CONTENT-DISTRIBUTION NETWORKS

May 20, 2025

George Porter

UC San Diego

ATTRIBUTION

UC San Diego

ANNOUNCEMENTS

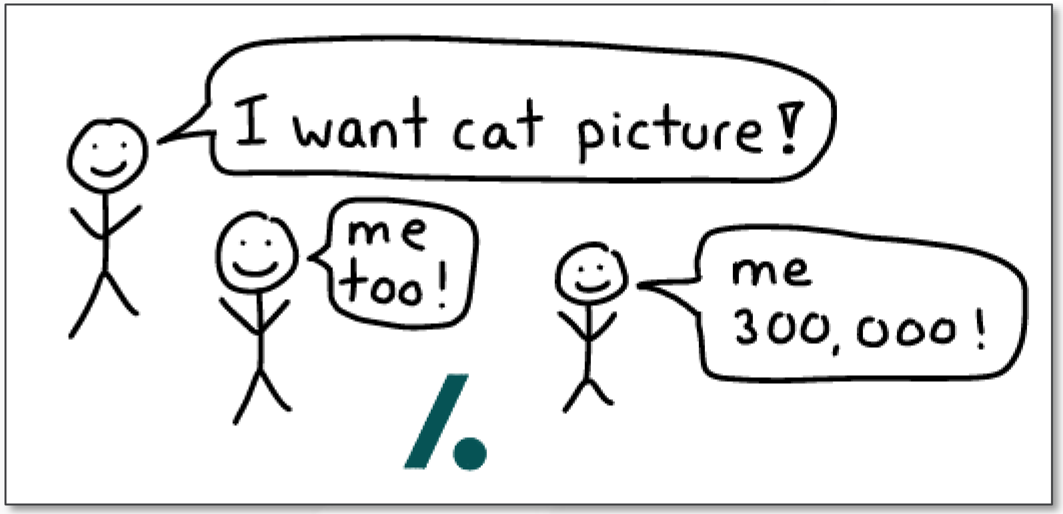
Required reading: “Algorithmic Nuggets in Content Delivery” (linked off canvas and the course schedule on the web page)

- Abstract, Introduction, and Section 2 “Stable Allocations” ONLY (for now)
- The Stable Marriage problem was formulated in 1962, so keep that in mind because discussions of that algorithm tend to be super heteronormative

OUTLINE

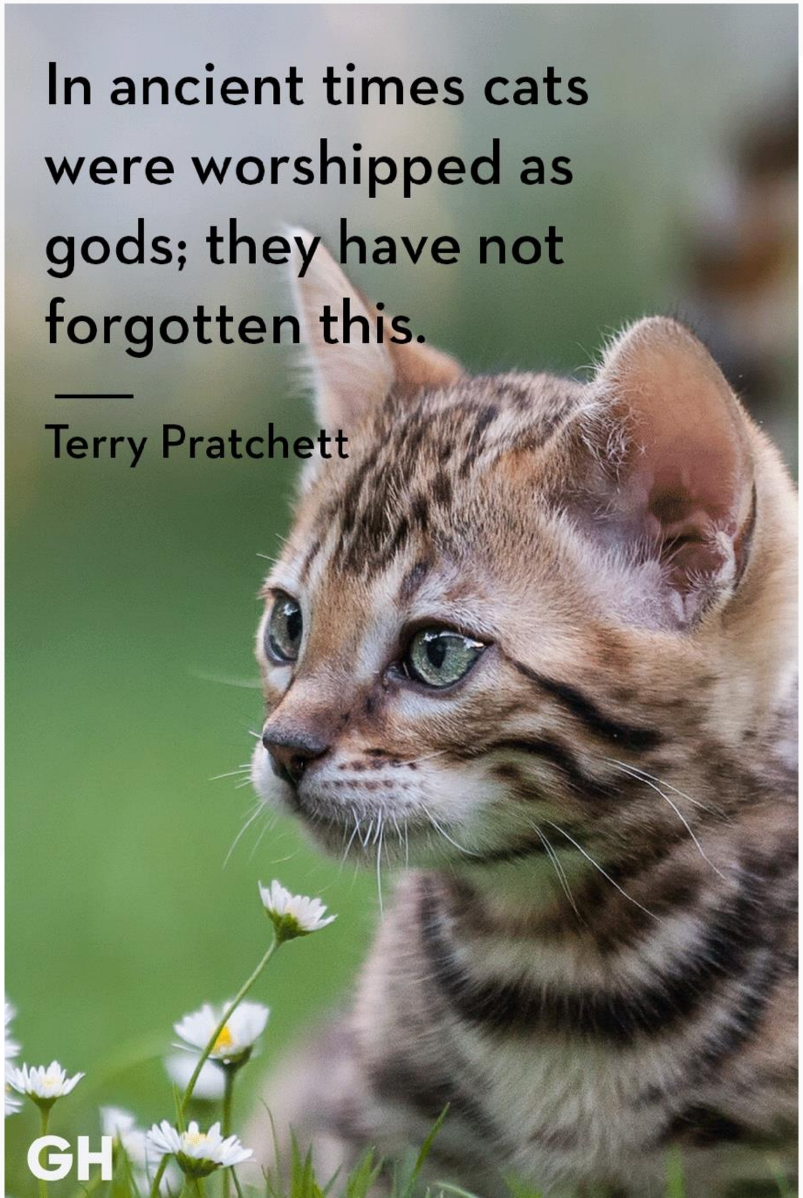
1. Web caching
2. Content-distribution networks
   - Featuring Akamai

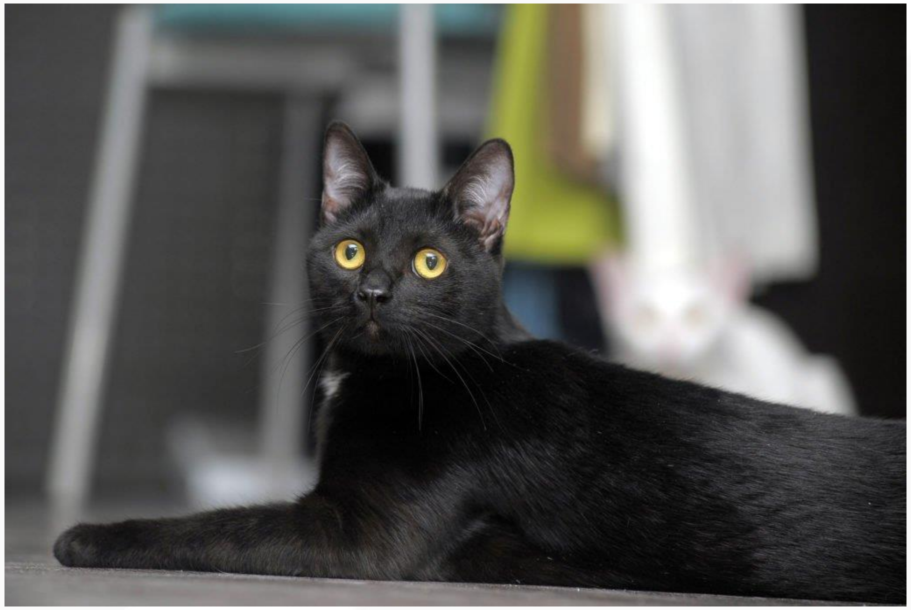
INTRO TO WEB REPLICATION



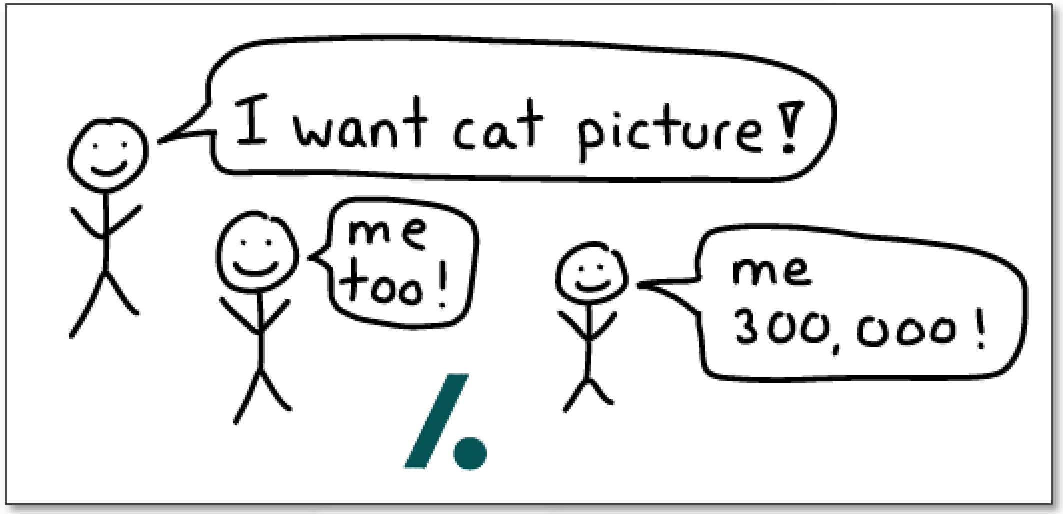
Images by Julia Evans

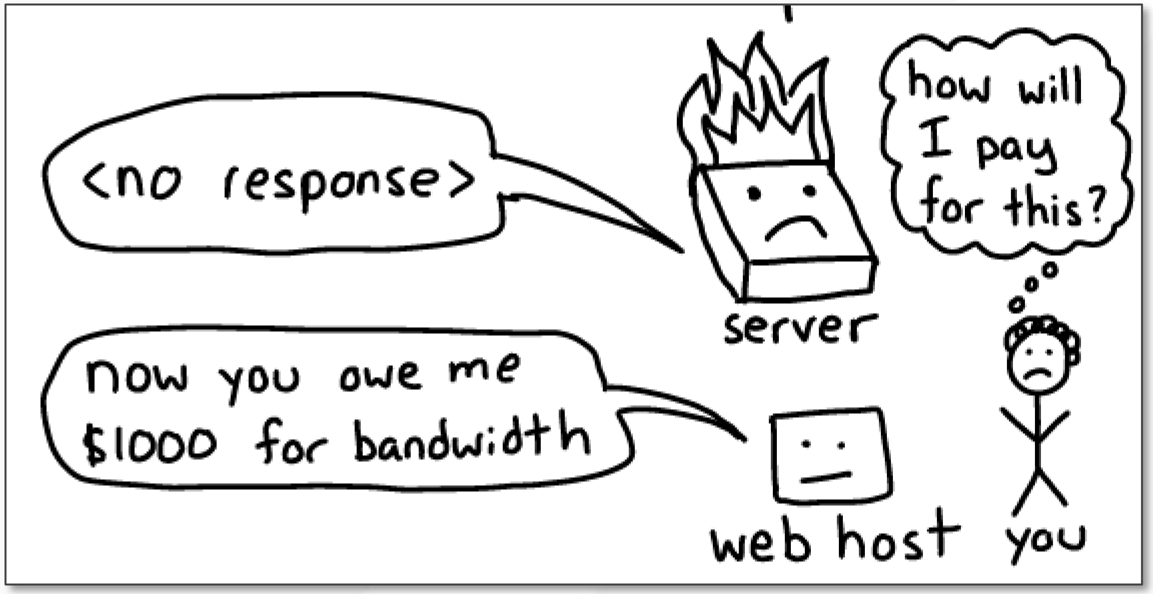
EXAMPLES OF CAT PICTURES FROM THE INTERNET


In ancient times cats were worshipped as gods; they have not forgotten this.
Terry Pratchett
GH



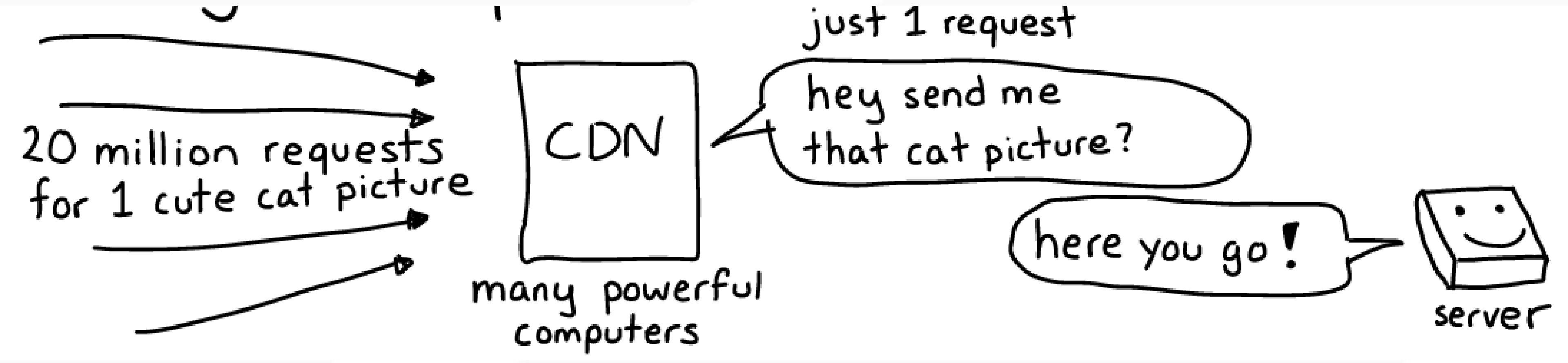
INTRO TO WEB REPLICATION





Images by Julia Evans

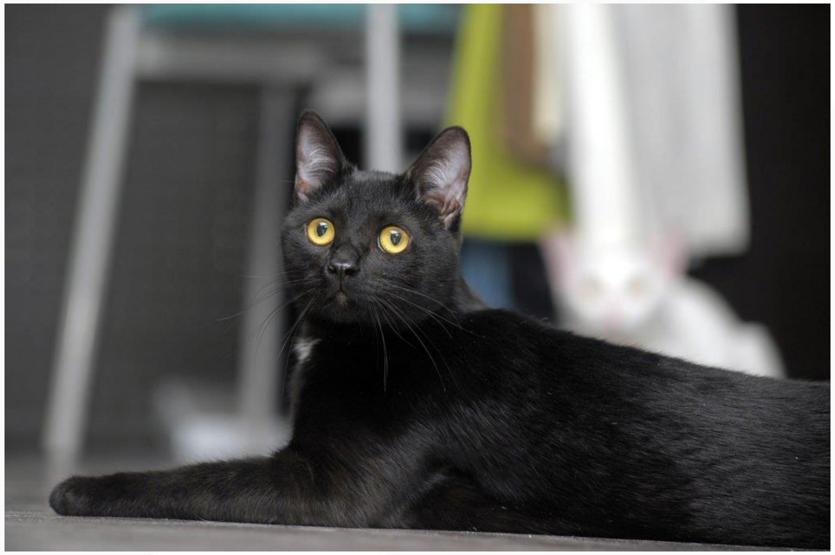
4 COMPONENTS TO CONTENT DISTRIBUTION NETWORKS


20 million requests for 1 cute cat picture
CDN
many powerful computers
just 1 request
hey send me that cat picture?
here you go!
server

1. **Proxies**: How to get web content from a server different than the original one?
2. **Caching**: OK, but then what if the original server updates/changes the content?
3. **Load balancing**: How do I choose which proxy/cache?
4. **Availability**: What if some of the proxies or caches fail?

Images by Julia Evans

CAT PHOTOS… TO MILLIONS OF USERS?



- Scenario:
    - Use HTTP to serve cat pictures to millions of users
    - How does that *scale*?

WEB CACHING

- Many clients rmation
  - Generates r ork load
  - Also, clients cy

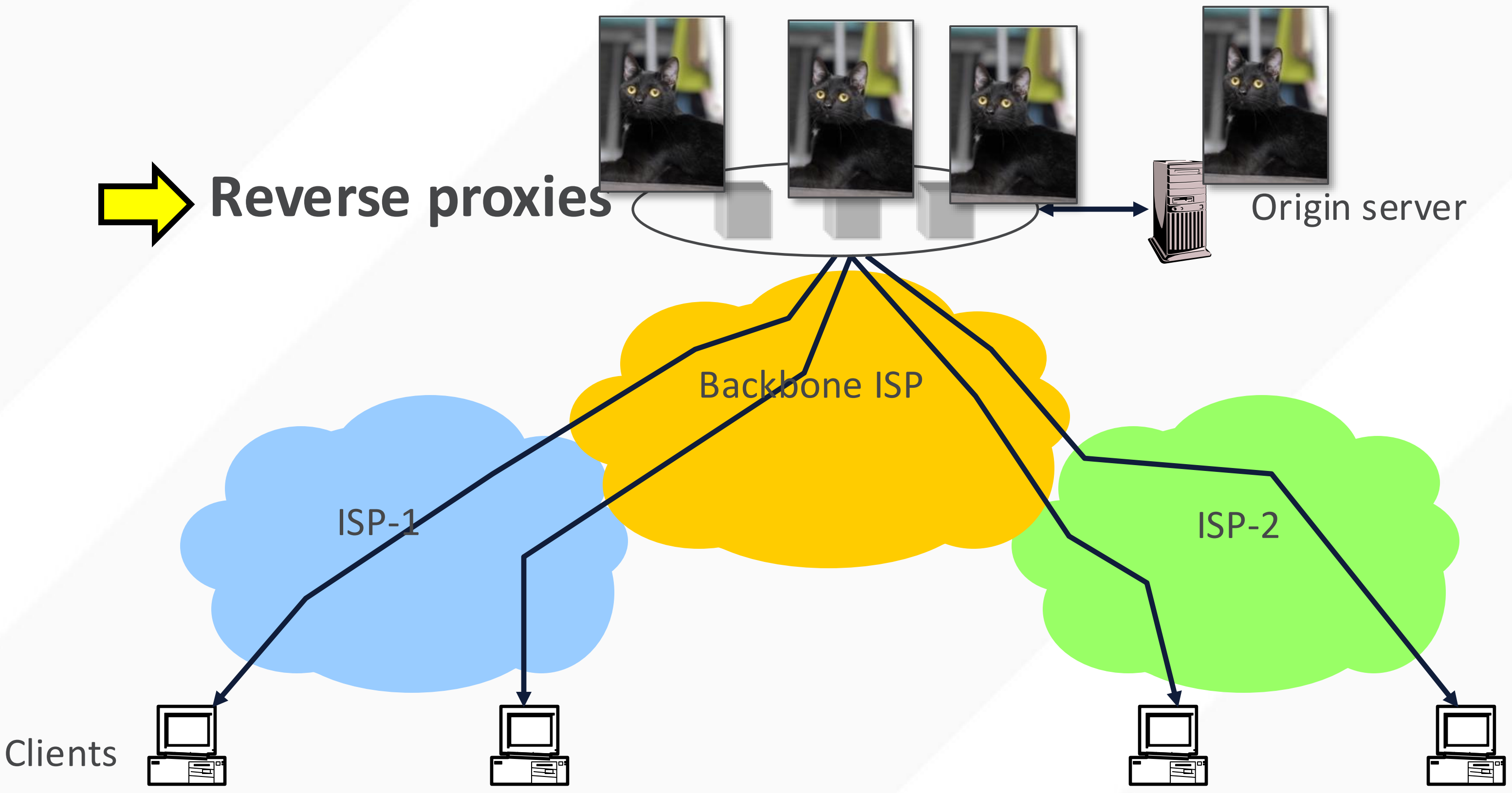
ISP-1
ISP-2
Clients

WHY WEB CACHING?

- Motivation for **placing content closer to client**:
  - User gets **better response time**
    - Content providers get happier users
  - Network gets **reduced load**
- Why does caching work? Exploits locality of reference
- How well does caching work?
  - Very well, **up to a limit**
  - Large overlap in content
  - But many unique requests

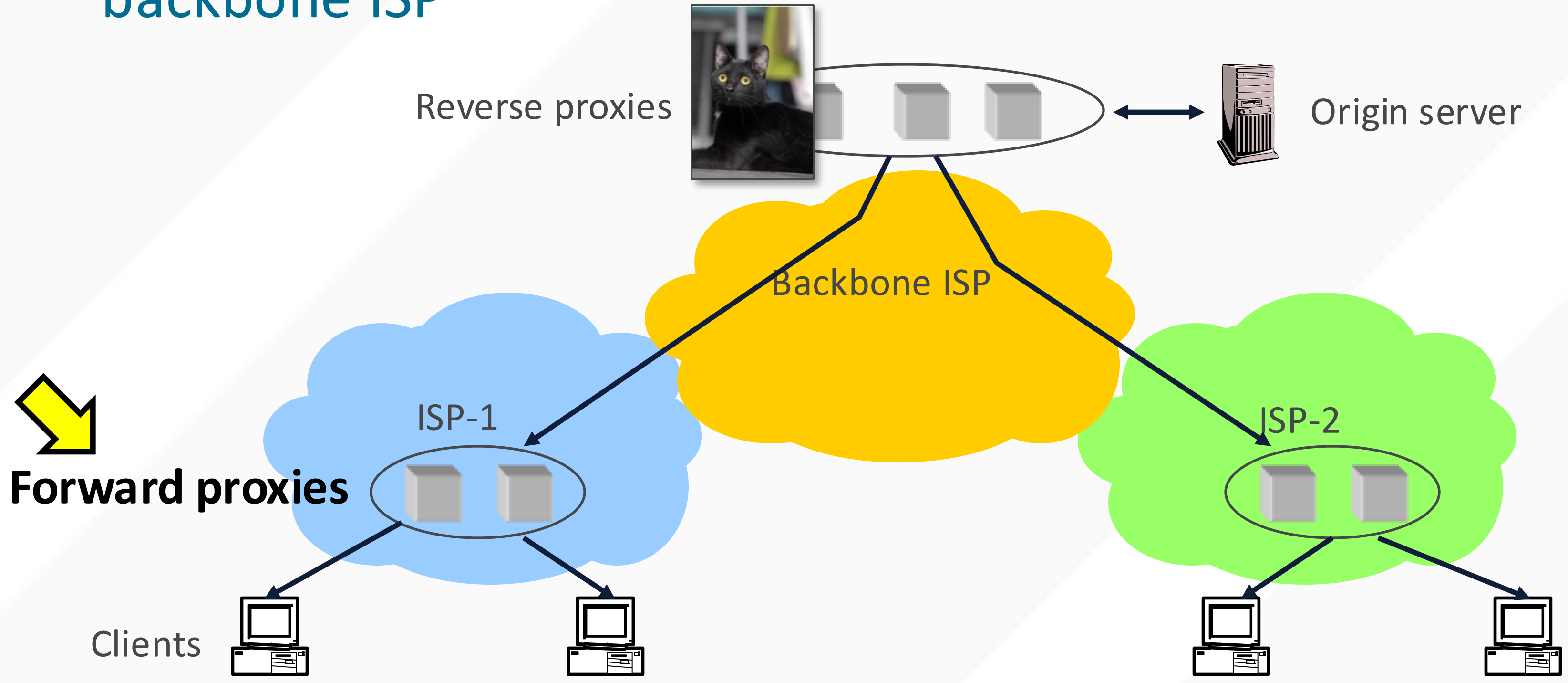
CACHING WITH REVERSE PROXIES

- Cache data close to origin server → decrease server load
  - Client thinks it is talking to the origin server (the server with content)
- Does not work for **dynamic content**



CACHING WITH FORWARD PROXIES

- Cache close to clients → less network traffic, less latency
  - Typically done by ISPs or corporate LANs
  - **Client configured** to send HTTP requests to forward proxy
- Reduces traffic on ISP-1's access link, origin server, and backbone ISP



CACHING & LOAD-BALANCING: OUTSTANDING PROBLEMS

- Problem *ca.* 2002: *How to reliably deliver large amounts of content to users worldwide?*
  - Popular event: **"Flash crowds" overwhelm** (replicated) web server, access link, or back-end database infrastructure
  - More rich content: audio, video, photos

- Web caching: Diversity causes **low cache hit rates (25–40%)**

GETTING CURL TO USE A WEB PROXY

- curl -v -x webproxy.ucsd.edu:3128 -o /dev/null https://cseweb.ucsd.edu/~gmporter/index.html

PROXY CACHES

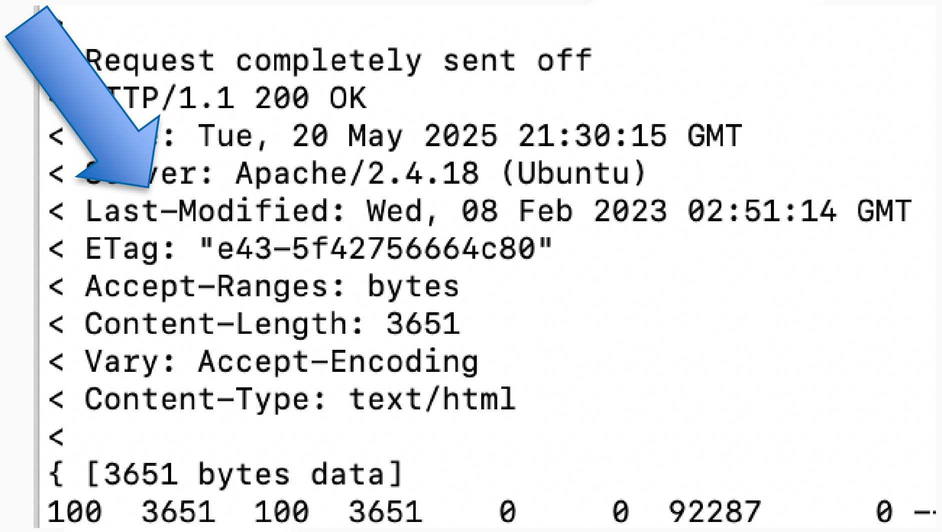
- Let's have the proxy also cache copies of documents
  - Reduce latency
  - Reduce bandwidth to origin server
  - Share document across local users

- But how does the proxy "know" if the original document has been updated?

APPROACH 1: IF-MODIFIED-SINCE (TIME-BASED)

```
curl --http1.1 -o /dev/null -v https://c3lab.net
```

- Response header:



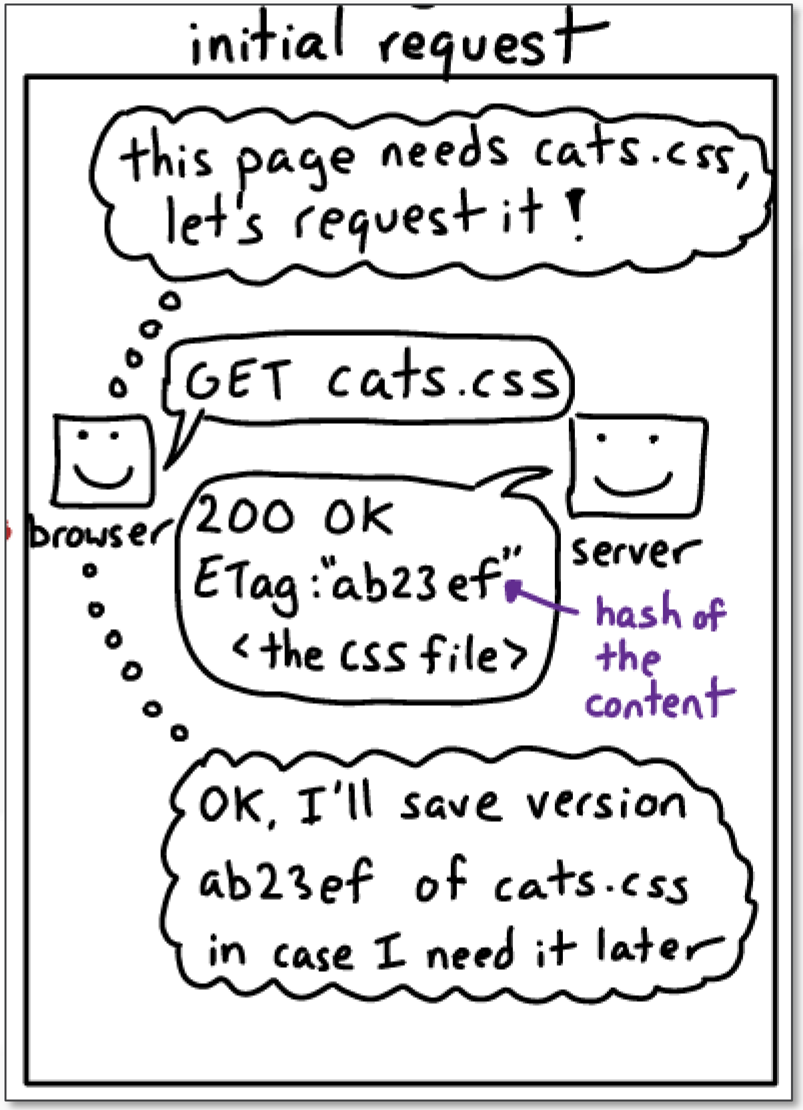
```
  Request completely sent off
   TTP/1.1 200 OK
<    : Tue, 20 May 2025 21:30:15 GMT
<    ver: Apache/2.4.18 (Ubuntu)
< Last-Modified: Wed, 08 Feb 2023 02:51:14 GMT
< ETag: "e43-5f42756664c80"
< Accept-Ranges: bytes
< Content-Length: 3651
< Vary: Accept-Encoding
< Content-Type: text/html
<
{ [3651 bytes data]
100  3651  100  3651    0     0  92287      0 -
```

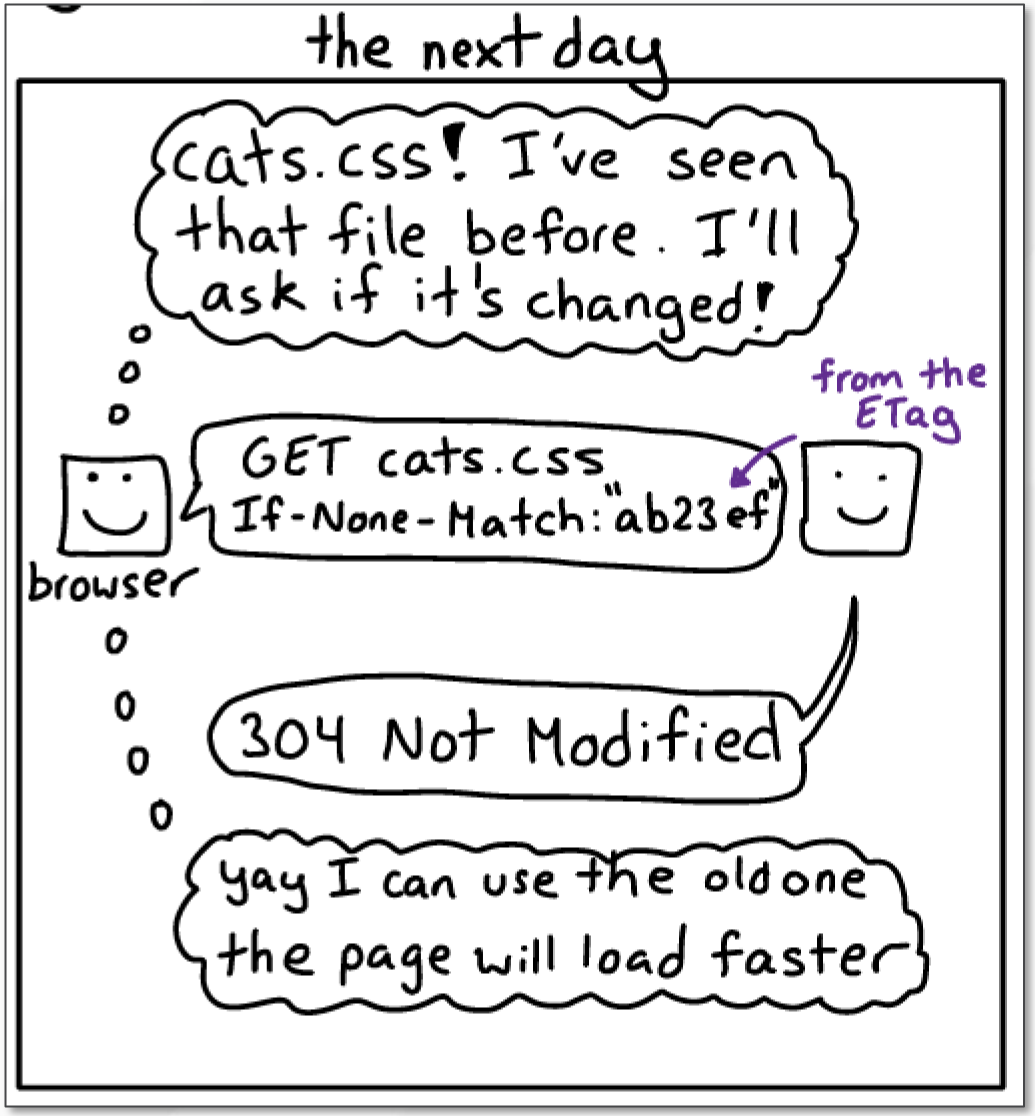
- Request header:
  - If-Modified-Since: <date>
    - Cache hit: 304 Not Modified
    - Cache miss: 200 OK

Question: If finding out it the browser's local cached copy is state or not requires a GET request, what is the benefit of returning 304 sometimes vs 200 every time?

```
curl --http1.1 -o /dev/null -H "If-Modified-Since: Thu, 02 Feb 2025 20:40:26 GMT" -v https://c3lab.net
```

APPROACH 2: ETAG/IF-NONE-MATCH (HASH-BASED)


initial request
this page needs cats.css, let's request it!
GET cats.css
browser
server
200 OK
ETag:"ab23ef"
<the css file>
hash of the content
OK, I'll save version ab23ef of cats.css in case I need it later


the next day
cats.css! I've seen that file before. I'll ask if it's changed!
from the ETag
GET cats.css
If-None-Match:"ab23ef"
browser
304 Not Modified
yay I can use the old one
the page will load faster

LET’S TRY IT

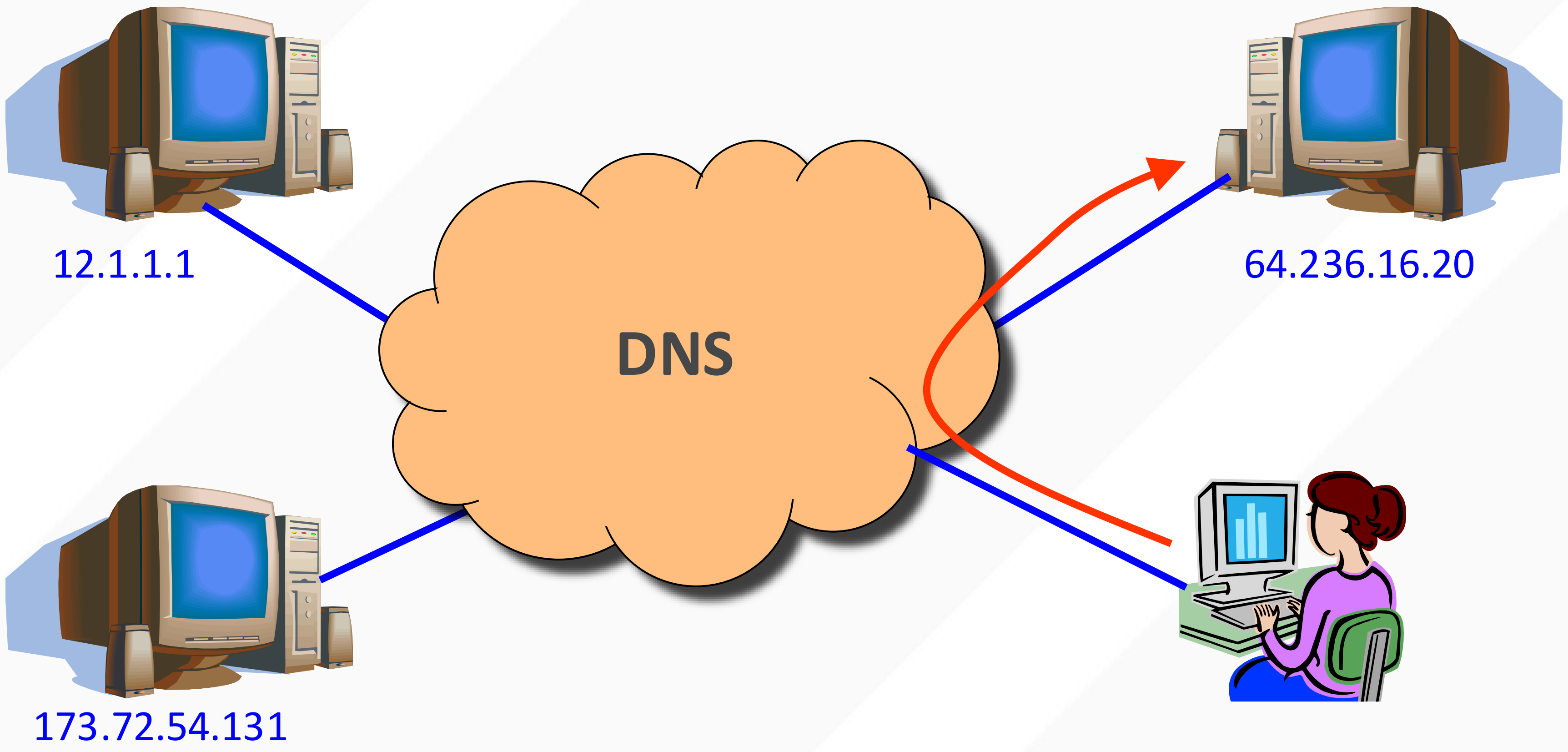
- Curl has a “-H” option to specify our own headers
- What happens if we pass in the previously seen Etag?
  1. curl --http1.1 -v -o /dev/null https://c3lab.net
  2. curl --http1.1 -v -o /dev/null -H "If-None-Match: \“<etag>\"" https://c3lab.net

REVERSE PROXIES: APPROACH 1 (STATIC)

- **Problem: Overloaded** popular web site
  - **Replicate** the site across multiple machines
    - Reverse proxies
- Want to direct client to a particular replica. Why?
  - **Balance load** across server replicas
- **Solution #1:** Manual selection by clients
  - Each replica has its own name (www1, www2, www3, etc)
  - Some Web page lists replicas (*e.g.*, by name, location), asks clients to click link to pick

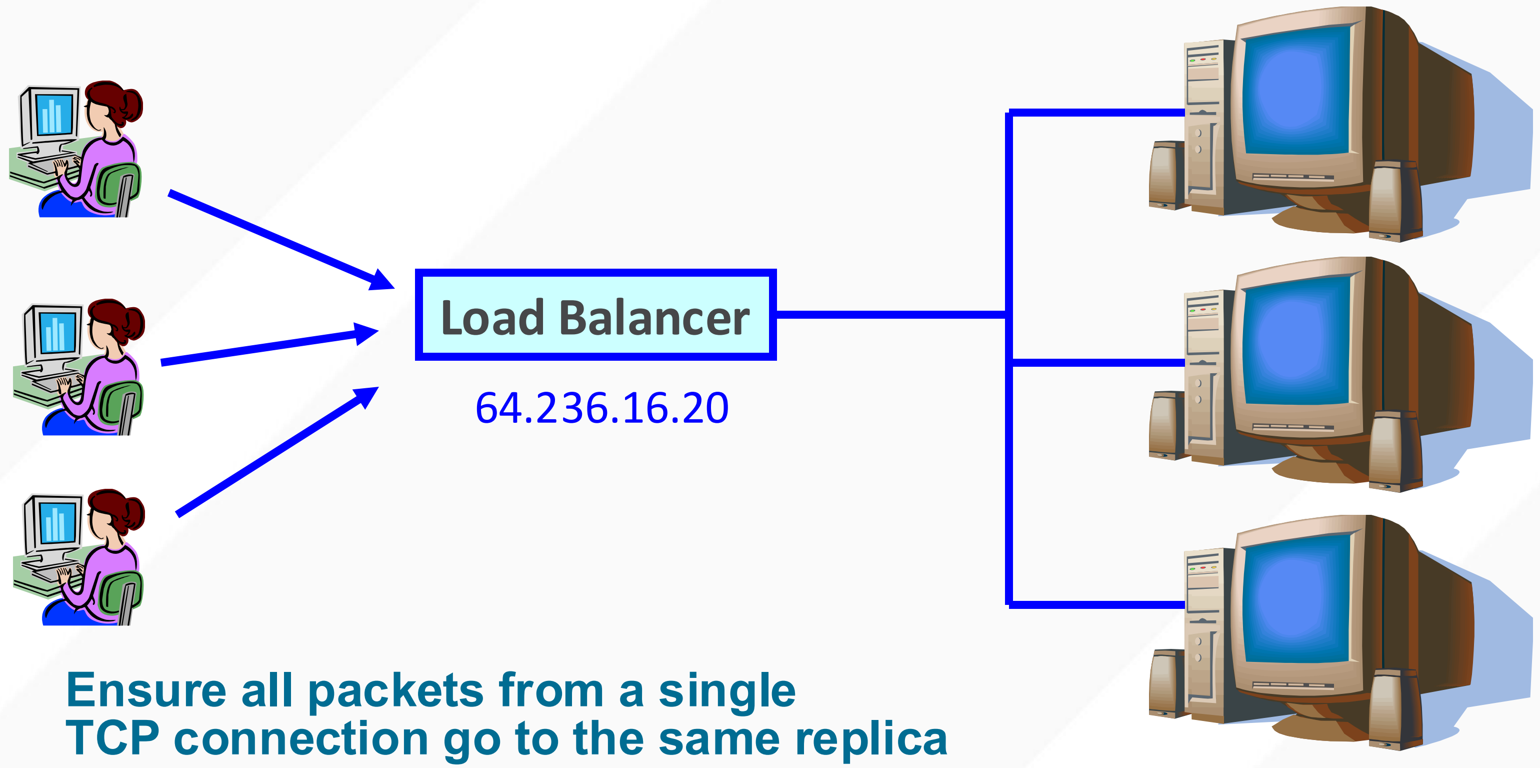
REVERSE PROXIES: APPROACH 2 (DNS)

- Multiple IP addresses, multiple machines
  - Same DNS name but different IP for each replica
    - DNS server returns IP addresses “round robin”



REVERSE PROXIES: APPROACH 3 (LOAD BALANCER)

- Single IP address, multiple machines
  - Run multiple machines behind a single IP address



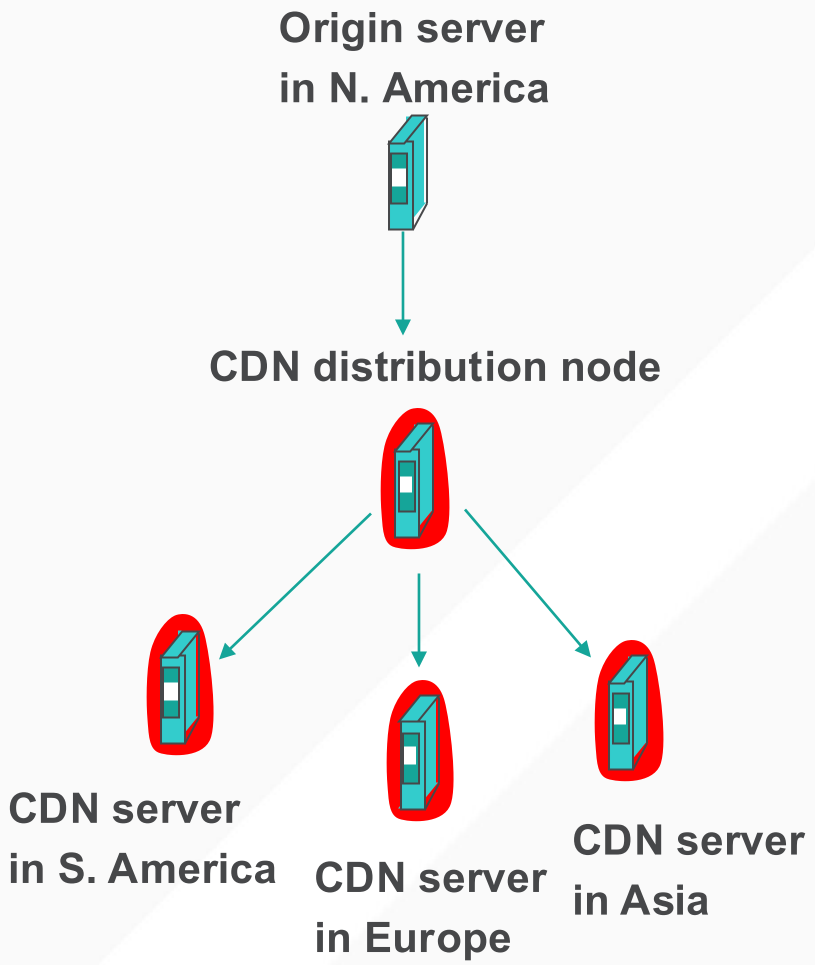
  - **Ensure all packets from a single TCP connection go to the same replica**
  - ***But how to share one IP across multiple backend servers?***

OUTLINE

1. Web caching
2. Content-distribution networks
   - Featuring Akamai

CONTENT DISTRIBUTION NETWORKS

- **Proactive content replication**
  - Content provider (*e.g.* CNN) pushes content out from its own ***origin server***
- CDN **replicates** the content
  - On many servers spread throughout the Internet
- Updating the replicas
  - Updates **pushed to replicas** when the content changes



REPLICA SELECTION: GOALS

- **Live** server
  - For availability

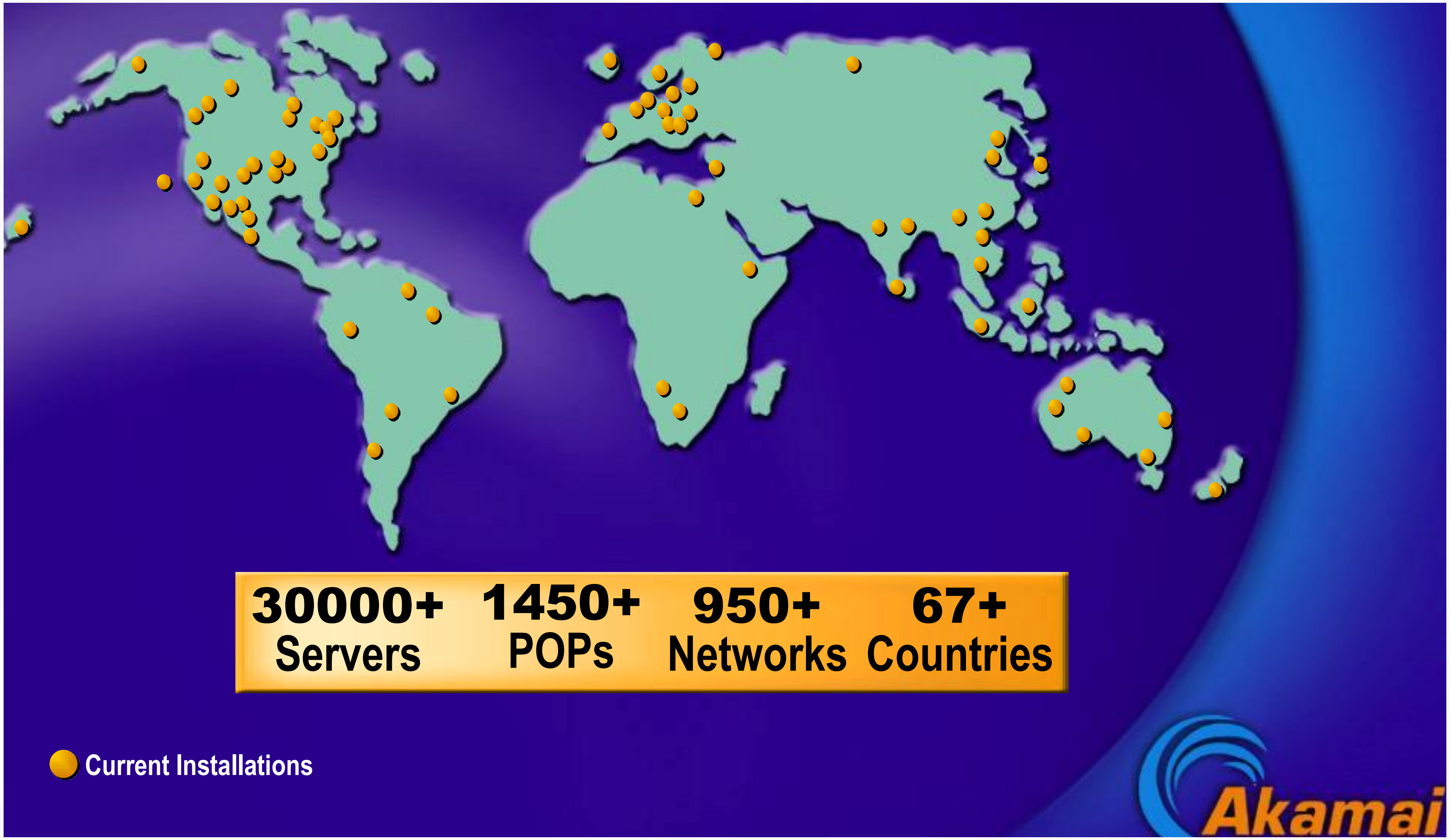
Requires continuous monitoring of liveness, load, and performance

- Lowest **load**
  - To balance load across the servers
- **Closest**
  - Nearest geographically, or in round-trip time
- Best **performance**
  - Throughput, latency, reliability…

AKAMAI

- Deployment
  - 147K+ servers, 1200+ networks, 650+ cities, 92 countries
  - highly hierarchical, caching depends on popularity
  - 4 yr depreciation of servers
  - Many servers inside ISPs, who are thrilled to have them
  - Deployed inside100 new networks in last few years
- Customers
  - 250K+ domains: all top 60 eCommerce sites, all top 30 M&E companies, 9 of 10 top banks, 13 of top 15 auto manufacturers
- Overall stats
  - 5+ terabits/second, 30+ million hits/second, 2+ trillion deliveries/day, 100+ PB/day, 10+ million concurrent streams
  - 15-30% of Web traffic

CIRCA 2007 OR SO

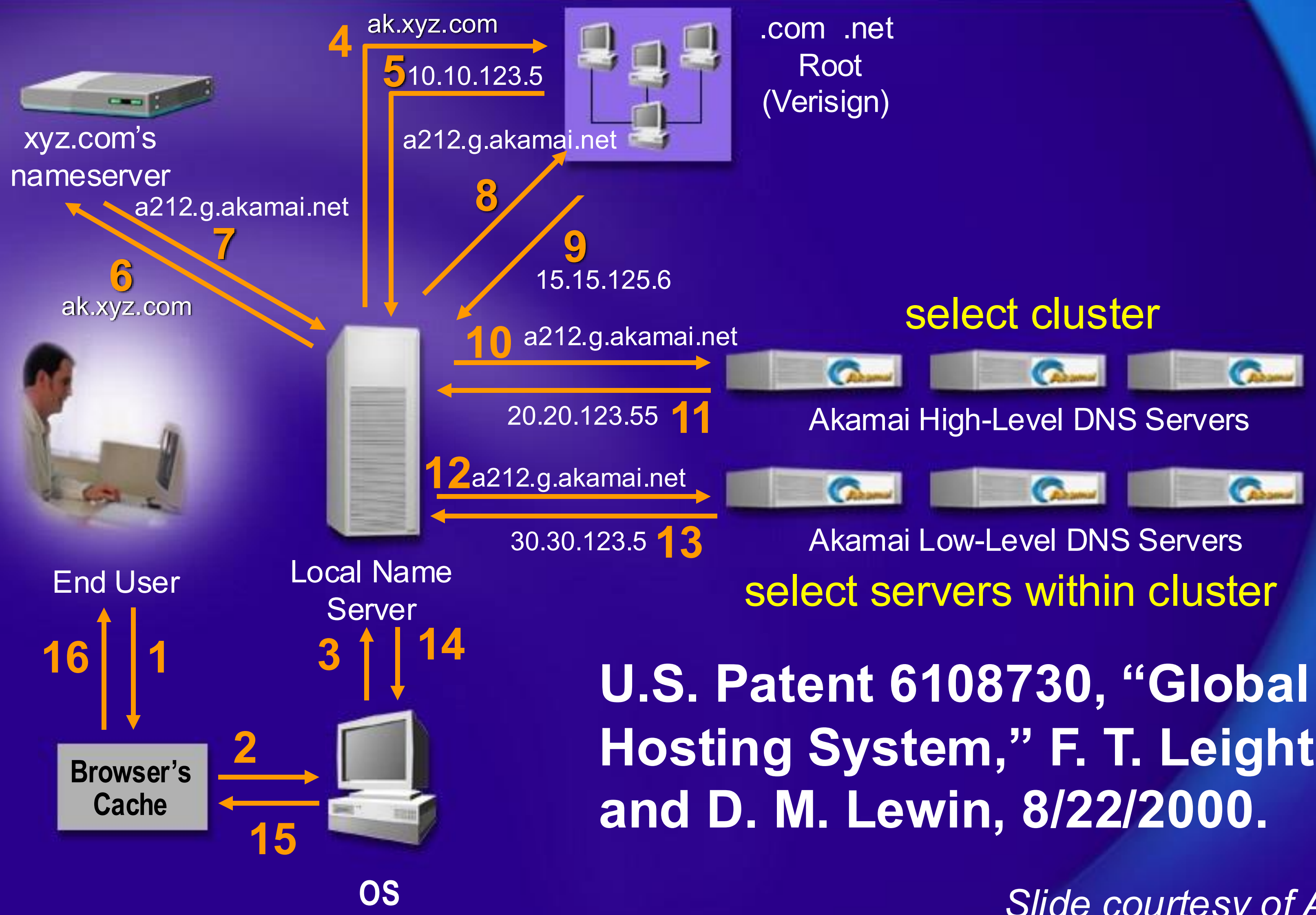


EMBEDDED IMAGE DELIVERY

```
<html>
<head>
<title>Welcome to xyz.com!</title>
</head>
<body>
<img src="http://www.xyz.com/logos/logo.gif">
<img src="http://www.xyz.com/jpgs/background.jpg">
<h1>Welcome to our Web site!</h1>
<a href="page2.html">Click here to enter</a>
</body>
</html>
```

Replace "www" with "ak"

Akamai DNS Resolution



U.S. Patent 6108730, "Global Hosting System," F. T. Leighton and D. M. Lewin, 8/22/2000.

Slide courtesy of Akamai

OPTIMIZING PERFORMANCE: NETWORK

- There are good solutions to server load and content
  - What about network performance?
- Key challenges for network performance
  - Measuring paths is hard
    - Traceroute gives us only the forward path
    - Shortest path != best path
  - RTT estimation is hard
    - Variable network conditions
    - May not represent end-to-end performance
  - No access to client-perceived performance

OPTIMIZING PERFORMANCE: NETWORK

- Example approximation strategies
  - Geographic mapping
    - Hard to map IP to location
    - Internet paths do not take shortest distance
  - Active measurement
    - Ping from all replicas to all routable prefixes
    - 56B * 100 servers * 500k prefixes = 500+MB of traffic per round
  - Passive measurement
    - Send fraction of clients to different servers, observe performance
    - Downside: Some clients get bad performance

MAPPING SYSTEM

- Equivalence classes of IP addresses
  - IP addresses experiencing similar performance
  - Quantify how well they connect to each other
- **Collect and combine** measurements
  - Ping, traceroute, BGP routes, server logs
    - *e.g.*, over 100 TB of logs per days
  - Network latency, loss, throughput, and connectivity

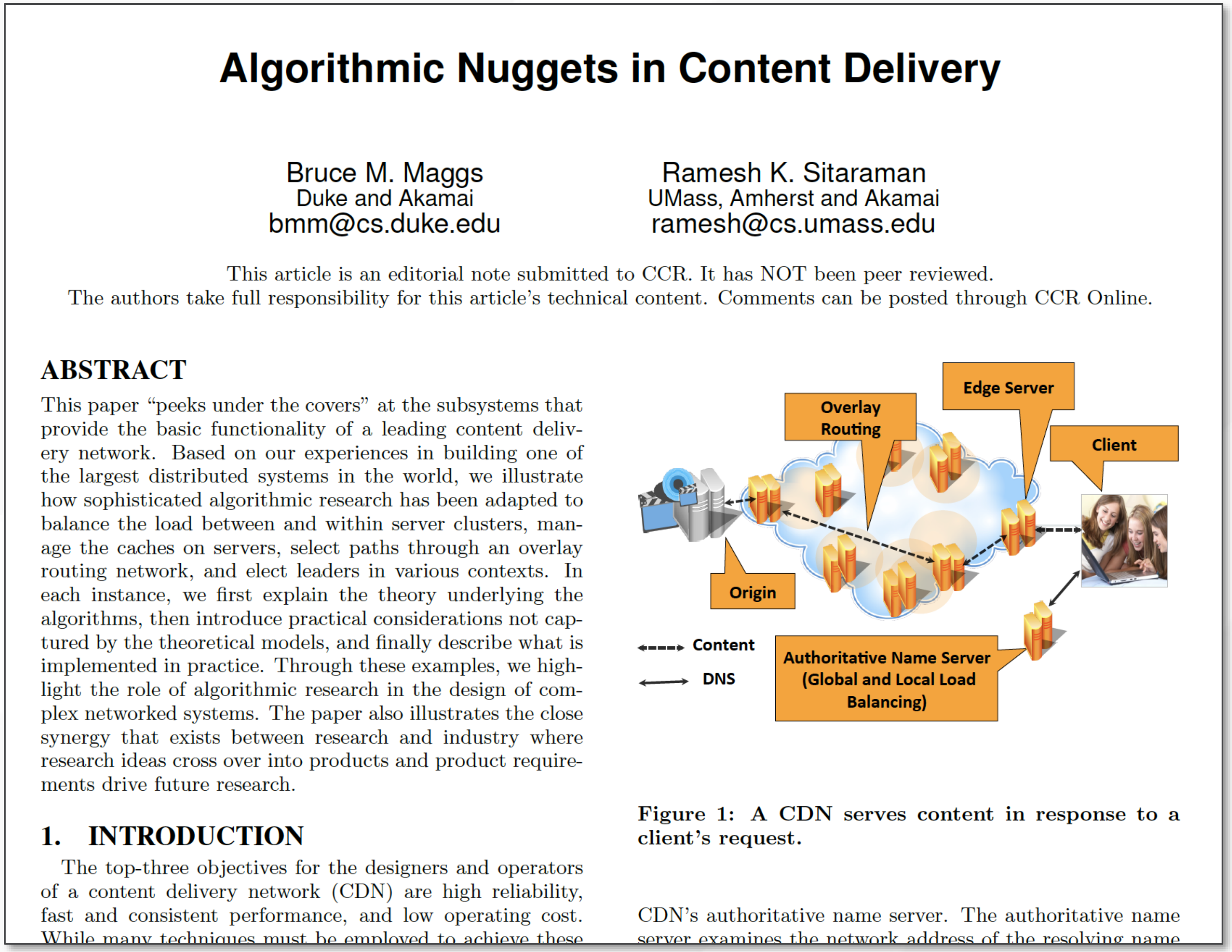
ROUTING CLIENT REQUESTS WITH THE MAP

- Map each **IP class** to a preferred **server cluster**
  - Based on performance, cluster health, etc.
  - Updated roughly every minute
    - **Short, 60-sec DNS TTLs** in Akamai regional DNS accomplish this
- Map client request to a server in the cluster
  - **Load balancer** selects a specific server
  - *e.g.*, to **maximize** the **cache hit rate**

ADAPTING TO FAILURES

- Failing **hard drive** on a server
  - Suspends after finishing “in progress” requests
- Failed **server**
  - Another server takes over for the IP address
  - Low-level map updated **quickly** (load balancer)
- Failed **cluster**, or **network path**
  - High-level map updated **quickly** (ping/traceroute)

AN IN-DEPTH LOOK AT AKAMAI’S ALGORITHMS



Algorithmic Nuggets in Content Delivery

Bruce M. Maggs
Duke and Akamai
bmm@cs.duke.edu

Ramesh K. Sitaraman
UMass, Amherst and Akamai
ramesh@cs.umass.edu

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article’s technical content. Comments can be posted through CCR Online.

ABSTRACT

This paper “peeks under the covers” at the subsystems that provide the basic functionality of a leading content delivery network. Based on our experiences in building one of the largest distributed systems in the world, we illustrate how sophisticated algorithmic research has been adapted to balance the load between and within server clusters, manage the caches on servers, select paths through an overlay routing network, and elect leaders in various contexts. In each instance, we first explain the theory underlying the algorithms, then introduce practical considerations not captured by the theoretical models, and finally describe what is implemented in practice. Through these examples, we highlight the role of algorithmic research in the design of complex networked systems. The paper also illustrates the close synergy that exists between research and industry where research ideas cross over into products and product requirements drive future research.

1. INTRODUCTION

The top-three objectives for the designers and operators of a content delivery network (CDN) are high reliability, fast and consistent performance, and low operating cost. While many techniques must be employed to achieve these

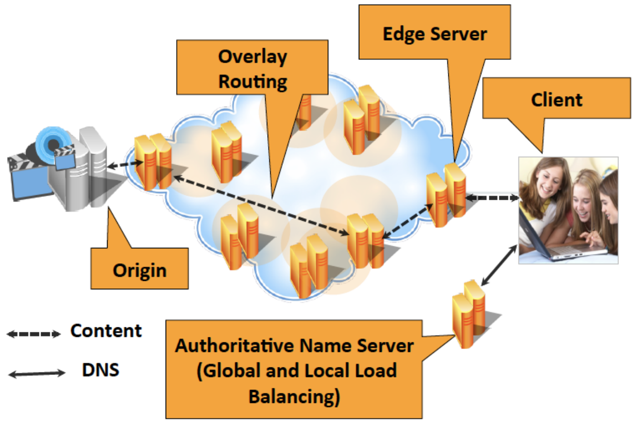


Figure 1: A CDN serves content in response to a client’s request.

CDN’s authoritative name server. The authoritative name server examines the network address of the resolving name

ALGORITHMIC TOPICS CRITICAL TO BUILDING CDNS

- Stable allocations w/ resource trees
- Consistent hashing w/ popular items
    - TritonTube project (lab 8)
- Bloom filters
- Overlay routing
- Leader election and consensus
    - Fault-tolerant TritonTube (lab 9)

TAKE-AWAY POINTS: CDNS

- Content distribution is hard
  - Many, diverse, changing objects
  - Clients distributed all over the world
- **Moving content to the client** is key
  - Reduces latency, improves throughput, reliability
- Content distribution solutions **evolved:**
  - Load balancing, reactive caching, to
  - Proactive content distribution networks

UC San Diego