Overlay Networks

Outline
Overlay networks overview
Routing overlays
Resilient Overlay Networks
Overlay Networks

- Logical network on top of the physical network
- Underlying idea: Let the applications make their own forwarding decisions
- Enables us to introduce new functionality into the network

How it works:
- Each node processes and forwards packets in an application specific way
- The links are implemented as tunnels

*Image from: http://www.cs.virginia.edu/~mngroup/hypercast/general.html*
Routing Overlays

- Can we try routing algorithms not supported by router vendors?
  - YES! Using Overlays.
- Experimental Versions of IP
  - IP multicast deployed in Mbone and 6BONE as an overlay
  - Multicast-aware routers tunnel through legacy routers
- End System Multicast
  - Lets the end hosts implement their own multicast trees
  - The goal is to find the embedded multicast tree that spans all group members
End System Multicast

Cont.

- Using unicast naïvely is not efficient:

- Optimal, by DVMRP:

- The best we can get with end system multicast:
Managing the Overlay

- For joining the network, the host needs to know the IP address of at least one other node in the network.
- A connected node sends periodic “keep alive” message to its neighbors.
- Each node exchanges the list of all nodes that it knows of with its neighbors.
  - These messages propagate through the mesh similar to the Distance Vector algorithm.
- Before leaving the mesh, a node sends a “leave mesh” message to its neighbors.
- Once a node detects a partition, it creates a new edge by sending a “join mesh” message.
- End hosts measure the round-trip latency to other nodes.
  - Add new links or drop links based on these measurements.
Resilient Overlay Networks

- Idea: To find alternative routes for traditional unicast applications
- Why do we need alternative routes?
  - BGP Does not guarantee shortest paths
- Tradeoff:
  - Getting optimal routes vs. scalability
  - BGP scales well, but does not give optimal routes
  - RON can give optimal routes, but does not scale well
- The bad: Needs active monitoring of links -> Expensive
  - Monitors Latency, available bandwidth, and loss probability
- The good: Fast recovery from network failures
Peer-to-Peer Networks

Outline
Overview
Gnutella
BitTorrent
Peer-to-Peer Networks Overview

- A peer-to-peer (P2P) network allows a community of users to pool their resources
  - Content
  - Storage
  - CPU,...
- A P2P network is both decentralized and self-organizing
  - Just like the Internet itself!
- Why do we care about these networks?
  - It is challenging to achieve decentralization and scalability at the same time.
Gnutella

- One of the first *decentralized* P2P networks for file sharing
- No central registry of objects.
- Example topology of a Gnutella P2P network

- Edges of the graph correspond to the relationship “A and B know each other”
Gnutella cont.

- The simple idea in Gnutella is to distribute the method for finding data
  - Great idea!
  - Lots of fun architectural possibilities!
- Gnutella is a distributed search protocol with a decentralized model
  - Clients can issue/view query results
  - Clients can serve/request data
  - Clients accept queries and respond with matches from their local data stores
Gnutella Protocol

- Protocol defines method of client communication
  - Set of descriptors used for communicating data
  - Set of rules governing inter-client exchange of descriptors
- Descriptors
  - Ping: active discovery of hosts on a network
  - Pong: response to Ping includes client address and metadata
  - Query: Ask for an object
  - Query Response: response to Query includes info necessary to get data
- A Gnutella client connects to network by establishing a connection with another client on the network
  - Finding another client is not part of Gnutella spec.
    - Host cache services are the typical way this is done
Gnutella Protocol

- New client then creates connection to the Gnutella client and thereby becomes part of the network
  - Gnutella client can reject the connect request
  - Successful new client can then send/receive descriptors
- Pings/pongs are then sent to establish network
  - No specification as to how much/often to probe
  - Network data can/is cached
- Message routing should be well behaved
  - Ping/Query descriptors should be sent to all directly connected clients
  - Pong/Query Response descriptors should be sent back along same path
  - TTL is mechanism to limit distance
- File downloads via HTTP/1.0 protocol via direct connect
Gnutella’s Downside

- Flooding does not scale well!
- Alternatives:
  - Forward queries randomly or according to probability of success based on past results
  - Proactively replicate objects to make them easier to find
Bittorrent

- Bittorrent is a peer-to-peer file sharing protocol devised by Bram Cohen.
- It is based on replicating the file, or rather, replicating segments of the file, which are called pieces.
- Any particular piece can usually be downloaded from multiple peers, even if only one peer has the entire file.
- The primary benefit of Bittorrent’s replication is avoiding the bottleneck of having only one source for a file. This is particularly useful when you consider that any given computer has a limited speed at which it can serve files over its uplink to the Internet, often quite a low limit due to the asymmetric nature of most broadband networks.
- Torrent File - File size, piece size, URL of tracker, SHA-1 for piece
- Tracker: Keeps current membership of swarm
Replication

- The beauty of BitTorrent is that replication is a natural side-effect of the downloading process: as soon as a peer downloads a particular piece, it becomes another source for that piece.

- The more peers downloading pieces of the file, the more piece replication occurs, distributing the load proportionately, and the more total bandwidth is available to share the file with others.

- Pieces are downloaded in random order to avoid a situation where peers find themselves lacking the same set of pieces.
Swarms

- Each file is shared via its own independent BitTorrent network, called a swarm. (A swarm could potentially share a set of files, but we describe the single file case for simplicity.)
- The lifecycle of a typical swarm is as follows. The swarm starts as a singleton peer with a complete copy of the file.
- A node that wants to download the file joins the swarm, becoming its second member, and begins downloading pieces of the file from the original peer.
- In doing so, it becomes another source for the pieces it has downloaded, even if it has not yet downloaded the entire file.
Peers in a BitTorrent swarm download from other peers that may not yet have the complete file.