Scalability

Generally

- Although adding multithreading capability to a server permits it to service multiple clients simultaneously, the capability of such a system is inherently limited by the constraints of a single server.
- In particular, the amount of memory and CPU's limit how many clients a server can service at one time and how quickly it can service them.
- At some point, multiple hosts are needed to handle client requests.
- Moreover, the same situation is presented when a host or connection to a host fails.
- The ability to address these problems is known generally as scalability.
- Generally, there are three means for scaling up:
  - load balancing
  - content distribution
  - peer-to-peer infrastructure

Load Balancing

- The use of physical infrastructure to handle large numbers of client requests ia referred to generally as load balancing. Note: the term "load balancing" has different definitions and is used in a variety of specific ways, although the general idea is the same.
- Load balancing is also used when a host or network connection fails. In the case of a failure, requests are redirected available physical infrastructure. This process is known as failover.

  The collection of destination application server hosts is known as a cluster; sometimes, it is referred to as a server farm.
- Although there are a number of specific load balancing architectures and solutions, they are generally implemented by two means:
  - IP address schemes
  - MAC address schemes

- A very simple means to distribute requests is known as DNS Round Robin:
  - The authoritative DNS server is configured with a list of IP addresses.
  - Each IP address represents a host running an application server. This collection of hosts comprises the cluster.
  - All of the IP addresses are mapped to a single domain name.
  - Each time a DNS request is made, the authoritative DNS server returns the next IP address in its list, starting over once the list has been exhausted.
- More sophisticated algorithms for selecting the destination host are handled by a different approach.
- A host can be added that runs a server known as a load balancer.
- Load balancing relies on a Virtual IP address, or VIP. The VIP does not map to a particular destination host; instead, it is mapped to the load balancer.
- Load balancing can utilize a variety of algorithms for selecting the destination application server host, such as weighting schemes, task queues, and even session data (such as HTTP cookies).

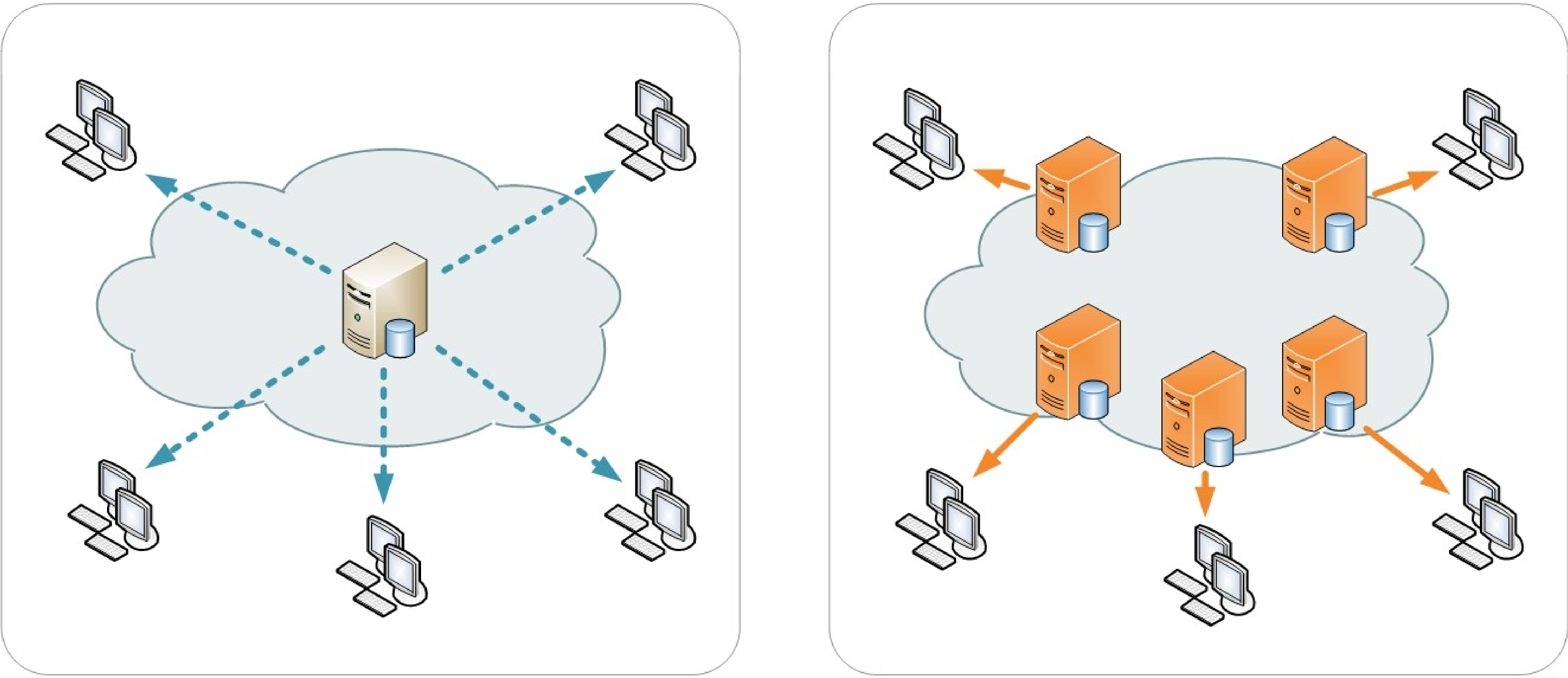
- Load balancers come in two flavors: two-arm and one-arm.
  - A two-arm load balancer acts as a router. Requests from clients are routed to the load balancer, which uses NAT to retransmit the request to the selected application server. See Two-Arm Load Balancer diagram.
  - A one-arm load balancer acts as a proxy. Requests from clients are translated using NAT, but the load balancer must also translate response packets using NAT, as well. See One-Arm Load Balancer diagram.
- A one-arm variation can use MAC addresses instead, known as a Direct Response load balancer. The load balancer uses ARP to direct client requests to the desired destination application server host. This approach is by far the fastest load balancer, but it disrupts ARP for the subnet.

- The implications of serving requests over a cluster arise from the architecture of the server program.
- Suppose that important data used by the application server is stored in a file.
- When a cluster is created, how is the file accessed? Is a copy made for each server in the cluster?
- What happens when the file is updated? The changes must be propagated to all of the servers in the cluster.
- What happens if the changes must be made quickly? Do the servers need to be shut down while the file changes are propagated?
- If, instead, the file is maintained on its own host and the other hosts connect to it over the network, what happens while changes are being made? Are the servers blocked? These issues arise from data replication.

- Another issue is state. Suppose that a distributed system requires that a client login to a server before making further requests.
- If the load balancer routes the login request to Server 1 and then routes a later request to Server 2. Can Server 2 learn whether the client successfully logged in? If so, how? Must the load balancer take into account the login state of the client when selecting a host for requests from that client?
- This concern is the basis for session data selection using HTTP cookies, but what about systems that do not use HTTP cookies? Must a custom load balancer be created?
- Can login or other session data be maintained in a central location that all of the servers in the cluster can connect to, such as a remote database?
- Bottom line: Your distributed system architecture must take into account load balancing if you want your system to scale!

Content Distribution

- Another technology for scaling distribution is the Content Delivery Network (CDN).
- A CDN deploys infrastructure near users and replicates data as needed



“Single server (left) versus Content Delivery Network (CDN) (right)” by Kanoha; Creative Commons, 22 September 2009

- CDN’s are used extensively for transmission of media, such as pictures and streaming video.
- The architecture of CDN's was designed originally by Dr. Tom Leighton, a professor at MIT, and his student, Dan Lewin. A prototype was created with the help of additional colleagues. Leighton and Lewin founded a company to deploy their solution, Akamai Technologies. Today, Akamai is one of the world's largest distributed systems. On Sept. 11, 2001, Lewin was a passenger on American Airlines Flight 11, which crashed into the World Trade Center.
- Another scaling technology is cloud computing.
  - Amazon Simple Storage Service (S3) is widely used.
  - See “Introduction to Amazon Simple Storage Service (S3) - Cloud Storage on AWS” on YouTube.

P2P Infrastructure

- Instead of scaling by adding clusters, others have scaled by distributing servers to client hosts. These solutions are known as Peer-to-Peer computing (P2P).
- Napster was one of the earliest popular P2P systems. It become embroiled in legal trouble due to copyright violations.
- A very popular P2P system used today is BitTorrent, which was covered in another lecture.
- As explained earlier, BitTorrent peers have both client and server capabilities.
- In practice, large files are distributed very quickly using BitTorrent.
- Moreover, the cryptographic hashes provide a great degree of assurance that the original torrent was transferred without tampering.
- Studies have shown, however, that a significant portion of torrents contain embedded malware.

- P2P systems can scale well beyond clustering-type solutions, often easily, and thereby eliminate considerable administrative overhead and cost.
- Nonetheless, P2P systems are generally impractical without some element of centralization, such as the tracker in the case of BitTorrent. Such centralization, though, is typically far less onerous than that of cluster systems.
- Updates, patches, configuration changes, and technical support are the responsibility of the community of users; there is no legally responsible entity generally (but see the case of Napster; A&M Records et al. v. Napster and RIAA lawsuits against users).