Streaming Live Media over a Peer-to-Peer Network

Technical Report

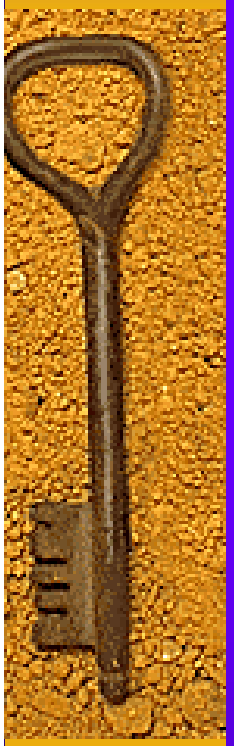
Stanford University

Deshpande, Hrishikesh

Bawa, Mayank

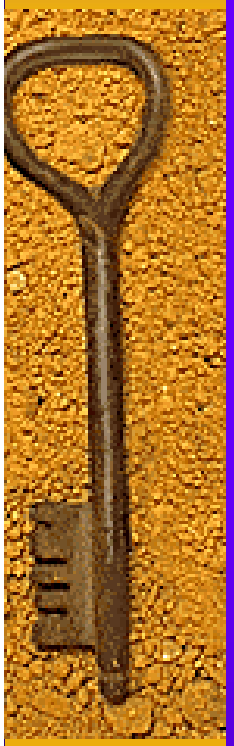
Garcia-Molina, Hector

Presenter: Kang, Feng



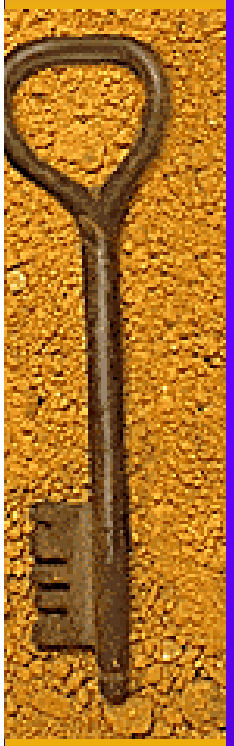
Outline

- Problem in media streaming and possible solutions
- Multicast tree over P2P to attack the problems in media sharing.
  - Advantages and challenges in P2P framework to handle this problem.
- SpreadIt framework to implement the multicast tree.
  - Peering layer.
  - Maintenance of topology
  - Performance evaluation
- Further Work



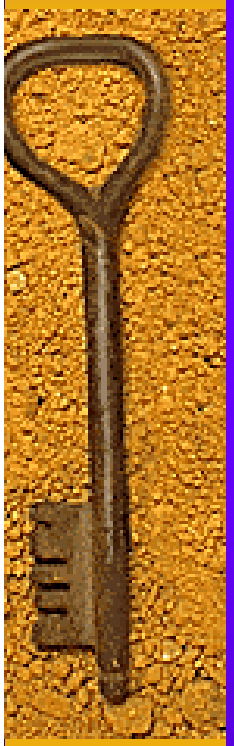
Problem in streaming media

- **Problem:** Because of the high bandwidth requirement of stream applications, a small number of clients are often sufficient to saturate the bandwidth at the source.
  - T3 has a 45Mbps capacity. A 30 fps, 320x240 pixels stream has a rate of 1Mbps. Only 45 clients can be served with a maximum resolution.



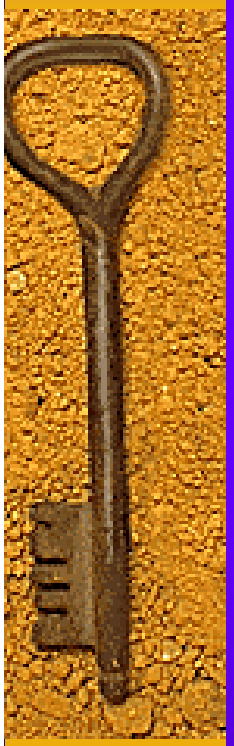
Solution

- Some solution
  - Turn away a large number of clients
  - Degrade QoS.
  - Distributed system to replicate content:
    - have a hard upper bound with the number of clients.
    - cost with the associated bandwidth.
  - IP Multicast: Feeding many clients from a single stream.
    - Require router support. The complexity costs of group management, distributed multicast address allocation, and support for network management have limited the commercial deployment of IP multicast.
  - Application layer multicast: Reliable servers are installed across Internet to act as routers by splitting and forwarding streams to the clients in the same Intranet.
    - Cost of deploying and maintaining the servers.
- Goal: Designing a content distribution system which can scale with the number of clients. Such a system need to be cost-effective.



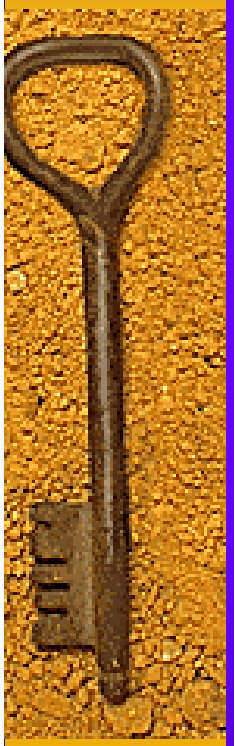
P2P Solution by Stanford.

- Use P2P to stream live media over a network of clients, using the resources of the clients themselves.
- Nodes form and multicast tree and each node acts as both a server and a client.
- Advantages:
  - Utilize bandwidth efficiently and support large number of clients.
  - Cheap deployment.
  - Although The delay and packet loss is increased, experiments show that effects of packet loss and delay are limited.
  - Good QoS (degrades gracefully as clients increases)
- Disadvantages:
  - Do not provide rigid guarantees of reliability and availability as provided by commercial vendors.



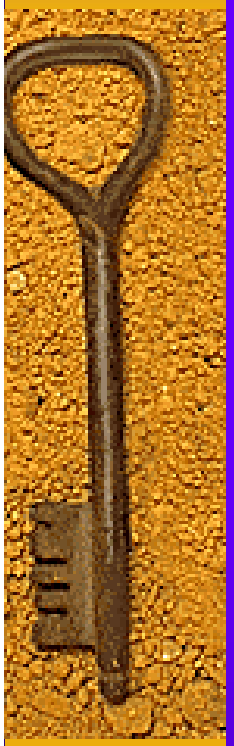
A more precise P2P model

- A stream is logically composed of two channels.
  - Data channel: RTP/UDP.
  - Control channel: RTSP/TCP.
- The stream has two end-points: Server and Client.
- Source of a live stream
  - Remains up to the duration of the stream.
- URI(Uniform Resource Identifier)
- Subscribe, Unsubscribe, and Failure.
- Data-transfer session and application session.
- Saturated and unsaturated.

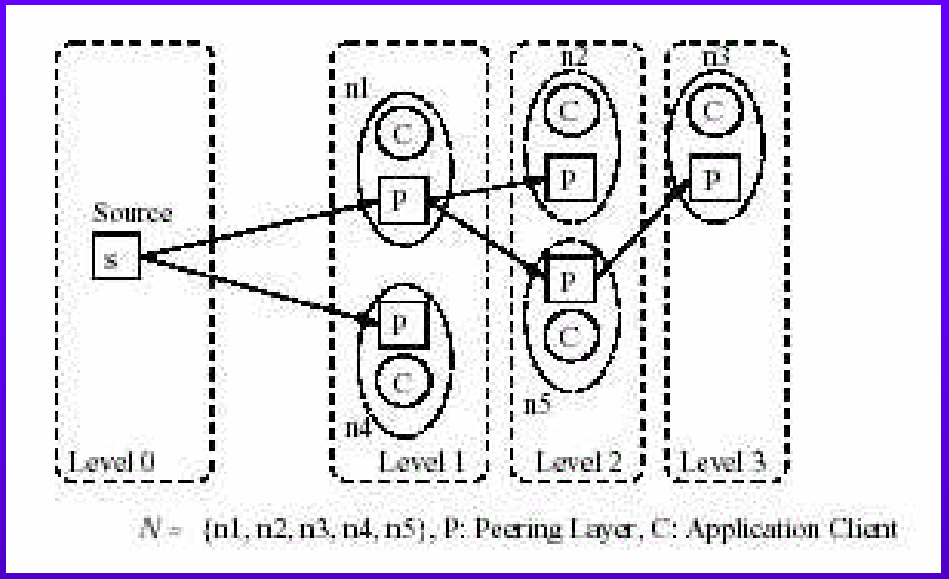


Key challenge of building a multicast tree over P2P.

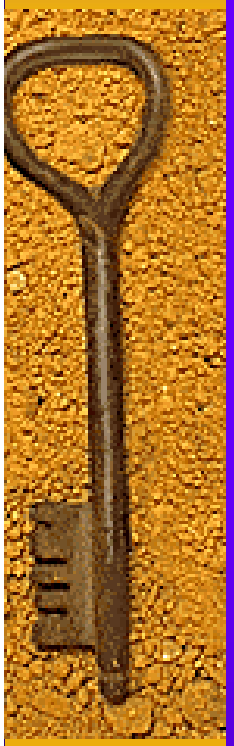
- *Discovering a server:* New nodes need to identify a parent when joining.
- *Managing Transience:*
  - Clients are autonomous, so subscribe and unsubscribe are unpredictable.
  - Disconnection of nodes causes all of the descendants to be stranded.
  - The repair time for such failure should be completed in a duration which has the minimal effect on the QoS, perceived by the descendants.
- *Managing load:* The clients are not intended to be a server. The resources of nodes should be judiciously used so as not to overload the nodes.
  - Processing power, memory resources
  - Bandwidth: the predominant constraining resource.



SpredIt: Address challenges in building a multicast tree:



- Current problems in Streaming applications.
  - Tight coupling of application layer and transport layer.
- Peering layer
  - A layer between application layer and transport layer.
  - Peering layers at different nodes coordinate among themselves to establish a maintain a multicast tree.
  - Advantages:
    - Application is masked from the failure of nodes
    - Existing applications don't need to be changed



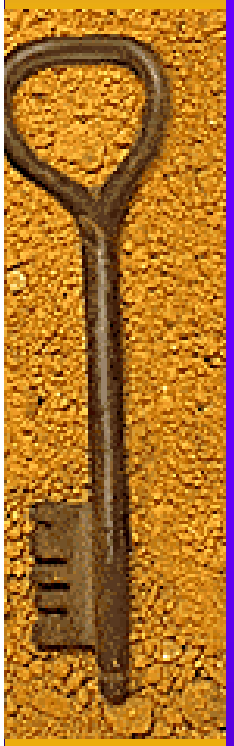
Key Ideas of Peer Layer

- The tuple to identify the end points.
  - <Server IP, S Port, Client IP, C Port, stream URI>.

Transport end points | Stream data

- Get-stream interface:
  - Application layer specify a stream to get.

- Redirect primitive:
  - One peer sends a redirect message to another peer and specifies the target peer to redirect to.
  - This message will change the topology of the multicast tree.
  - Good building block for tree maintaining algorithms.



Maintenance of multicast tree topology

- Adding nodes.
  - Procedure
  - Policies
- Delete nodes
  - Procedure
  - Policies
- Handle failure of nodes.
  - Detection of failure
  - Handle failures
- Different policies will
  - Influence the topology of the multicast tree and balance of the tree.
    - The time to discover an unsaturated node.
    - The time to first packet
    - The transience of time of a node.

Maintenance of topology: Adding a new node to multicast tree(1)

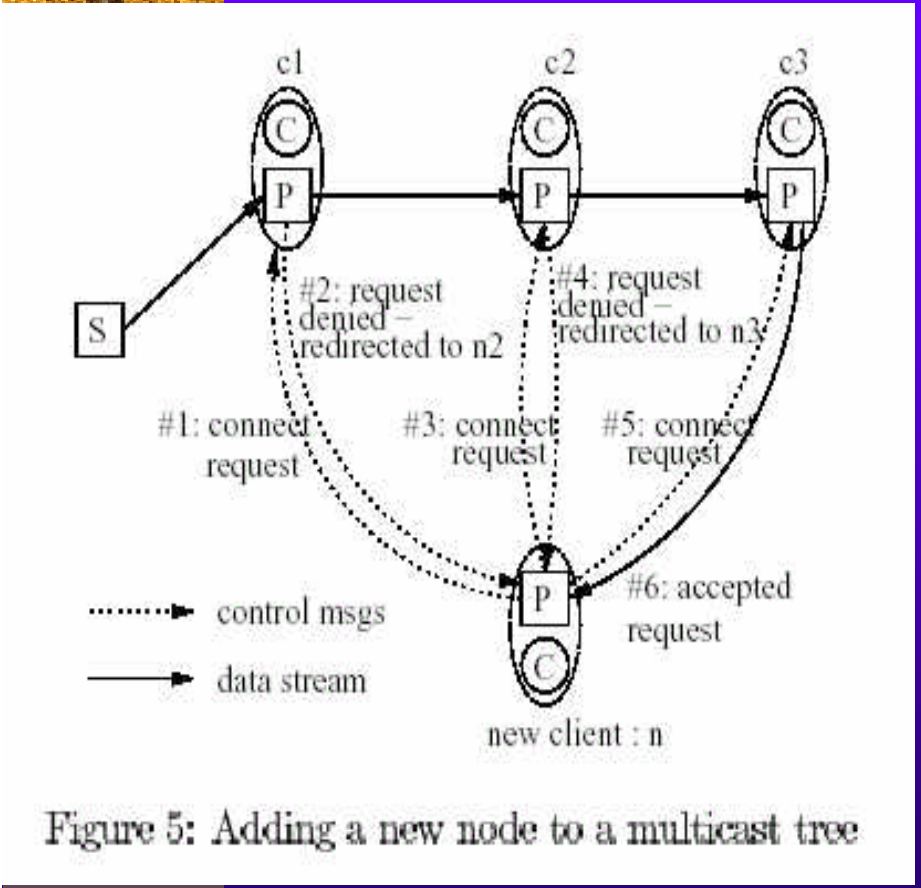
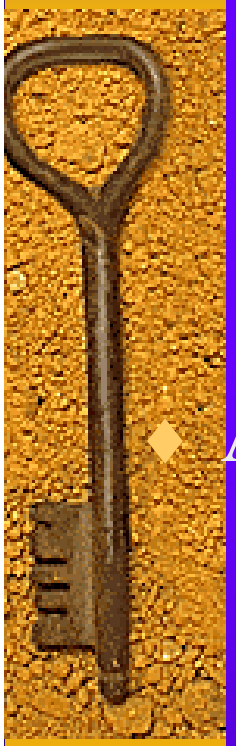


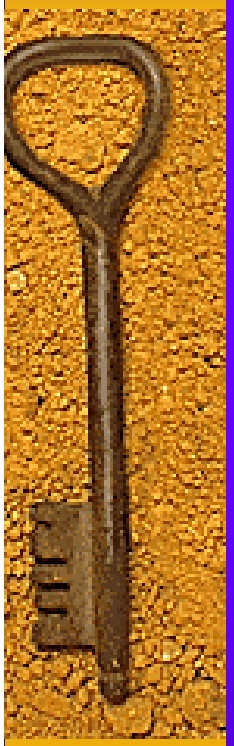
Figure 5: Adding a new node to a multicast tree

- Finding an unsaturated node with redirect primitive
- Iterative resource finding process is executed,
  - starting from the known root of the stream
  - If the node is saturated, select one child as target and return redirect message to the requesting node. N->s->c1-> c2->c3.
  - Until reaching a node that is unsaturated and could accommodate the request.



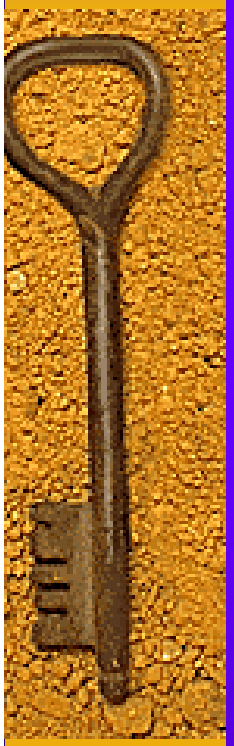
Maintenance of topology: Adding of a node(2)

- Addition policy
  - If a node is unsaturated, it will establish the connection.
  - If it is saturated, it needs to tell the requesting node the target to redirect to. Which node is selected as target is addition policy.
    - Random: On average, the tree is balanced.
    - Round-robin: requires some state information.
    - Smart-placement: The receiving node will specify the requesting node's closest node. It result in less traffic.
    - Knock-downs: If the requesting node is closer to the source node, the source node will accept the requesting node as its child, but it will let the farthest node redirect.
    - Smart-bandwidth: If the requesting node X's bandwidth is greater than the receiving node Y's bandwidth. Y will redirect X to its parent and it will redirect itself to X..



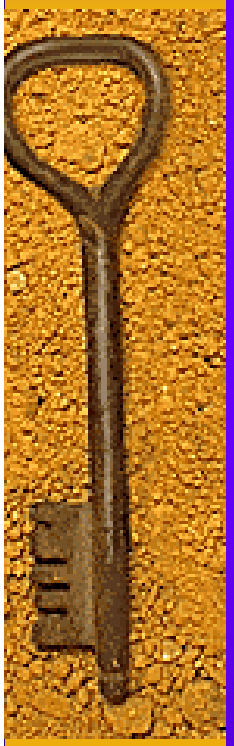
Maintenance of Topology: Deleting a node(1)

- When deleting a node, a redirect message is sent to the children, specifying the target. Then a redirect process begins. All descendants will be grafted back to the source-rooted multicast tree.
- The deleted node at least knows its parent and the source, these are at least two candidates of target.



Maintenance of Topology: Deleting a node(2)

- Policy:
  - Grandfather-All(GFA): It will send redirect messages to all of its descendants with its parent as the target.
  - Root-All(RTA): Choose the source as target for all its descendants.
  - Grandfather(GF): Choose grandfather as target but only send redirect messages to the immediate children.
  - Root(RT): Choose source as target.

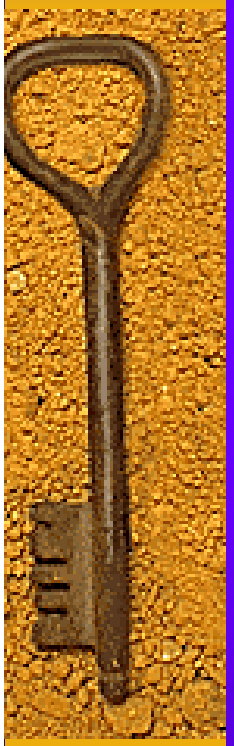


Topology maintenance: Handling failures of node

- Failures do not unsubscribe and do not send redirect messages.

- Detection of failure of nodes
  - Heart-beats mechanism, in which a node sends alive messages periodically to its parent and children.

- Handle node failure
  - If a parent detects a child failure, delete information.
  - If a child detects the parent failure, it will send to itself redirect message with source as the target.



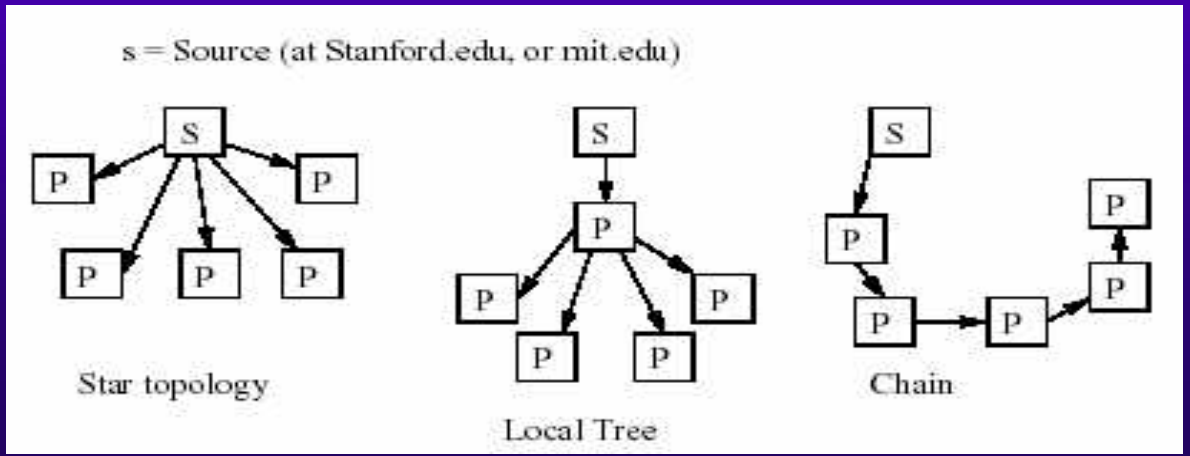
Performance Evaluation

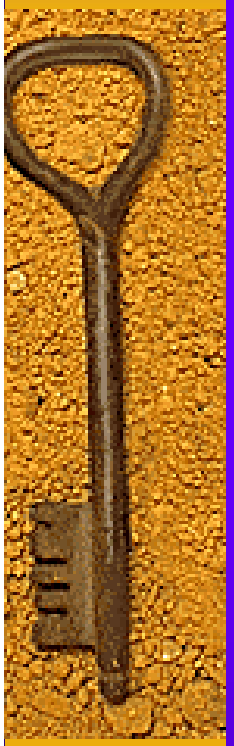
- Empirical tests to get values for parameters and then use these parameters to evaluate various policies for QoS metrics by simulation.

- Experimental Testbed
  - Apple's open source Darwin Streaming Server at the source.
  - QuickTime as the application client at the peer nodes.
  - Peering layer is implemented in Python.
  - Use redirect mechanism to discover unsaturated nodes during addition, deletion, and failure.
  - Failures are detected by the absence of heart-beats.

Empirical Tests:

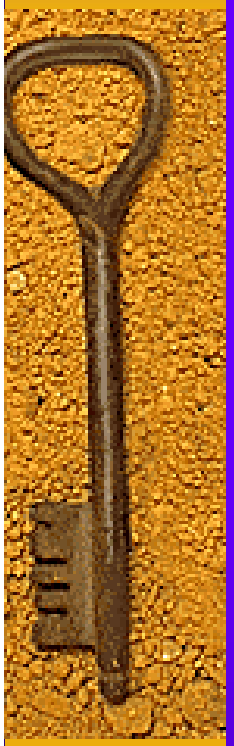
- Two sets of experiments:
  - All the peers and server are put in the same Intranet.(All in Stanford Intranet)
  - The source is placed across Internet. The peers are in the same Intranet. (Server in MIT, clients are in Stanford Intranet.)
- Three kinds of topology for peers are evaluated.
  - Star. Models that the clients get data through direct connections to the source
  - Local tree. This allows to study the effects of network proximity.
  - Chain. The effects of increasing number of hops of a peer from the source.





Result from Empirical Tests

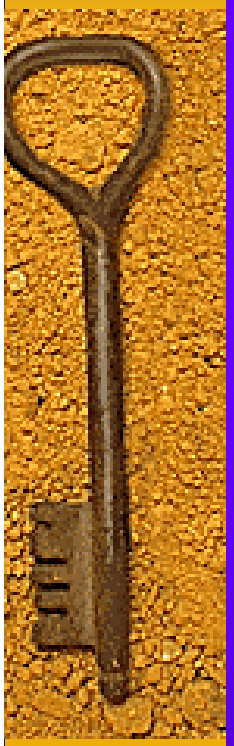
- Packet Delays:
  - Def: Time difference between the instance when a packet is sent and when a packet is received.
  - Experiment:
    - Chain topology, all within Stanford.
    - Timestamps from the same computer, after reaching the final client, form a loop to return to the original node.
  - Result:
    - five hops: the order 0.1s.
    - Linear growth: every hop 0.02s, which means a tree of height 15 to reach 1% of buffer delay.
    - Height 15 means thousands of clients.
- Packet Loss:
  - Rule: Consecutive packet losses are more important than isolated packet losses.
  - Experiment:
    - Chain topology.
    - Packets are logged at the node and at the source. The difference is the packets loss.
  - Result:
    - Across Internet: 1%.
    - Within Intranet: 1% of the the connection across the Internet.
    - Preferable to put all the peers in the Intranet.



Result from Empirical Tests

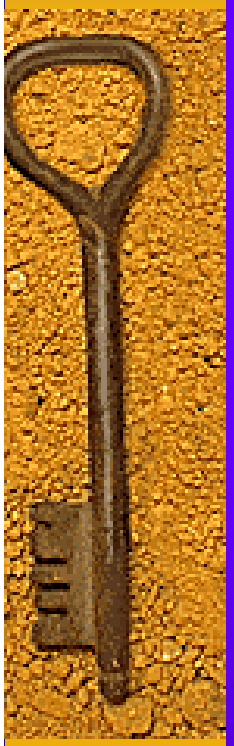
- Time to discover unsaturated node:
  - Def: Interval between the time a new node sends a subscribe request to the source and to an unsaturated node.
  - Result:
    - Time to traverse levels in chain is linear in the number of hops, rate is approximately 0.1s.
    - This is a good indication to time to discover the unsaturated node, because finding also required traversing.
- Time to first packet:
  - Def: time from sending a request to receiving the first packet.
  - Not measured in this empirical test but in later simulation as a performance parameter.

- Inference: Time to add, delete and handle of failure should only be part of the time the receiving buffer could hold.
  - The time for recover should be less than 5% of an application level buffer, which means 1.5s.
  - It can support 10 redirect, a tree of height 10.



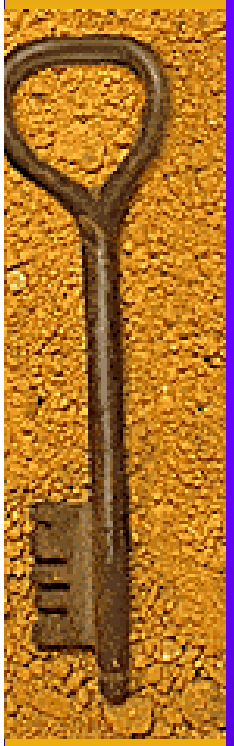
Simulation: Model

- Start from an Intranet model and extends it to Internet model.
  - The simulation is done with each combination of add and delete policy for the multiple Intranet model.
- A client in three states:
  - Inactive: when a node is not subscribed to the stream.
  - Active: when a node is subscribed to the stream and is getting the stream.
  - Transient: if a node is subscribed to the stream, but is not getting the stream because it has not discovered an unsaturated node.
- Two aspects:
  - STAR vs. SpreadIt
  - Different combinations of policies.



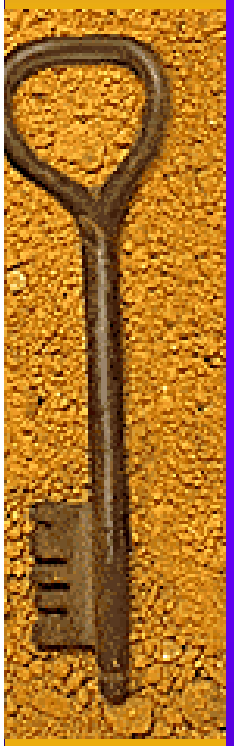
Simulation: Parameters(1)

- Time
  - Time: discrete, monotonically increasing quantity.
  - The length of the media is t_{len}
  - The source sends packets once every t_s time steps.
  - A node only acts on one event at each time step. It takes t_c t_{hs} t_{ss} time step to handle DESCRIBE, SETUP, PLAY message.
  - A node send and receive a heart-beat messages from/to its parent and children every t_{hb}



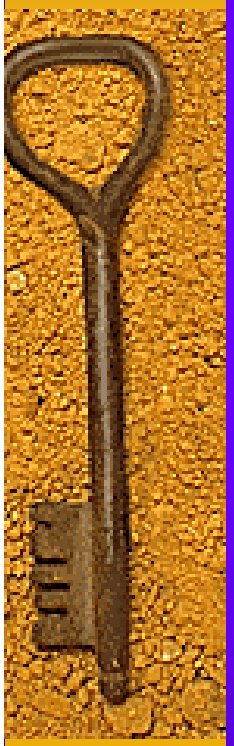
Simulation : Parameters(2)

- Maximum nodes to support.
  - The source can support $c_{\max}$
  - In the same intranet c_{mi}
  - Across intranet c_{mo}
- Packet loss probability:
  - A packet loss on a hop with probability: p_{loss}
  - In the same Intranet: p_{li}
  - Across the Internet: p_{lo}
- State transition probability(Add, Delete, Fail probability)
  - A new node is added: p_{add}
  - A node is deleted: p_{unsub}
  - A node failed: p_{fail}



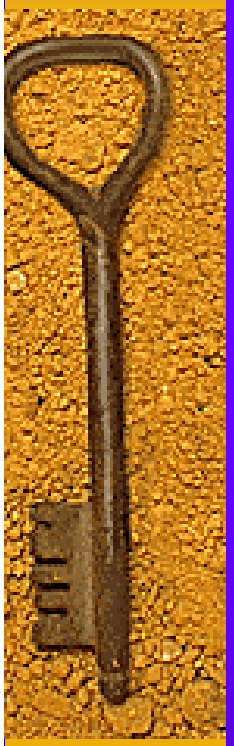
Simulation results(1)

- Packet loss for different add/delete policies:
  - Among add policies: SP outperforms RR.
    - SP uses the intranet information of a node to minimize across intranet hops.
    - Chances that packets loss are more for hops across an Intranet, than within an intranet.
  - In active state, Random performs similarly with RR. So in future comparison, only RR is considered.
    - The loss increase with the height of the tree in both of the two schemes. This is because of accumulative loss.
  - Among delete policies: GFA and RTA suffers more in the transient state than GF and RT. Among 200 nodes failure, GFA took 8 times to recover than GF.
    - In GFA and RTA, the Grandfather/Root node is swamped.
    - Losses in transient state are more important than those in the active state. GF and RT are recommended.



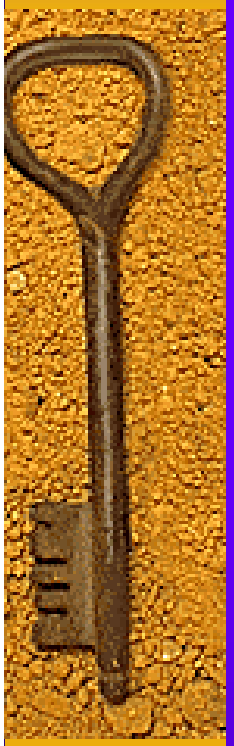
Simulation results(2)

- Average loss of packets by one node: STAR and multicast tree with different combinations of add/delete policies:
  - STAR performs best.
  - In active state: SP/GF and SP/RT suffers only 2.5 more than STAR.
  - In transient state: STAR losses 4 times more than SP/RT
- Inference:
  - Losses in transient state are more important than those in the active state. So SpreadIt is better than STAR.
  - STAR imposes higher requirement on nodes. So SpreadIt is better than STAR.
  - Reason: In STAR, the source can process one message at a time. The clients' requests add up to a long queue, causing each client wait longer, while SpreadIt can balance the load away from a single node.



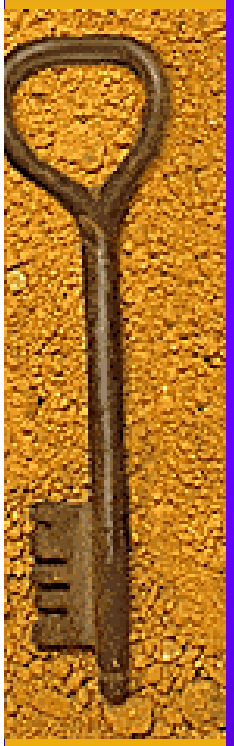
Simulation results(3)

- Performance in flash crowd.
  - The time to first packet
    - increases rapidly for STAR as p_{add} increases. When there are 1000 nodes simultaneously, STAR performs 10X worse than RR.
    - In STAR, the source becomes the bottleneck to establish a data-session.
    - RR distributes load from the root quickly.
  - The packet loss follows the similar pattern.



Simulation results(4)

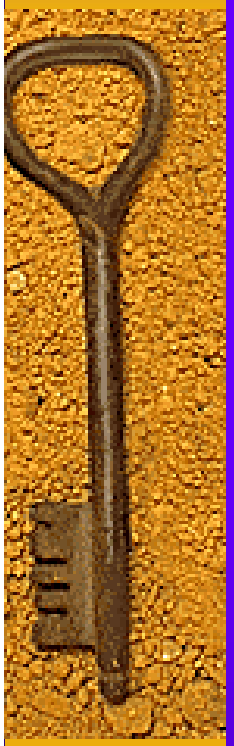
- Performance when the maximum number of nodes $c_{\max}$ increases.
  - For SpreadIt, as $c_{\max}$ increases, mean packet loss is decreased.
    - Increasing of maximum allowable nodes lead to decreased tree height and more unsaturated nodes are available at lower levels, which leads to a faster discovery of parent.
  - The mean time of outage is also decreased
    - As height of tree decreases, less nodes are affected.
  - STAR has higher packet loss and smaller mean length of outage than SpreadIt
    - Maximum allowable nodes mean more nodes at the source
    - In star, failure of one node does not affect other nodes.



Simulation results(5)—across Intranet

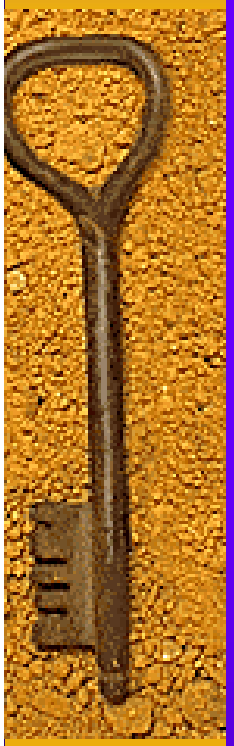
- For average packet loss, SP/GF is consistently below the RR/GF. The gap is increased as loss rate across Internet increases.
  - SP is aware of the Intranet placement of nodes and minimize the across intranet loss rate, which is more important than loss within the Intranet.

- Performance of increasing the maximum allowable external nodes.
  - Loss rate is decreased.
    - the stream reaches a given Intranet in a smaller number of hops.
    - Tree becomes more compact.
    - After arriving the Intranet, the loss rate is greatly reduced.



Some Highlights of Simulation

- The number of packet loss in the active state 2.5 time more for SpreadIt than for Star.
- Star performs 4 times worse than SpreadIt on the time to first packet, and packet losses in the transient state metrics. The difference increases to 10 times in flash crowd.
- Among the various policies, SP performs well for addition, and GF performs well for deletion.
- All the QoS metrics increase linearly with the depth of the multicast tree. However, with the increasing of depth of the tree, the number of supported clients are exponentially increased.



Some Further Work

- Build multiple distribution tree.
  - Coopnet by Microsoft.
- Use a cluster of clients instead of one client to construct the distribution tree.
  - UCF paper.