CE693: Adv. Computer Networking

L-17 Naming

Acknowledgments: Lecture slides are from the graduate level Computer Networks course thought by Srinivasan Seshan at CMU. When slides are obtained from other sources, a a reference will be noted on the bottom of that slide. A full list of references is provided on the last slide.

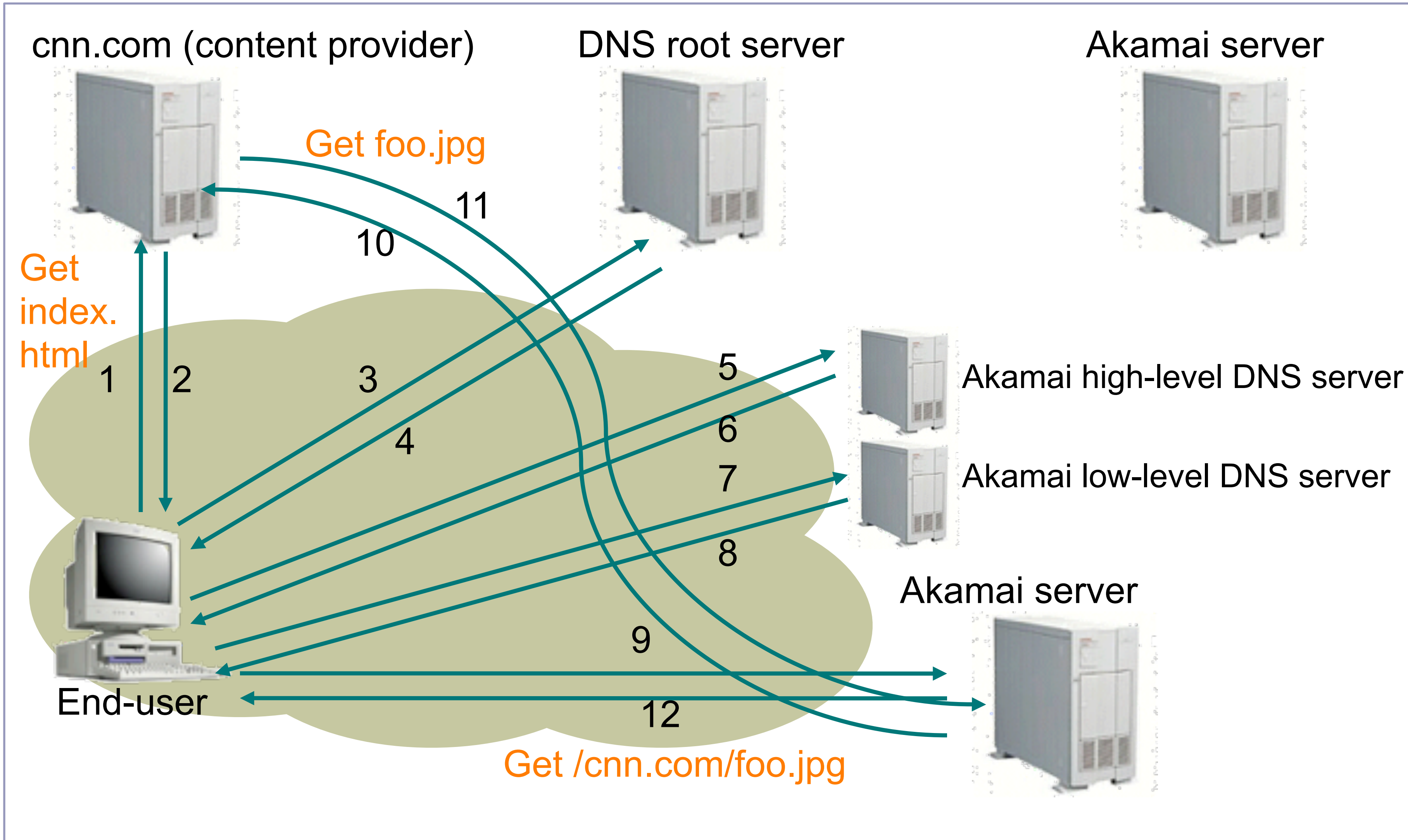
Today's Lecture

- Naming and CDNs
- Required readings
  - Middleboxes No Longer Considered Harmful
  - Internet Indirection Infrastructure

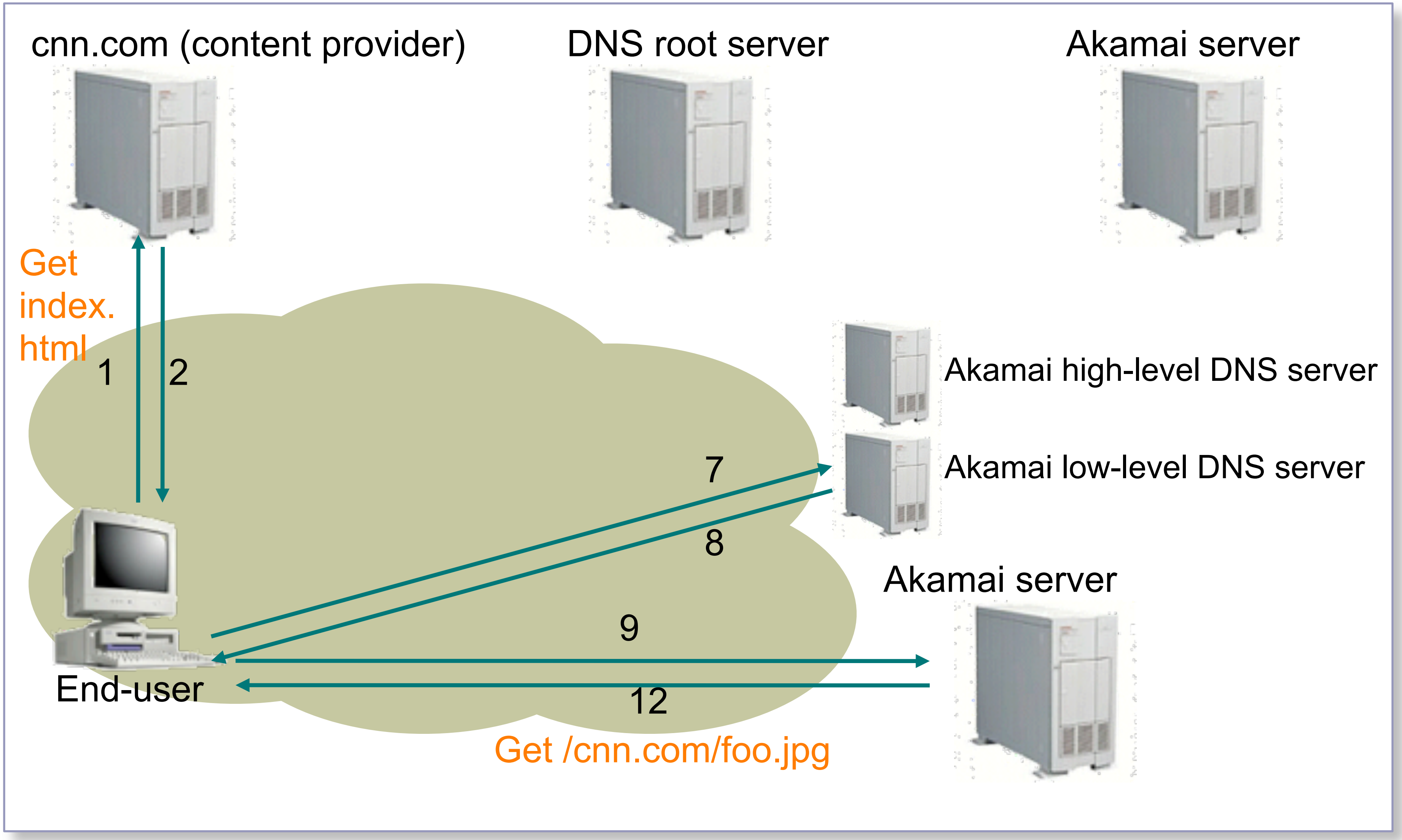
Overview

- Akamai (CDNs)
- I3
- DOA

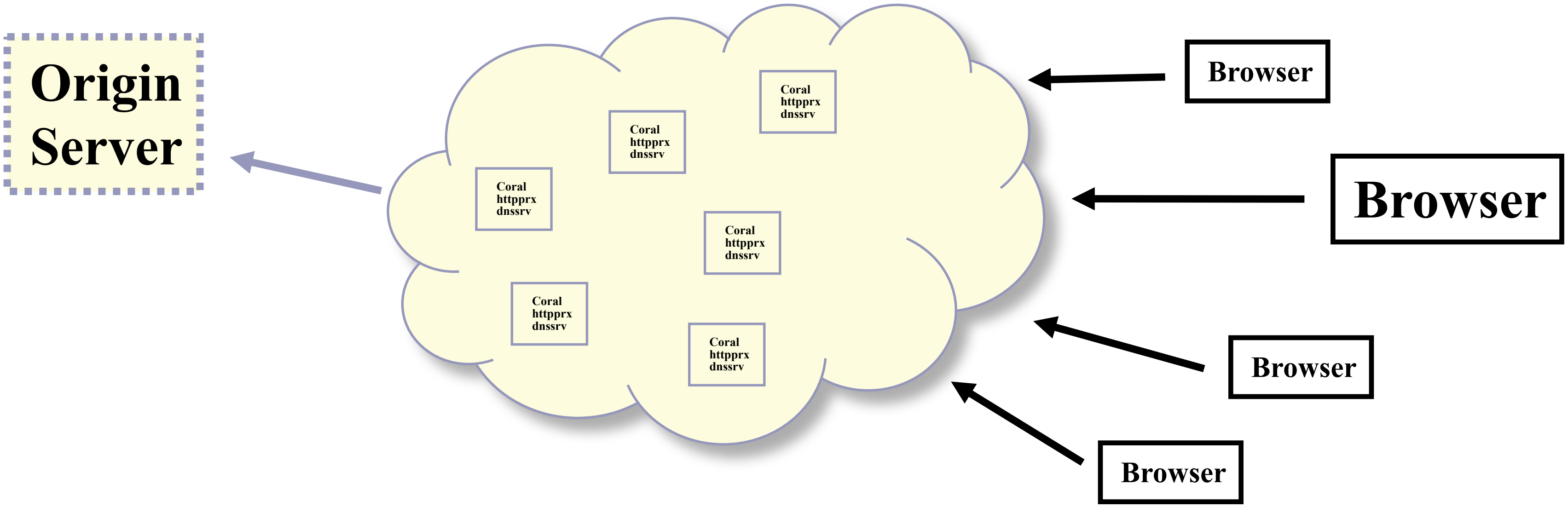
How Akamai Works



Akamai – Subsequent Requests



Coral: An Open CDN



Pool resources to dissipate flash crowds

- Implement an open CDN
- Allow anybody to contribute
- Works with unmodified clients
- CDN only fetches once from origin server

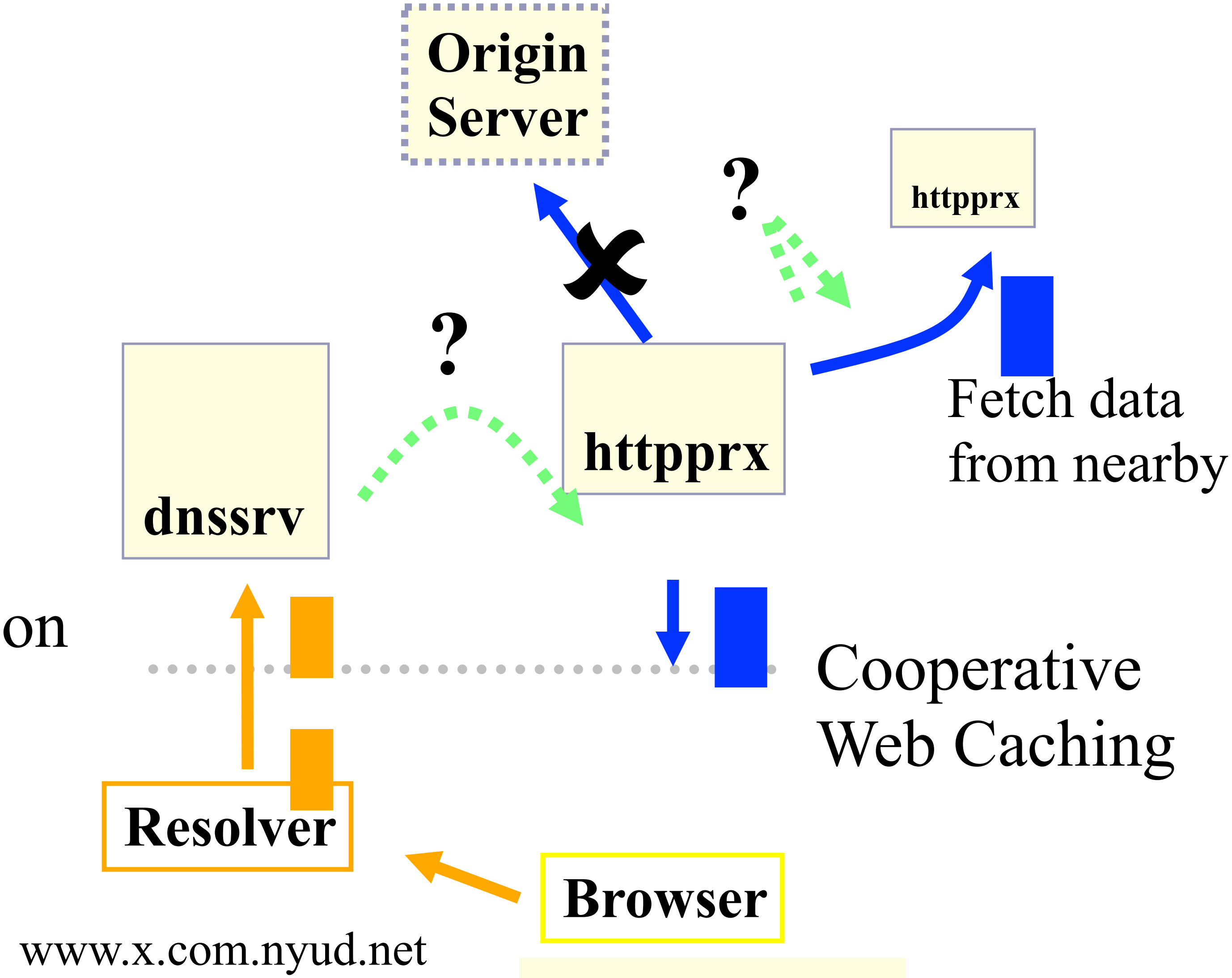
Using CoralCDN

- Rewrite URLs into “Coralized” URLs
- www.x.com → www.x.com.nyud.net:8090
  - Directs clients to Coral, which absorbs load
- Who might “Coralize” URLs?
  - Web server operators Coralize URLs
  - Coralized URLs posted to portals, mailing lists
  - Users explicitly Coralize URLs

CoralCDN components

DNS Redirection

Return proxy, preferably one near client



Functionality needed

- DNS: Given network location of resolver, return a proxy near the client

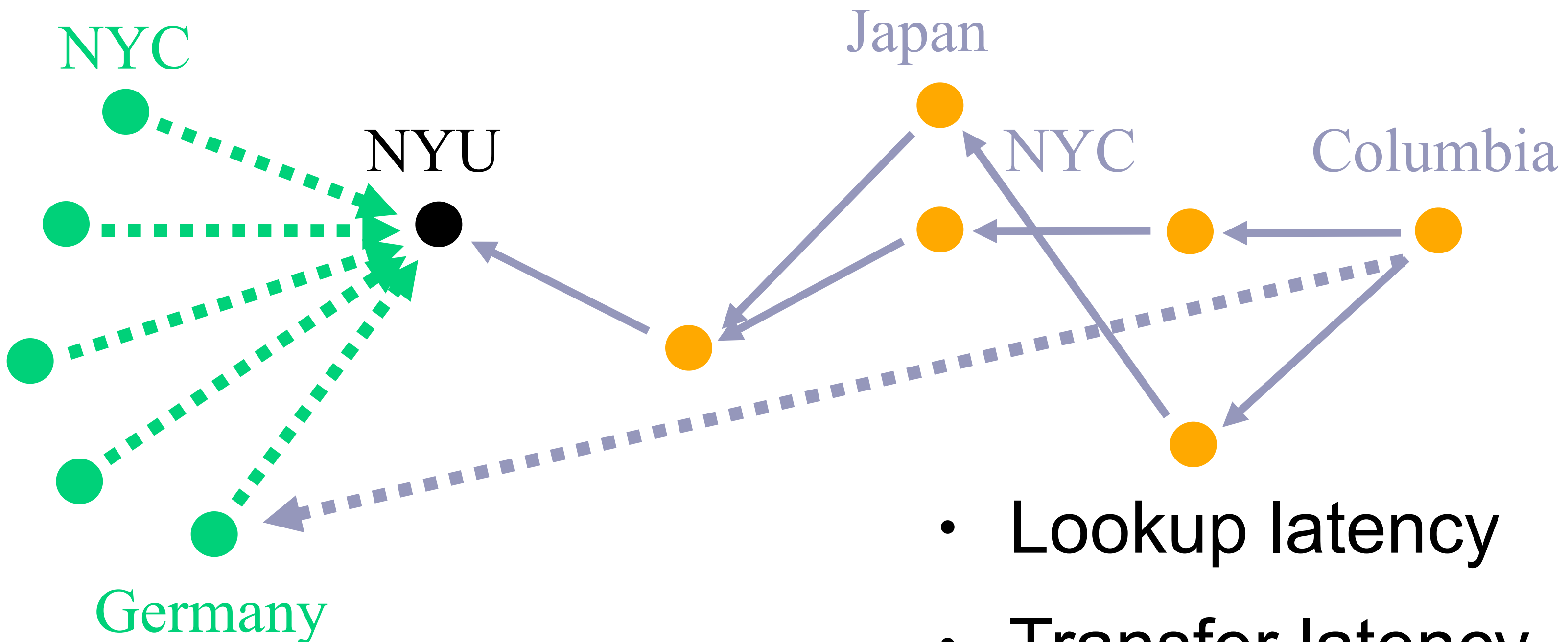
  put (network info, self)
  get (resolver info) → {proxies}

- HTTP: Given URL, find proxy caching object, preferably one nearby

  put (URL, self)
  get (URL) → {proxies}

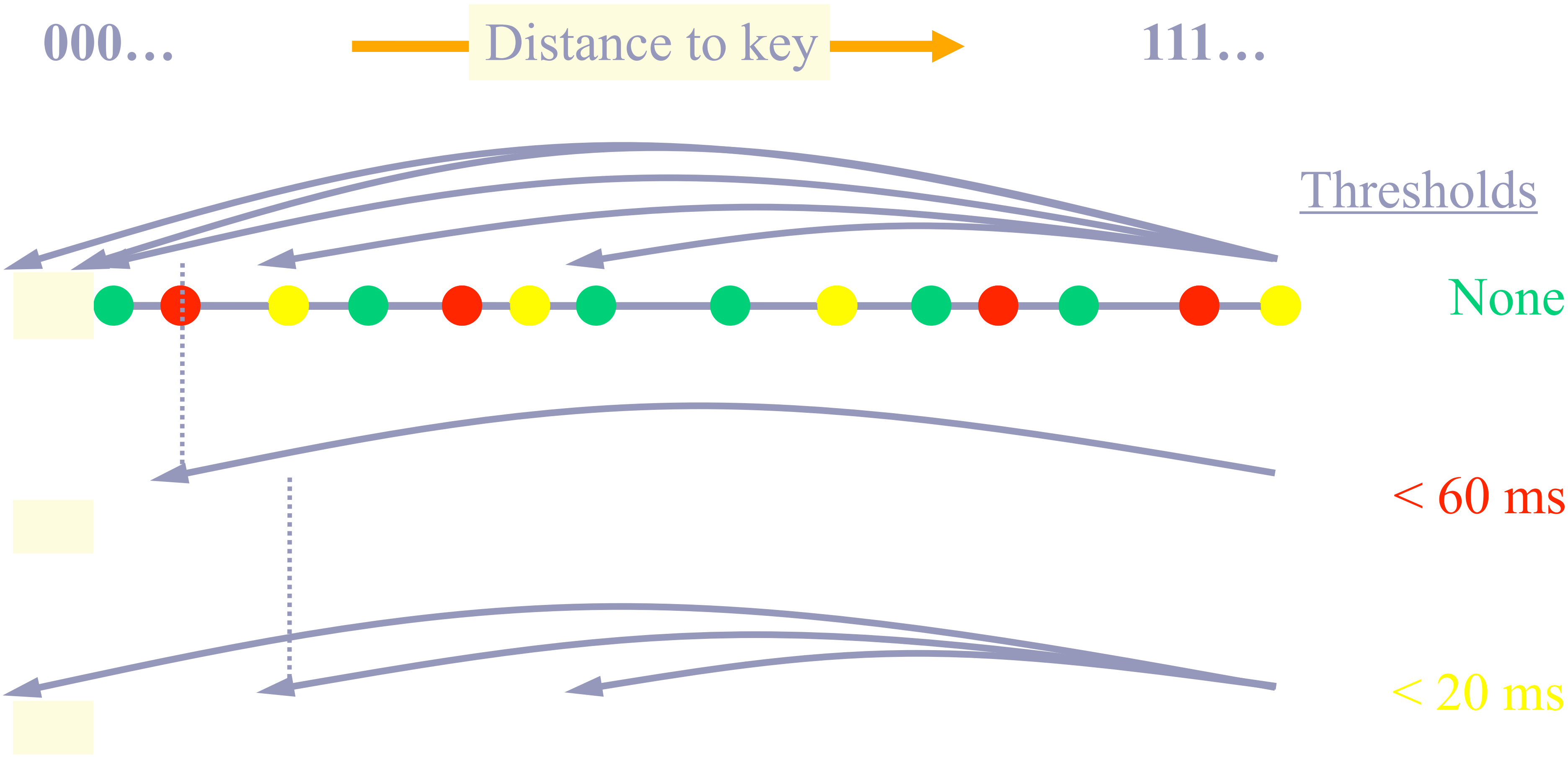
Use a DHT?

- Supports put/get interface using key-based routing
- Problems with using DHTs as given



- Lookup latency
- Transfer latency
- Hotspots

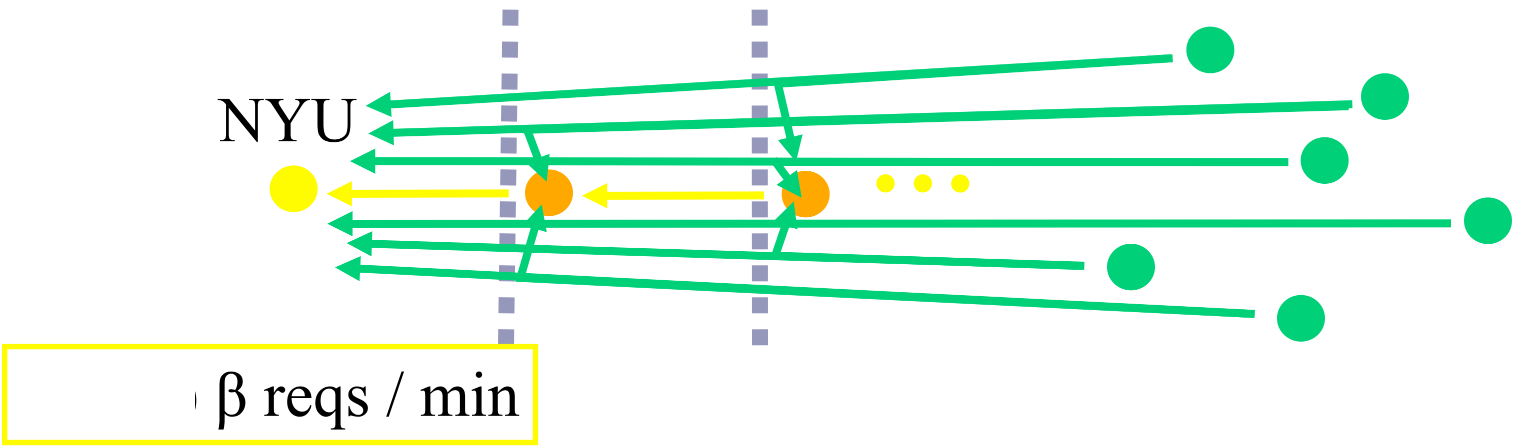
Key-based XOR routing



- Minimizes lookup latency
- Prefer values stored by nodes within faster clusters

Prevent insertion hotspots

- Store value once in each level cluster
  - Always storing at closest node causes hotspot

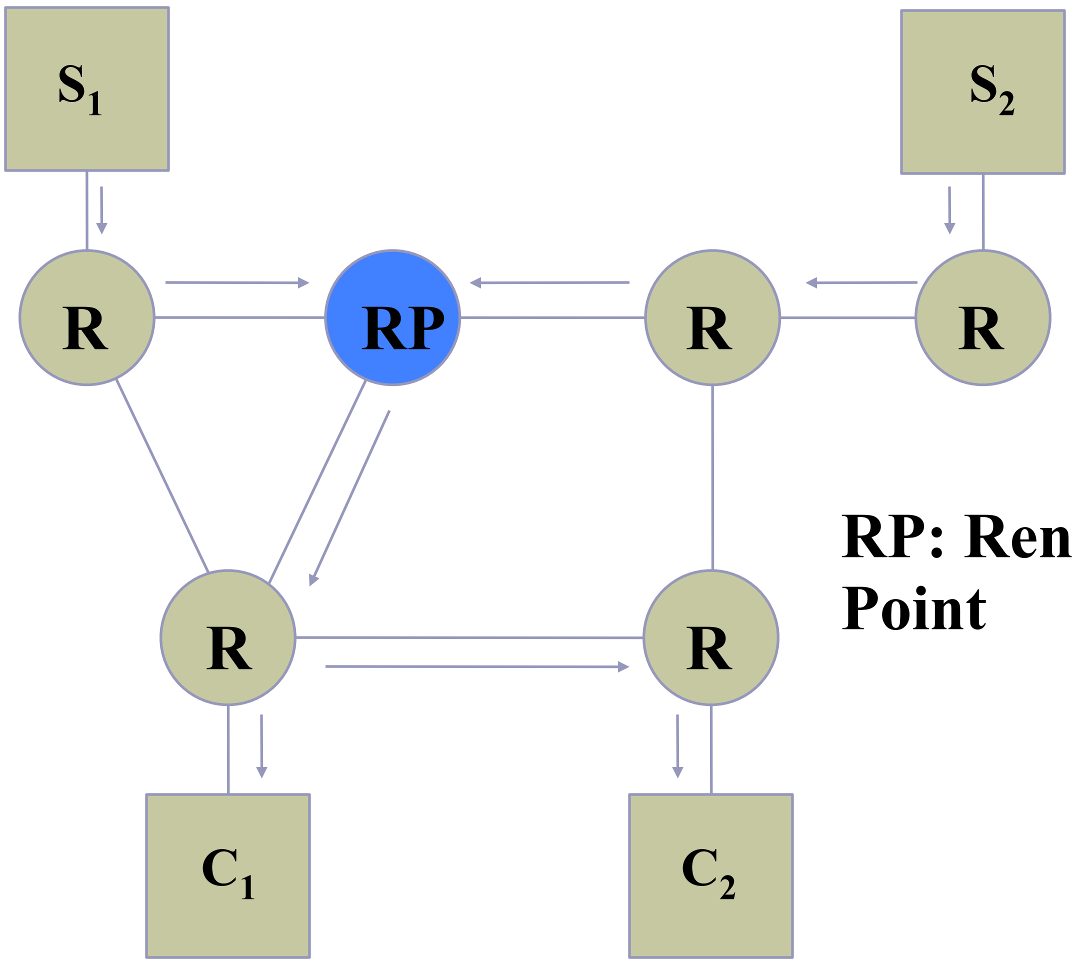


- Halt put routing at full and loaded node
  - Full → M vals/key with TTL > ½ insertion TTL
  - Loaded → β puts traverse node in past minute
- Store at furthest, non-full node seen

Overview

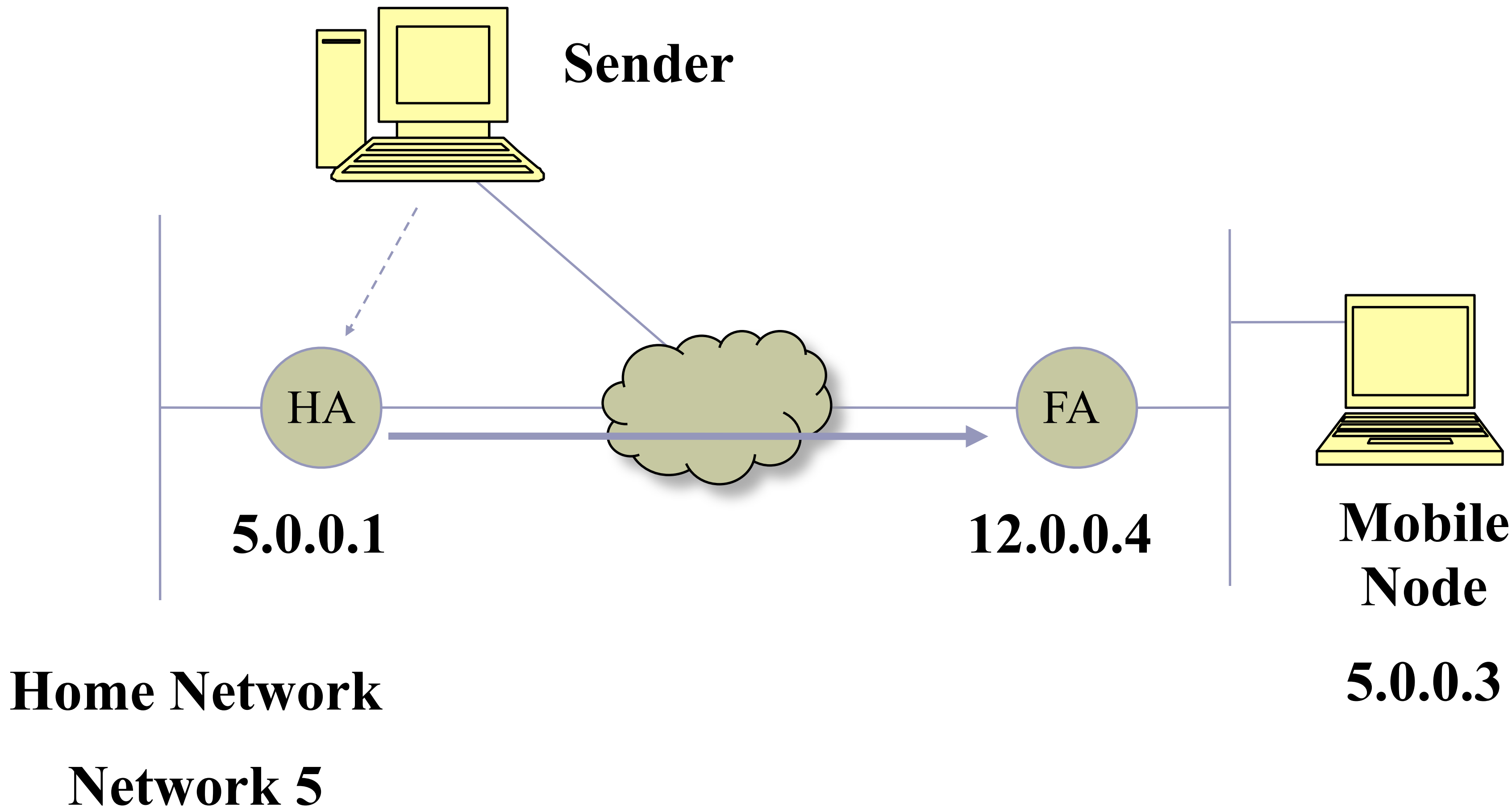
- Akamai
- I3
- DOA

Multicast



RP: Rendezvous Point

Mobility



Sender
HA
FA
5.0.0.1
12.0.0.4
Mobile
Node
5.0.0.3
Home Network
Network 5

i3: Motivation

- Today's Internet based on point-to-point abstraction
- Applications need more:
  - Multicast
  - Mobility
  - Anycast
- Existing solutions:
  - Change IP layer
  - Overlays

So, what's the problem?
A different solution for each service

The i3 solution

- Solution:
  - Add an indirection layer on top of IP
  - Implement using overlay networks

- Solution Components:
  - Naming using “identifiers”
  - Subscriptions using “triggers”
  - DHT as the gluing substrate

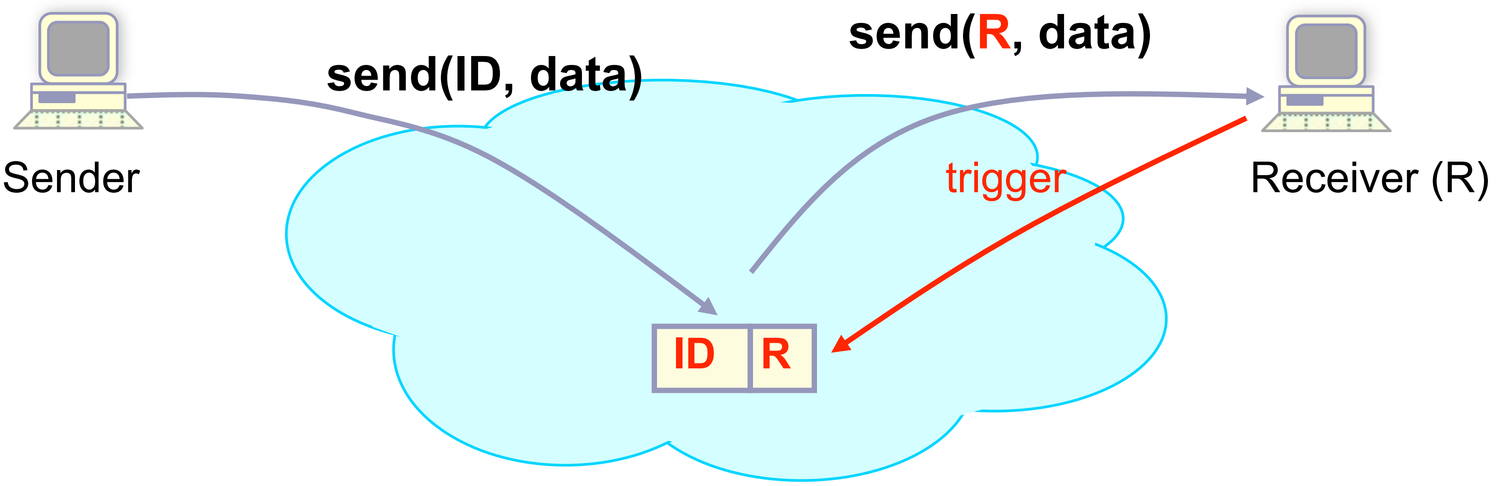
Only primitive needed

Indirection

Every problem in CS … ☺

i3: Rendezvous Communication

- Packets addressed to identifiers ("names")
- Trigger=(Identifier, IP address): inserted by receiver



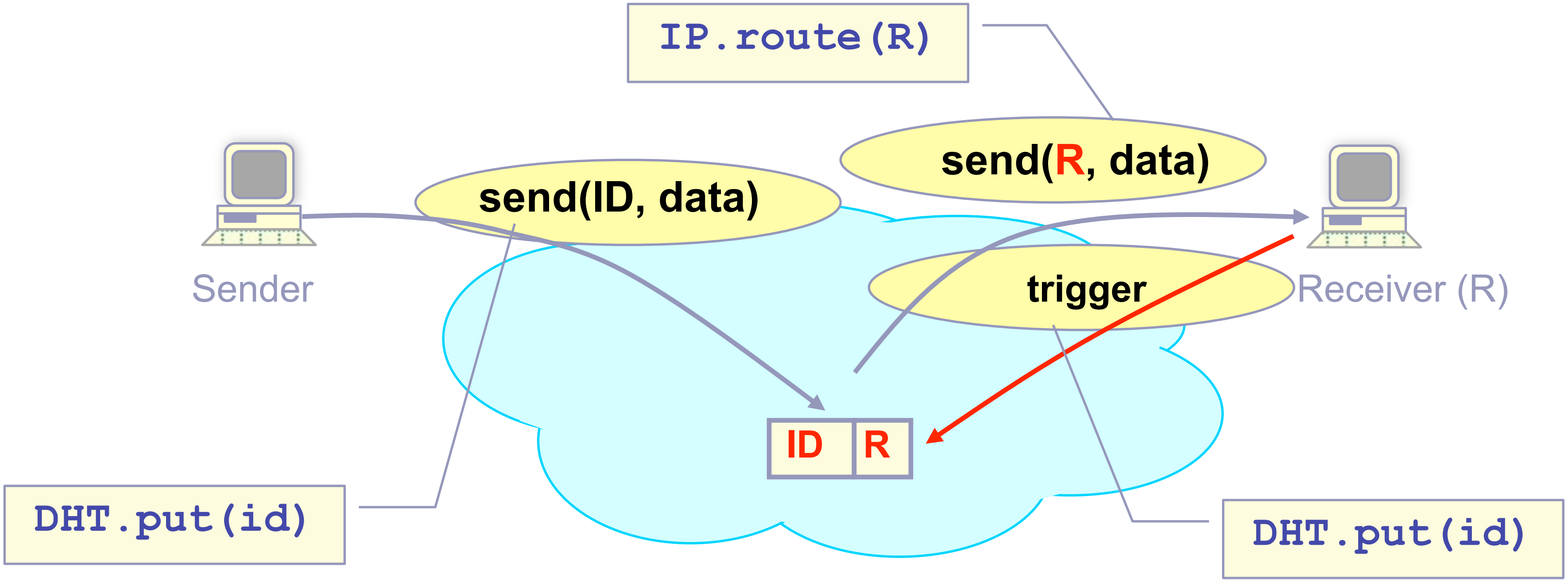
Senders *decoupled* from receivers

i3: Service Model

- API
  - `sendPacket(id, p);`
  - `insertTrigger(id, addr);`
  - `removeTrigger(id, addr); // optional`
- Best-effort service model (like IP)
- Triggers periodically refreshed by end-hosts
- Reliability, congestion control, and flow-control implemented at end-hosts

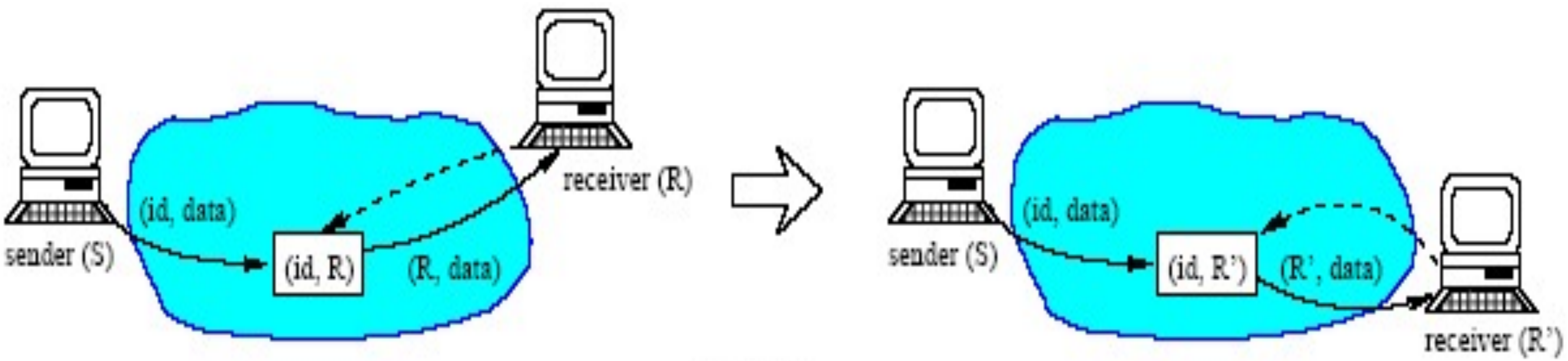
i3: Implementation

- Use a Distributed Hash Table
  - Scalable, self-organizing, robust
  - Suitable as a substrate for the Internet



Mobility

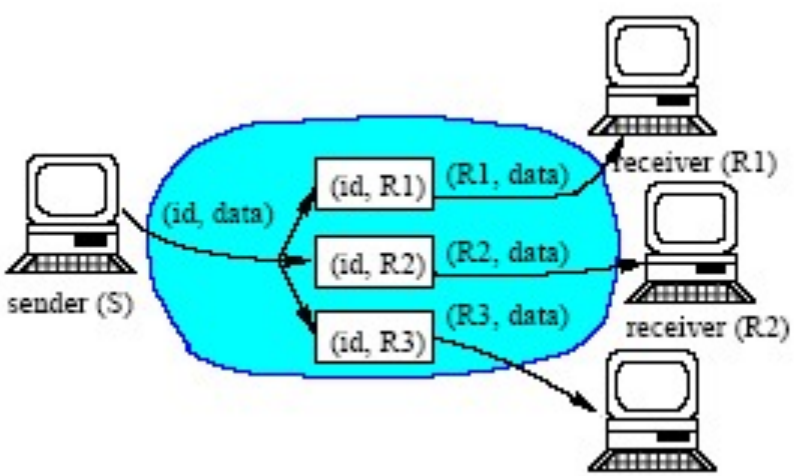
- The change of the receiver's address
- from R to R' is transparent to the sender



(a) Mobility

Multicast

- Every packet $(id, data)$ is forwarded to each receiver R_i that inserts the trigger (id, R_i)



(b) Multicast

Generalization: Identifier Stack

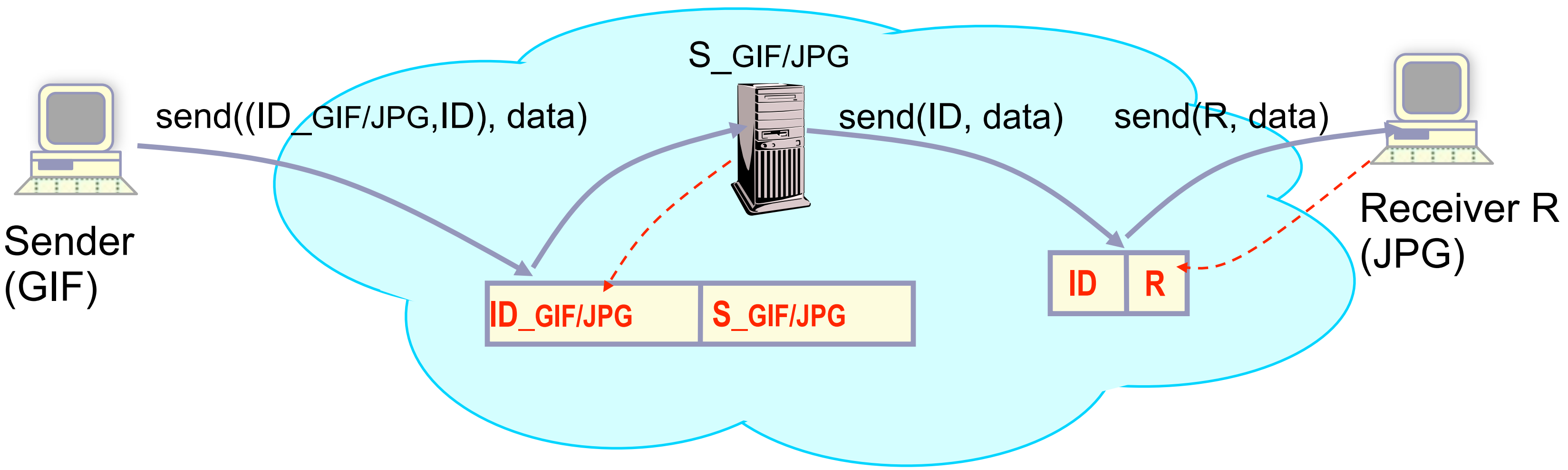
- Stack of identifiers
  - i3 routes packet through these identifiers

- Receivers
  - trigger maps id to <stack of ids>
- Sender can also specify id-stack in packet

- Mechanism:
  - first id used to match trigger
  - rest added to the RHS of trigger
  - recursively continued

Service Composition

- Receiver mediated: R sets up chain and passes id_gif/jpg to sender: sender oblivious
- Sender-mediated: S can include (id_gif/jpg, ID) in his packet: receiver oblivious



Public, Private Triggers

- Servers publish their public ids: e.g., via DNS
- Clients contact server using public ids, and negotiate private ids used thereafter

- Useful:
  - Efficiency -- private ids chosen on “close-by” i3-servers
  - Security -- private ids are shared-secrets

Overview

- Akamai
- I3
- DOA

Architectural Brittleness

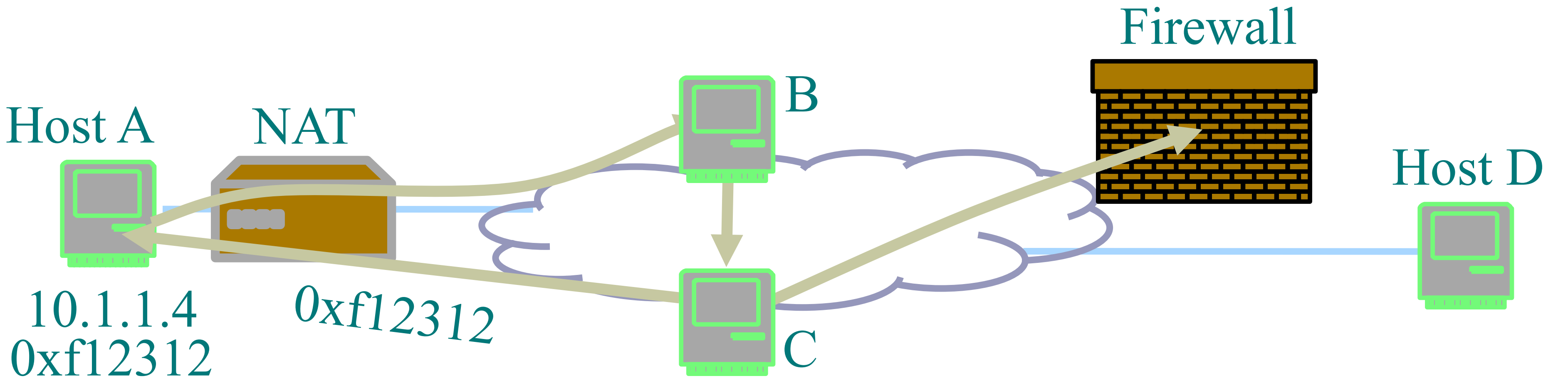
- Hosts are tied to IP addresses
  - Mobility and multi-homing pose problems
- Services are tied to hosts
  - A service is more than just one host: replication, migration, composition
- Packets might require processing at intermediaries before reaching destination
  - “Middleboxes” (NATs, firewalls, …)

Reactions to the Problem

- Purist: can’t live with middleboxes
- Pragmatist: can’t live without middleboxes
- Pluralist (us): purist, pragmatist both right

- DOA goal: Architectural extension in which:
  - Middleboxes first-class Internet citizens
  - Harmful effects reduced, good effects kept
  - New functions arise

DOA: Delegation-Oriented Architecture



- Architectural extension to Internet. Core properties:
  1. Restore globally unique identifiers for hosts
  2. Let receivers, senders invoke (and revoke) off-path boxes: delegation primitive

Naming Can Help

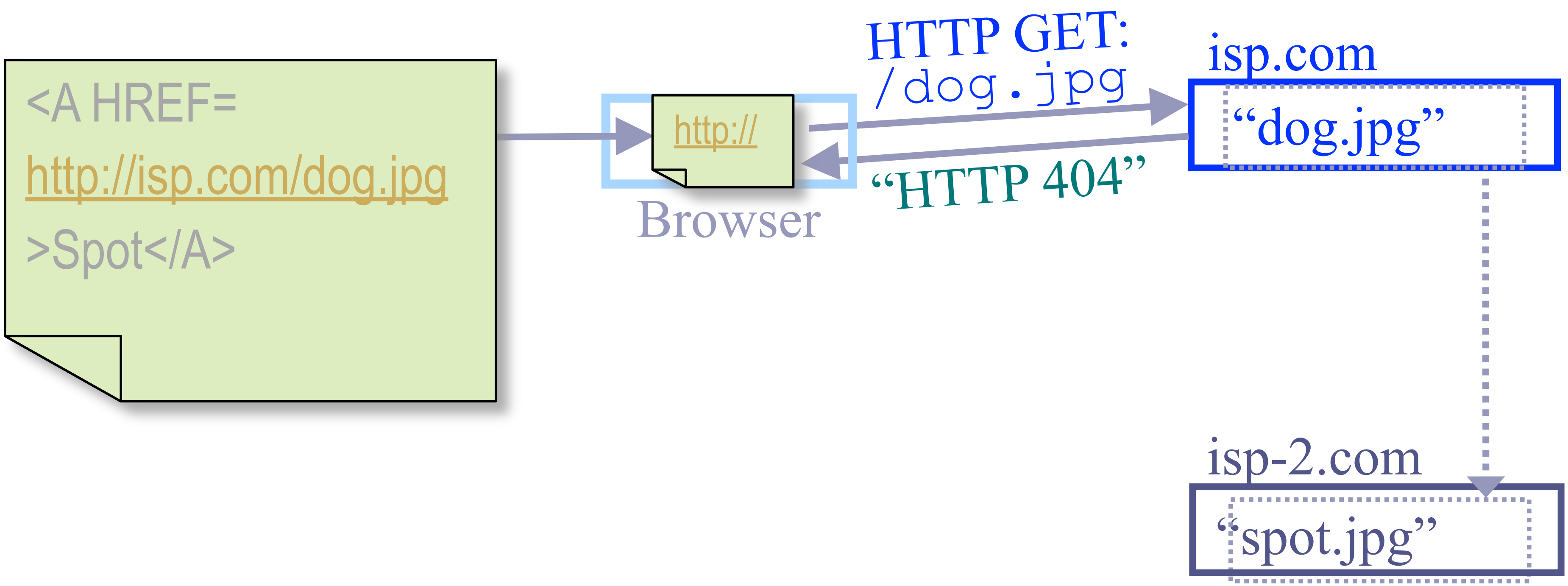
- Thesis: proper naming can cure some ills
  - Layered naming provides layers of indirection and shielding
- Many proposals advocate large-scale, overarching architectural change
  - Routers, end-hosts, services
- Proposal:
  - Changes “only” hosts and name resolution
  - Synthesis of much previous work

Internet Naming is Host-Centric

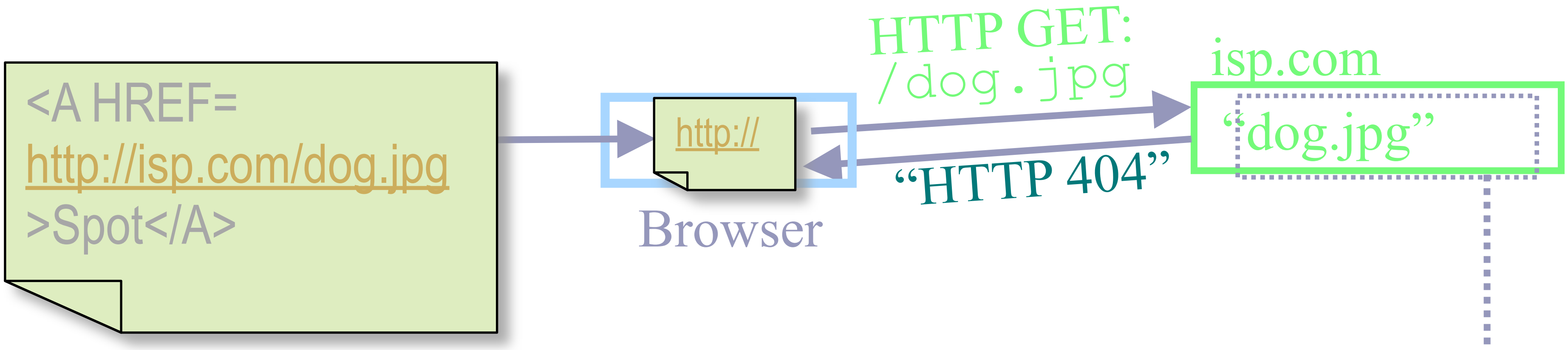
- Two global namespaces: DNS and IP addresses
- These namespaces are host-centric
  - IP addresses: network location of host
  - DNS names: domain of host
  - Both closely tied to an underlying structure
  - Motivated by host-centric application
- Such names constrain movement/replication

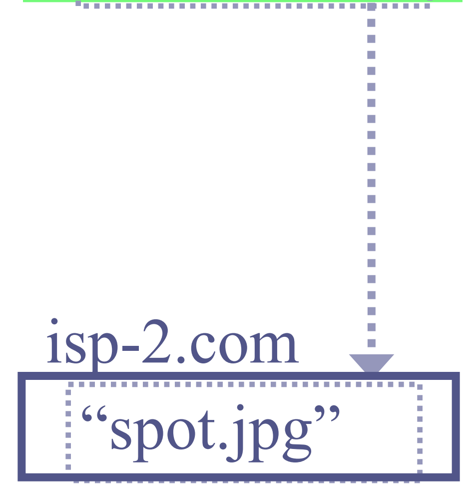
Object Movement Breaks Links

- URLs hard-code a domain and a path



Object Movement Breaks Links, Cont'd





- Today's solutions not stable:
- HTTP redirects
  - need cooperation of original host

Supporting Object Replication

- Host replication relatively easy today
- But per-object replication requires:
  - separate DNS name for each object
  - virtual hosting so replica servers recognize names
  - configuring DNS to refer to replica servers

Key Architectural Questions

- Which entities should be named?
- What should names look like?
- What should names resolve to?

Delegation

- Names usually resolve to "location" of entity
- Packets might require processing at *intermediaries* before reaching destination
- Such processing today violates layering
  - Only element identified by packet's IP destination should inspect higher layers

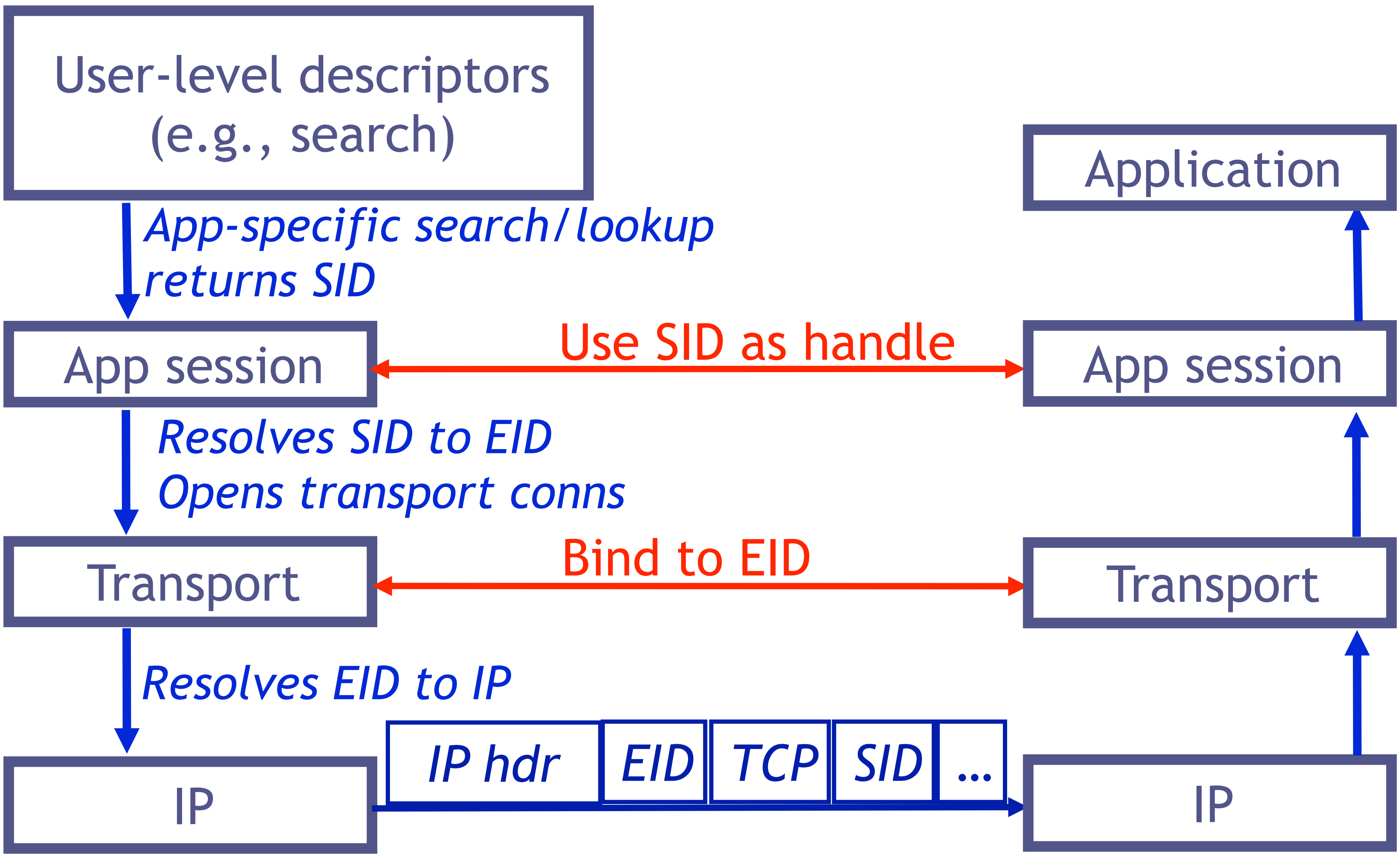
Delegation principle: *A network entity should be able to direct resolutions of its name not only to its own location, but also to chosen delegates*

Name Services and Hosts Separately

- *Service identifiers* (*SIDs*) are host-independent data names
- *End-point identifiers* (*EIDs*) are location-independent host names
- Protocols bind to names, and resolve them
  - Apps should use SIDs as data handles
  - Transport connections should bind to EIDs

Binding principle: *Names should bind protocols only to relevant aspects of underlying structure*

The Naming Layers



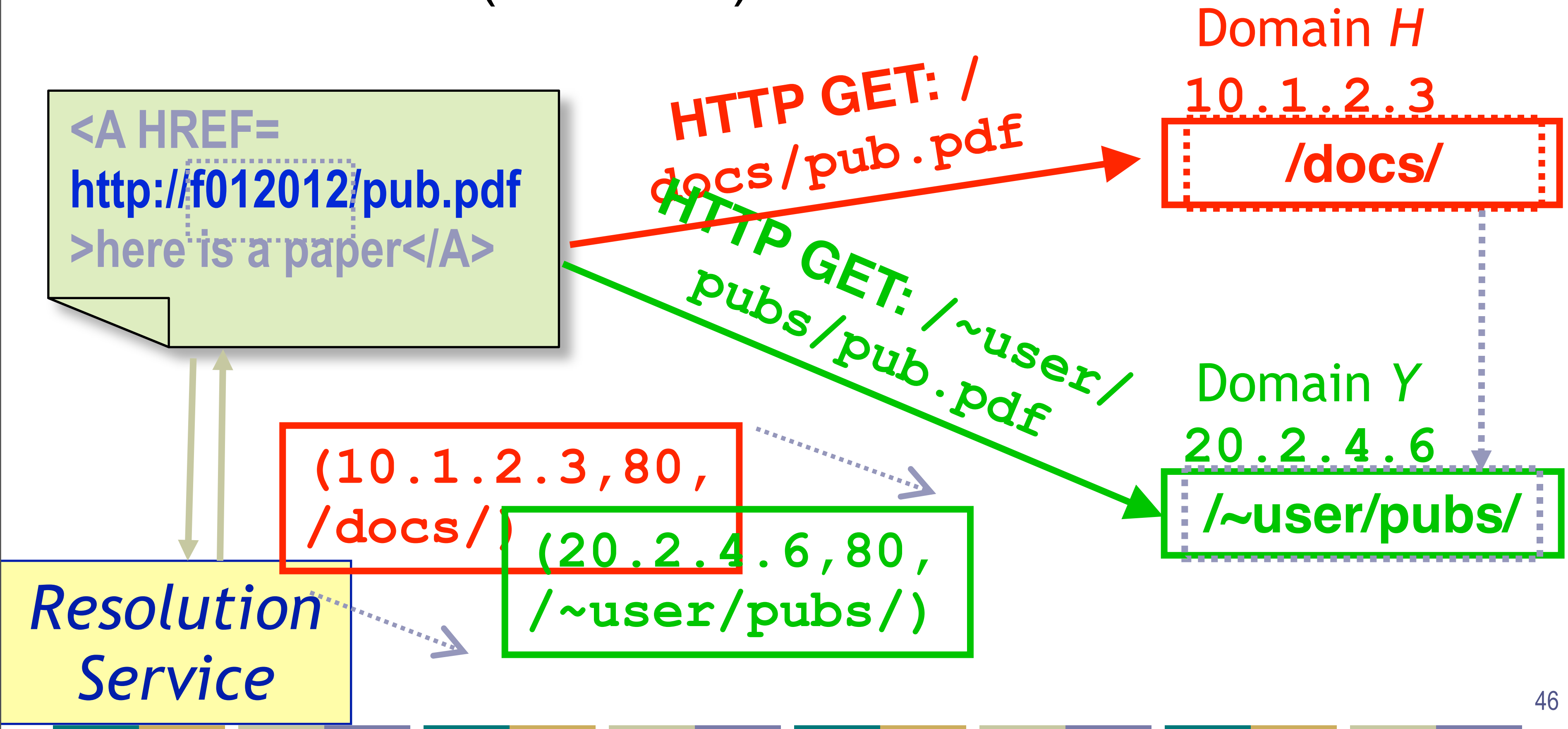
SIDs and EIDs should be *Flat*
0xf436f0ab527bac9e8b100afeff394300

Stable-name principle: ***A stable name should not impose restrictions on the entity it names***

- Flat names impose no structure on entities
  - Structured names stable only if name structure matches natural structure of entities
  - Can be resolved scalably using, e.g., DHTs
- Flat names can be used to name *anything*
  - Once you have a large flat namespace, you never need other global “handles”

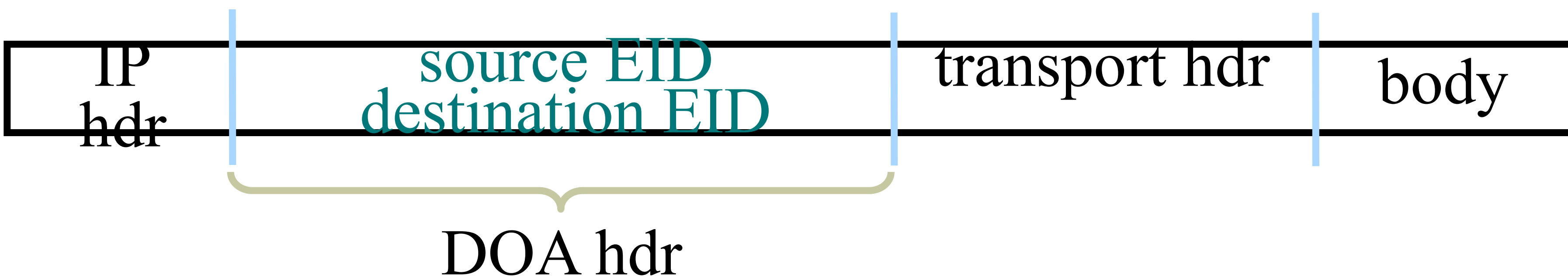
Flat Names Enable Flexible Migration

- SID abstracts all object reachability information
- Objects: any granularity (files, directories)
- Benefit: Links (referrers) don't break



Globally Unique Identifiers for Hosts

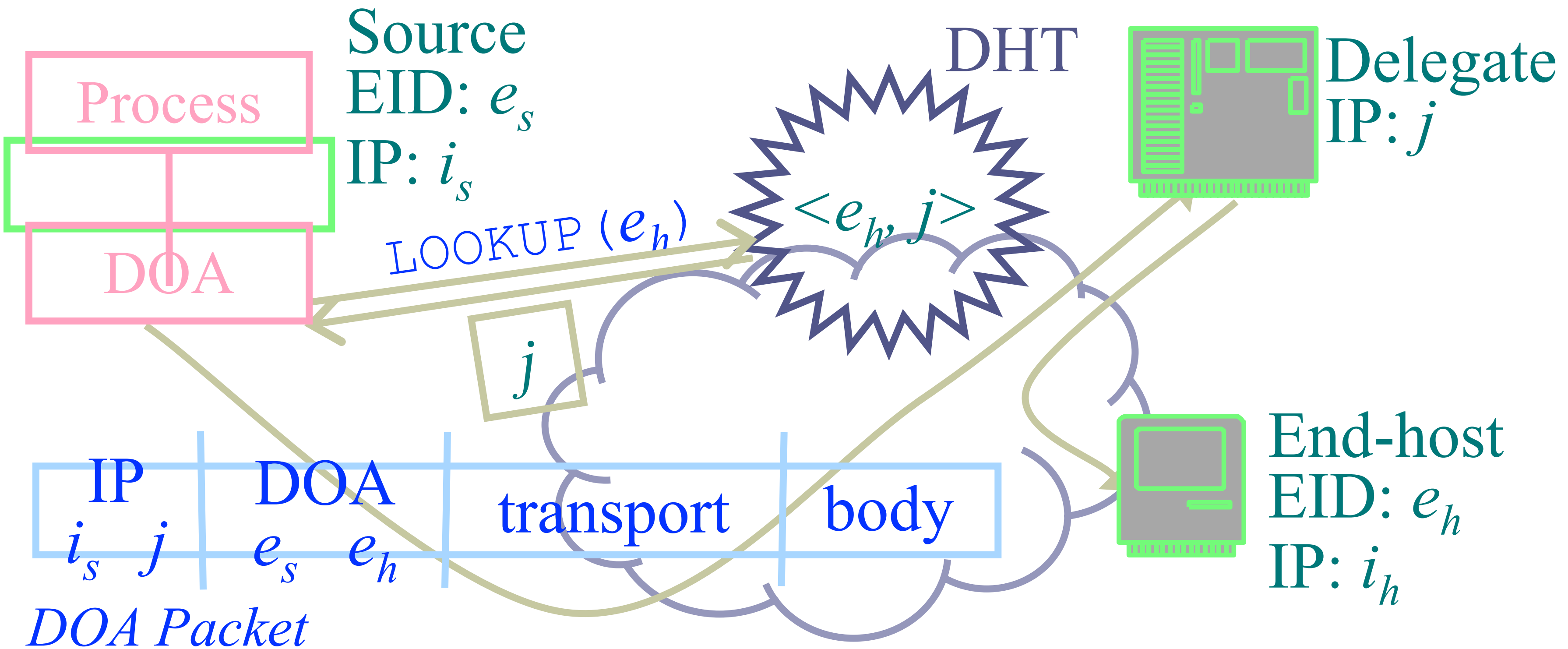
- Location-independent, flat, big namespace
- Hash of a public key
- These are called EIDs (e.g., 0xf12abc…)
- Carried in packets



Delegation Primitive

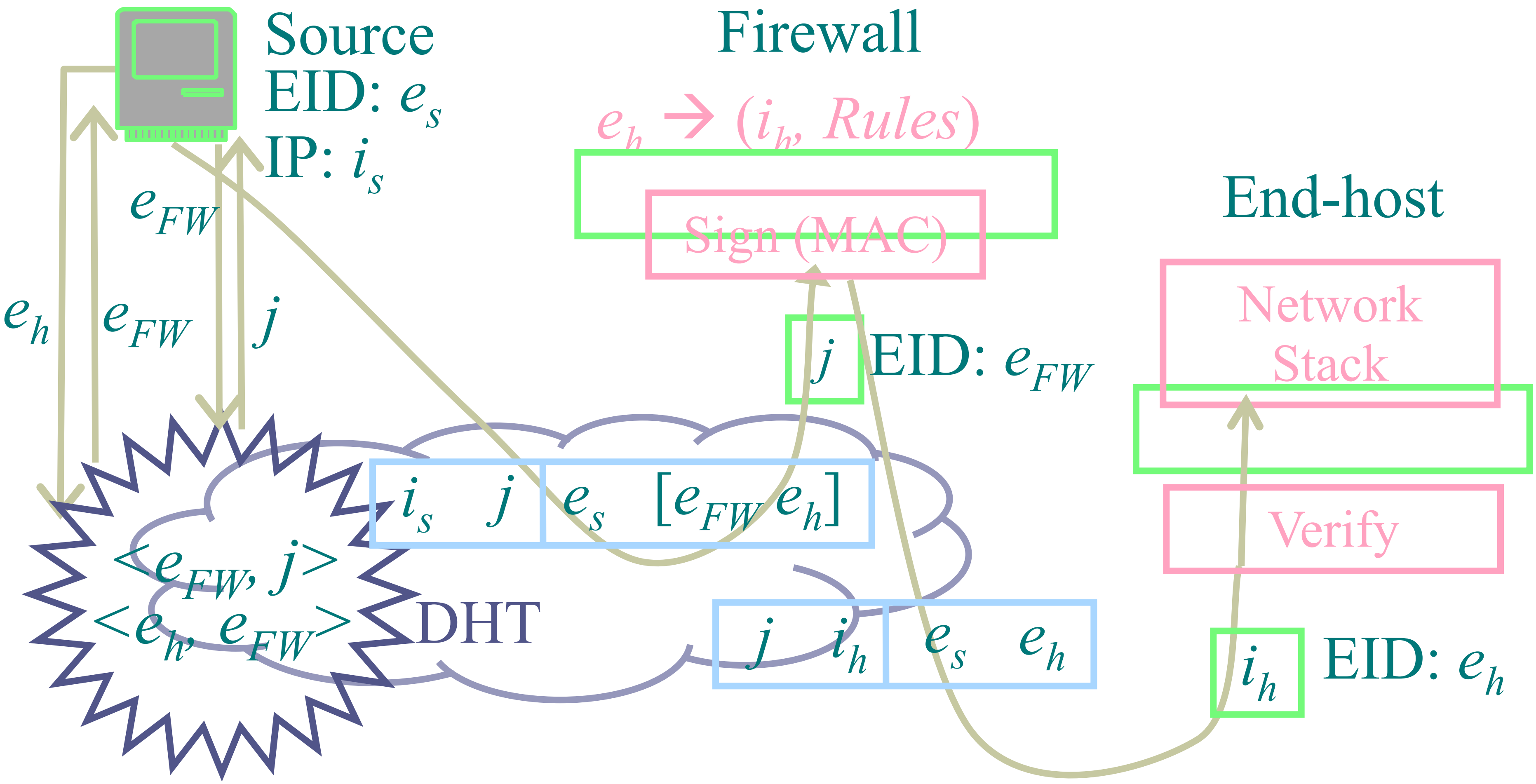
- Let hosts invoke, revoke off-path boxes
- Receiver-invoked: sender resolves receiver's EID to
  - An IP address or
  - An EID or sequence of EIDs
- DOA header has destination stack of EIDs
- Sender-invoked: push EID onto this stack

DOA in a Nutshell



- End-host replies to source by resolving e_s
- Authenticity, performance: discussed in the paper

Off-path Firewall



Off-path Firewall: Benefits

- Simplification for end-users who want it
  - Instead of a set of rules, one rule:
  - “Was this packet vetted by my FW provider?”
- Firewall can be anywhere, leading to:
  - Third-party service providers
  - Possible market for such services
  - Providers keeping abreast of new applications

- DOA enables this; doesn’t mandate it.

Next Lecture

- Data-oriented networking and DTNs
- Required reading:
  - Networking Named Content
  - A Delay-Tolerant Network Architecture for Challenged Internets