Data distribution: web servers, CDN, cloud, data center

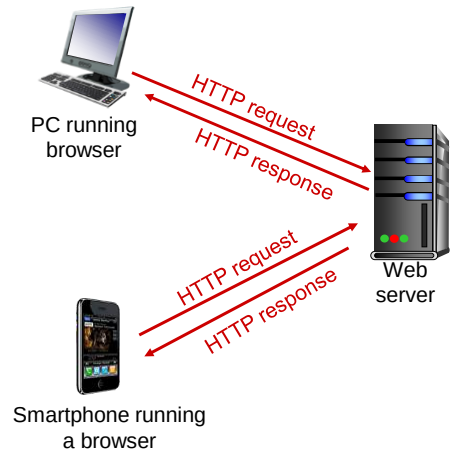
Andrea Bianco
Telecommunication Network Group
firstname.lastname@polito.it
http://www.telematica.polito.it/

Contents in the Web

- Most infos in the Internet are Web contents that use HTTP (or HTTPS)
  - Web is more than half the total traffic
- A web page consists of a of base HTML-file which includes several objects
  - An object can be HTML file, JPEG image, Java applet, audio file,…
  - Each object is addressable by a URL, composed of host name and path (e.g., www.telematica.polito.it/public/faculty)
  - Object size is small (average is less than 100KB, with median around 3KB)

HTTP overview

- HTTP: hypertext transfer protocol
- Web's application layer protocol
- client/server model
  - client: browser that requests, receives, (using HTTP protocol) and "displays" Web objects
  - server: Web server sends (using HTTP protocol) objects in response to requests



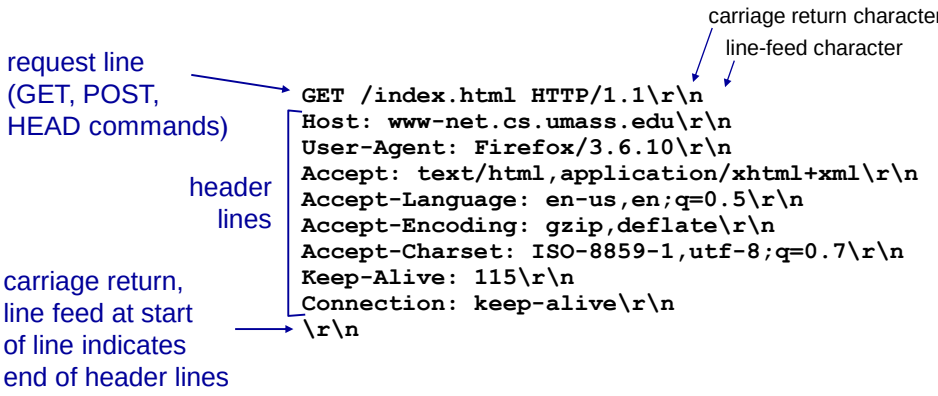
HTTP overview

- Uses TCP:
- Client initiates TCP connection (creates socket) to server, port 80
- Server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed

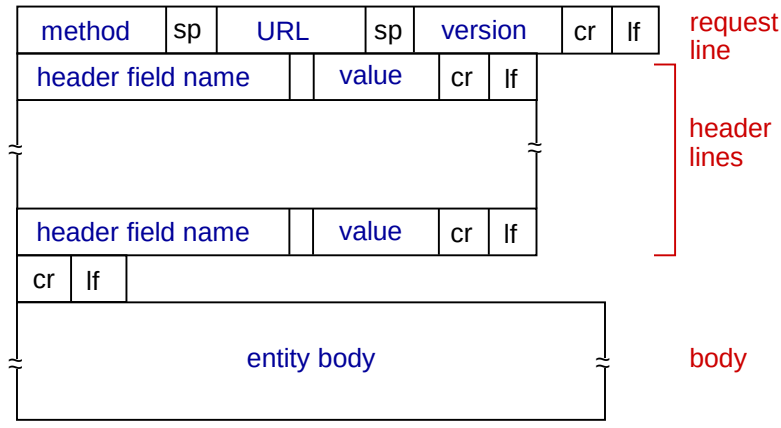
- HTTP is "stateless"
- Server maintains no information about past client requests
- Stateful operation is complex
  - Need to keep history (state)
  - In case of crash, inconsistent views may be solved

HTTP request message

- Two types of HTTP messages: request, response
- HTTP request message:
  - ASCII (human-readable format)



HTTP request message: general format



HTTP response message

status line (protocol status code status phrase)

header lines

data, e.g., requested HTML file

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-1\r\n
\r\n
data data data data data ...
```

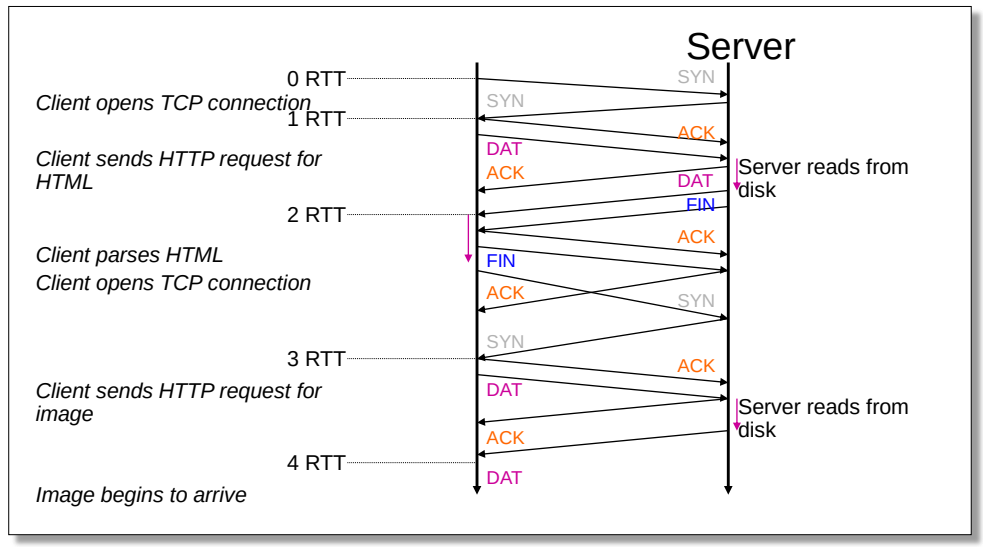

HTTP status code

Codice	Significato
100	Continue, prosegui con operazioni
101	Switching protocol, cambiato protocollo di comunicazione
200	OK
201	Created, URL creata
202	Accepted, comando accettato (ma non ancora eseguito)
300	Multiple choice, possibile scegliere tra più formati
301	Mover parmanently, URL dislocata altrove
303	Not modified, risorsa non modificata (risposta a GET condizionale)
400	Bad request, il server non capisce
401	Unauthorized
402	Payment required
404	Not found
405	Method not allowed
410	Gone, la risorsa non è disponibile
415	Unsupported media type
500	Internal server error
503	Service unavailable
504	Gateway timetout

HTTP connections

- Non-persistent HTTP
  - at most one object sent over TCP connection
  - connection then closed
  - downloading multiple objects requires multiple connections
- Persistent HTTP
  - multiple objects can be sent over single TCP connection between client and server

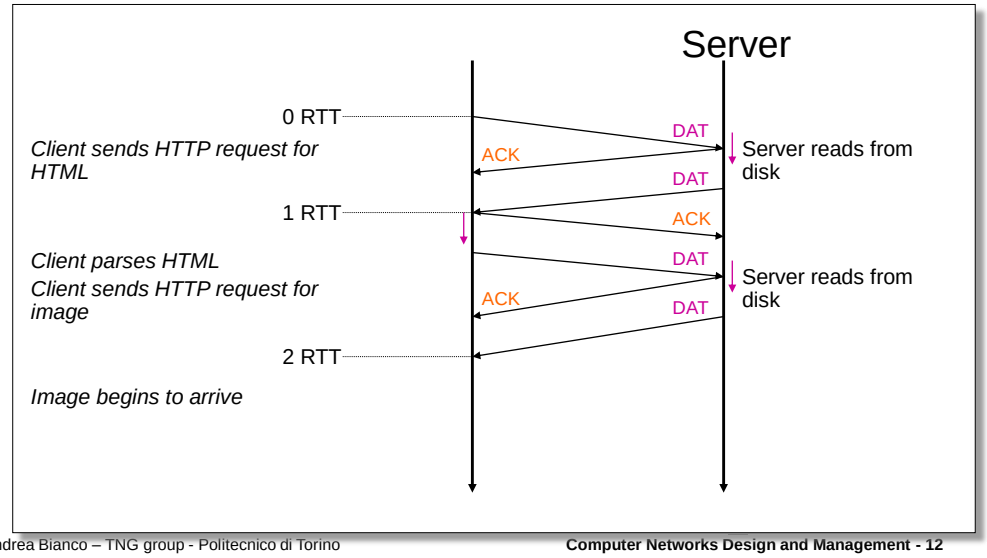
Non-persistent example



Problems with non-persistent connections

- Multiple connection setups
  - Three-way handshake each time
- Round-trip time estimation
  - Maybe large at the start of a connection (e.g., 3 seconds)
  - Leads to latency in detecting lost packets
- Short transfers are hard on TCP
  - Stuck in slow start (beginning of TCP connection is slow, at low bitrate)
  - Loss recovery is poor when windows are small
- Lots of extra connections

Persistent connection example



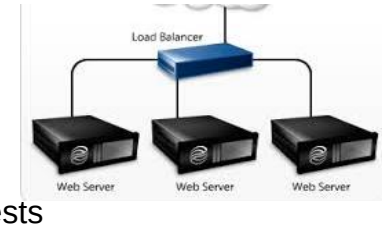
Non-persistent vs persistent HTTP

- Non-persistent HTTP issues:
  - Requires 2 RTTs per object
  - OS must allocate resources for each TCP connection
- Persistent HTTP
  - Server leaves connection open after sending response
  - Subsequent HTTP messages between same client/server are sent over connection
  - Must serialize
  - Browsers often open parallel TCP connections to fetch referenced objects
    - Potential throughput improvement
    - Especially used for non persistent HTTP

Replicated Web service

- For site reliability and scalability
  - Use multiple servers
- Disadvantages
  - How do you decide which server to use?
  - How to synchronize state among servers?

Load balancers



- Device that multiplexes requests across a collection of servers
  - All servers share one public IP
  - Balancer transparently directs requests to different servers
- The balancer assigns clients to servers
  - Random / round-robin
  - Load-based
- Challenges
  - Scalability (must support traffic for n hosts)
  - State (must keep track of previous decisions)

Load balancing

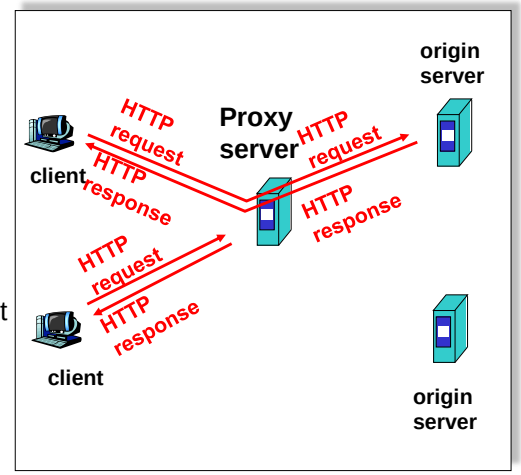
- Advantages
  - Allows scaling of hardware independent of IPs
  - Relatively easy to maintain
- Disadvantages
  - Expensive
  - Still a single point of failure
  - Difficult to choose the placement

HTTP performance

- Where should the server go?
  - For Web pages, RTT matters most
  - For video, available bandwidth matters most (not only)
- Is there one location that is best for everyone?
- Idea: caching!
  - Clients often cache documents
    - When should origin be checked for changes?
    - Every time? Every session? Date?
  - HTTP includes caching information in headers and includes rules for document expiration
  - If not expired, use cached copy
  - If expired, use GET request to origin

Web Proxy Caches

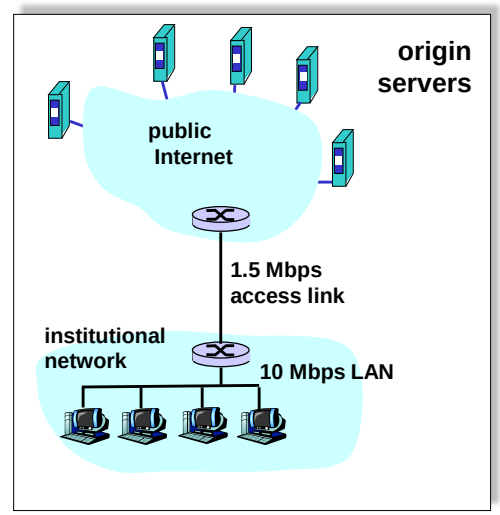
- GOAL: satisfy client request without involving origin server
- User configures browser: Web accesses via cache
- Browser sends all HTTP requests to cache
  - Object in cache: cache returns object
  - Else: cache requests object from origin, then returns to client



Caching example

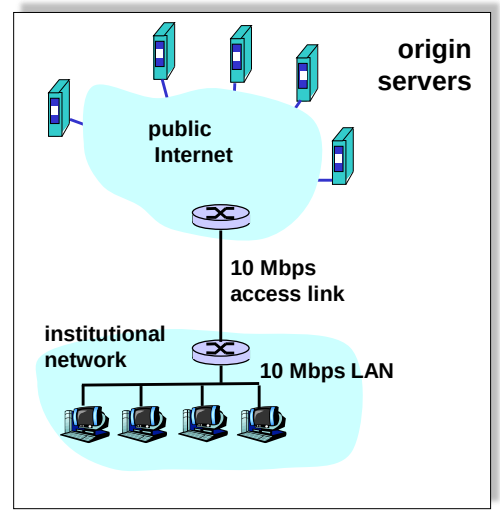
- Assumptions
  - Avg object size = 100K bits
  - Avg. request rate from browsers to origin servers = 15/sec
  - Delay from institutional router to any origin server and back to router = 2 sec
- Consequences
  - Utilization on LAN = 15%
  - Utilization on access link = 100%
  - Total delay = Internet delay + access delay + LAN delay
  - = 2 sec + minutes + ms

Congestion



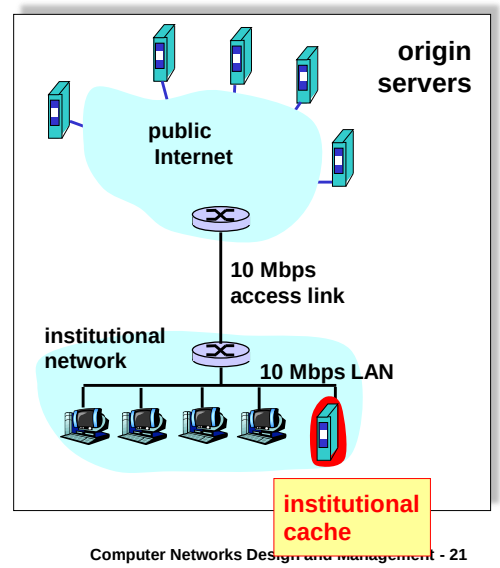
Caching example

- Possible Solution
  - Increase bandwidth of access link to, say, 10 Mbps
  - Costly upgrade
- Consequences
  - Utilization on LAN = 15%
  - Utilization on access link = 15%
  - Total delay = Internet delay + access delay + LAN delay
  - = 2 sec + ms + ms ≈ 2 s



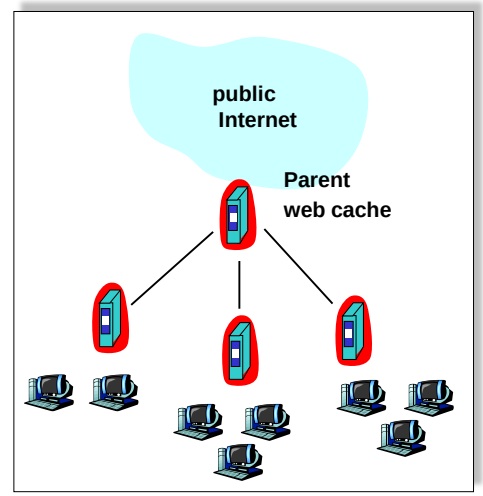
Caching example

- Install Cache
  - Support hit rate is 40%
- Consequences
  - 40% requests satisfied almost immediately (say 10 msec)
  - 60% requests satisfied by origin
  - Utilization of access link down to 60%, yielding negligible delays
  - Weighted average of delays
  - = .6*2 s + .4*10 ms < 1.3 s
  - Cheaper and more effective



Multiple caches

- What if the working set is > proxy disk?
- A static hierarchy
  - Check local
  - If miss, check parent
  - If miss, fetch through parent
- Internet Cache Protocol (ICP)
  - ICPv2 in RFC 2186 (& 2187)
  - UDP-based, short timeout



Web caches

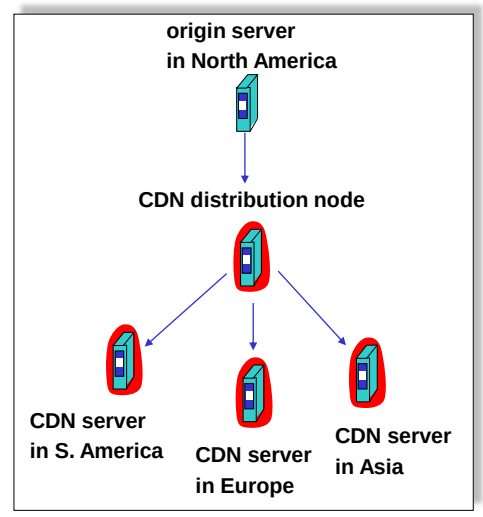
- Pros
  - Better performance: content is closer to users
  - Lower cost: content traverses network boundary once
- Problems
  - Size of all Web content is too large
    - Zipf distribution limits cache hit rate
  - Significant fraction (>50%?) of HTTP objects un-cachable
    - Web content is **dynamic** and **customized**
      - Can't cache banking content, stock price, web cams
      - Results based on parameters of passed data
      - Encrypted data

Content Delivery Networks: CDNs

- Content Delivery (or Distribution) Networks
  - At least half of the world's bits are delivered by a CDN
- Primary Goals
  - Create replicas of content throughout the Internet
  - Ensure that replicas are always available
  - Direct clients to replicas that will give good performance

CDNs

- Content providers are CDN customers
- Content replication
- CDN company installs thousands of servers throughout Internet
  - In large datacenters
  - or close to users
- CDN replicates customers' content
- When provider updates content, CDN updates servers



Examples of CDNs

- Akamai
  - 147K+ servers, 1200+ networks, 650+ cities, 92 countries
- Limelight
  - Well provisioned delivery centers, interconnected via a private fiber-optic connected to 700+ access networks
- Edgecast
  - 30+ PoPs, 5 continents, 2000+ direct connections
- Others
  - Google, Facebook, AWS, AT&T, Level3, Brokers

Inside a CDN

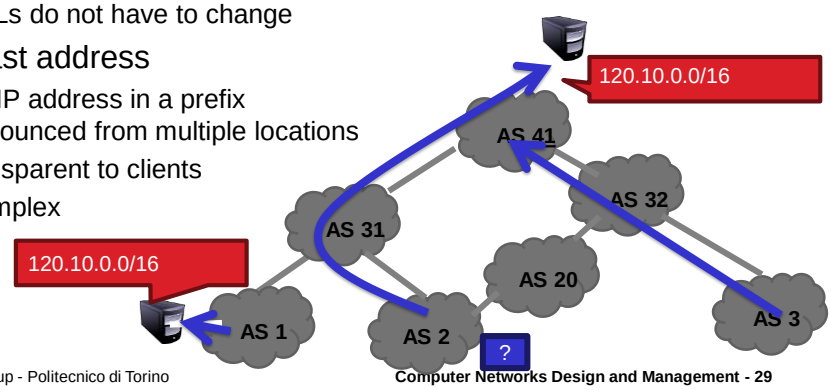
- Servers are deployed in clusters for reliability
  - Some may be offline
    - Could be due to failure
    - Also could be “suspended” (e.g., to save power or for upgrade)
- Could be multiple clusters per location (e.g., in multiple racks)
- Server locations
  - Well-connected points of presence (PoPs)
  - Inside of ISPs

CDN challenges

- Replicate content on many servers
  - How to replicate content
  - Where to replicate content
- Which server? The “best”
  - Least loaded: to balance load on servers
  - Best performance: to improve client performance
    - Based on Geography? RTT? Throughput? Load?
  - Any alive node: to provide fault tolerance
  - The best server can change over time
    - It depends on client location, network conditions, server load, …
- Existing technology that can be used to direct clients towards replica
  - As part of routing: anycast, cluster load balancing
  - As part of application: HTTP redirect
  - As part of naming: DNS

Mapping clients to servers

- DNS-based redirection
  - DNS server directs client to one or more IPs based on requested content (url)
  - Use short expiration of DNS entries to limit the effect of caching
  - Good to use existing DNS infrastructure
  - URLs do not have to change
- Anycast address
  - An IP address in a prefix announced from multiple locations
  - transparent to clients
  - Complex

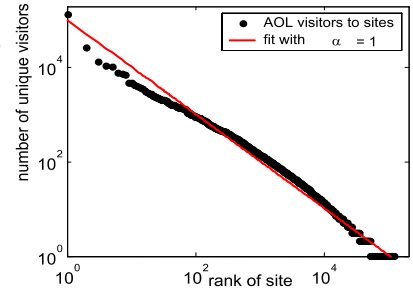


Optimizing performance

- Key goal
  - Send clients to server with the best end-to-end performance
- Performance depends on
  - Server load
  - Content at that server
  - Network conditions
- Optimizing for server load
  - Load balancing, monitoring at servers
  - Generally solved

Optimizing performance: caching

- Where to cache content?
  - Popularity of Web objects is Zipf-like
    - Also called heavy-tailed and power law
  - $N_r \sim r^{-1}$
  - Small number of sites cover large fraction of requests



- Given this observation, how should cache-replacement work?

Optimizing performance: network

- Key challenges for network performance
  - Measuring paths is hard
    - Traceroute gives us only the forward path
    - Shortest path is not always the best path
  - RTT estimation is hard
    - Variable network conditions
    - May not represent end-to-end performance
  - No access to client-perceived performance

The Network is the Computer

- Network computing has been around for time
  - Grid computing
  - High-performance computing
  - Clusters
- Used to be highly specialized
  - Nuclear simulation
  - Stock trading
  - Weather prediction
- Then, they evolved into facilities that are not highly specialized and can be used for several purposes

The Cloud

- What is “the cloud”?
- Everything as a service
  - Hardware
  - Storage
  - Platform
  - Software
- Anyone can rent computing resources
  - Cheaply
  - At large scale
  - On demand

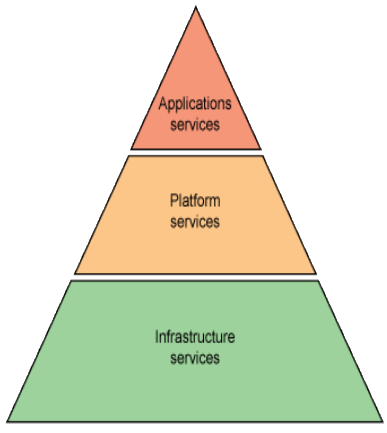
Cloud computing

- USA National Institute of Standards and Technologies (NIST) definition

"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

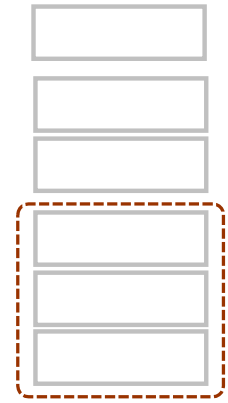
Cloud computing services



- Application as Service (AaaS)
- Platform as Service (PaaS)
- Infrastructure as Service (IaaS)

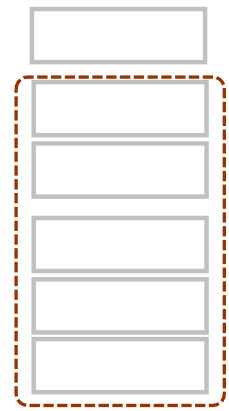
Cloud computing services

- Infrastructure as Service (IaaS): Provides on-demand infrastructural resources: servers, storage, network
- Enable consumers to deploy and run software, OS and applications
- Clients have control of virtual resources
- e.g., Amazon Elastic Compute Cloud (EC2), Microsoft Windows Azure



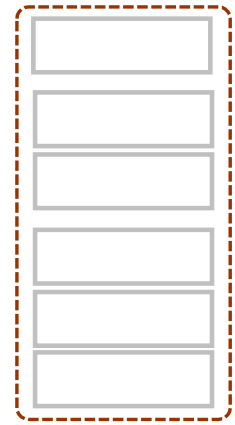
Cloud computing services

- Platform as Service (PaaS): Provides platform layer resources, e.g. operating system support and software development frameworks
- Consumer has control over
  - Deployed applications
  - Possible application hosting environment configurations
- Google App Engine (Go, Java, Python, PHP), Microsoft Windows Azure (C#, Visual Basic, C++)



Cloud computing services

- Application as Service (AaaS): Provides on-demand applications over the Internet
- Consumer use provider's applications running in the Cloud infrastructure
- Application is accessible from various devices (e.g., web browser)
- Online sales application, dropbox, google calendar, gmail, ...



Cloud services

- Cloud providers offer guarantee on the use of certain services/resources, in the form of Service Level Agreements (SLAs)
  - For example, 99% guarantee of availability
- Users can require/release resources on-demand
- IT resources can be scaled quickly, highly dynamic allocation
- To the client, cloud resources seem infinite
- Customers can pay for the resources they actually use
- Pricing and billing are related to SLAs

Advantages of cloud computing

- Reduces IT cost, by reducing capital expenditure
- Flexible scaling of the resources, that can be easily and quickly activated when needed
- Ensures the availability of a wide set of resources at varying levels (from infrastructure to application)
- Ubiquitous access to resources
- Makes it easier to use software without installation issues and to maintain/update it

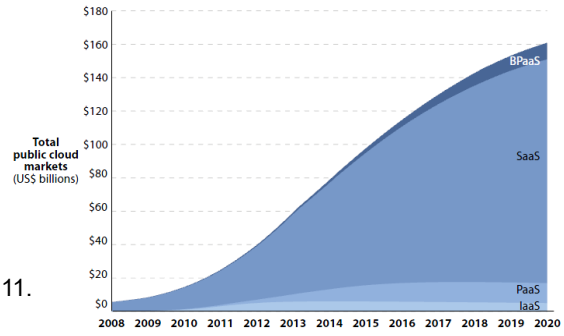
Grid vs Cloud

Grid	Cloud
Uniform distribution of resources	On-demand allocation of resources
Focus on a specific task (often, scientific tasks)	Focus on commercial tasks
Grid security infrastructure	No specific security model

Extremely large numbers

- Flickr has more than 6 billion pictures
- Google serves more than 1.2 billion queries a day on more than 27 billion items
- More than 2 billion video watched on Youtube a day

Huge market!

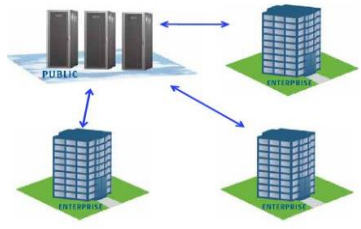


Source: Larry Dignan. "Cloud Computing Market", ZDNet 2011.

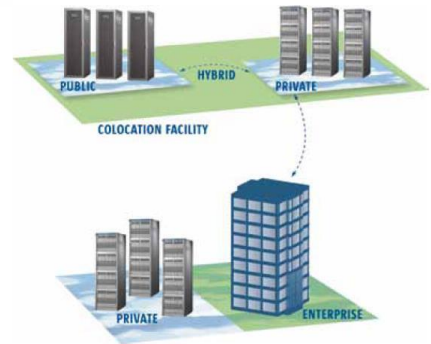
Types of clouds

Private

Public



Hybrid



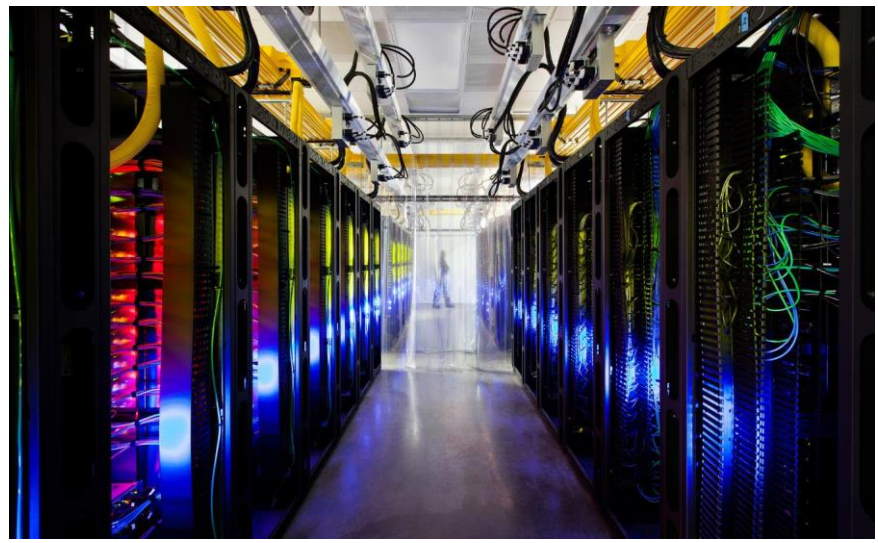
Virtualization

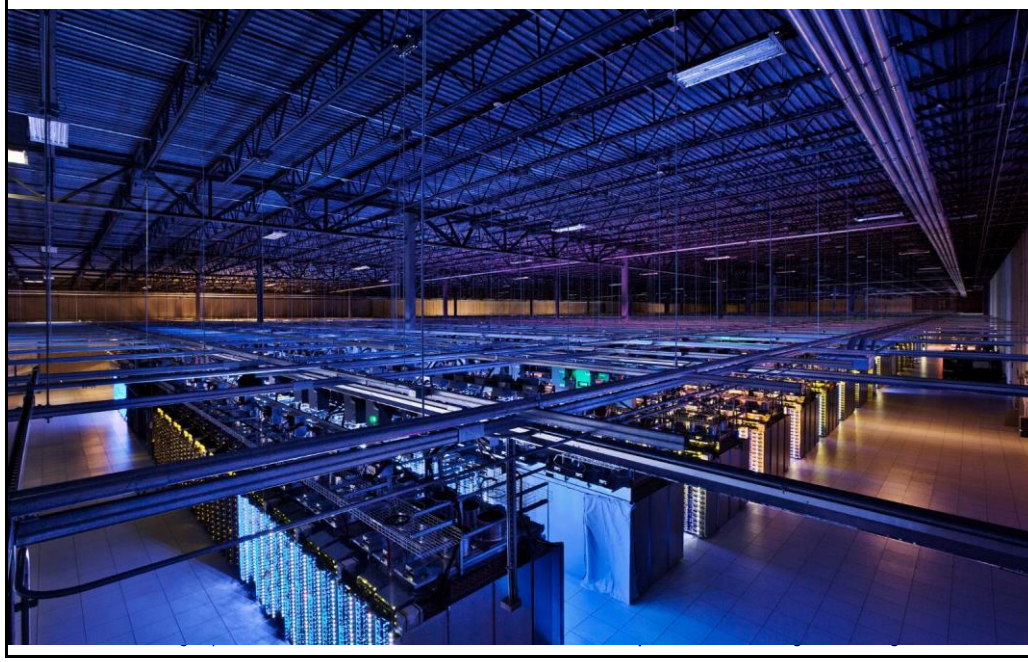
- The key enabling technology for cloud computing is virtualization
- Virtual machines are
  - software implementation of a machine that executes programs like a physical machine
  - VMs are the elements that implement the desired service in a cloud computing environment
  - they are emulation of a computer system and include the desired layers of computing facilities

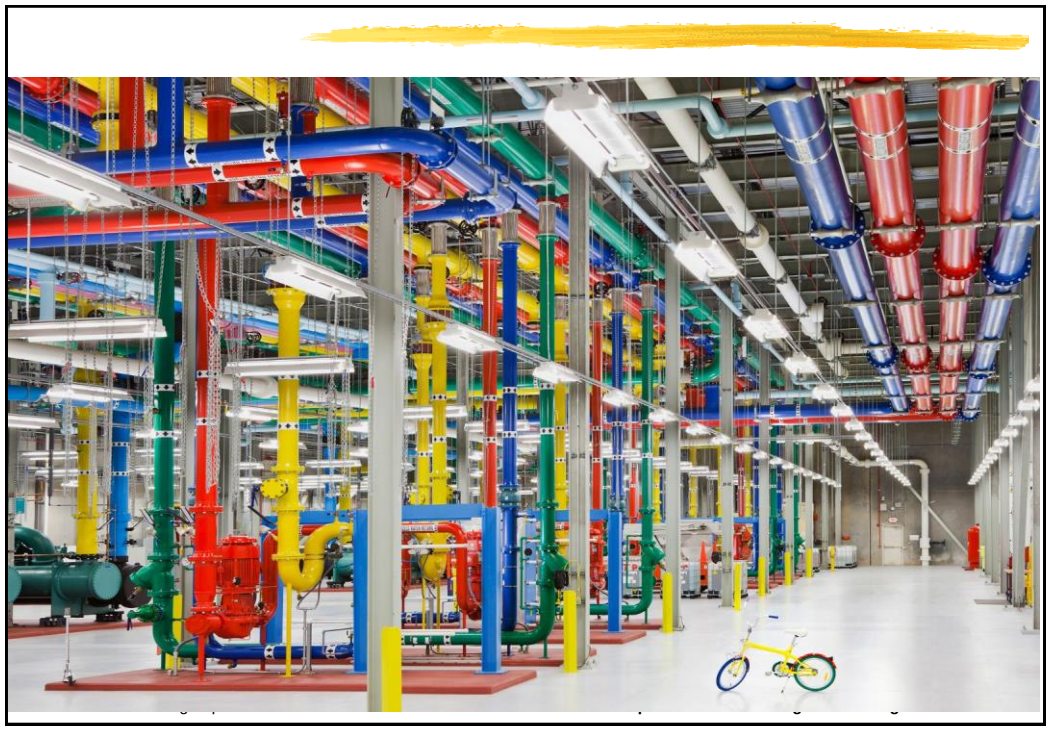
Data Centers

- Cloud computing and CDNs requires large (or huge) aggregates of resources (computing and storage)
- Cloud services are often provided through Data Centers
  - The term Data Center, in general, refers to the computing and storage facilities of a company or an organization
- Data Centers for cloud computing can include up to 100,000 servers
  - The limiting factor is often the power
  - 100,000 servers consuming 500 W require 50 MW power supply!
- Depending on the types of VM that are provided and on the servers, from 10 to 100 of VMs are allocated in each server of a Data Center

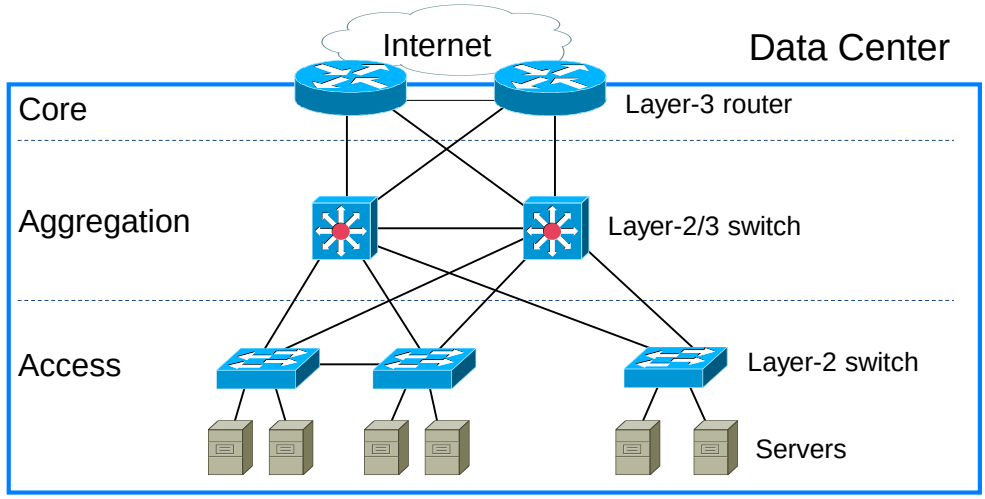
Google data center





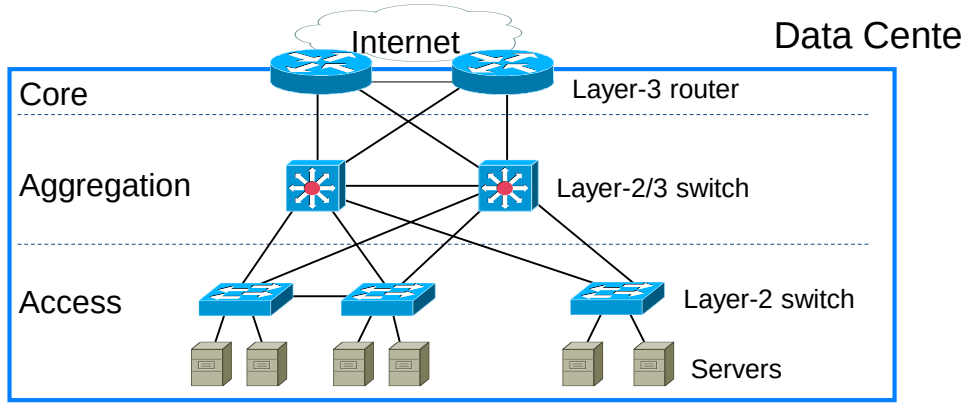


Typical data center topology



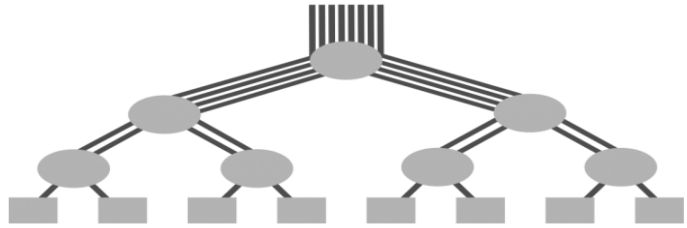
Problem: oversubscription

- Bandwidth gets scarce as you move up the tree
- Links are shared among a growing number of servers (typical ratios 1:2 to 1:20)



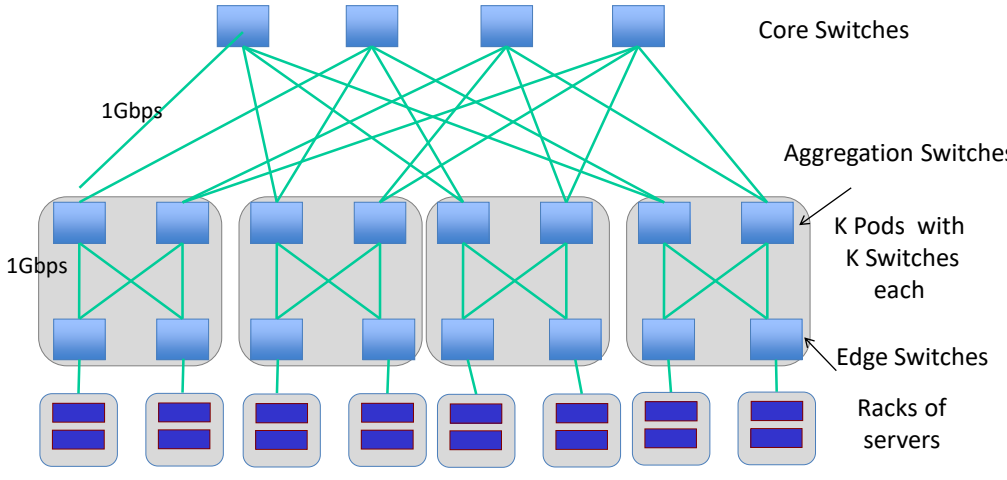
FAT Tree based solution

- Use a *fat tree* topology
  - The number of links going up or down from a node in the tree is the same
  - Links become “fatter” as we go up



Fat Tree topology

Example with K=4



FAT Tree based solution

- 3 layer solution (edge, aggregation, core):
  - K Points of Delivery (PODs) with K switches organized in two layers each, aggregate and edge layers, with K/2 each
  - Each edge switch connects to K/2 nodes K/2 aggregate switches
  - Each aggregate switch connects to K/2 edge switches and K/2 core switches
  - Supports $K^3/4$ nodes
- Other regular topologies are also possible

Modular Datacenters

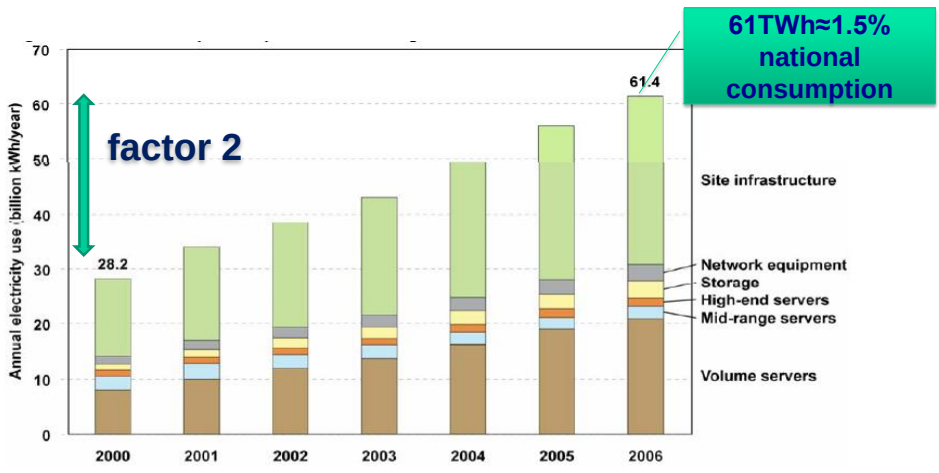
- Shipping container "datacenter in a box"
  - Around 1000 servers per container

- Easy to deploy and update amount of resources

Issues

- Datacenters have
  - Heterogeneous, unpredictable traffic patterns
  - Competition over resources
  - Need for high reliability
  - Privacy
- Heat and power
  - 30 billion watts per year, worldwide
  - May cost more than the machines
  - Not environmentally friendly
- All actively being researched

Data centers energy consumption



Source: Report to Congress on Server and Data Center Energy Efficiency Public Law 109-431. U.S. Environmental Protection Agency ENERGY STAR Program , August 2007

Power Usage Effectiveness (PUE)

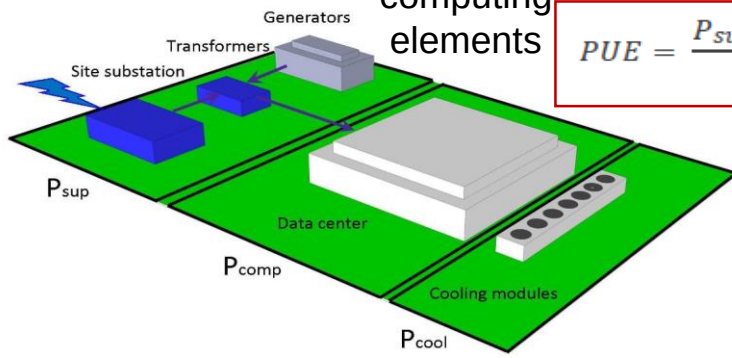
Metric used to evaluate the efficiency of the Data Center

$$PUE = \frac{P_{DC}}{P_{comp}}$$

P_{DC} ← power to the DC

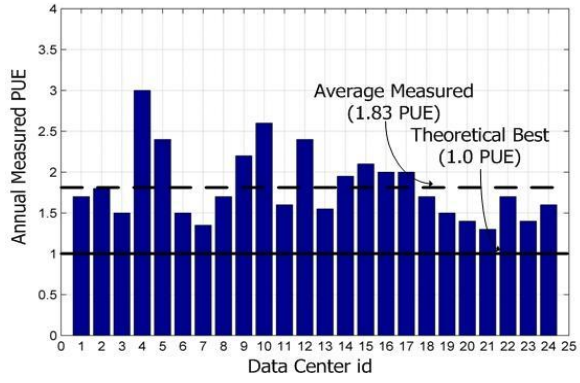
P_{comp} ← power for computing elements

$$PUE = \frac{P_{sup} + P_{cool} + P_{comp}}{P_{comp}}$$



Power Usage Effectiveness (PUE)

High values of PUE in many data centers

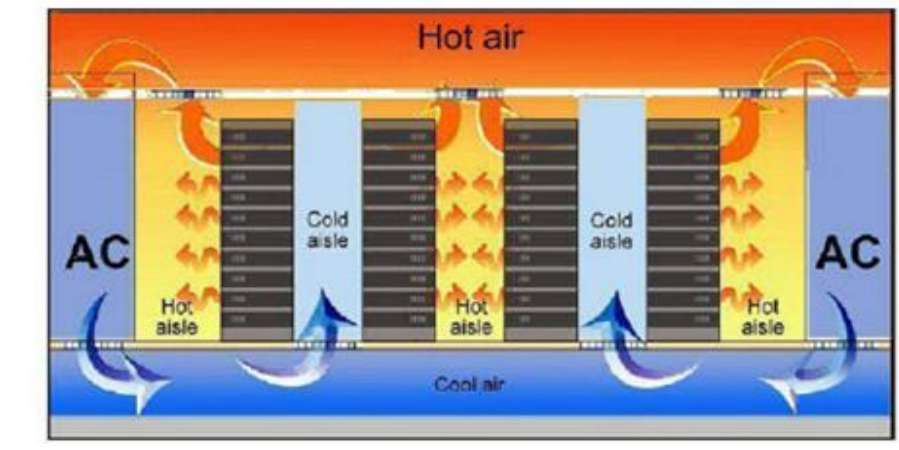


Source: "Self-benchmarking guide for high-tech buildings," data from the LBLN data base centers in the LBNL database, http://hightech.lbl.gov/

Solutions for improving PUE

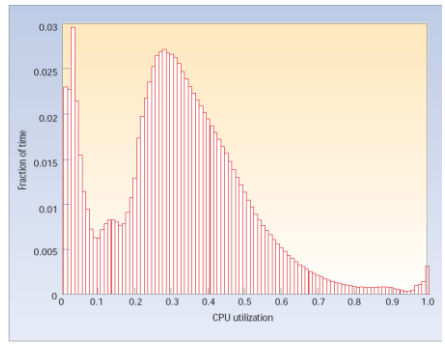
- Improve infrastructure (power and cooling)
  - Liquid cooling
  - Improve efficiency of chillers, funs and pump
  - Improve transformers and power supplies
- Reduce cooling needs (cooling consumes as much as 40% of the operating costs) through specific physical layouts

Solutions for improving PUE



Beyond PUE

Servers generally operate in a low utilization region

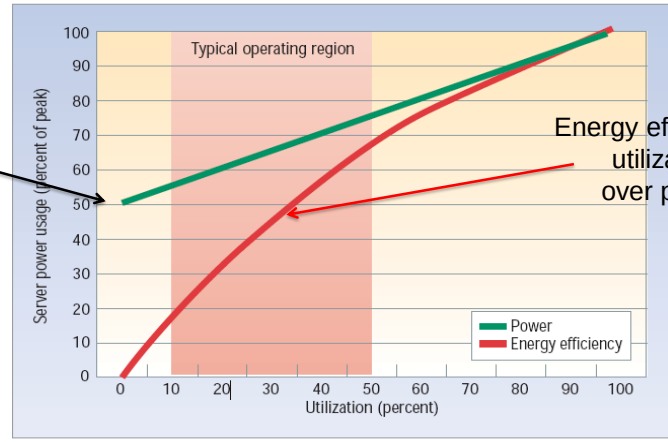


Most mass is in 20% to 40% range

Source: L.Barroso, U.Holzle, The case of energy proportional computing, ACM Computer Journal, Volume 40 Issue 12, December 2007.

Server current design

When idle, power is 50% of full load



Energy efficiency = utilization over power

Source: L.Barroso, U.Holzle, The case of energy proportional computing, ACM Computer Journal, Volume 40 Issue 12, December 2007.

Current solutions for data centers

- Consolidate servers and storage & eliminate unused servers
  - Algorithms to free up servers and put them into sleep mode or to manage loads on the servers in a more energy-efficient way
  - Sensors identify which servers would be best to shut down based on the environmental conditions
- Adopt “energy-efficient” servers or more efficient components
- Enable power management at level of applications, servers, and equipment for networking and storage