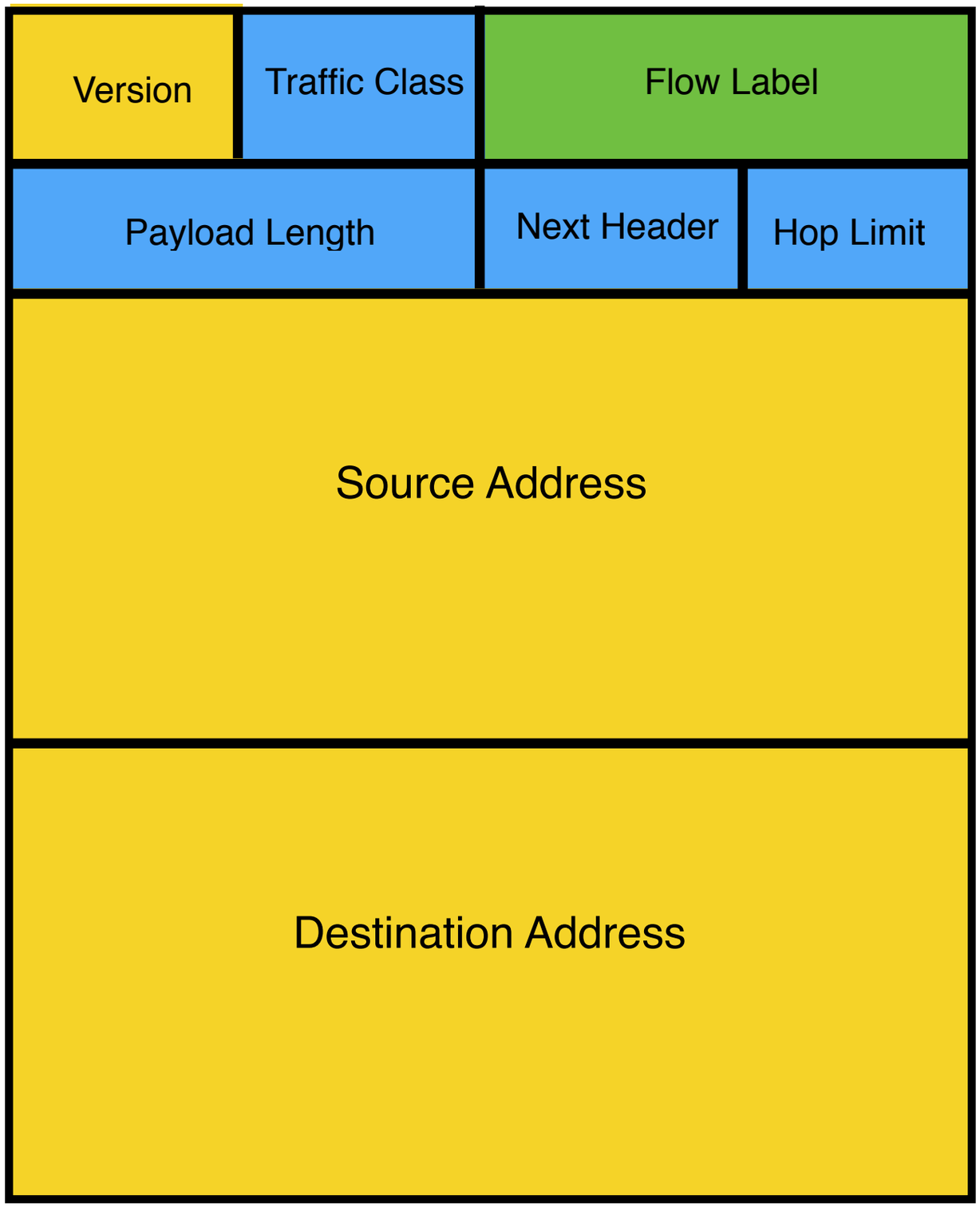
IPv6

IPv6

- Motivated (prematurely) by address exhaustion
    - Address **four** times as big
- Steve Deering focused on simplifying IP
    - Got rid of all fields that were not absolutely necessary
    - “Spring Cleaning” for IP
- Result is an elegant, if unambitious, protocol

IPv4 and IPv6 Header Comparison





Field name kept from IPv4 to IPv6

Fields not kept in IPv6

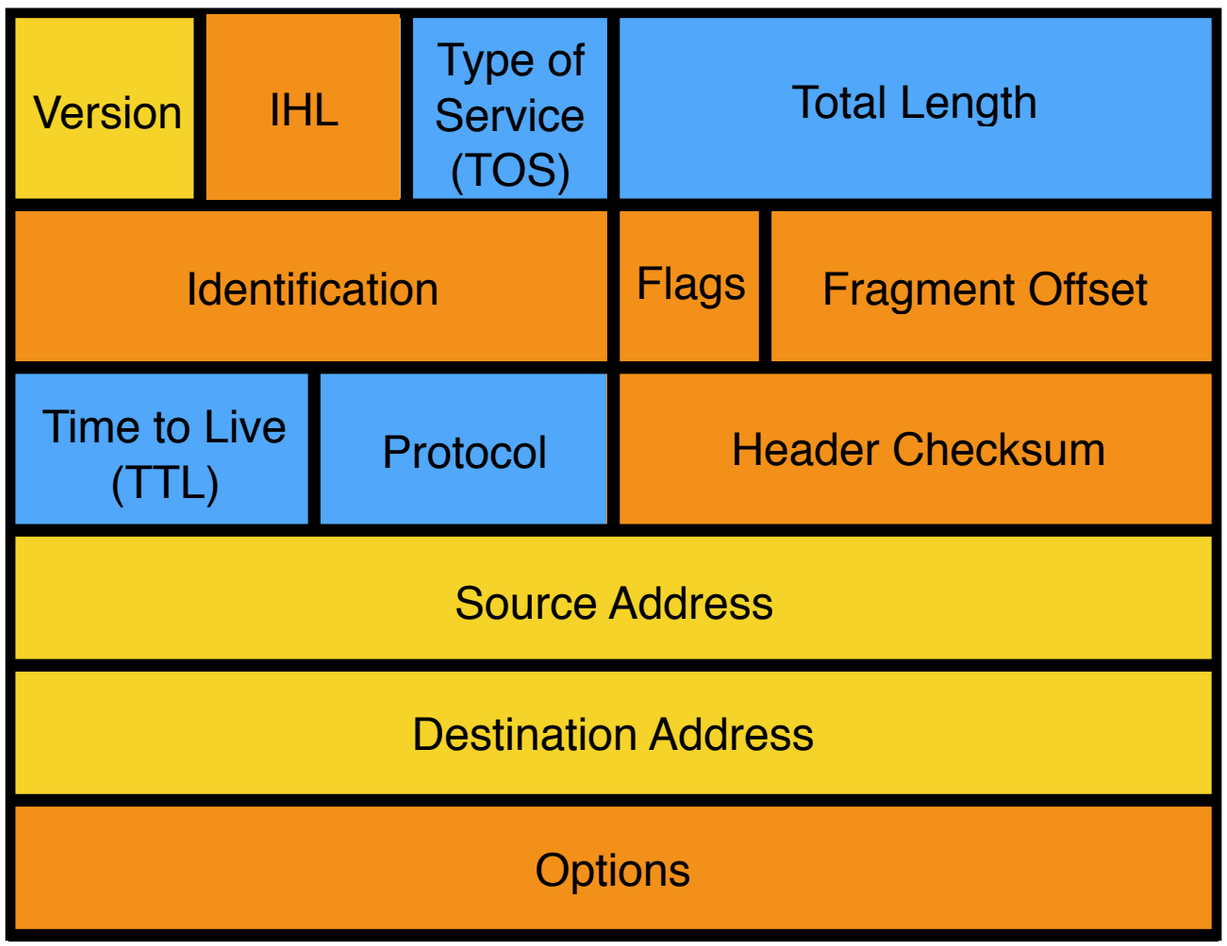
Name and position changed in IPv6

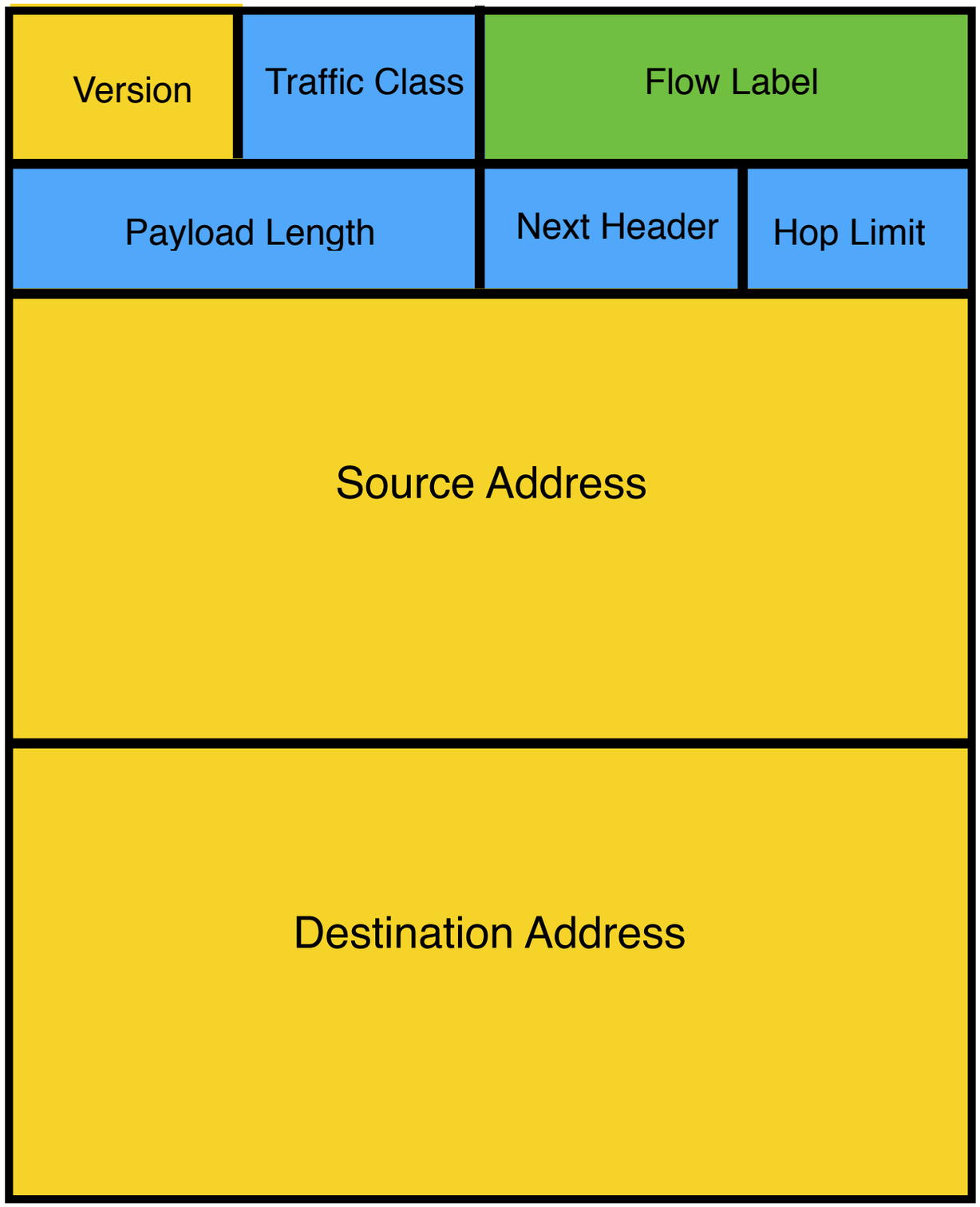
New field in IPv6

Summary of Changes

- Eliminated Fragmentation
- Eliminated header length
- Eliminated Checksum
- New options mechanism (next header)
- Expanded address
- Added Flow Label

IPv4 and IPv6 Header Comparison





Field name kept from IPv4 to IPv6

Fields not kept in IPv6

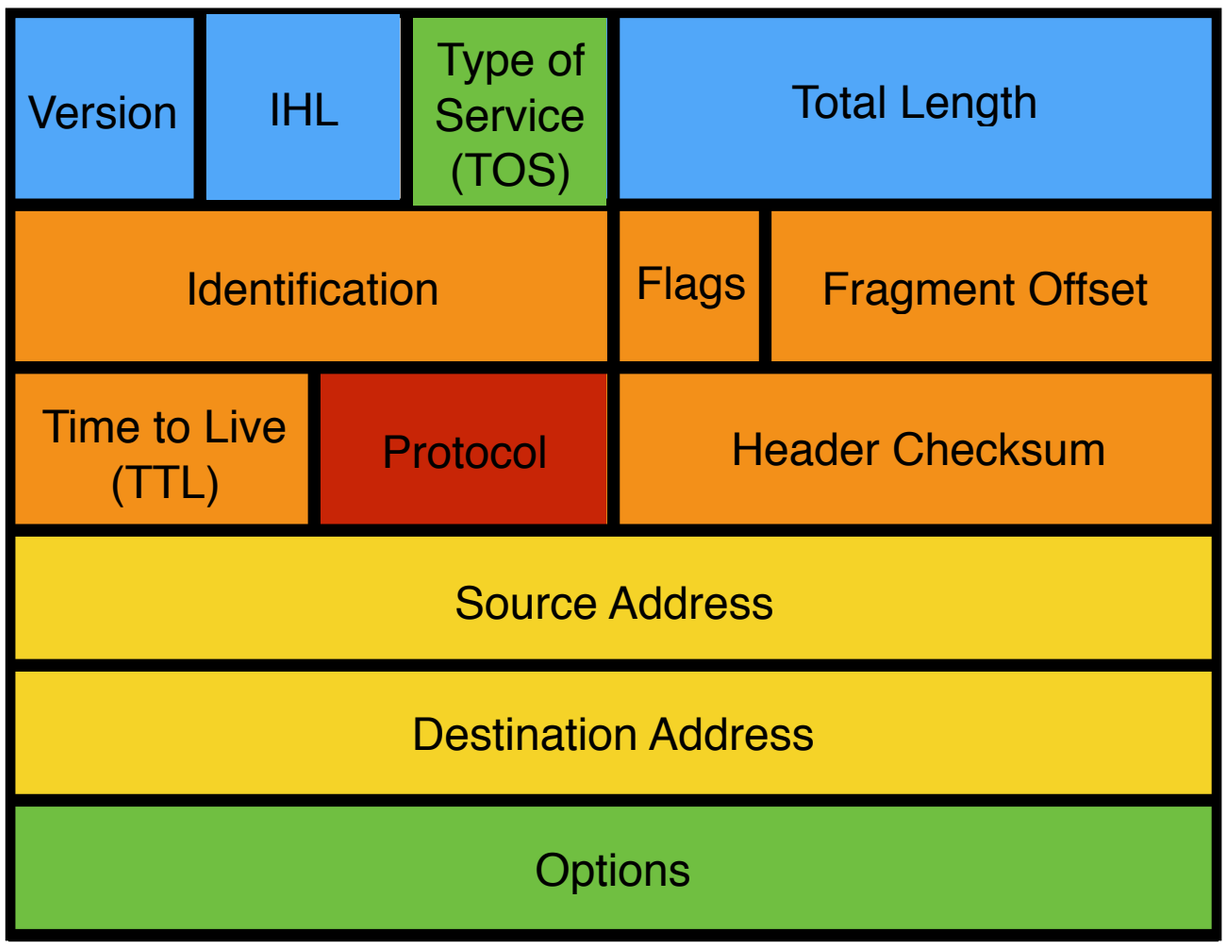
Name and position changed in IPv6

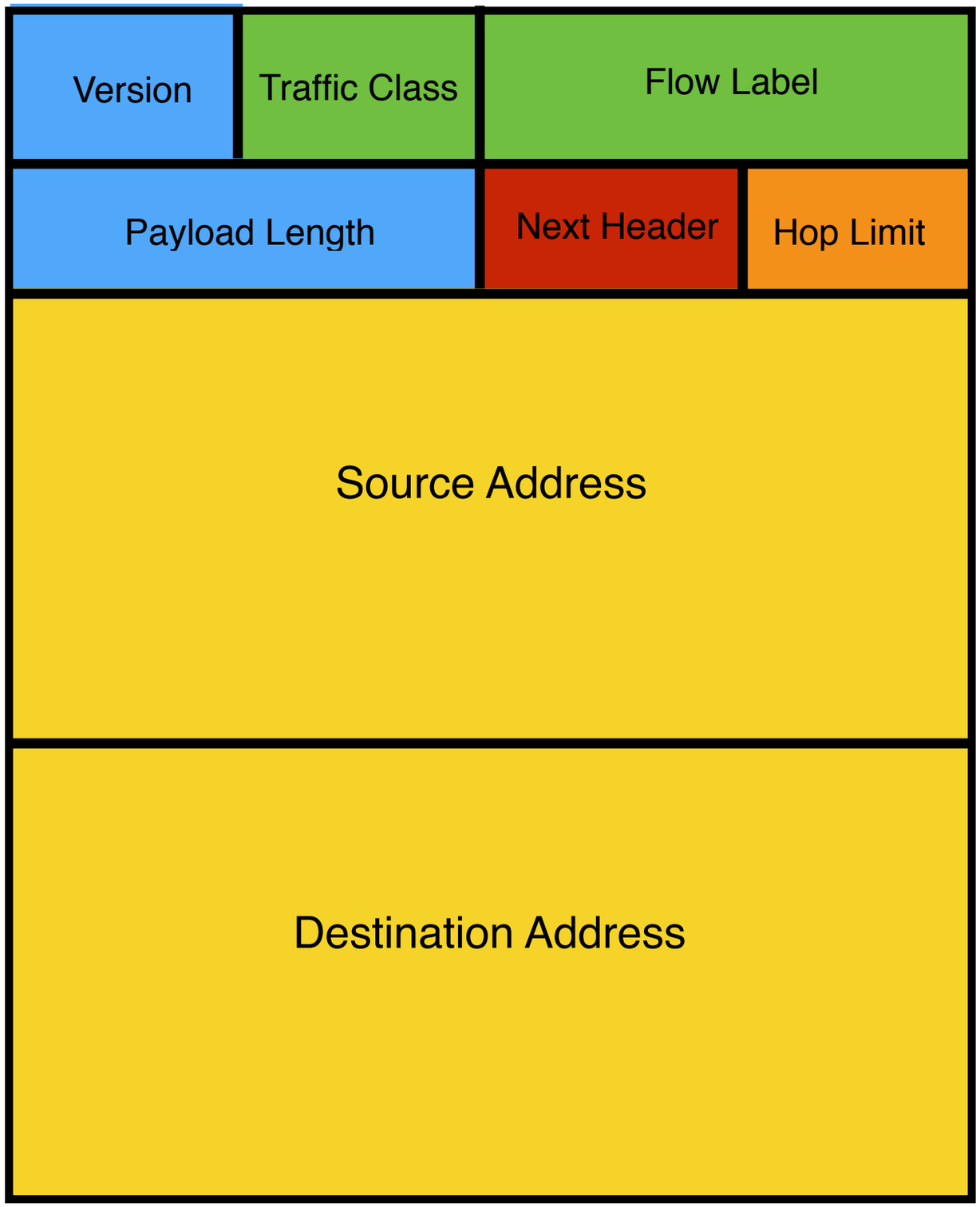
New field in IPv6

Philosophy of Changes

- Don’t deal with problems: leave to ends
    - Eliminated fragmentation
    - Eliminated checksum
    - **Why retain TTL?**
- Simplify handling
    - New options mechanism (uses next header approach)
    - Eliminated header length
        - **Why couldn’t IPv4 do this?**
- Provide general flow label for packet
    - Not tied to semantics
    - Provides great flexibility

IPv4 and IPv6 Header Comparison





To Destination and Back (expanded)

Deal with Problems (greatly reduced)

Read Correctly (reduced)

Special Handling (Similar)