Network Applications: Load Balancing among Homogeneous Servers; Application Overlays (P2P)

Qiao Xiang

https://qiaoxiang.me/courses/cnns-xmuf22/index.shtml

10/13/2022

This deck of slides are heavily based on CPSC 433/533 at Yale University, by courtesy of Dr. Y. Richard Yang.

Outline

- Admin and recap
- Multi-servers
  - Basic issues
  - Load direction
    - DNS (IP level)
    - Load balancer/smart switch (sub-IP level)
- Application overlays (distributed network applications) to
  - scale bandwidth/resource (BitTorrent)

Admin

- Lab assignment two due today
- Lab assignment three posted today
  - Due on Nov. 8th

Recap: Operational Quantities

- T: observation interval
- Bi: busy time of device i
- i = 0 denotes system

Ai: # arrivals to device i

Ci: # completions at device i

$$\text{arrival rate } \lambda_i = \frac{A_i}{T}$$

$$\text{Throughput } X_i = \frac{C_i}{T}$$

$$\text{Utilization } U_i = \frac{B_i}{T}$$

$$\text{Mean service time } S_i = \frac{B_i}{C_i}$$

Recap: Operational Laws

- Utilization law: U = XS
- Forced flow law: Xi = Vi X
- Bottleneck device: largest Di = Vi Si
- Little's Law: Qi = Xi Ri
- Bottleneck bound of interactive response (for the given closed model):

$$X(N) \le \min\{\frac{1}{D_{\max}}, \frac{N}{D+Z}\}$$

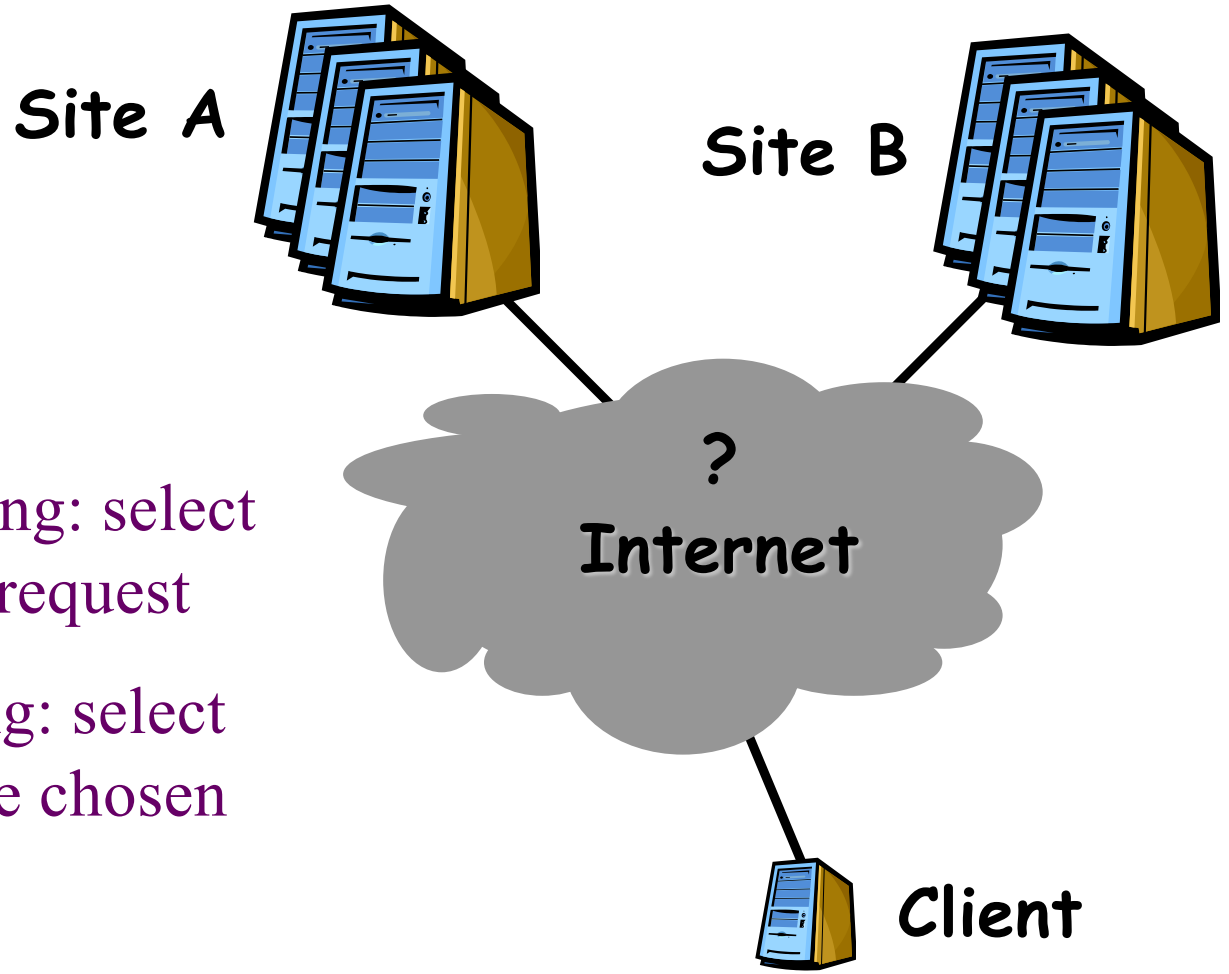
$$R(N) \ge \max\{D, ND_{\max} - Z\}$$

Discussion: Key Technical Challenges in Using Multiple Servers

Outline

- Recap
- Single network server
- Multiple network servers
  - Why multiple servers
  - *Multiple servers basic mechanism: request routing*

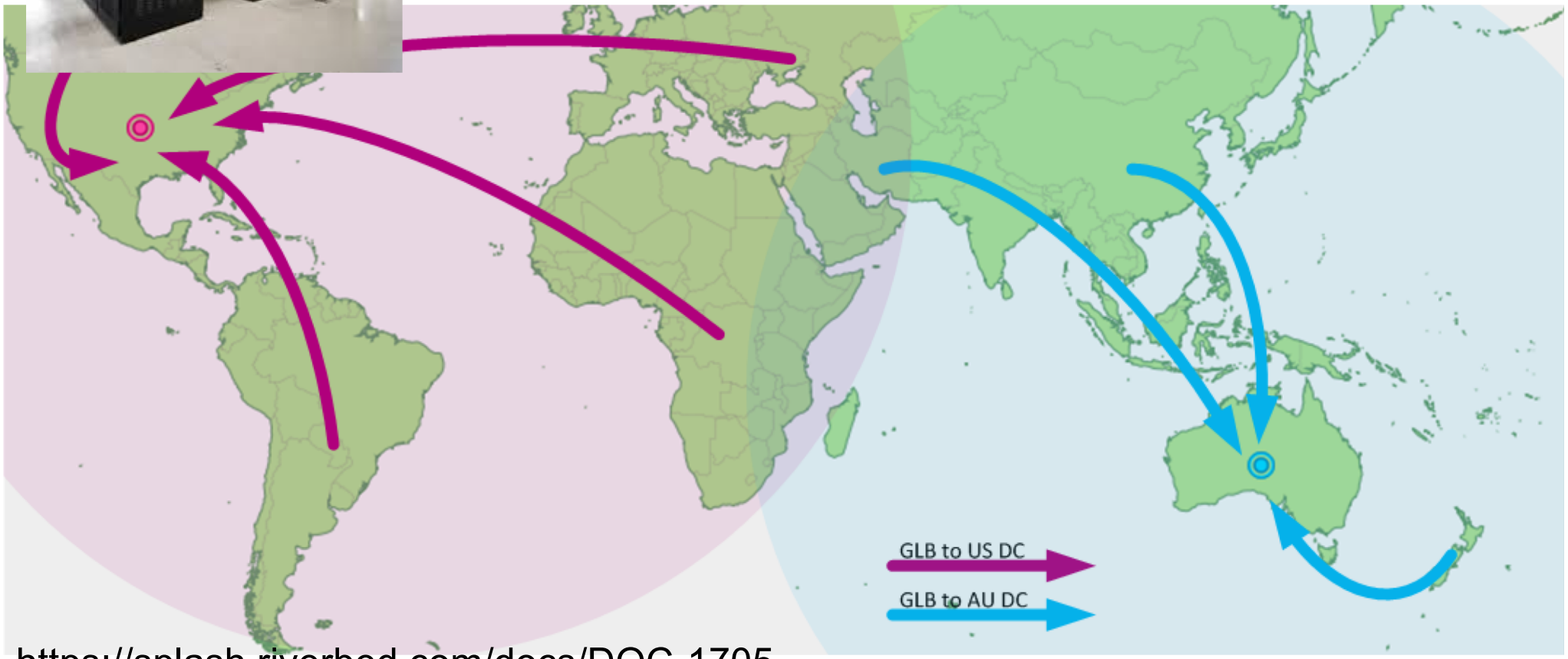
Request Routing: Overview



- Global request routing: select a server site for each request

- Local request routing: select a specific server at the chosen site

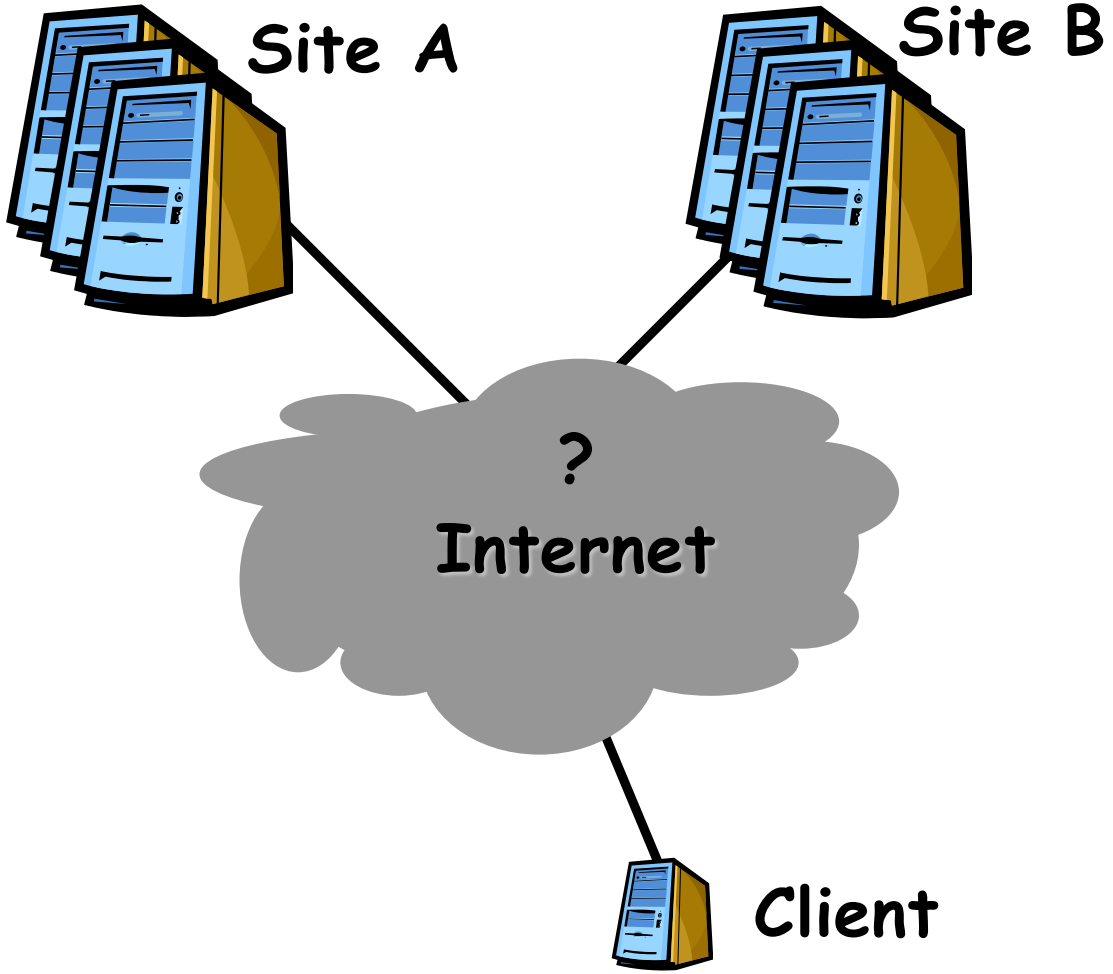
Request Routing: Overview



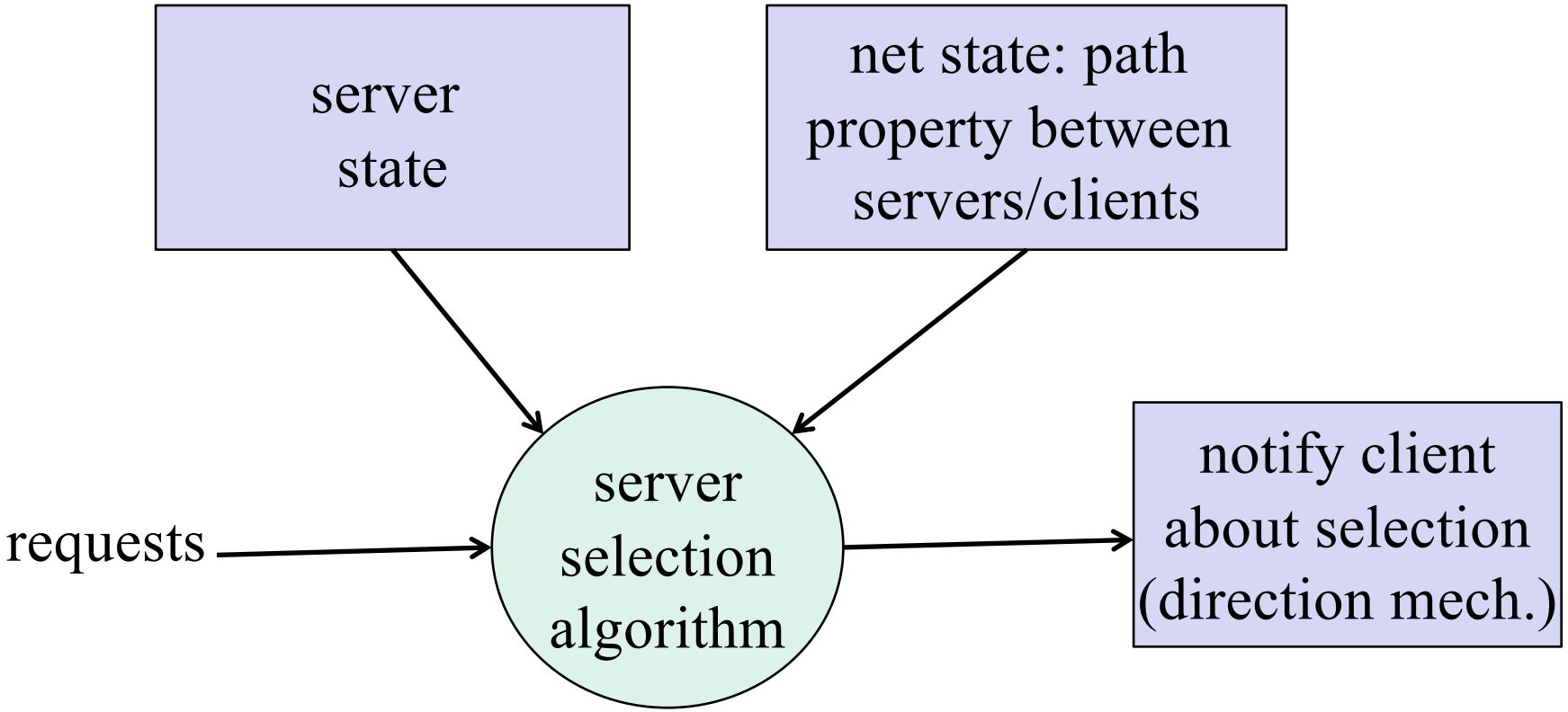
https://splash.riverbed.com/docs/DOC-1705

Request Routing: Basic Architecture

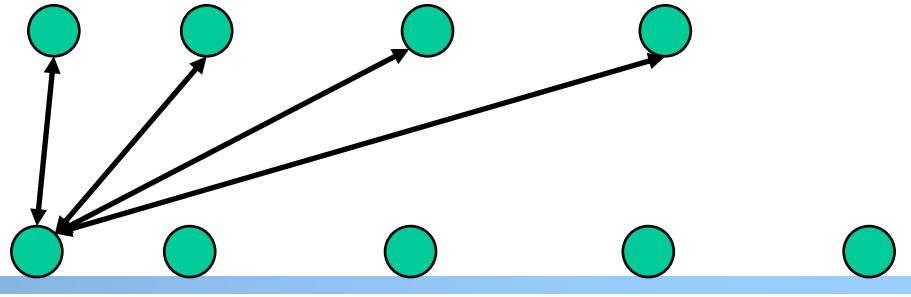
- Major components
  - Server state monitoring
    - Load (incl. failed or not); what requests it can serve
  - Network path properties estimation
    - E.g., bw, delay, loss, network cost between clients and servers
  - Server assignment alg.
  - Request direction mechanism



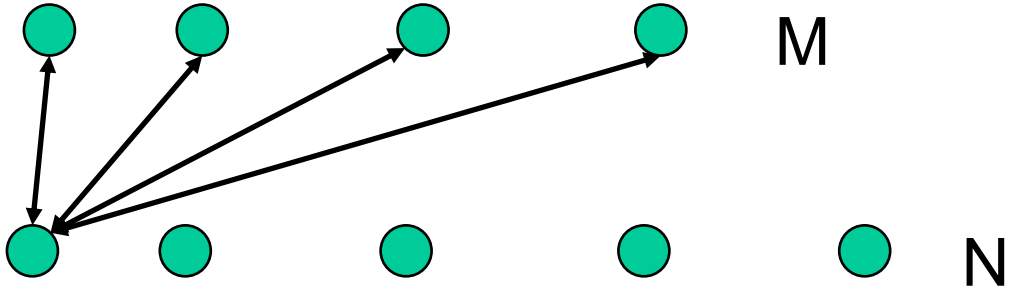
Request Routing: Basic Architecture



Network Path Properties



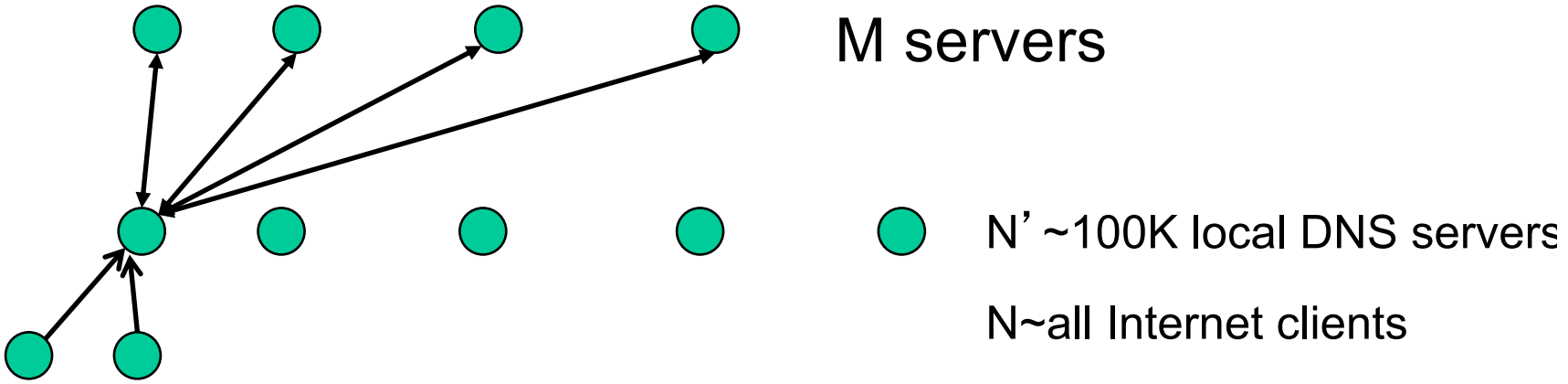
- Why is the problem difficult?
  - Scalability: if do measurements, complete measurements grow with N * M, where
    - N is # of clients
    - M is # of servers



  - Complexity/feasibility in computing path metrics

Network Path Properties: Improve Scalability

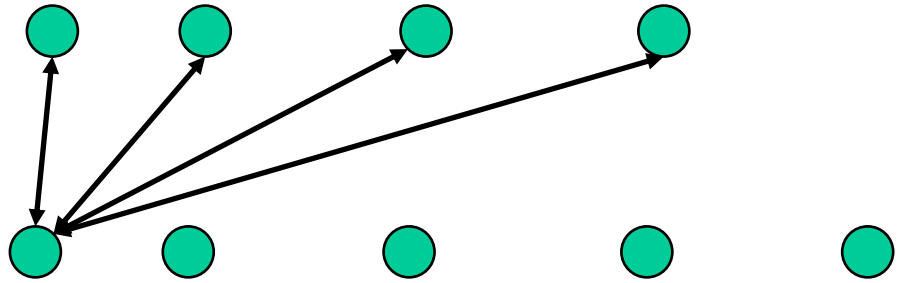
- Aggregation:
  - merge a set of IP addresses (reduce N and M)
    - E.g., when computing path properties, aggregates all clients sharing the same local DNS server



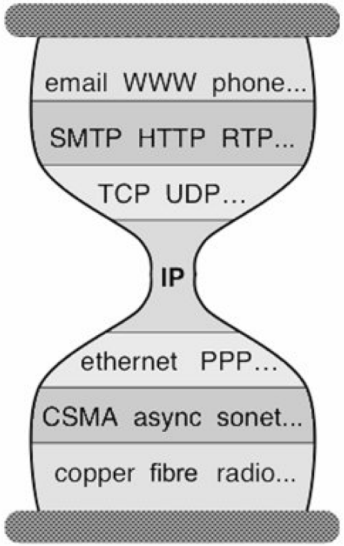
- Sampling and prediction
  - Instead of measuring N*M entries, we measure a subset and predict the unmeasured paths
  - We will cover it later in the course

Server Assignment

- Why is the problem difficult?
  - What are potential problems of just sending each new client to the lightest load server?

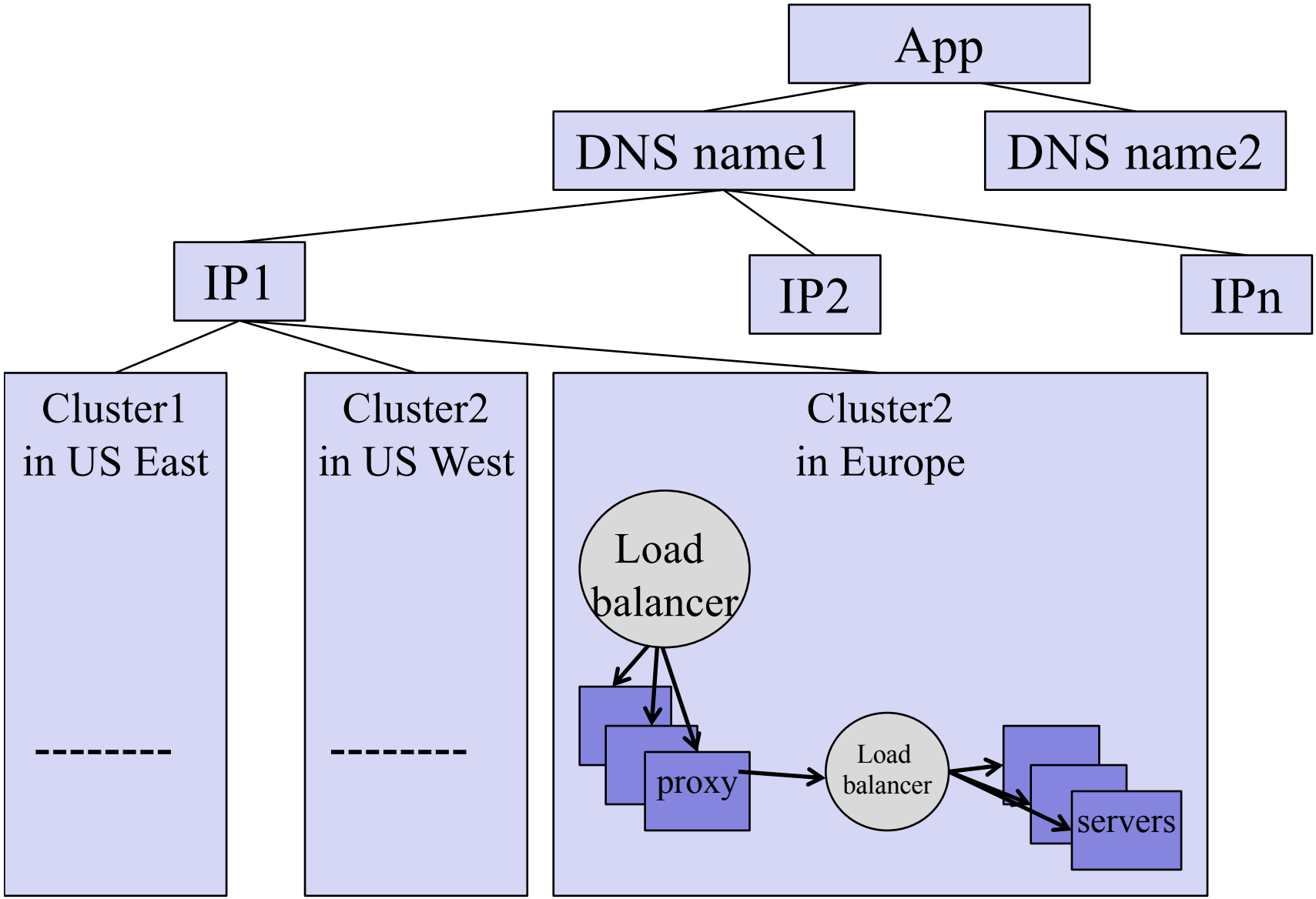


Client Direction Mechanisms

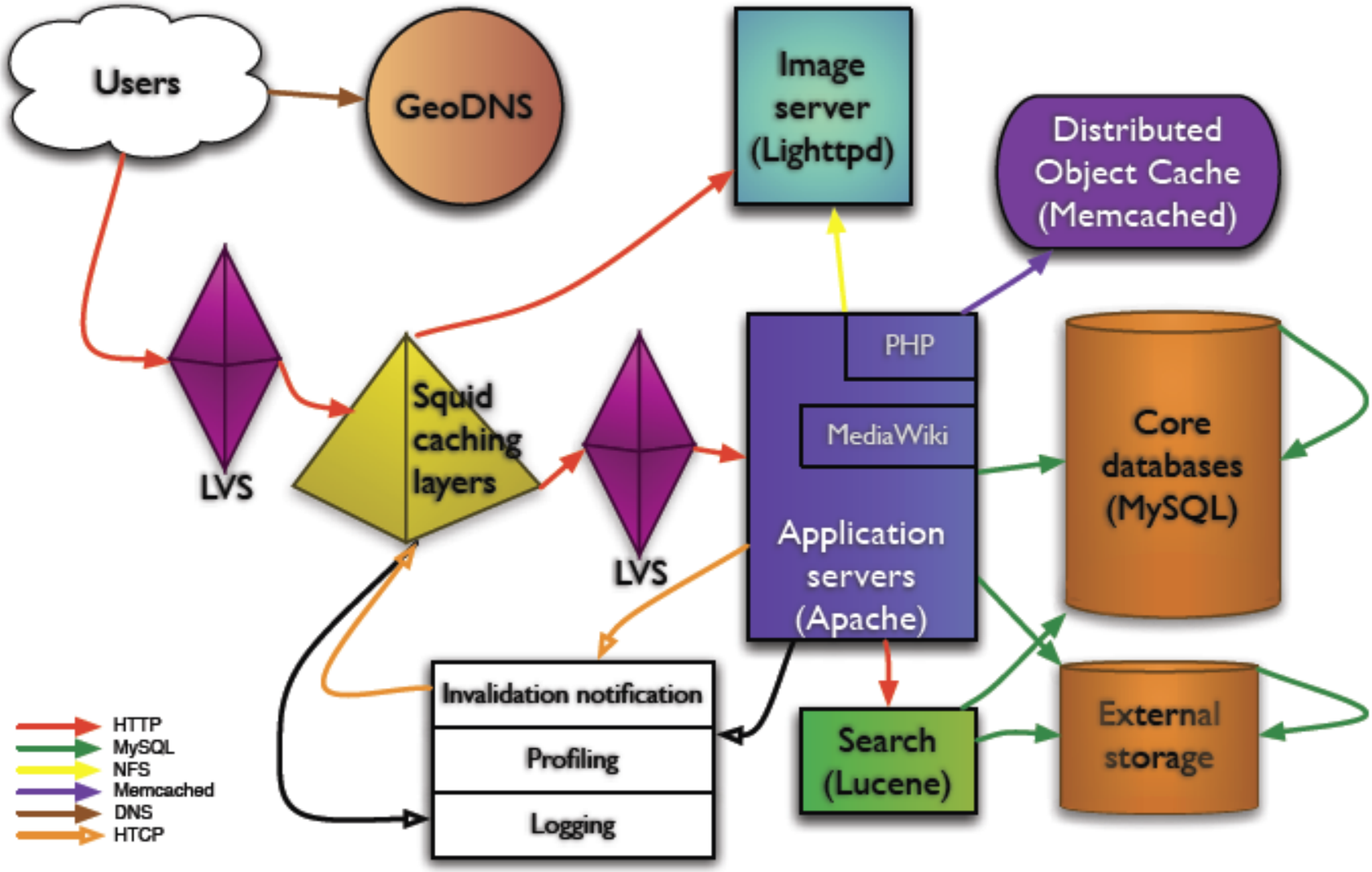


- Key difficulty
  - May need to handle a large of clients
- Basic types of mechanisms
  - Application layer, e.g.,
    - App/user is given a list of candidate server names
    - HTTP redirector
  - DNS: name resolution gives a list of server addresses
  - IP layer: Same IP address represents multiple physical servers
    - IP anycast: Same IP address shared by multiple servers and announced at different parts of the Internet. Network directs different clients to different servers
    - Smart-switch indirection: a server IP address may be a virtual IP address for a cluster of physical servers

Direction Mechanisms are Often Combined



Example: Wikipedia Architecture



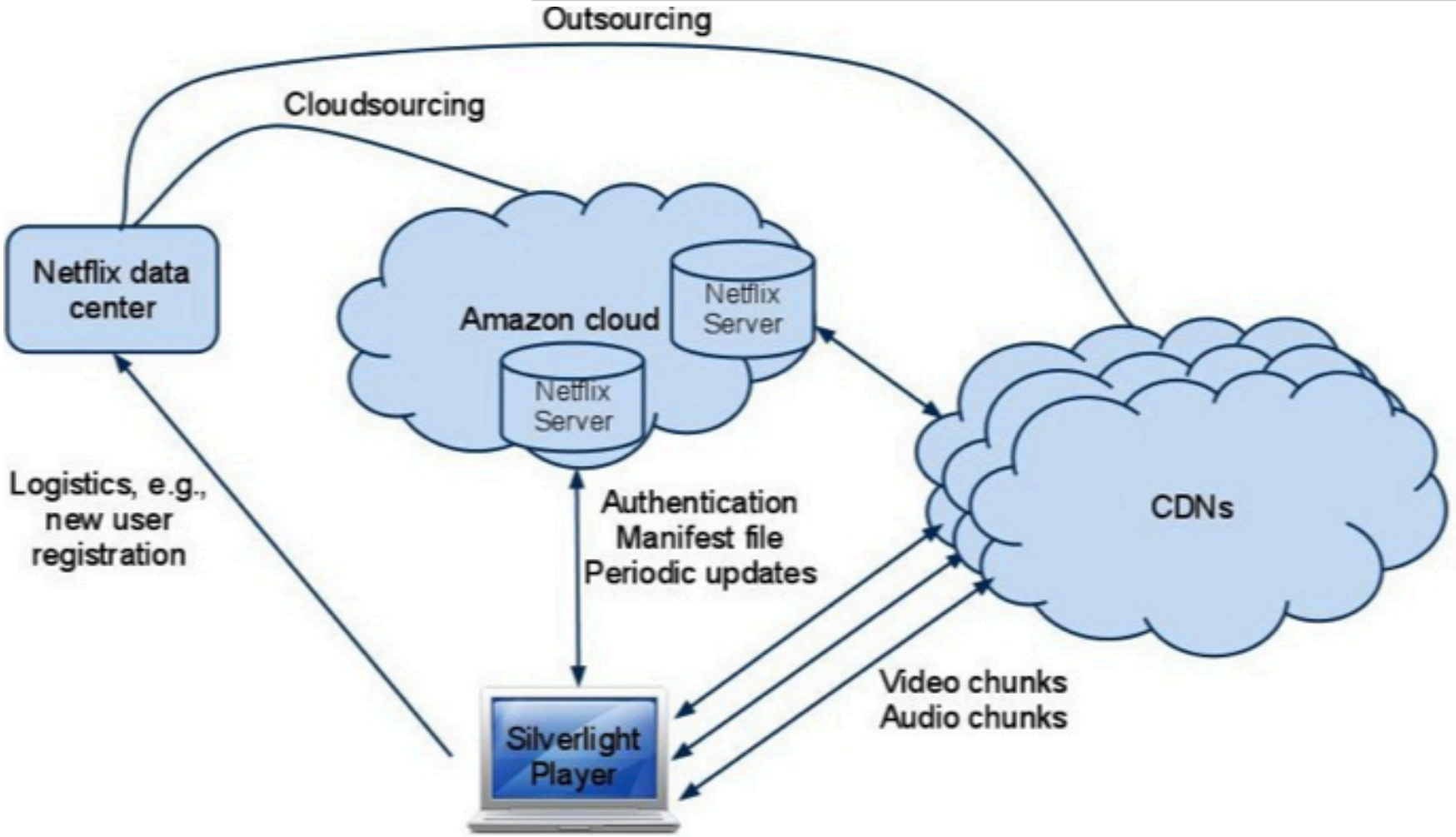
http://wikitech.wikimedia.org/images/8/81/Bergsma_-_Wikimedia_architecture_-_2007.pdf

Outline

- Recap
- Single, high-performance network server
- Multiple network servers
  - Why multiple servers
  - Request routing mechanisms
    - Overview
    - *Application-layer*

Example: Netflix

Hostname	Organization
www.netflix.com	Netflix
signup.netflix.com	Amazon
movies.netflix.com	Amazon
agmoviecontrol.netflix.com	Amazon
nflx.i.87f50a04.x.lcdn.nflximg.com	Level 3
netflix-753.vo.llnwd.net	Limelight
netflix753.as.nflximg.com.edgesuite.net	Akamai



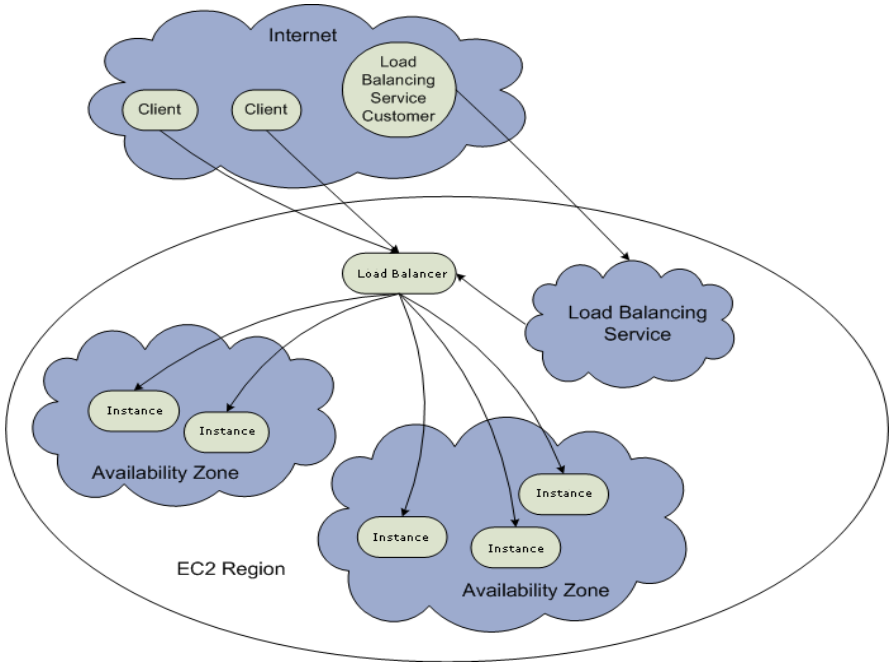
Example: Netflix Manifest File

- Client player authenticates and then downloads manifest file from servers at Amazon Cloud

```
<nccp:cdns>
    <nccp:cdn>
        <nccp:name>level3</nccp:name>
        <nccp:cdnid>6</nccp:cdnid>
        <nccp:rank>1</nccp:rank>
        <nccp:weight>140</nccp:weight>
    </nccp:cdn>
    <nccp:cdn>
        <nccp:name>limelight</nccp:name>
        <nccp:cdnid>4</nccp:cdnid>
        <nccp:rank>2</nccp:rank>
        <nccp:weight>120</nccp:weight>
    </nccp:cdn>
    <nccp:cdn>
        <nccp:name>akamai</nccp:name>
        <nccp:cdnid>9</nccp:cdnid>
        <nccp:rank>3</nccp:rank>
        <nccp:weight>100</nccp:weight>
    </nccp:cdn>
</nccp:cdns>
```

Example: Amazon Elastic Cloud 2 (EC2) Elastic Load Balancing

- Use the *create-load-balancer* command to create an Elastic Load Balancer.
- Use the *register-instances -with-load-balancer* command to register the Amazon EC2 instances that you want to load balance with the Elastic Load Balancer.
- Elastic Load Balancing automatically checks the health of your load balancing Amazon EC2 instances. You can optionally customize the health checks by using the *configure-healthcheck* command.
- Traffic to the DNS name provided by the Elastic Load Balancer is automatically distributed across healthy Amazon EC2 instances.



http://docs.aws.amazon.com/ElasticLoadBalancing/latest/DeveloperGuide/elastic-load-balancing.html

Details: Create Load Balancer

The operation returns the DNS name of your LoadBalancer. You can then map that to any other domain name (such as www.mywebsite.com)
(how?)

```
%aws elb create-load-balancer --load-
balancer-name my-load-balancer --listeners
"Protocol=HTTP,LoadBalancerPort=80,InstanceP
rotocol=HTTP,InstancePort=80" --
availability-zones us-west-2a us-west-2b
```

Result:

{ "DNSName": "my-load-balancer-123456789.us-west-2.elb.amazonaws.com”}

http://docs.aws.amazon.com/cli/latest/reference/elb/create-load-balancer.html

Details: Configure Health Check

The operation configures how instances are monitored, e.g.,

```
%aws elb configure-health-check --load-
balancer-name my-load-balancer --health-
check
Target=HTTP:80/png,Interval=30,UnhealthyThre
shold=2,HealthyThreshold=2,Timeout=3
```

Result:

```
{
   "HealthCheck": {
       "HealthyThreshold": 2,
       "Interval": 30,
       "Target": "HTTP:80/png",
       "Timeout": 3,
       "UnhealthyThreshold": 2
 }
```

http://docs.aws.amazon.com/cli/latest/reference/elb/configure-health-check.html

Details: Register Instances

The operation registers instances that can receive traffic,

```
%aws elb register-instances-with-load-
balancer --load-balancer-name my-load-
balancer --instances i-d6f6fae3
```

Result:

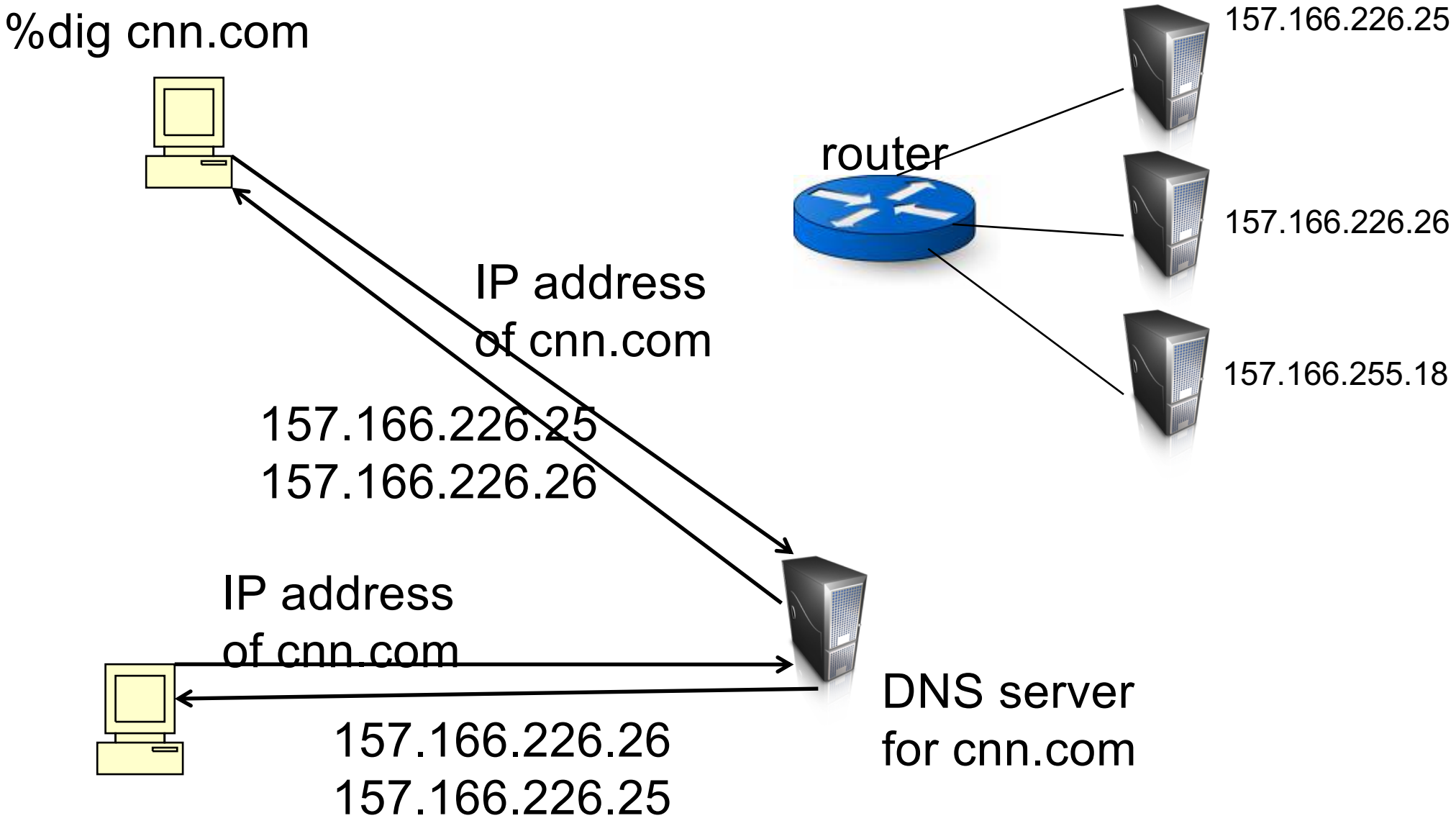
```
{   "Instances": [
     {"InstanceId": "i-d6f6fae3"},
     {"InstanceId": "i-207d9717"},
     {"InstanceId": "i-afefb49b"}
    ]
}
```

http://docs.aws.amazon.com/cli/latest/reference/elb/register-instances-with-load-balancer.html

Outline

- Recap
- Single network server
- Multiple network servers
  - Why multiple servers
  - Request routing mechanisms
    - Overview
    - Application-layer
      - *DNS*

Basic DNS Indirection and Rotation



CDN Using DNS (Akamai Architecture as an Example)

- Content publisher (e.g., cnn)
  - provides base HTML documents
  - runs origin server(s); but delegates heavy-weight content (e.g., images) to CDN
- Akamai runs
  - (~240,000) edge servers for hosting content
    - Deployment into 130 countries and 1600 networks
    - Claims 85% Internet users are within a single "network hop" of an Akamai CDN server.
  - customized DNS redirection servers to select edge servers based on
    - closeness to client browser, server load

Source: https://www.akamai.com/us/en/about/facts-figures.jsp

Linking to Akamai

- Originally, URL Akamaization of embedded content: e.g.,

  <IMG SRC= http://www.provider.com/image.gif >
  changed to

  <IMG SRC = http://a661. g.akamai.net/hash/image.gif>

  Note that this DNS redirection unit is per customer, not individual files.

- URL Akamaization is becoming obsolete and supported mostly for legacy reasons

Exercise

- Check any web page of nba.com and find a page with an image
- Find the URL
- Use

  `%dig [+trace]`

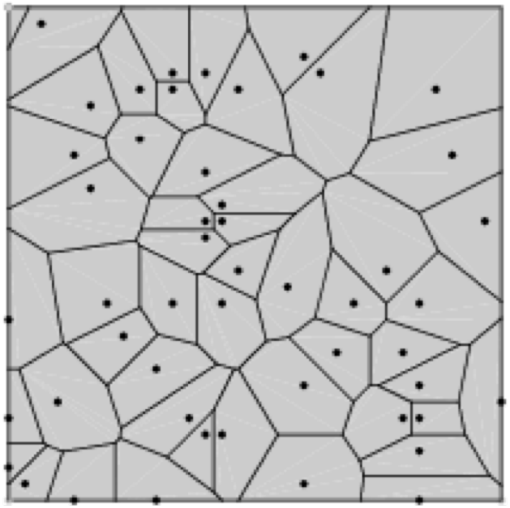
  to see DNS load direction

https://www.nba.com/news/reports-lakers-extend-gm-rob-pelinka-through-2025-26-season

Recap: CNAME based DNS Name

- Typical design
  - Use cname to create aliases, e.g., https://cdn.nba.com/manage/2021/08/GettyImages-1234610091-scaled-e1665274852371-1568x882.jpg
  - cname: e8017.dscb.akamaiedge.net
    - why two levels in the name?

Two-Level Direction



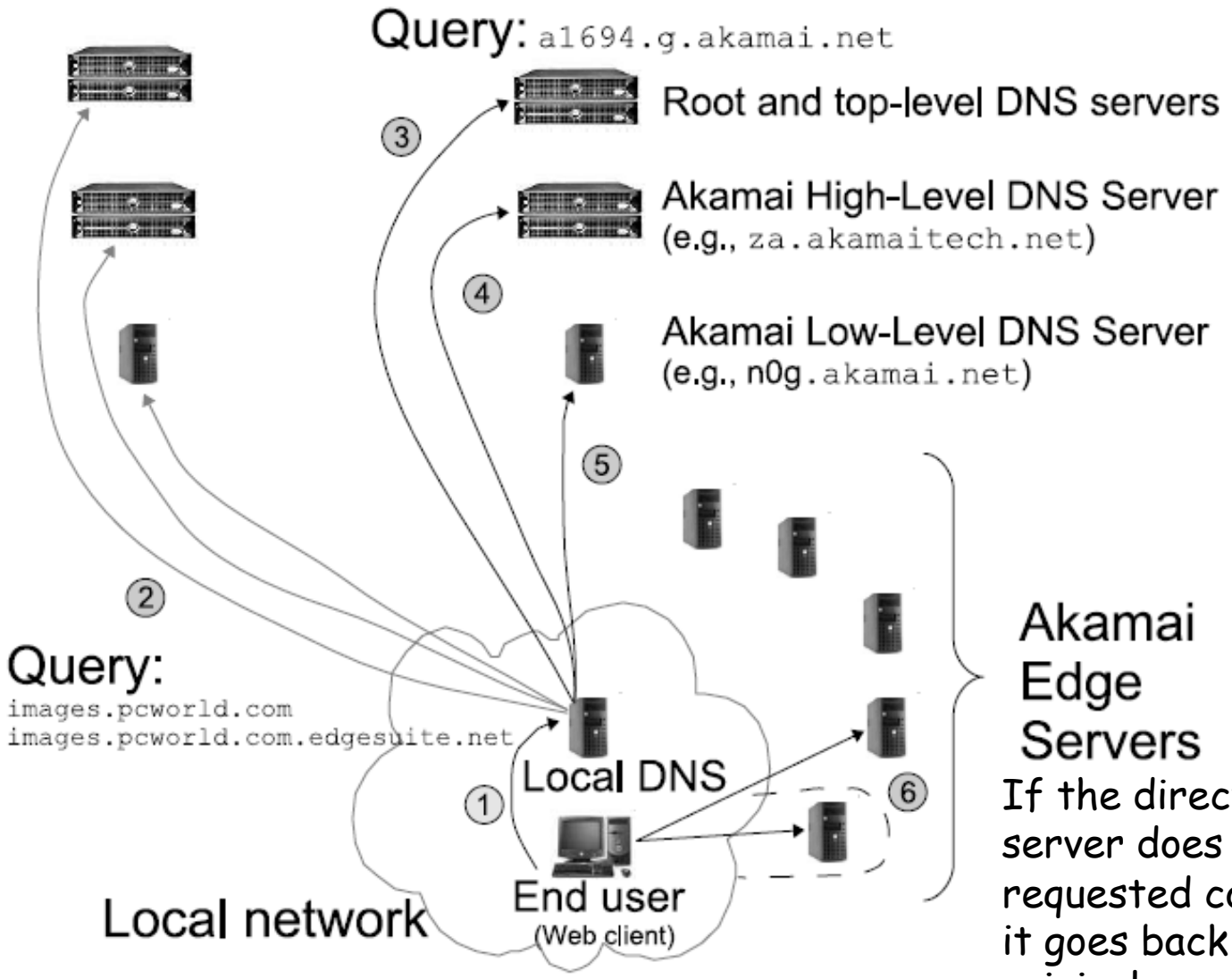
- high-level DNS determines proximity, directs to low-level DNS;

Input: dscb.akamaiedge.net &
and client IP,
Output: region (low-level) DNS

- low-level DNS: who manages a close-by cluster of servers with different IP addresses

Input: e8017.dscb.akamaiedge.net &
and client IP
Output: specific servers

Akamai Load Direction



If the directed edge server does not have requested content, it goes back to the original server (source) .

Two-Level DNS Mapping Alg

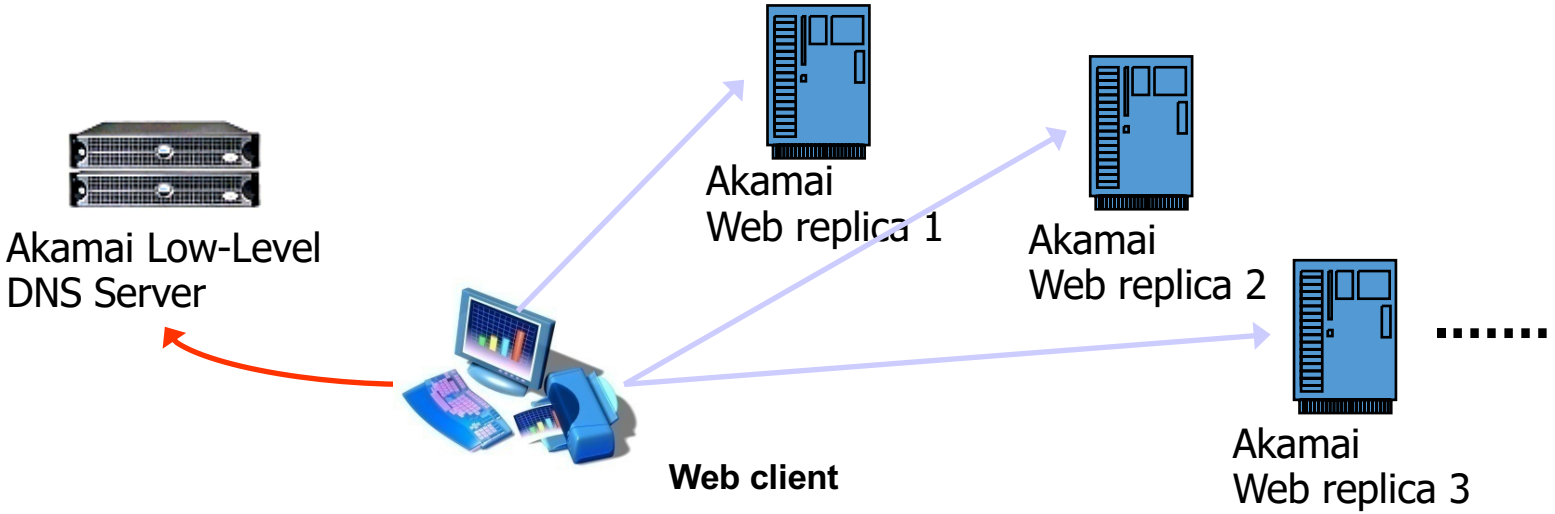
- High-level
  - Typically geo-location matching from client to reg
- Low-level
  - Typically secret, e.g., details of Akamai algorithms are proprietary
  - Typical goal: load balancing among servers

Akamai Local DNS LB Alg

- A Bin-Packing algorithm (column 12 of Akamai Patent) every T second
  - Compute the load to each publisher k (called serial number)
  - Sort the publishers from increasing load
  - For each publisher, associate a list of random servers generated by a hash function
    - Hash range may be increasing, e.g., first hash [0,1], second [0, 3]
  - Assign the publisher to the first server that does not overload

Experimental Study of Akamai Load Balancing

- Methodology
  - 2-months long measurement
  - 140 PlanetLab nodes (clients)
    - 50 US and Canada, 35 Europe, 18 Asia, 8 South America, the rest randomly scattered
  - Every 20 sec, each client queries an appropriate CNAME for Yahoo, CNN, Fox News, NY Times, etc.

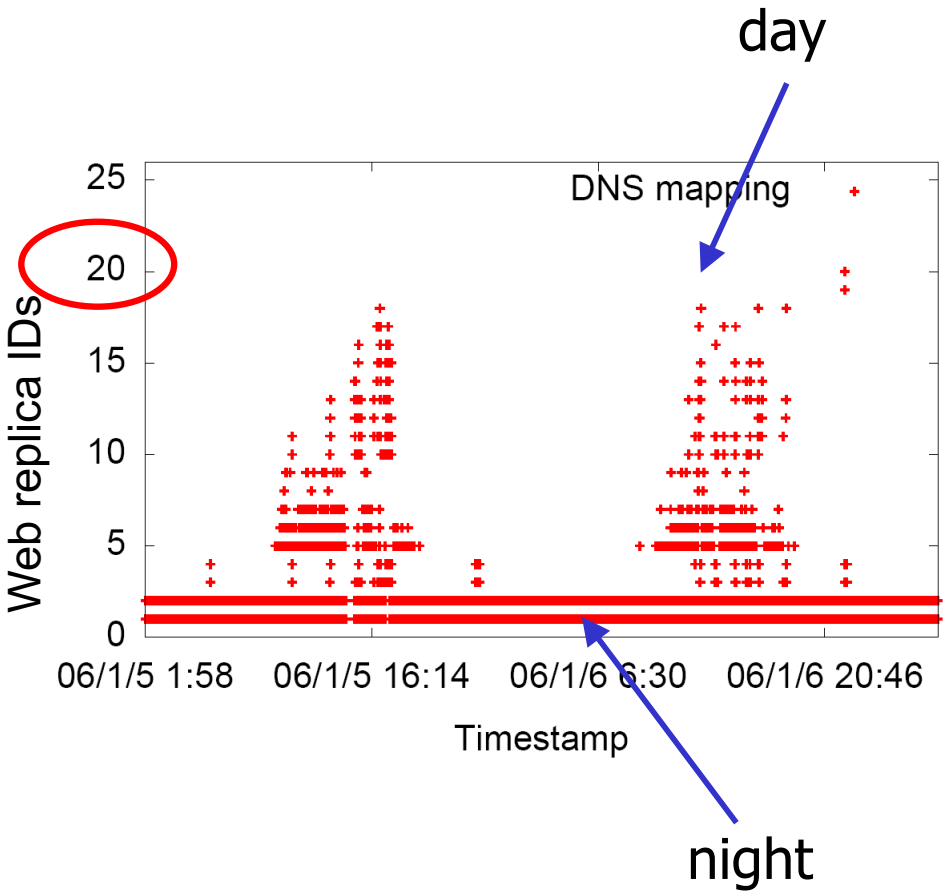


See http://www.aqualab.cs.northwestern.edu/publications/Ajsu06DBA.pdf

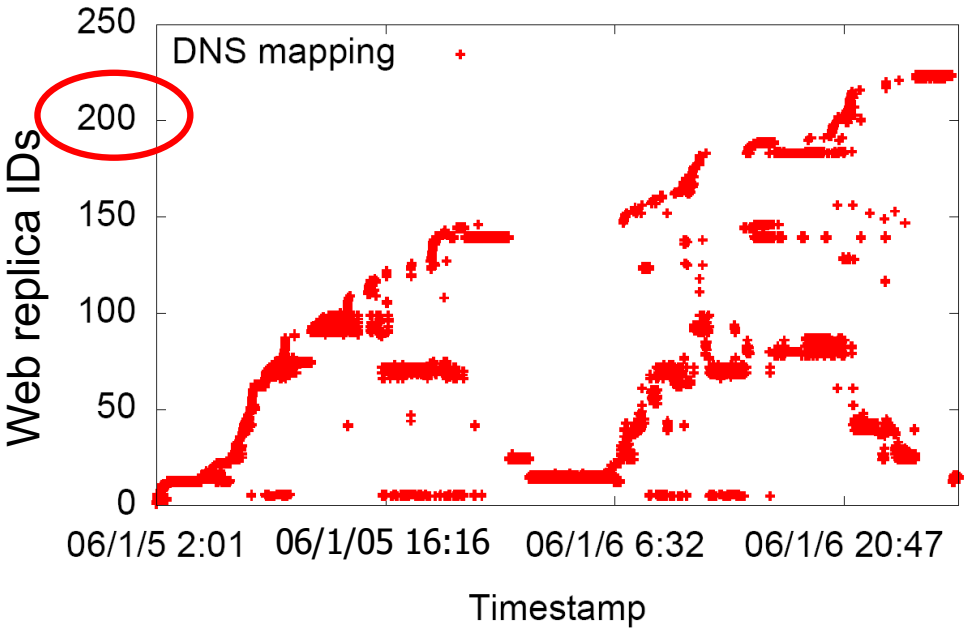
Server Pool: to Yahoo

Target: a943.x.a.yimg.com (Yahoo)

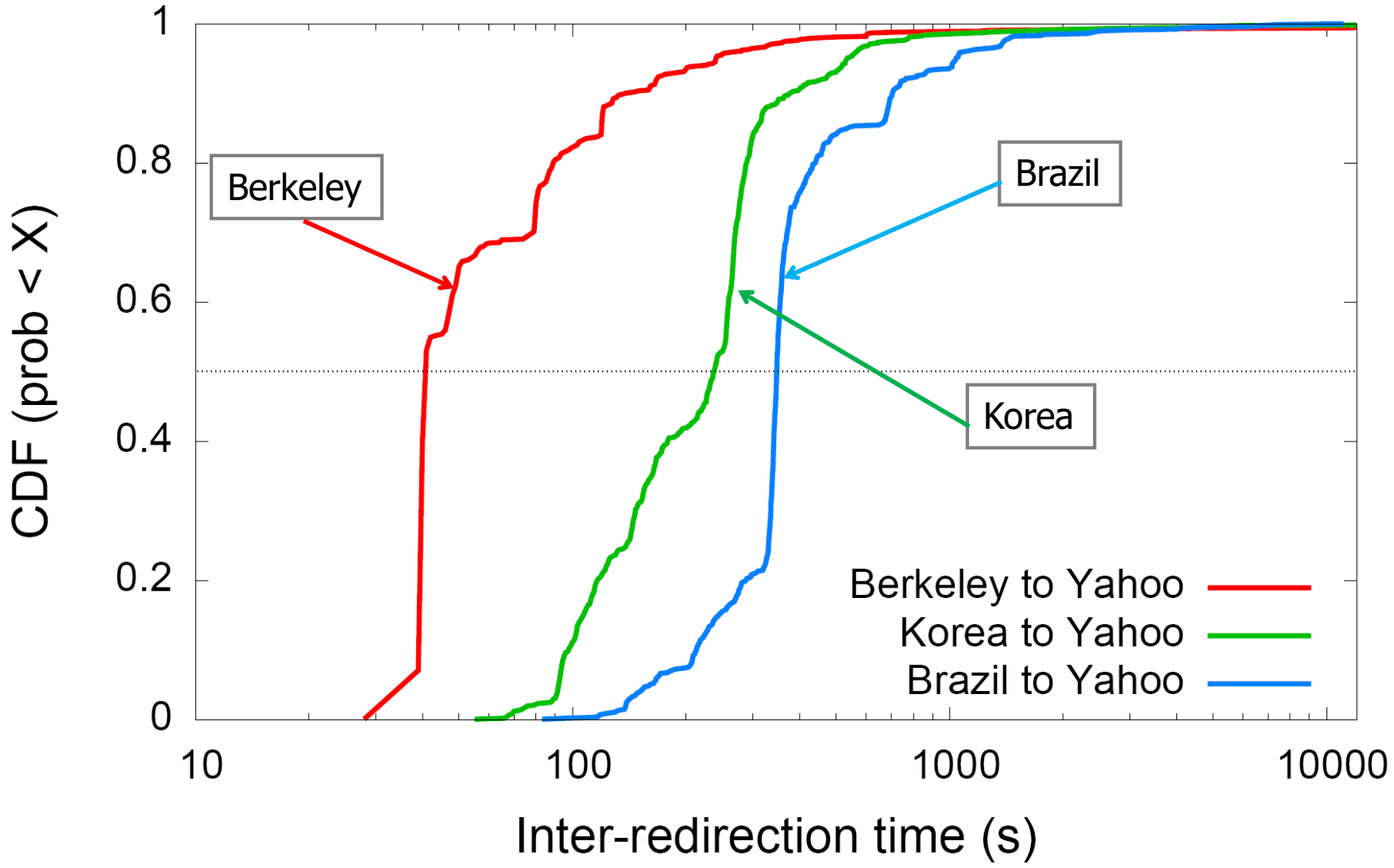
Client 1: Berkeley



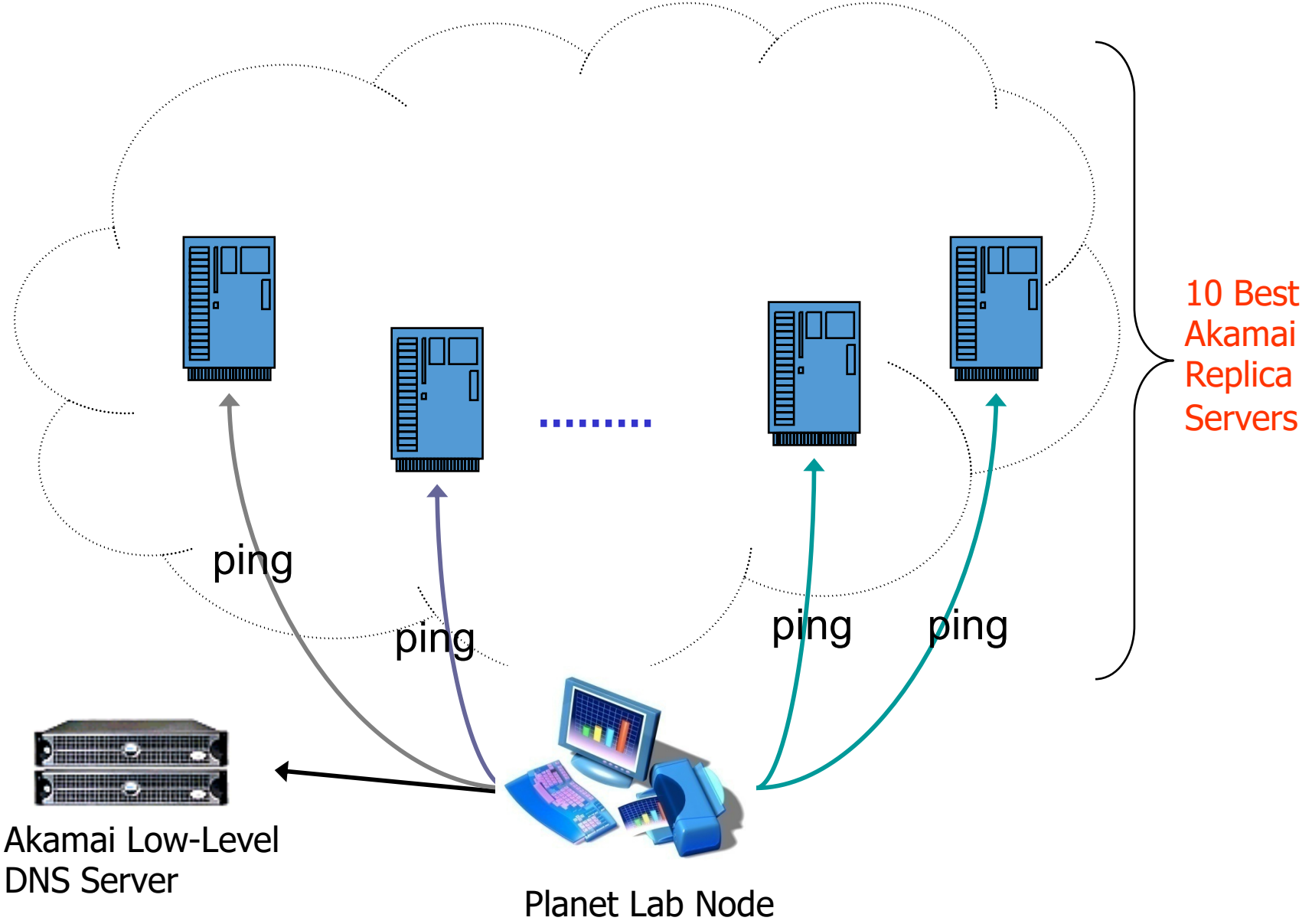
Client 2: Purdue



Load Balancing Dynamics

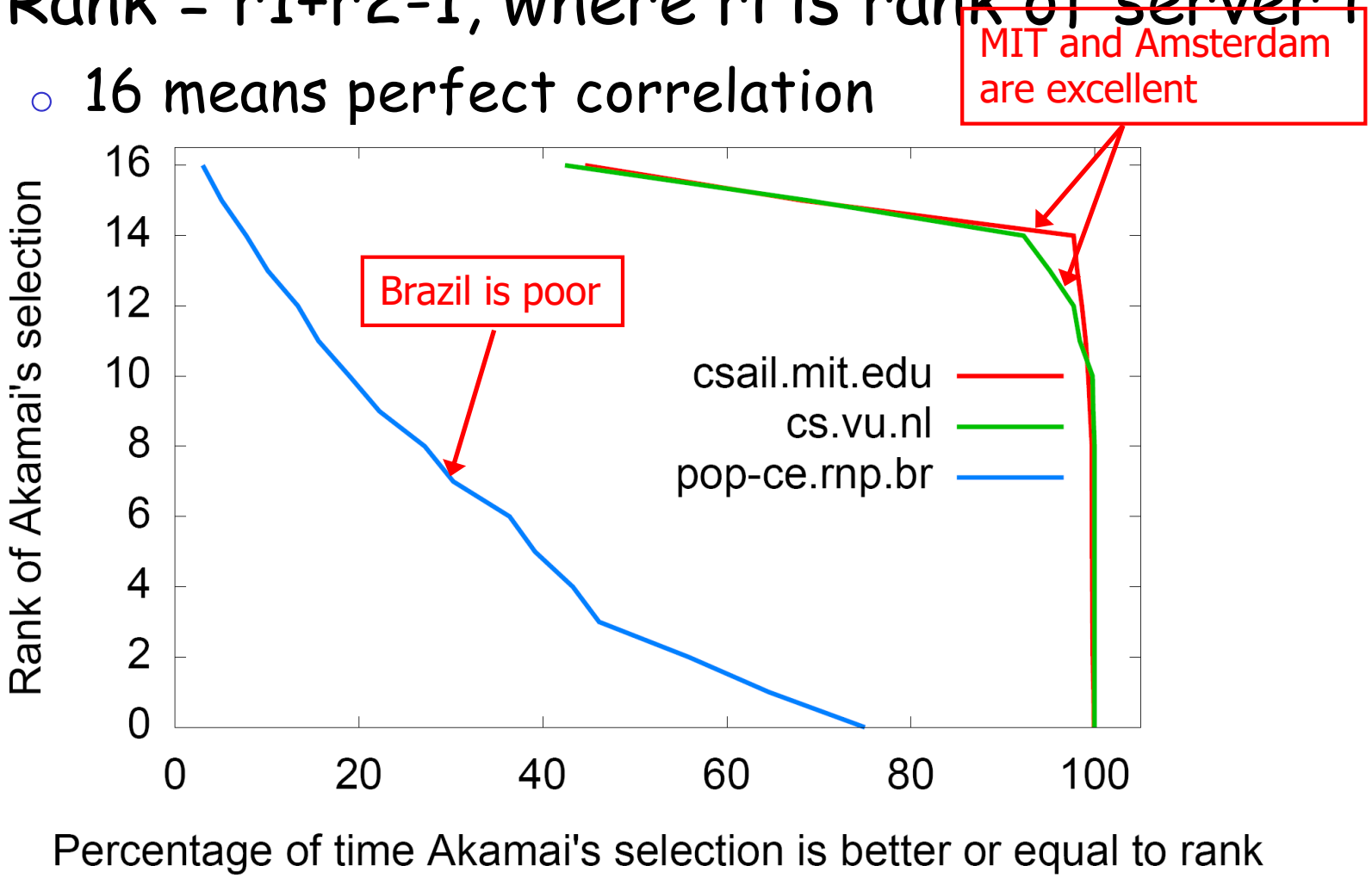


Redirection Effectiveness: Measurement Methodology



Do redirections reveal network conditions?

- Rank = r1+r2-1, where ri is rank of server i
  - 16 means perfect correlation



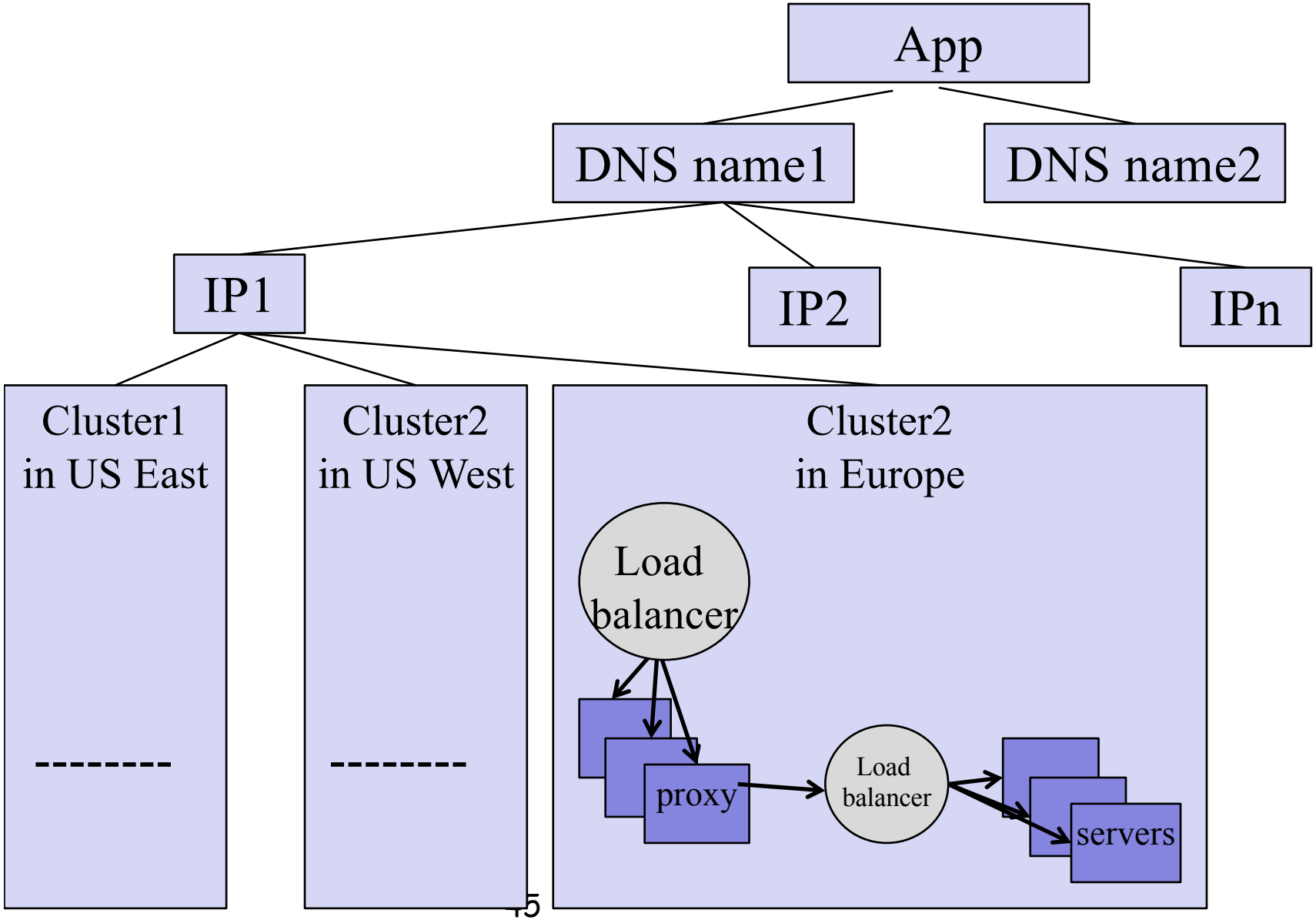
(Offline Read) Facebook DNS Load Direction

- A system named Cartographer (written in Python) processes measurement data and configures the DNS maps of individual DNS servers (open source tinydns)

Discussion

- Advantages of using DNS for using multiple servers (LB)
  - Leveraging existing DNS features (e.g., cname, hierarchy name for natural hierarchical redirection)
  - Leveraging existing DNS deployment/optimization
- Disadvantages of using DNS
  - Distributed caching may lead to slow response
  - Only in the unit of IP addresses

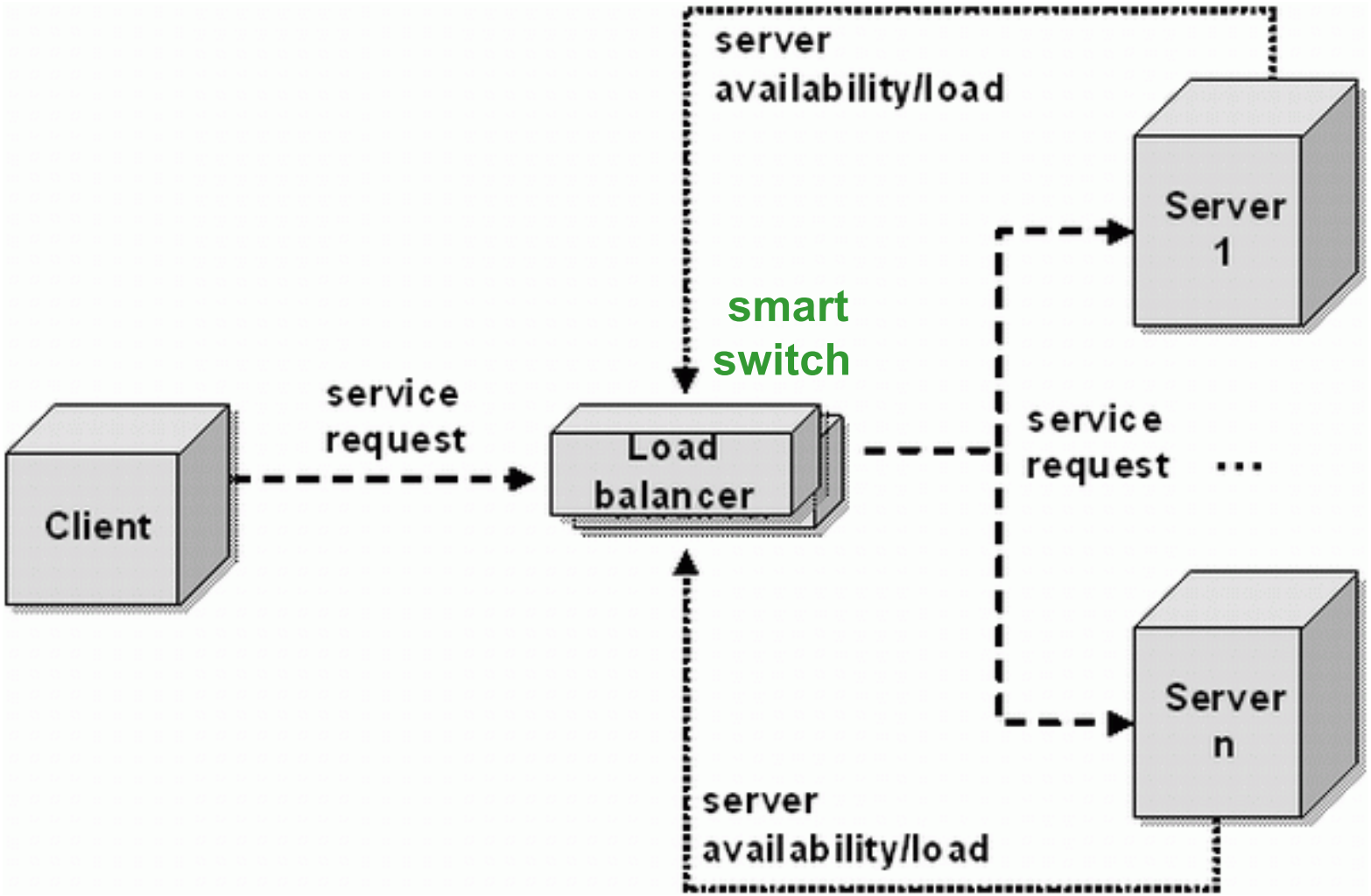
Recap: Direction Mechanisms



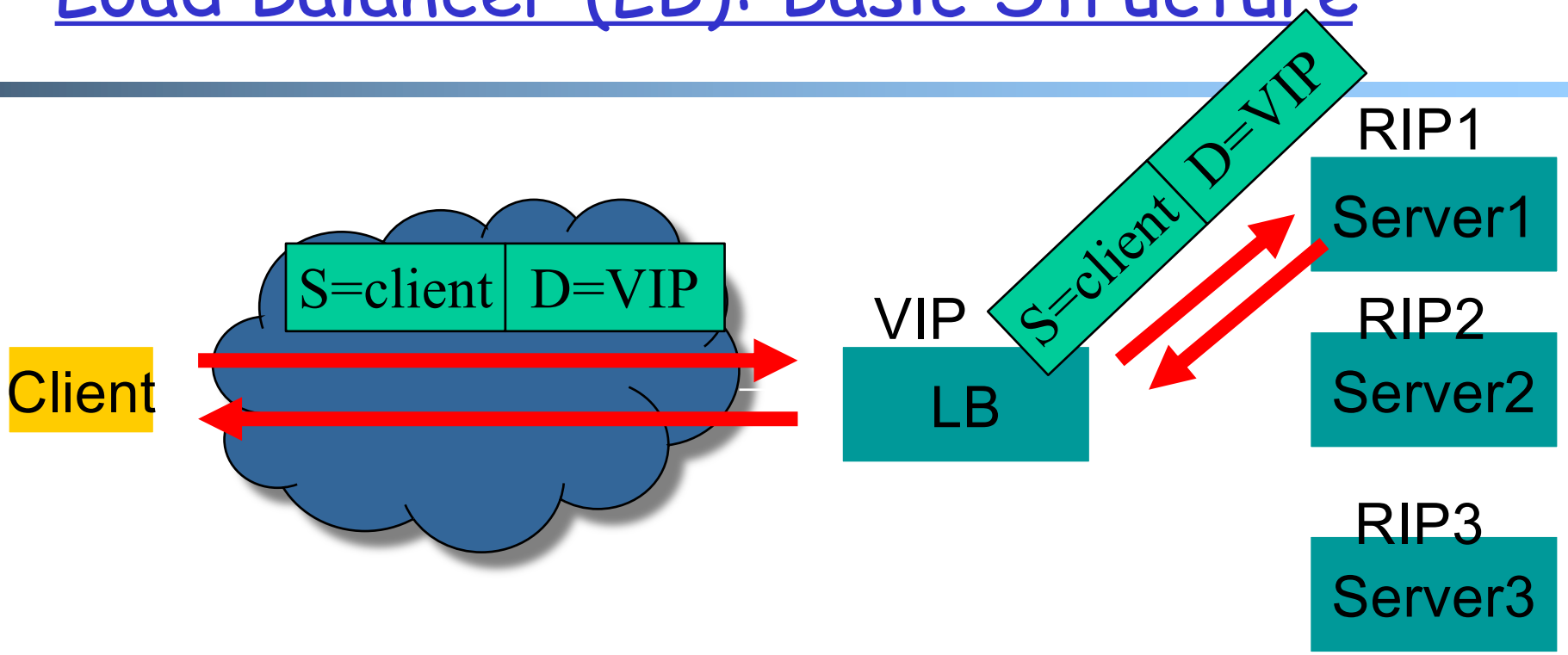
Outline

- Recap
- Single, high-performance network server
- Multiple network servers
  - Basic issues
  - Load direction
    - DNS (IP level)
    - Load balancer/smart switch (sub-IP level)

Smart Switch: Big Picture



Load Balancer (LB): Basic Structure

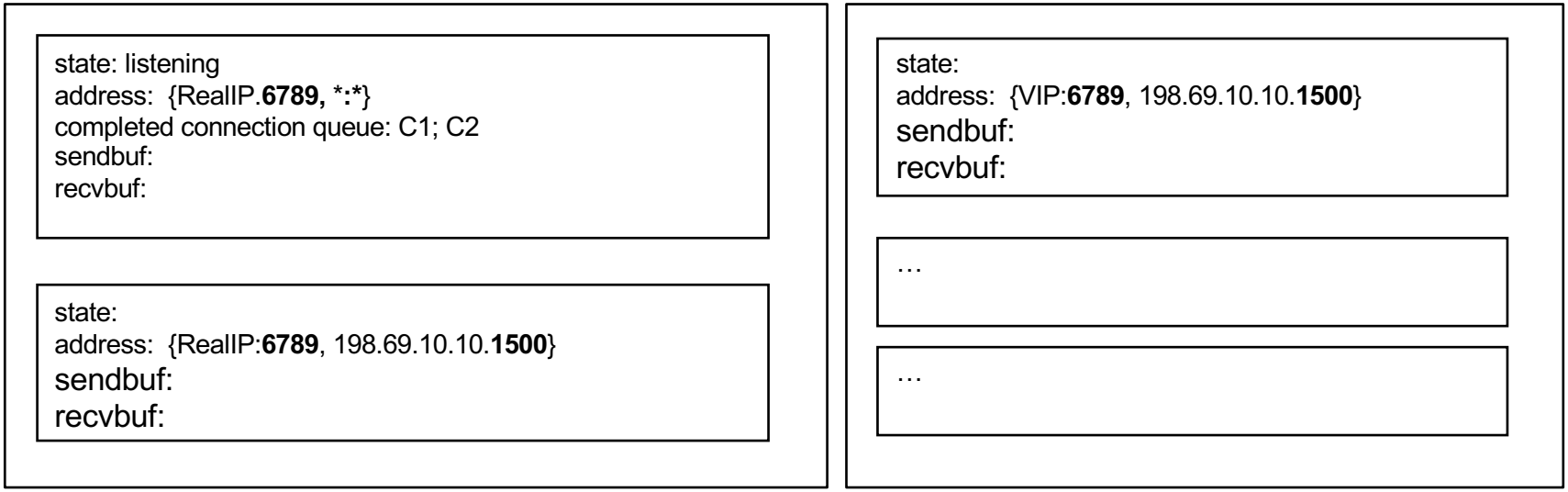


Problem of the basic structure?

Problem

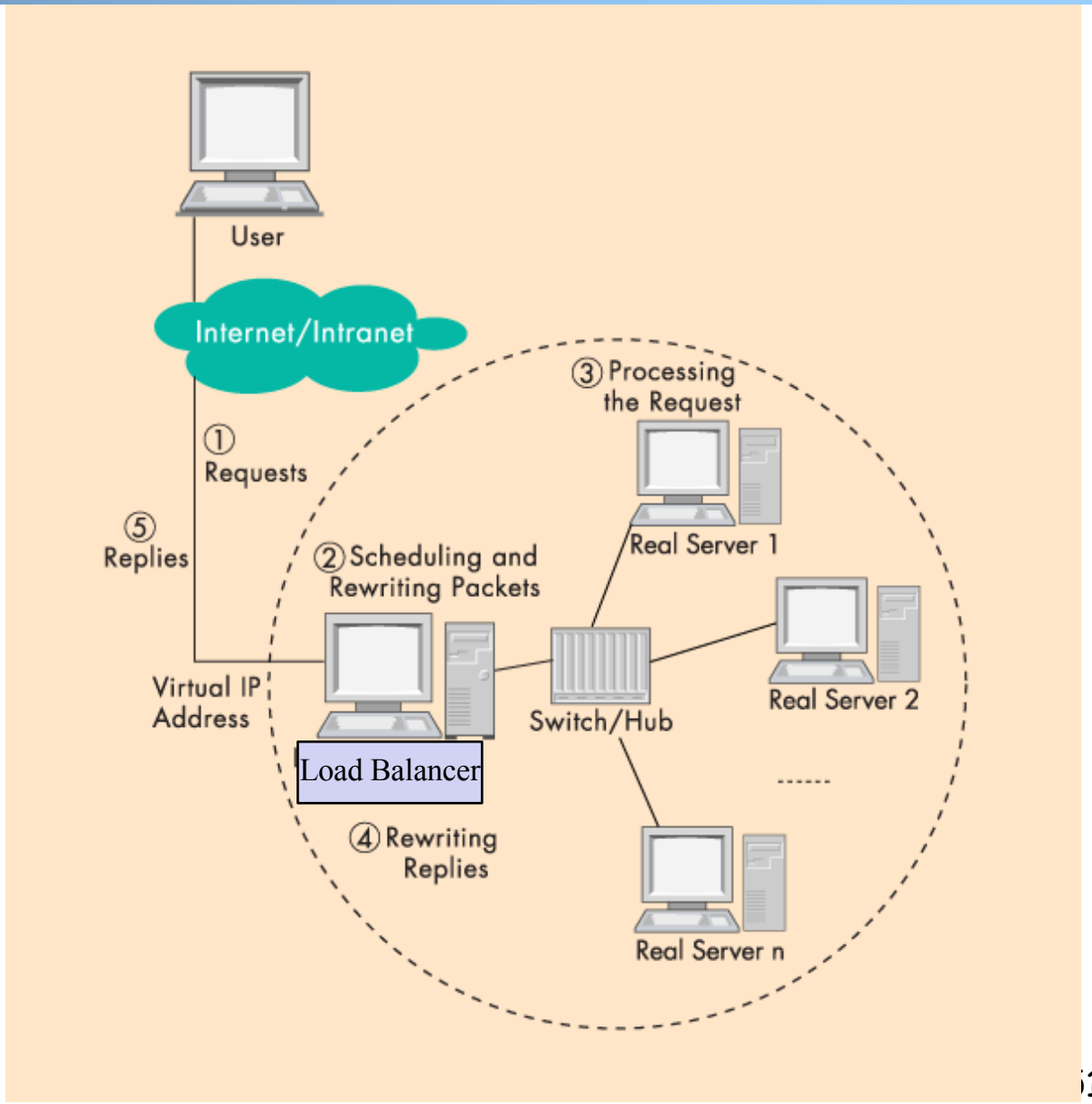
S=client	D=VIP

- Client to server packet has VIP as destination address, but real servers use RIPs
  - if LB just forwards the packet from client to a real server, the real server drops the packet
  - reply from real server to client has real server IP as source -> client will drop the packet



Solution 1: Network Address Translation (NAT)

- LB does rewriting/ translation
- Thus, the LB is similar to a typical NAT gateway with an additional scheduling function



Example Virtual Server via NAT

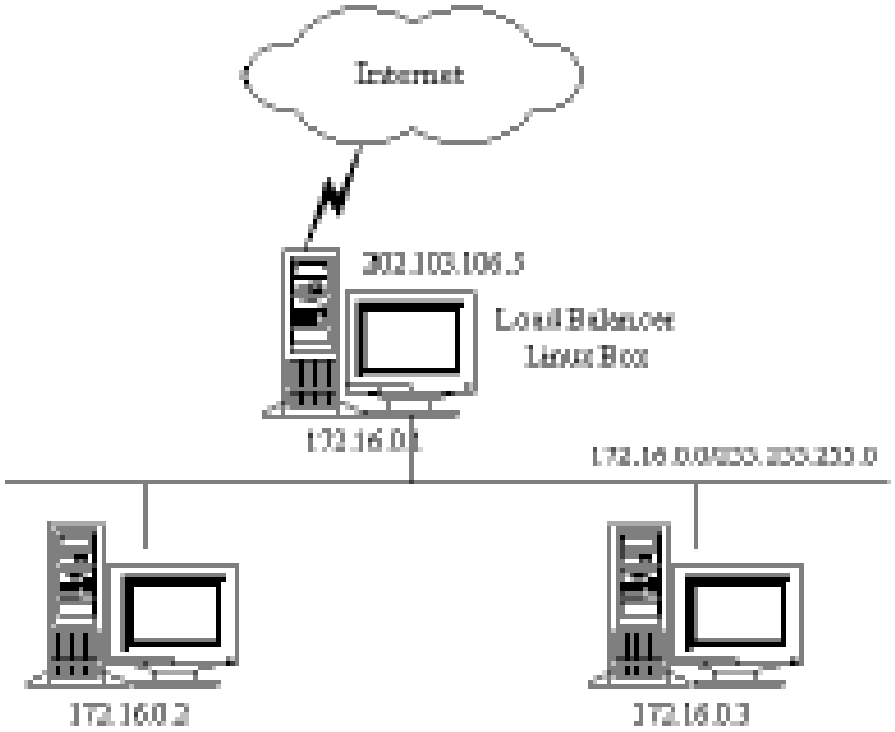
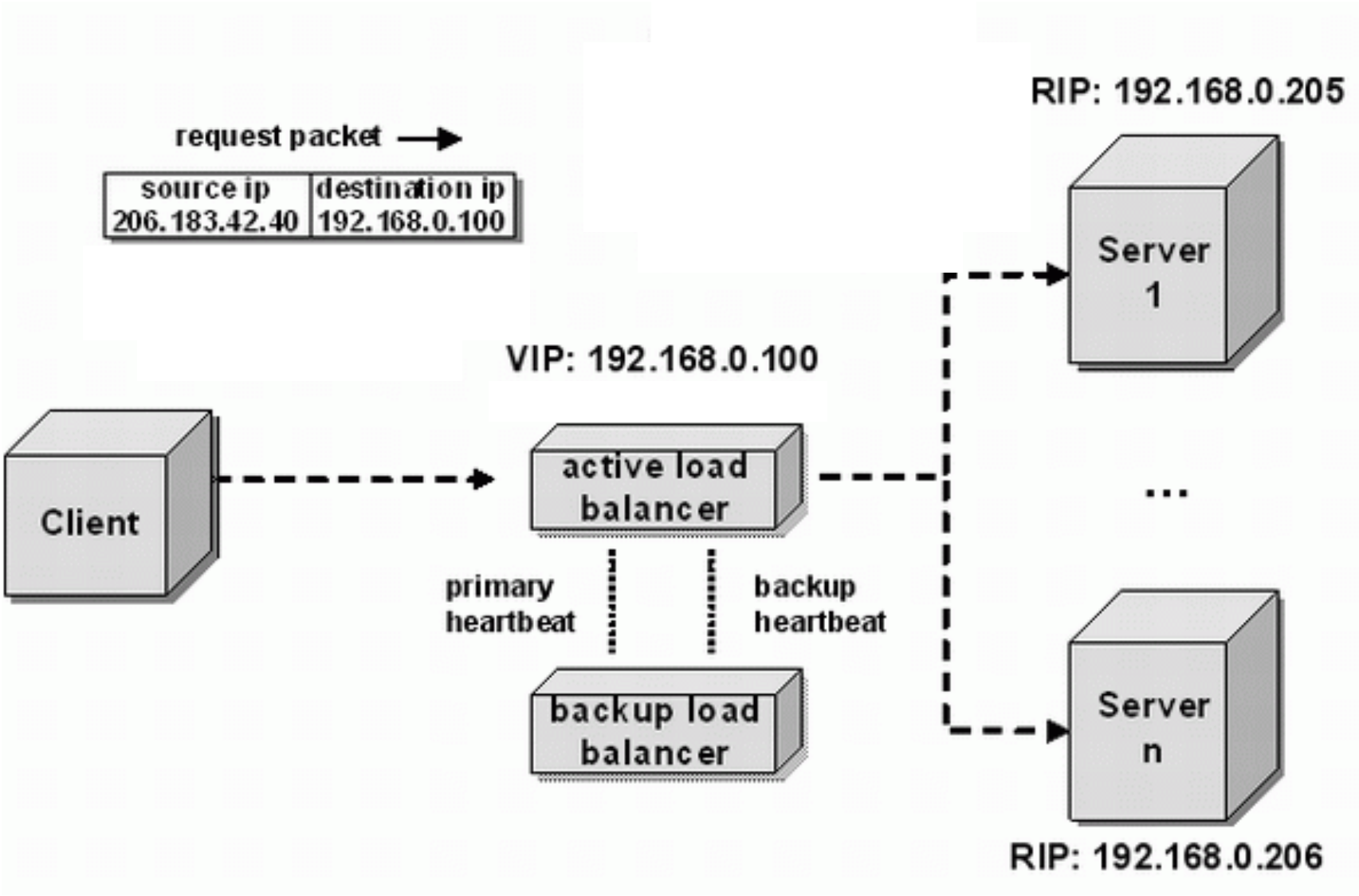


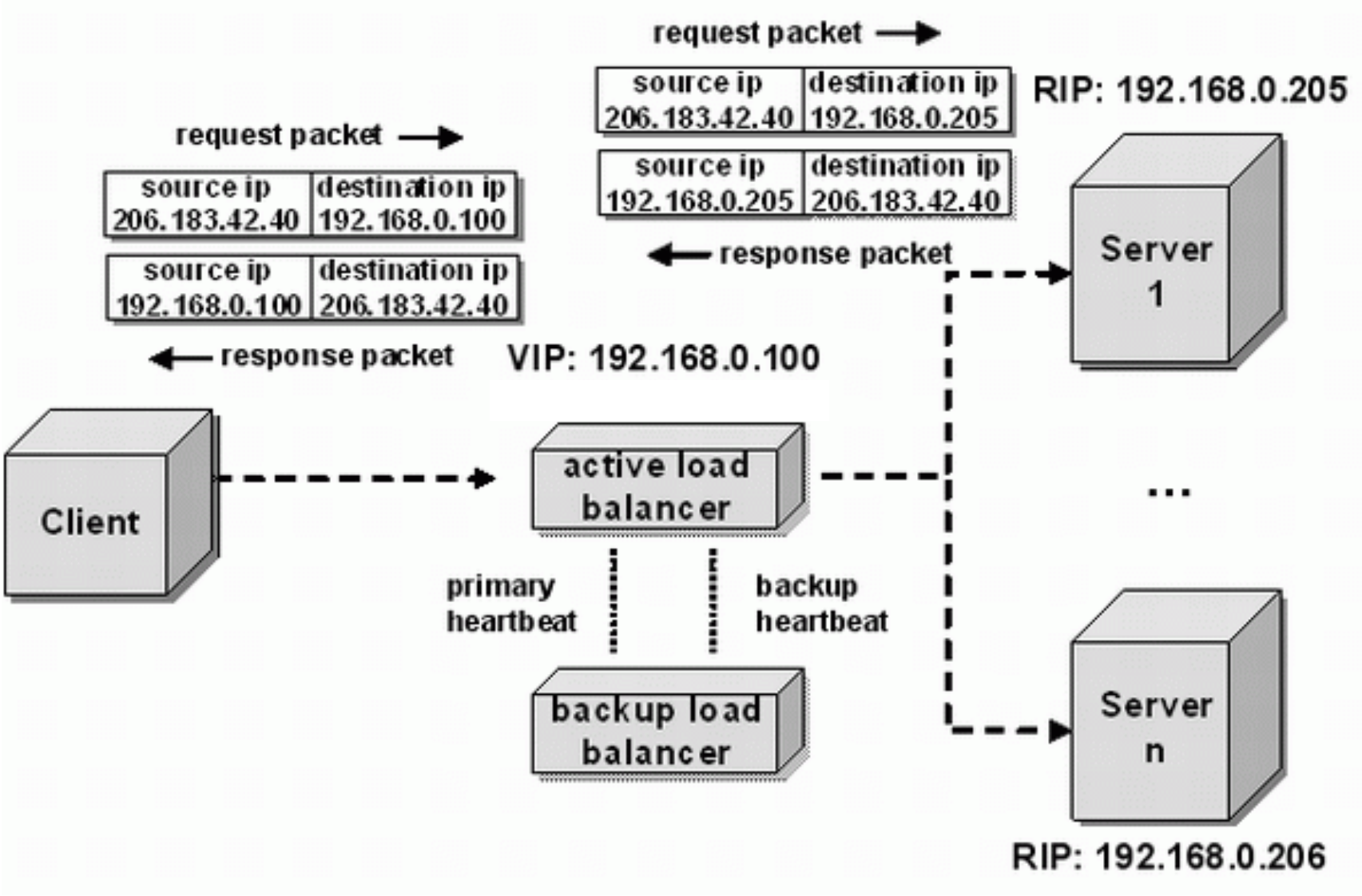
Table 1: an example of virtual server rules

Protocol	Virtual IP Address	Port	Real IP Address	Port	Weight
TCP	202.103.106.5	80	172.16.0.2	80	1
			172.16.0.3	8000	2
TCP	202.103.106.5	21	172.16.0.3	21	1

LB/NAT Flow


RIP: 192.168.0.205
request packet
source ip
206.183.42.40
destination ip
192.168.0.100
Server
1
VIP: 192.168.0.100
Client
active load
balancer
...
primary
heartbeat
backup
heartbeat
backup load
balancer
Server
n
RIP: 192.168.0.206

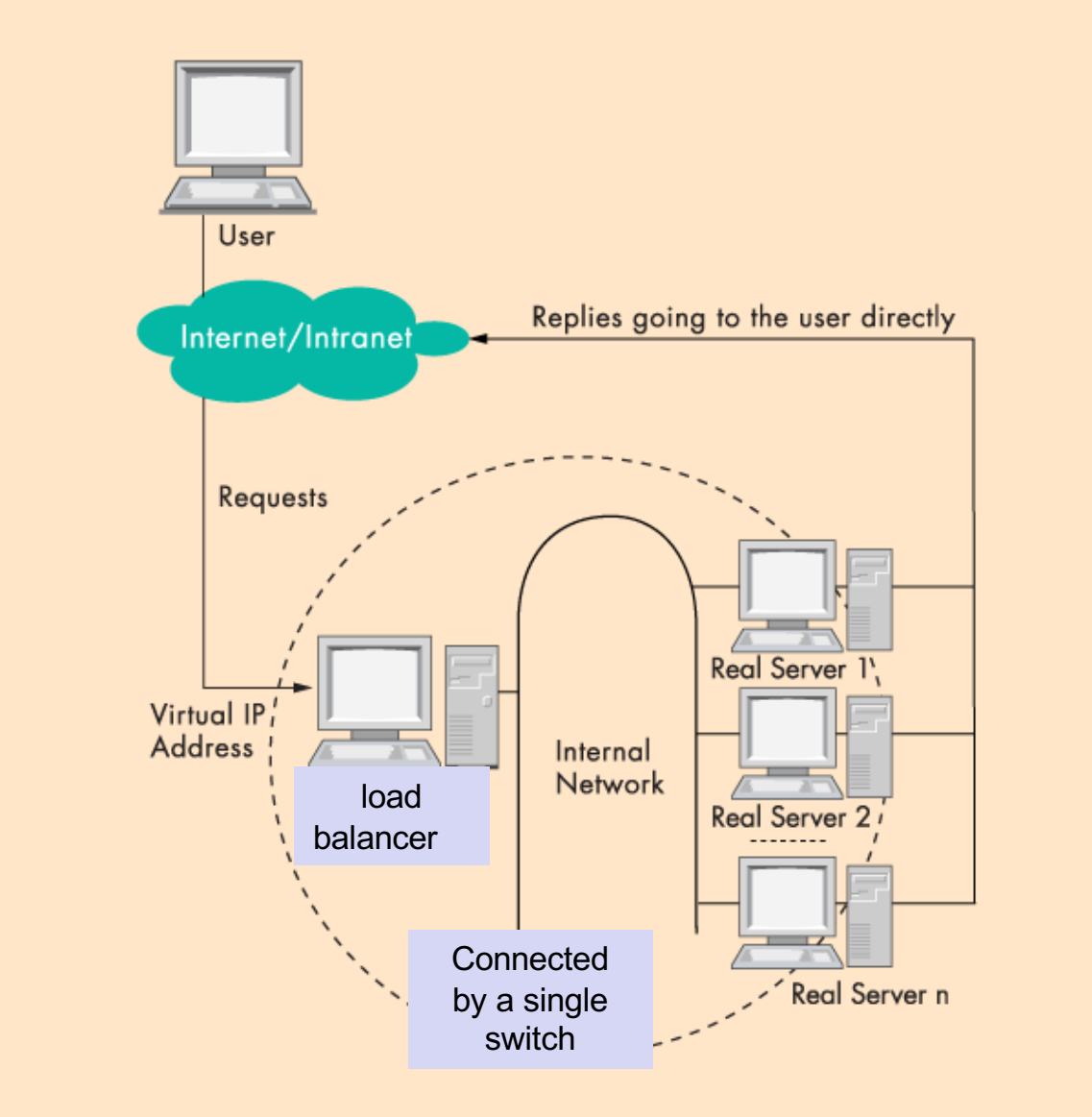
LB/NAT Flow


request packet
source ip 206.183.42.40
destination ip 192.168.0.205
RIP: 192.168.0.205
request packet
source ip 206.183.42.40
destination ip 192.168.0.100
source ip 192.168.0.205
destination ip 206.183.42.40
response packet
Server 1
source ip 192.168.0.100
destination ip 206.183.42.40
response packet
VIP: 192.168.0.100
Client
active load balancer
...
primary heartbeat
backup heartbeat
backup load balancer
Server n
RIP: 192.168.0.206

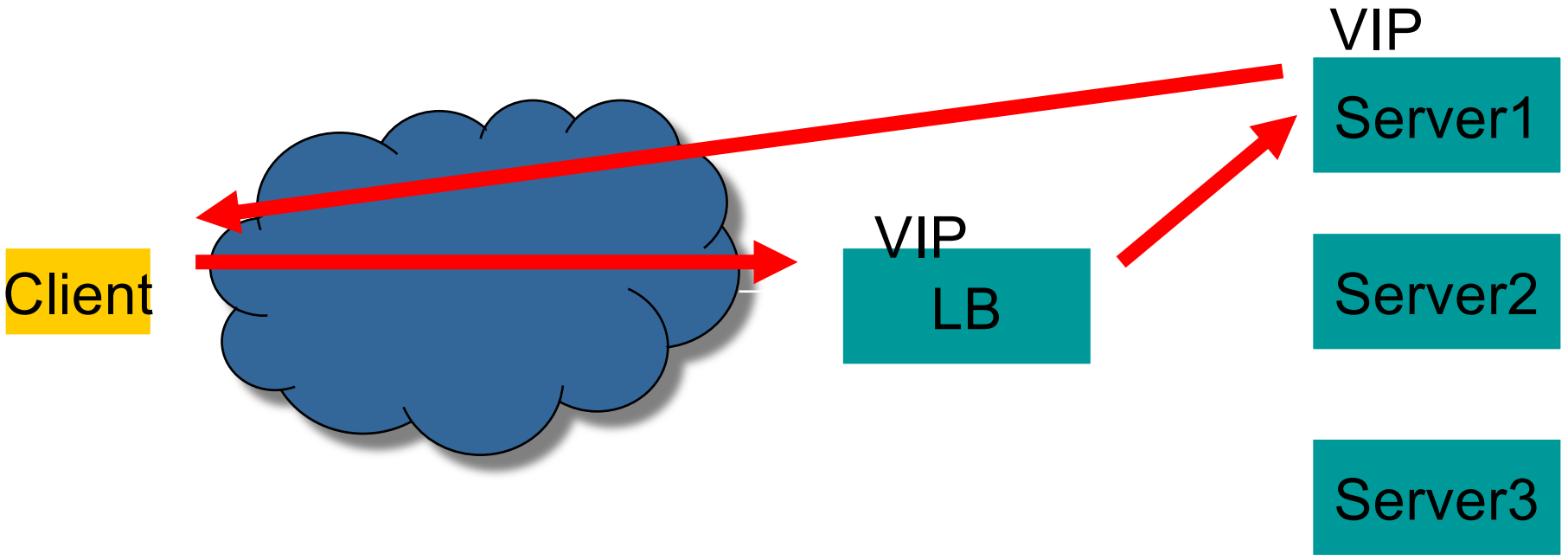
LB/NAT Advantages and Disadvantages

- Advantages:
  - Only one public IP address is needed for the load balancer; real servers can use private IP addresses
  - Real servers need no change and are not aware of load balancing
- Problem
  - The load balancer must be on the critical path and hence may become the bottleneck due to load to rewrite request and response packets
    - Typically, rewriting responses has more load because there are more response packets

Goal: LB w/ Direct Reply



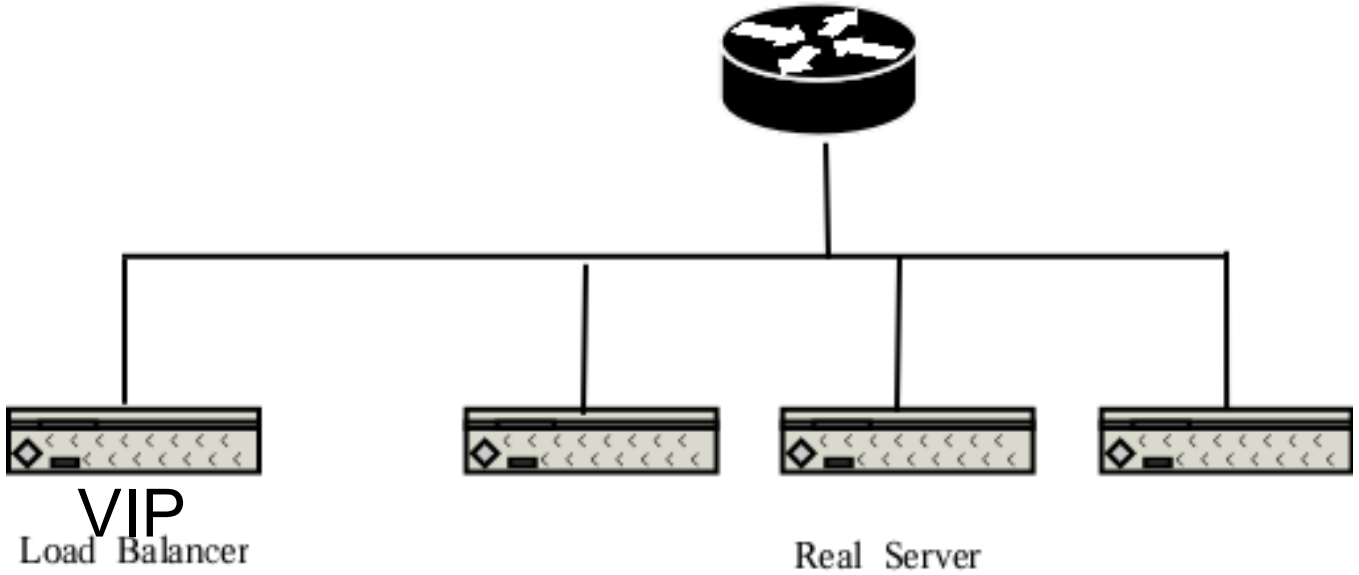
LB with Direct Reply: Implication



Direct reply → Each real server uses VIP as its IP address

→ Address conflict: multiple devices w/ the same IP addr

Why IP Address Matters?



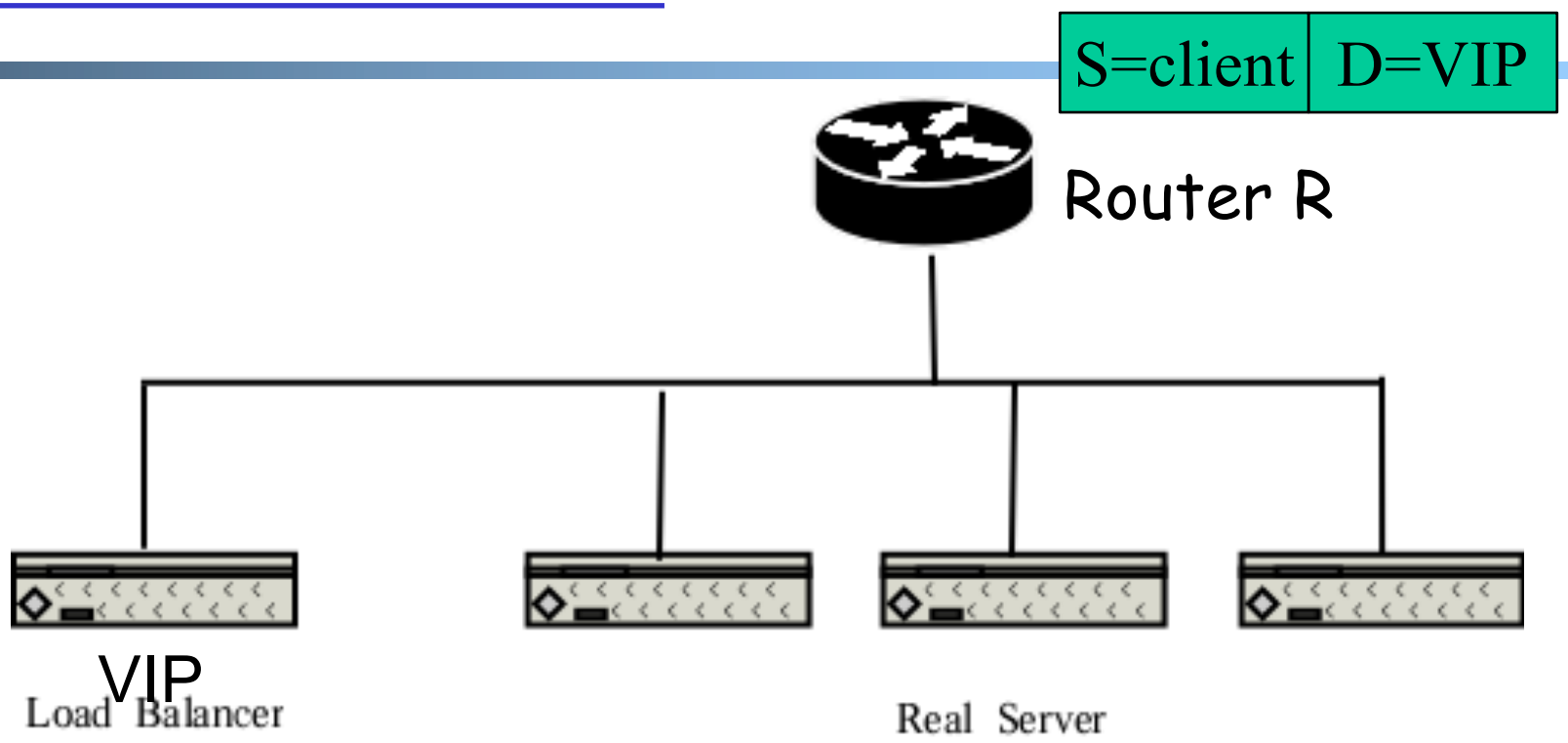
- Each network interface card listens to an assigned MAC address
- A router is configured with the range of IP addresses connected to each interface (NIC)
- To send to a device with a given IP, the router needs to translate IP to MAC (device) address
- The translation is done by the Address Resolution Protocol (ARP)

ARP Protocol

- ARP is "plug-and-play":
  - nodes create their ARP tables without intervention from net administrator

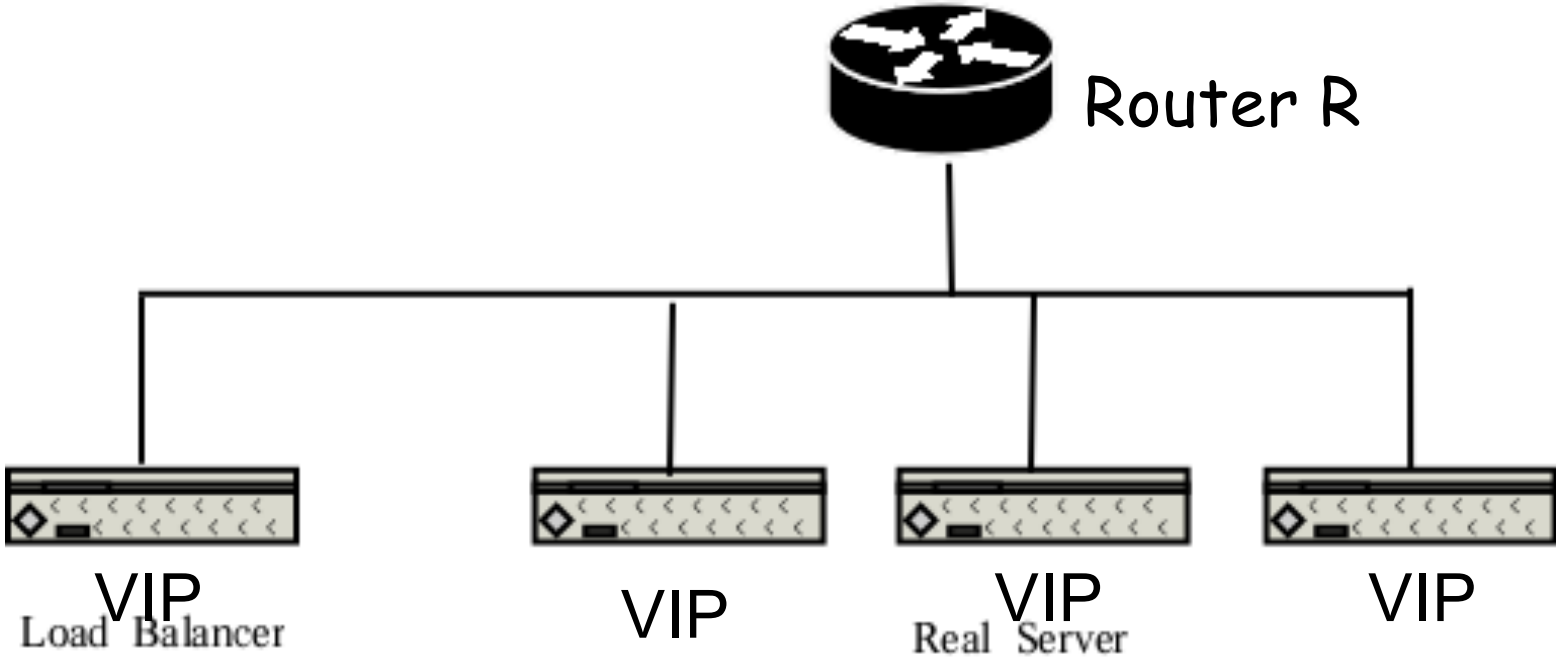
- A broadcast protocol:
  - Router broadcasts query frame, containing queried IP address
    - all machines on LAN receive ARP query
  - Node with queried IP receives ARP frame, replies its MAC address

ARP in Action



- Router broadcasts ARP broadcast query: who has VIP?
- ARP reply from LB: I have VIP; my MAC is MAC_{LB}
- Data packet from R to LB: destination MAC = MAC_{LB}

LB/DR Problem



ARP and race condition:

- When router R gets a packet with dest. address VIP, it broadcasts an Address Resolution Protocol (ARP) request: who has VIP?
- One of the real servers may reply before load balancer

Solution: configure real servers to not respond to ARP request

LB via Direct Routing

- The virtual IP address is shared by real servers and the load balancer.
- Each real server has a non-ARPing, loopback alias interface configured with the virtual IP address, and the load balancer has an interface configured with the virtual IP address to accept incoming packets.
- The workflow of LB/DR is similar to that of LB/NAT:
  - the load balancer directly routes a packet to the selected server
    - the load balancer simply changes the MAC address of the data frame to that of the server and retransmits it on the LAN (how to know the real server's MAC?)
  - When the server receives the forwarded packet, the server determines that the packet is for the address on its loopback alias interface, processes the request, and finally returns the result directly to the user

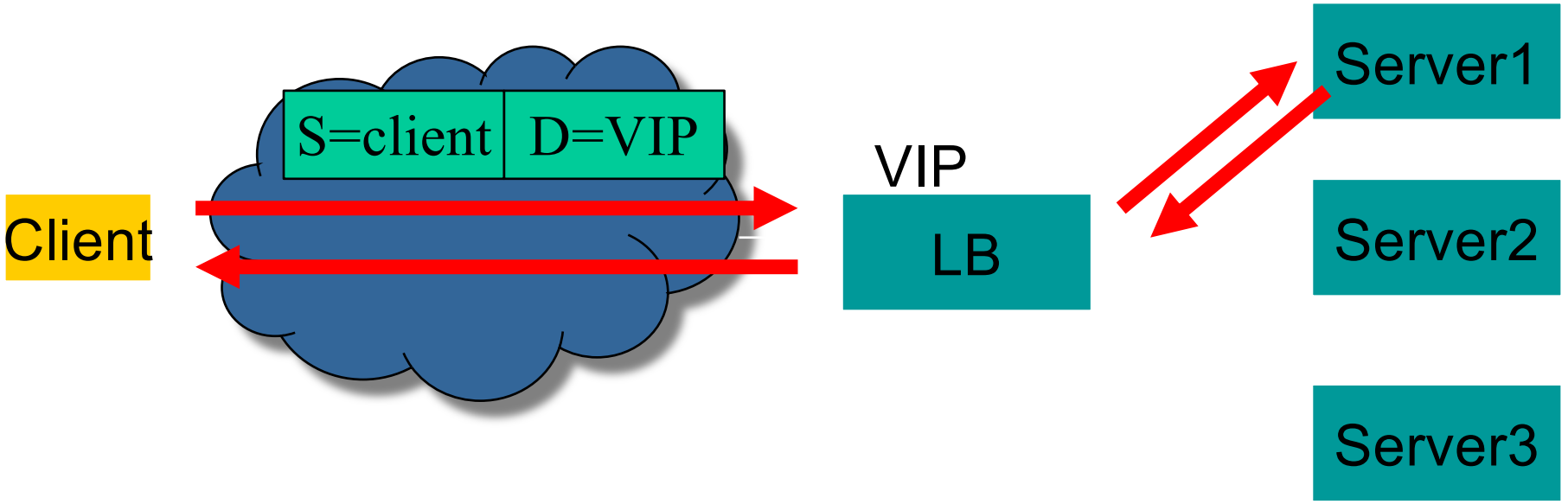
LB/DR Advantages and Disadvantages

- Advantages:
  - Real servers send response packets to clients directly, avoiding LB as bottleneck
- Disadvantages:
  - Servers must have non-arp alias interface
  - The load balancer and server must have one of their interfaces in the same LAN segment
  - Considered by some as a hack, not a clean architecture

Example Implementation of LB

- An example open source implementation is Linux virtual server (linux-vs.org)
  - Used by
    - www.linux.com
    - sourceforge.net
    - wikipedia.org
  - More details on ARP problem: http://www.austintek.com/LVS/LVS-HOWTO/HOWTO/LVS-HOWTO.arp_problem.html
  - Many commercial LB servers from F5, Cisco, …
- More details please read chapter 2 of Load Balancing Servers, Firewalls, and Caches

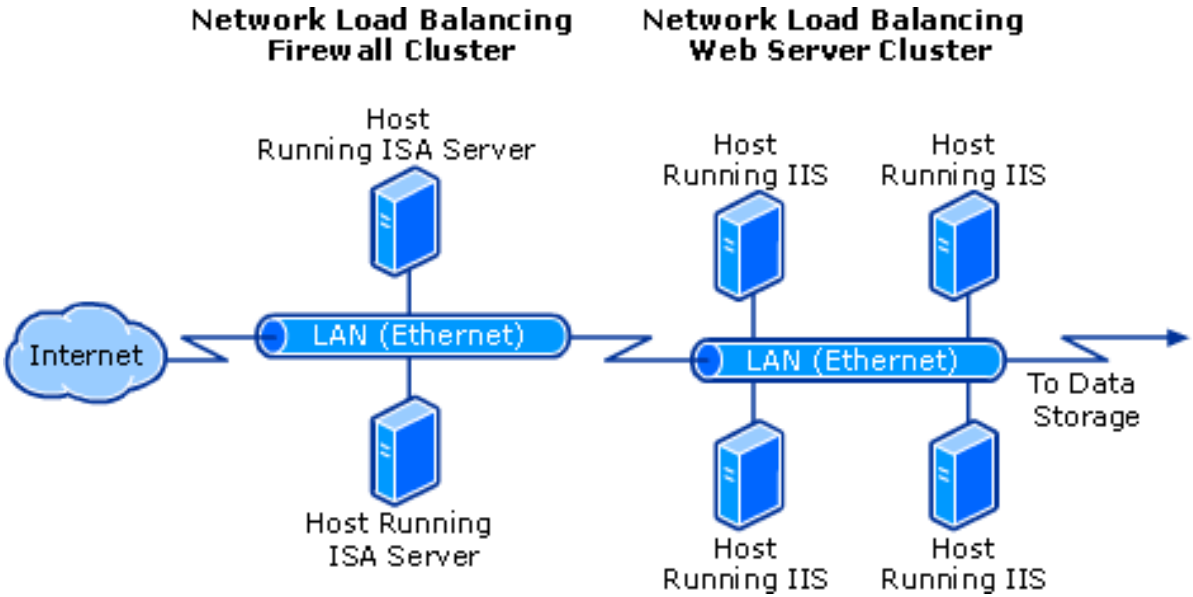
Problem of the Load Balancer Architecture



One major problem is that the LB becomes a single point of failure (SPOF).

Solutions

- Redundant load balancers
  - E.g., two load balancers (a good question to think offline)
- Fully distributed load balancing
  - e.g., Microsoft Network Load Balancing (NLB)



Microsoft NLB

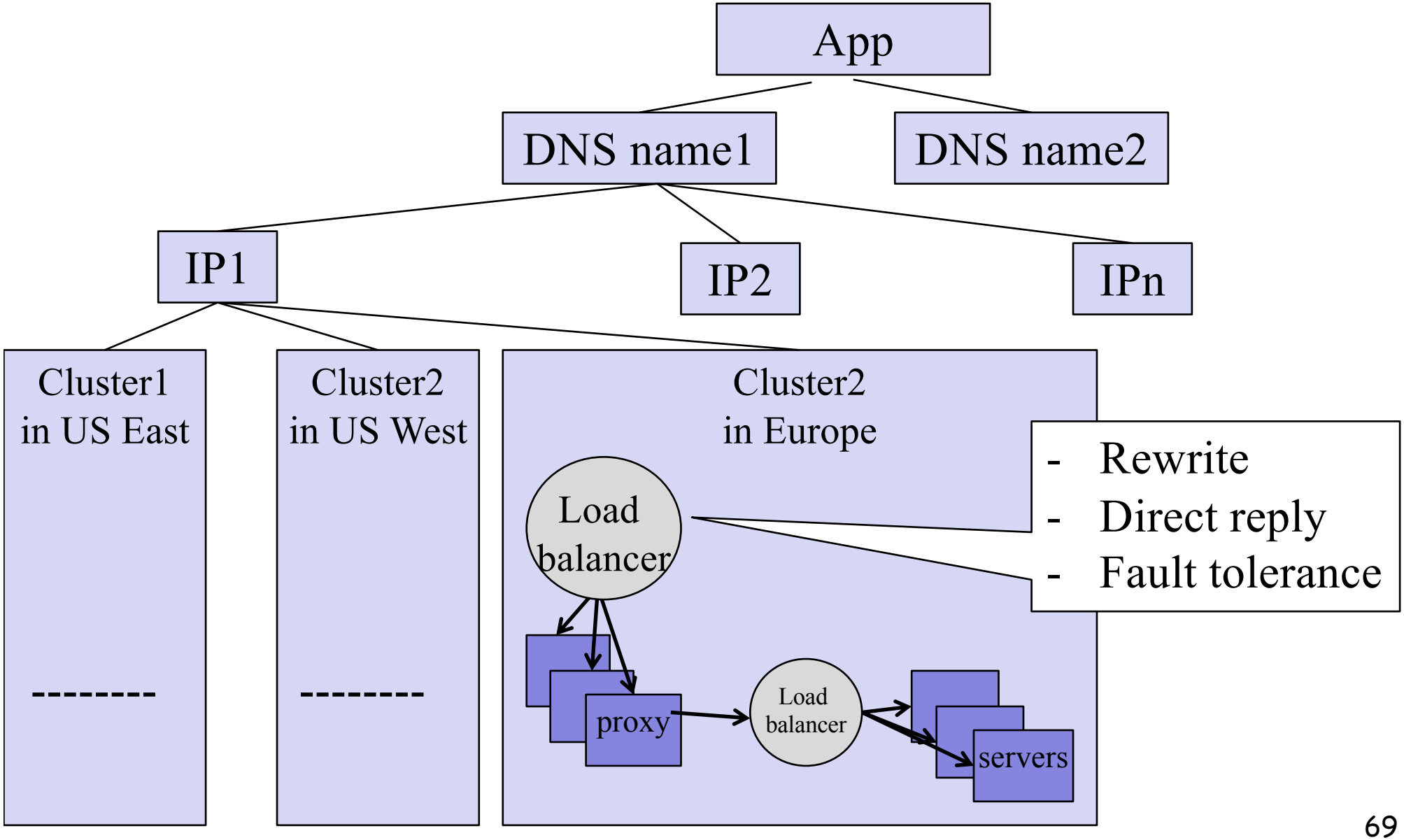
- No dedicated load balancer
- All servers in the cluster receive all packets
- Key issue: one and only one server processes each packet
  - All servers within the cluster simultaneously run a mapping algorithm to determine which server should handle the packet. Those servers not required to service the packet simply discard it.
    - Mapping (ranking) algorithm: computing the "winning" server according to host priorities, multicast or unicast mode, port rules, affinity, load percentage distribution, client IP address, client port number, other internal load information

http://technet.microsoft.com/en-us/library/cc739506%28WS.10%29.aspx

Discussion

- Compare the design of using Load Balancer vs Microsoft NLB

Recap: Direction Mechanisms



Outline

- Admin and recap
- Single, high-performance network server
- Multiple servers
  - Overview
  - Basic mechanisms
  - Example: YouTube (offline read)

http://video.google.com/videoplay?docid=-6304964351441328559#

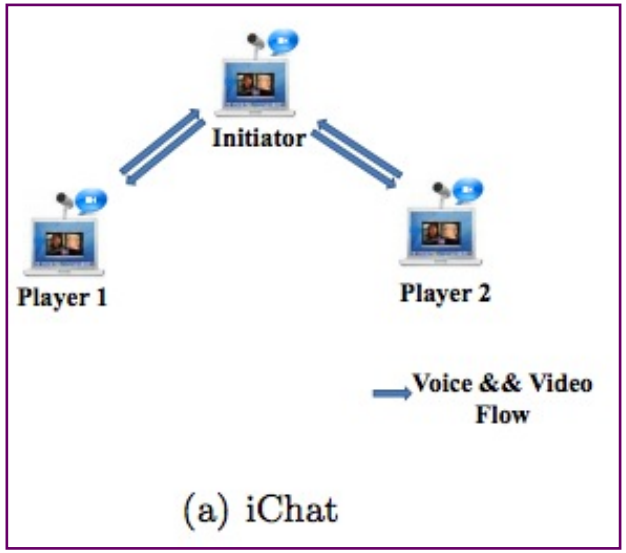
You Tube

- 02/2005: Founded by Chad Hurley, Steve Chen and Jawed Karim, who were all early employees of PayPal.
- 10/2005: First round of funding ($11.5 M)
- 03/2006: 30 M video views/day
- 07/2006: 100 M video views/day
- 11/2006: acquired by Google
- 10/2009: Chad Hurley announced in a blog that YouTube serving well over 1 B video views/day (avg = 11,574 video views /sec)

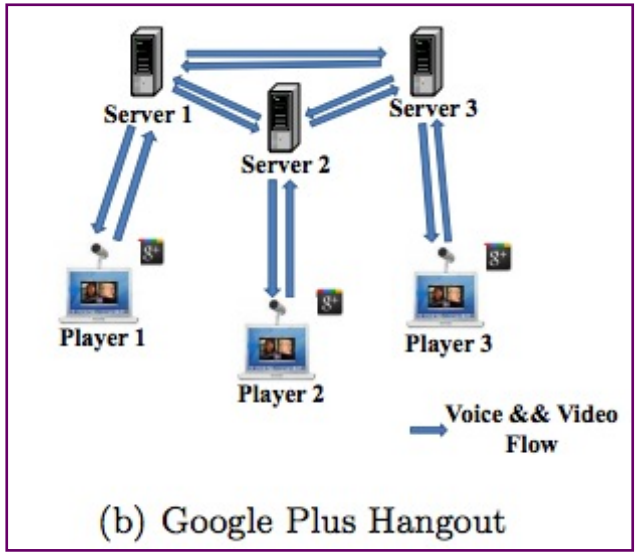
Pre-Google Team Size

- 2 Sysadmins
- 2 Scalability software architects
- 2 feature developers
- 2 network engineers
- 1 DBA
- 0 chefs

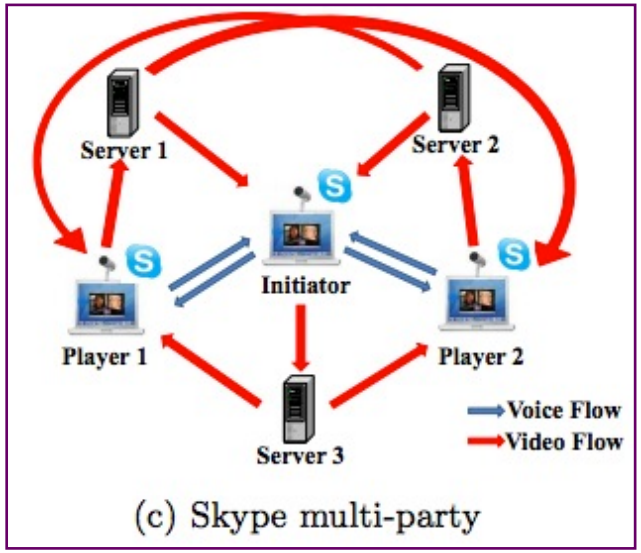
Example Server Systems



(a) iChat



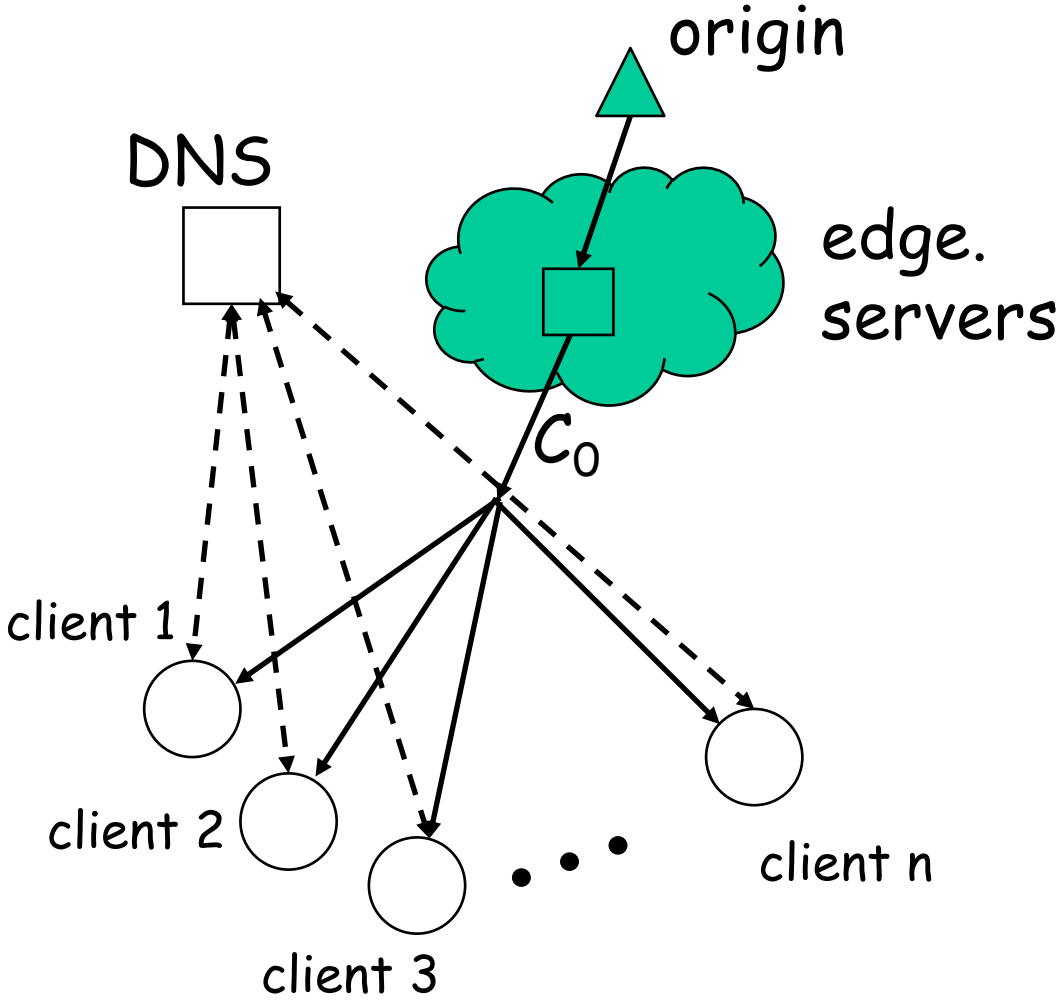
(b) Google Plus Hangout



(c) Skype multi-party

http://eeweb.poly.edu/faculty/yongliu/docs/imc12.pdf

Scalability of Server-Only Approaches



Outline

- Admin and recap
- Multiple servers
- Application overlays
  - potential

An Upper Bound on Scalability

- Idea: use resources from both clients and the server
- Assume
  - need to achieve same rate to all clients
  - only uplinks can be bottlenecks
- What is an upper bound on scalability?