Local and Wide-area Server Selection: Techniques and Challenges

Arup Acharya, Anees Shaikh, Renu Tewari

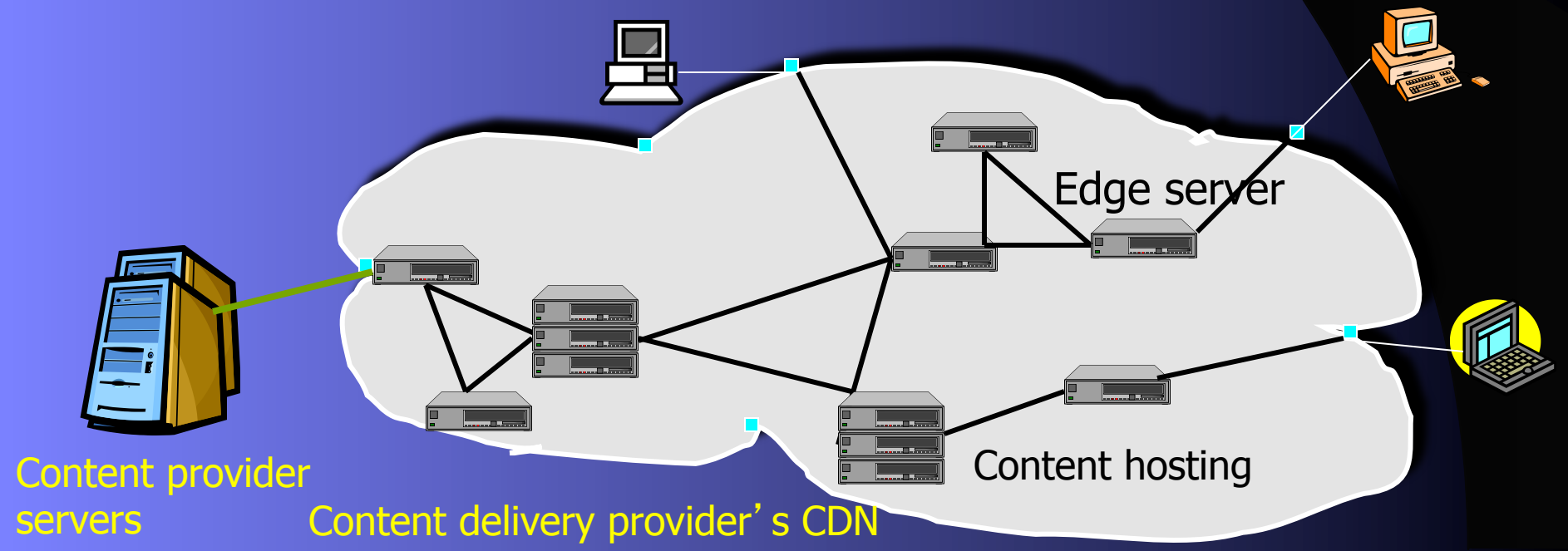
(arup, aashaikh, tewarir)@watson.ibm.com

IBM T.J. Watson Research Center

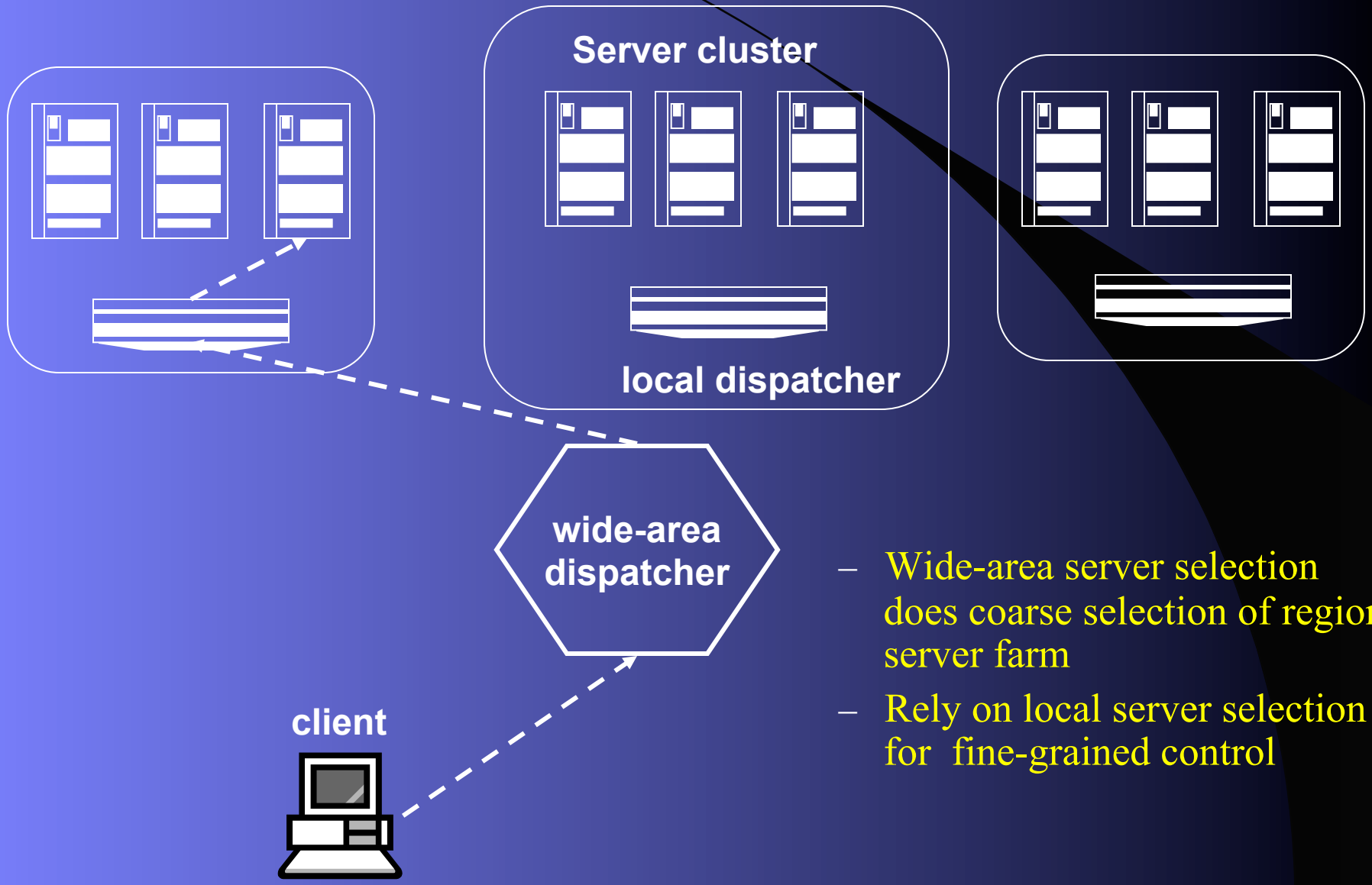
Trends

- Content distribution providers distribute content to the network edges for better scalability, performance, QoS
- Server clusters at large sites and web hosting service providers for better scalability and consolidation

- Content everywhere but not a good way to find it
- *Need to direct client to "best" content location*



Two-level Server Selection



- Wide-area server selection does coarse selection of region/ server farm
- Rely on local server selection for fine-grained control

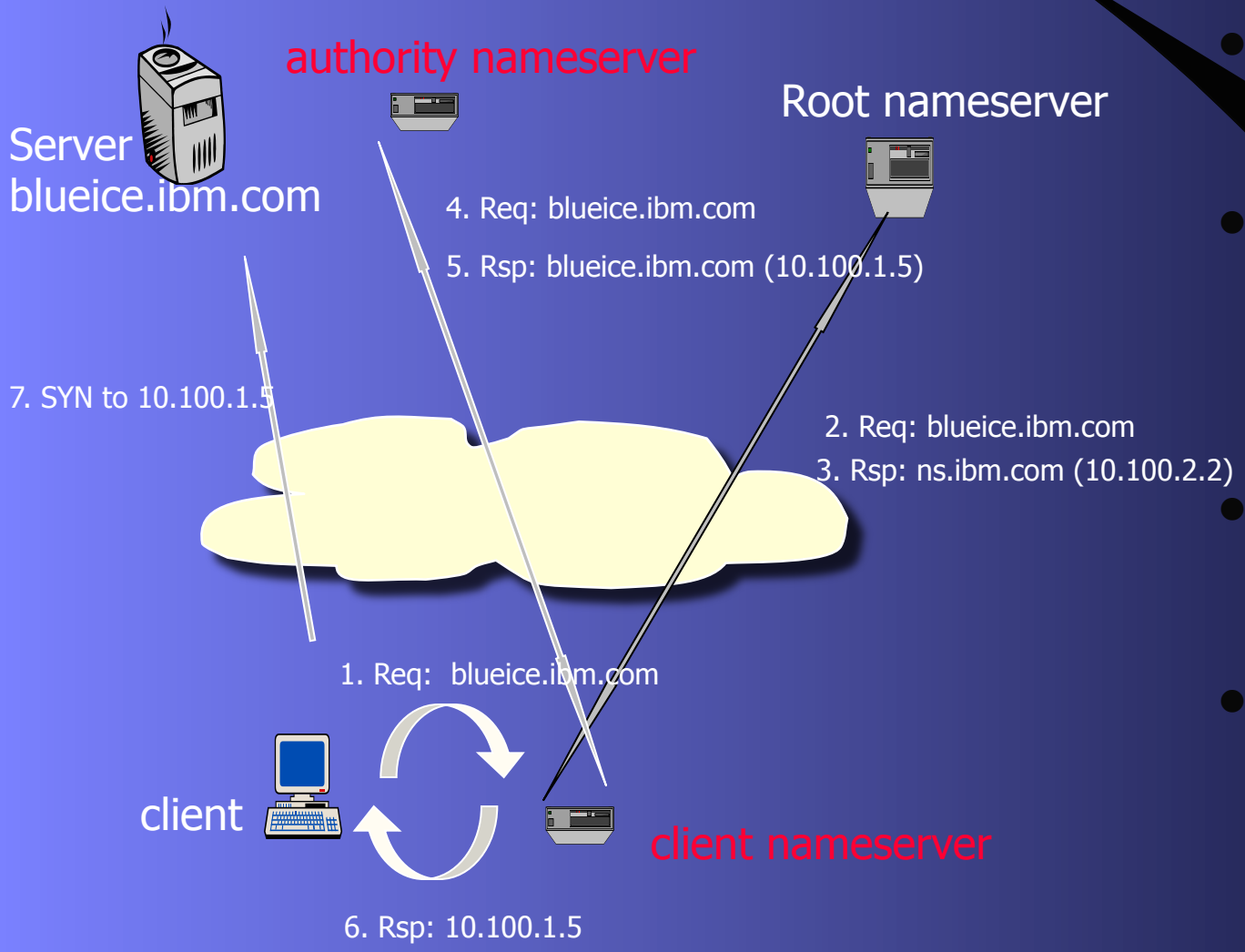
Talk Overview

- Wide-area server selection
  - Techniques and metrics used
  - Issues in DNS-based server selection
  - Evaluation of DNS-based techniques
- Local server selection
  - Overview and new challenges
  - MPLS-based local dispatching (a preview)
- Conclusions

Wide-area Server Selection

- Metrics
  - server load, network conditions
  - client location
  - QoS specification, content requested (images etc.)
- Approaches
  - DNS based
    - nameserver dynamically maps names to addresses
    - transparent, general, widely used
    - Akamai, Cisco, F5 3DNS, Alteon, Digital Island ….
  - HTTP redirect
  - BGP-based (e.g., MSIPR)
  - Application/IP layer anycast

DNS Operation: Overview



- Map names to IP addresses
- Nameserver address configured statically or dynamically (DHCP, PPP)
- Mapping is cached for TTL period
- Remote DNS sees Client NS' Address

Requirements and Challenges

- Dynamic server selection
  - limit client-side DNS caching with low TTL values
  - effects of limiting DNS caching
    - End-user performance decreases (latency increase ~24%)
    - Scalability decreases (nameserver load and network load higher)
- Location (or proximity) based server selection
  - need to identify the client
  - is the client nameserver representative of client location?
    - client's local DNS mis-configured
    - few nameservers across a large ISP
    - nameserver in different AS domain
    - clients and nameservers median cluster size ~8 hops

Factors in End-user Latency

- Name resolution latency
  - varies with level of address caching
    - No cache, nameserver address, server address
- Number of resolutions required
  - number of embedded objects (e.g., images)
  - location of objects (co-located)
  - HTTP version (1.1 with keep-alive)
- Page size and transfer time

Name Resolution Time

DNS Cache level	Median Resolution Time
No local DNS cache	200 ms
75 percentile (popular sites)	3 sec
85 percentile (popular sites)	5 sec
Cached authority nameserver IP	60 ms
Cached server IP	2.3 ms

- Data sets (for hostnames)
  - –Proxy logs: medium-sized ISP, single pop
  - –Popular sites (Media Metrix Top 50)
- Measured name resolution time from multiple sites
  - –Massachusetts, NY, Michigan, California

End-user Latency

Page Statistics	Mean Value
Page download time (popular sites)	6.3 secs
Total Page size	30.9 KB
# of objects per page	35 (median 25)

Cache Level	Resolution Overhead
No caching (25 objects 200 ms. each)	5 sec.
NS address cached (25 objects. 60 ms)	1.5 sec.

Summary

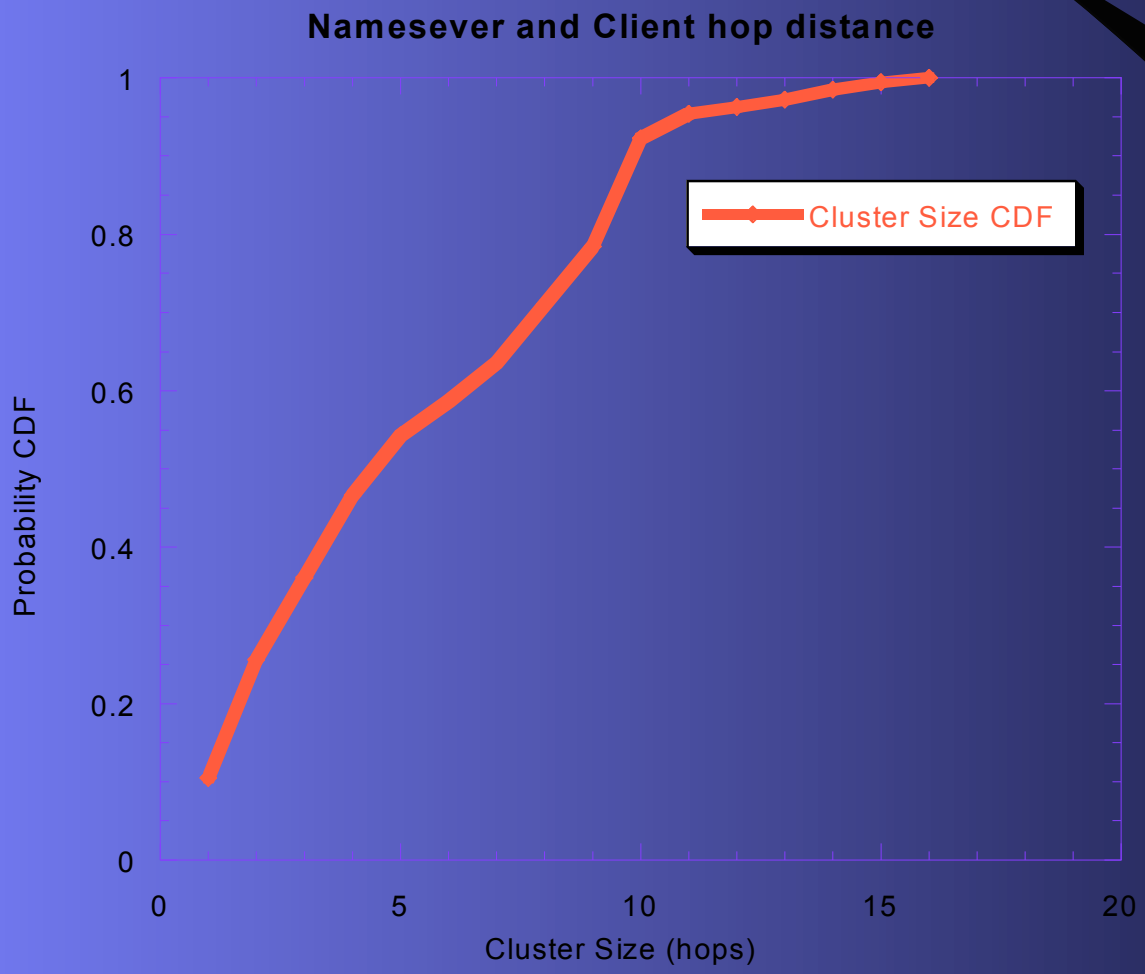
Complete lookup per object costly

Higher TTL preferred

Embedded objects located on same server

HTTP Keep-Alive

Client Proximity Mismatch



IGS DNS and HTTP logs

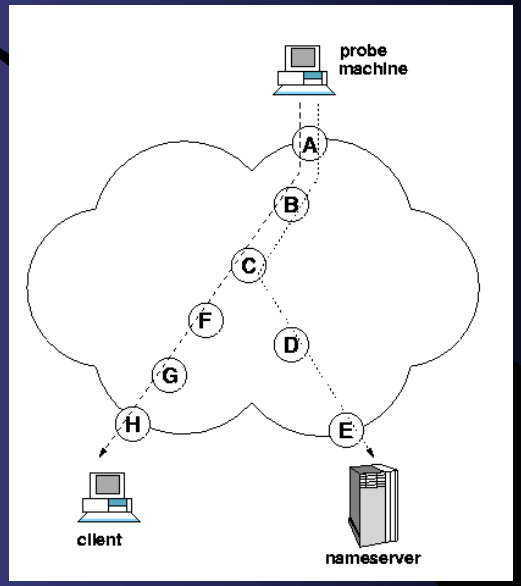
Cluster Size: Distance from first common ancestor

Average: 5.7
Max: 16
Median: 5 hops
65 percentile: 8 hops

Optimistic since removed mismatches

Dial-Up Proximity Mismatch

- Direct distance:
  - Mean 7.6 hops; median 8 hops
  - Avg RTT: 234 ms (first hop 188 ms)
- Cluster sizes
  - Median: 8 hops (NY probe),
- Common/disjoint path ratio
  - High ratio ⇒ long common path
  - Median ratio: 0.25 (NY probe)



ISP accounts	9 national retail; 2 free
Unique nameserver addrs	54
Nameserver addrs per ISP	2-15; avg 7.4

Summary of DNS Based Server Selection

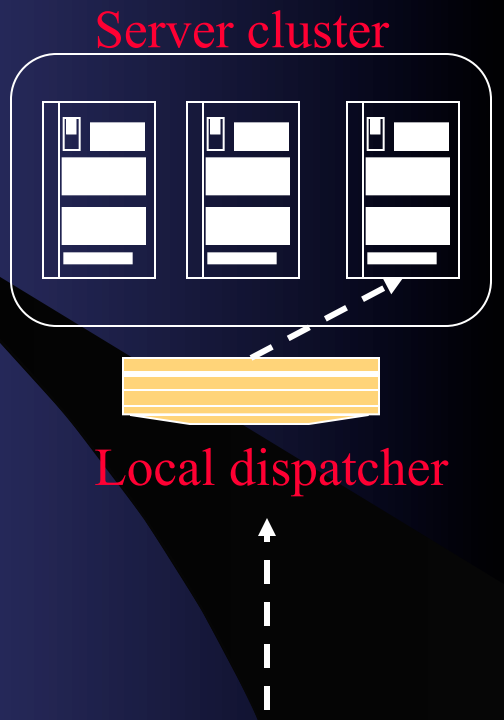
- Limiting caching
  - Increases resolution latency by two orders of magnitude
  - Increase end-user latency 24%
    - Larger TTL better TTL
    - Embedded objects co-located on same server
    - HTTP Keep-Alive
- Client/local nameserver proximity
  - Clients often far from their nameservers (8 hops or more)
  - Correlation between delay to client and local nameserver: positive, but small
  - Proposal: include client address in DNS query

Talk Overview

- Wide-area server selection
  - Techniques and metrics used
  - Issues in DNS-based server selection
  - Evaluation of DNS-based techniques
- Local server selection
  - Overview and new challenges
  - MPLS-based local dispatching (a preview)
- Conclusions

Local Server Selection

- Local dispatcher for a cluster of servers
  - Arrowpoint (Cisco), Nortel, F5, Alteon, Foundry, IBM
- L4 switches
  - use TCP/IP header for simple client based dispatch
  - provide session persistence, QoS
  - use server load information for load balancing
- L5-L7 switches
  - TCP termination for content based routing
  - dispatching based on HTTP headers, SSL id, cookies, tags
    - connection splicing (LD overhead per connection)
    - connection handoff (modify server kernel)
- Our goals
  - a common solution for all web-switching functions
  - avoid the TCP termination bottleneck
  - use "commodity" hardware



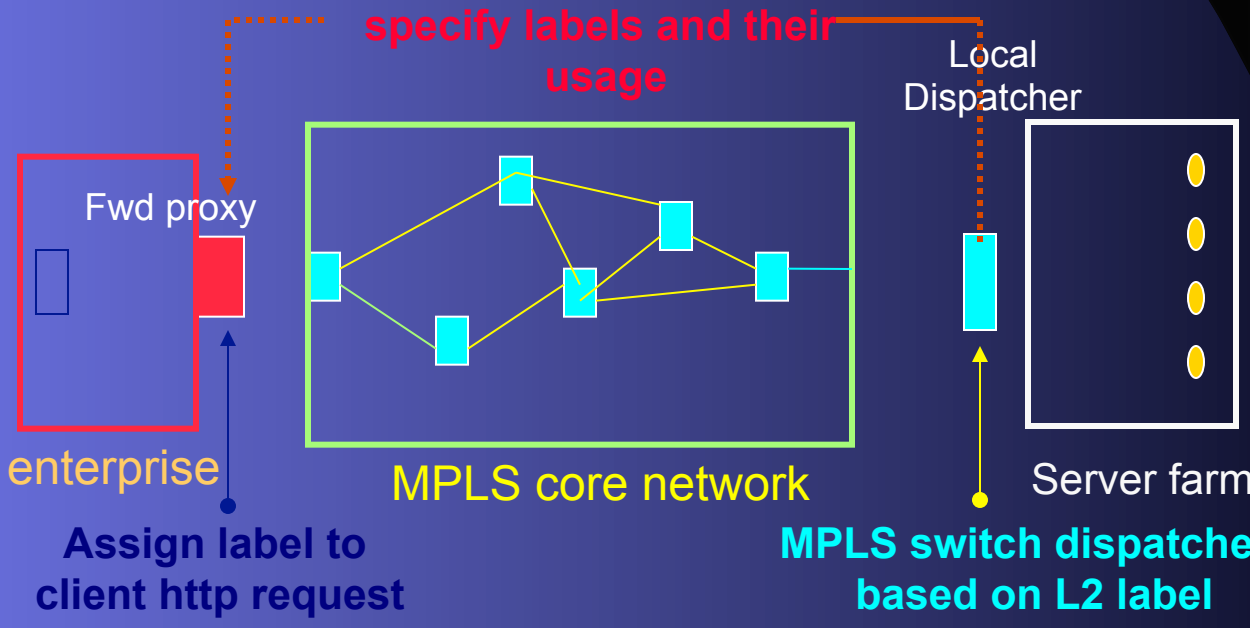
Open Questions

- Can local dispatching be done at layer 2 switching speeds?
  - –replace with MPLS switch
- Can we replace a web switch (L7) with commodity h/w?
  - –conjecture: MPLS switches (L2/ L3) will be "commodity"
- Can we provide the same functionality (content routing, load balancing, session affinity, QoS) ?
  - –application layer information encoded in layer 2/3/4 headers
  - –L3/routing semantics applied to L2 labels

Yes with MPLS?

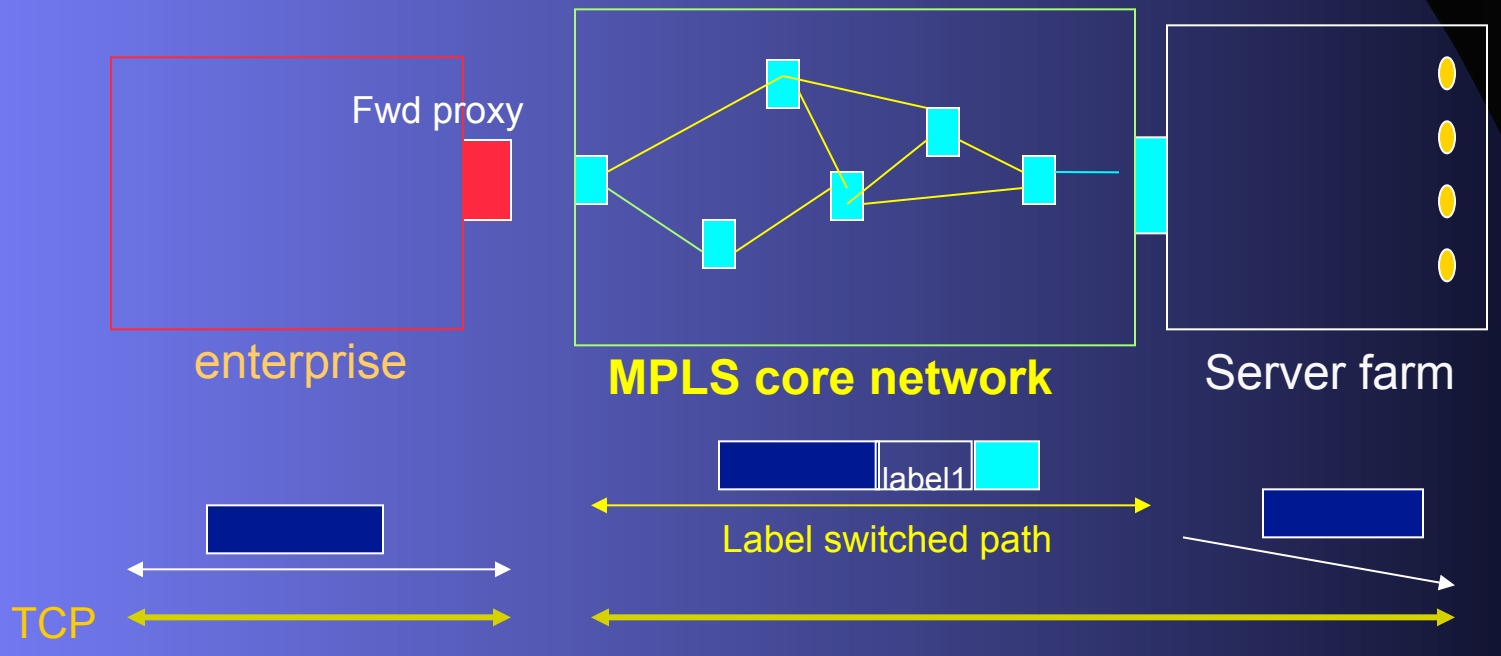
Proposed Approach

- Use MPLS label stacking feature
- Encode L4-L7 semantics onto MPLS inner label
  - Outer label used for routing
- No TCP termination at dispatcher for content routing, load balancing, affinity
  - Out of path return allowed
- Maintain control connection for label distribution with forward proxies



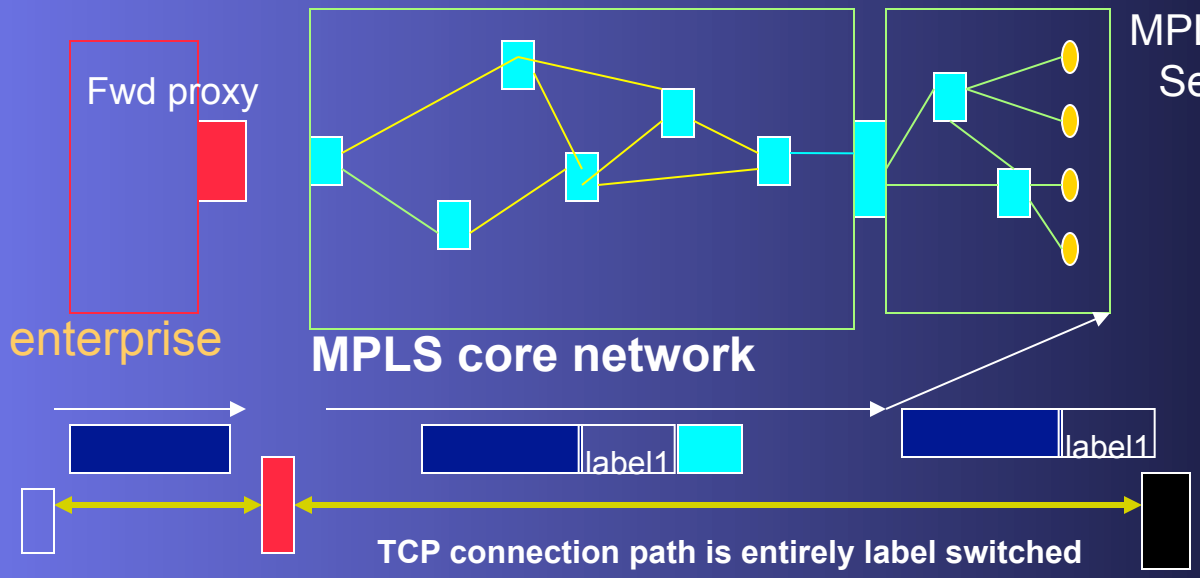
MPLS-based Content Routing

- Local dispatcher communicates semantics of (inner) label to fwd proxy
  - Label1 → www.cnn.com/ headlinenews/
  - Label2 → www.cnn.com/ fn/
- Fwd proxy assigns inner label based on URL
  - Outer label assigned based on route within the core
  - Dispatcher routes to the right server based on inner label



MPLS-based Load Balancing

- Local dispatcher communicates set of labels to fwd proxies
  - label1, label2, label3
  - assign weights to labels (w1, w2, w3)
  - different label sets to different proxies
- Fwd proxy assigns inner label per connection based on weights
- Dispatcher can re-map labels for temporary load imbalance
- Redistribute labels/weights for long term changes
- If the server farm network MPLS enabled, then entire path is label switched



Label Distribution Scenarios

- Control connection between dispatcher and proxies
  - Content based routing
    - table of URL/ label mappings
  - Load balancing
    - set of labels and weights, proxy round-robins
  - Affinity
    - set of labels, proxy assigns same label to all client requests for a session
  - Service differentiation
    - sets of labels, one set per class (gold/…/..)
    - assumes pre-defined service agreement
- Dispatcher populates label/server mapping table at layer2

Deployment Issues

- What is the incentive for proxies to participate?
  - Benefits to dispatcher is better performance
  - Proxies could belong to same Web hosting/ISP providers (mutual benefit)
  - Profit sharing between ISP proxies and the content hosting companies
- How many proxies need to participate?
  - #enterprises/ISP proxies accessing a given server is very small (~250) even for a large client access base (~60 million)
  - limited participation adequate to derive large performance gains (?)
- If proxy and dispatcher are in different MPLS domains, how label stacking will work needs to be resolved

Conclusions

- DNS-based wide-area server selection promising but flawed
  - Need larger TTL for scalability and client latency
  - Need to solve proximity mismatch issues
- Label encoding techniques for local server selection
  - MPLS inner label encodes higher layer information
  - Layer 7 switching at price/performance of layer 2
- Open Research Issue:
  - Can label encoding work in wide-area server selection too?