#adobemax151

P2P on the Flash Platform with RTMFP

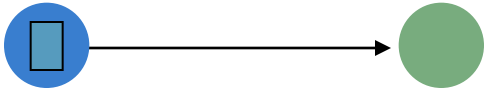
Matthew Kaufman

Sr. Computer Scientist, Project Lead

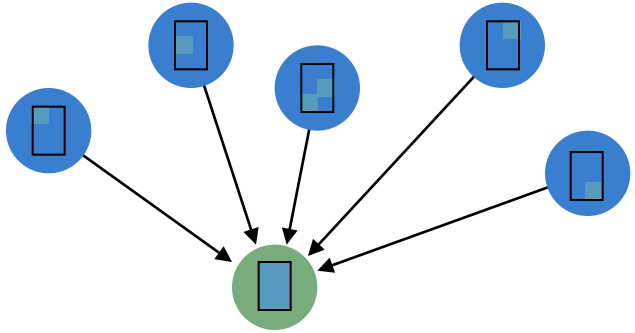
Adobe

What is “Peer-to-Peer”?

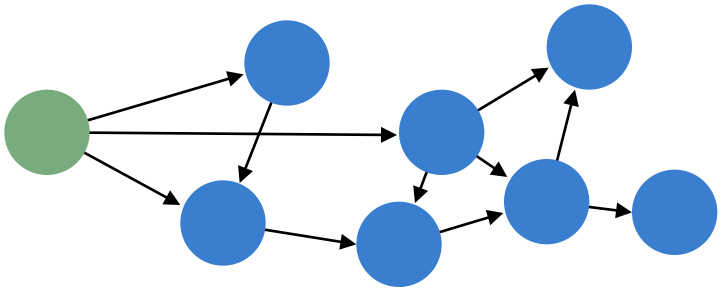
- Point-to-point
  - Live streaming
  - Document delivery



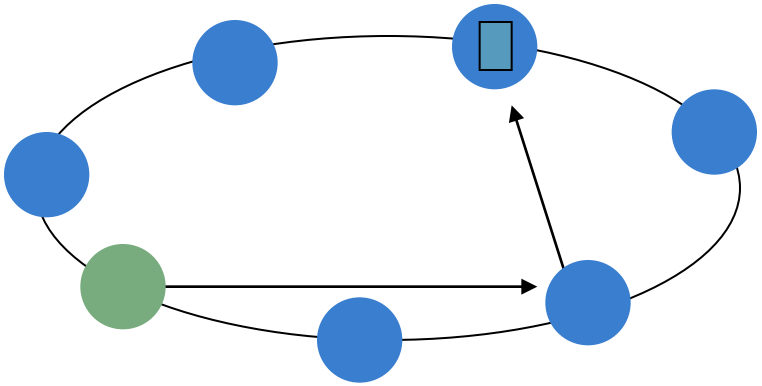
- “Swarming”
  - Large-file download (possibly progressive)



- Live Application-Level Multicast
  - Broadcast (1 to many, some latency tolerable)
  - Interactive (many to many, or 1 to many with feedback, low latency required)



- Distributed Data Storage
  - DHT-like structures to form distributed database

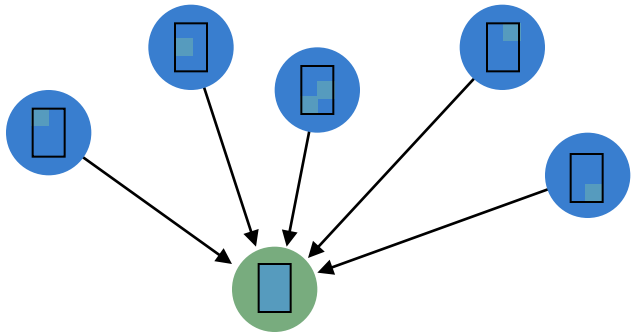


Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

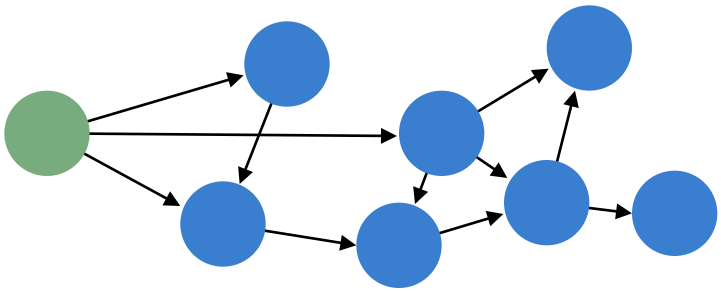
Peer-to-Peer Capabilities in Flash Player 10.0

- Point-to-point
  - Live streaming
  - Document delivery

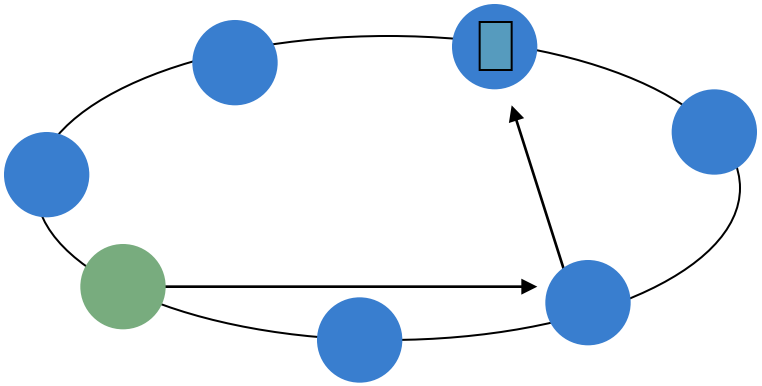
- "Swarming"
  - Large-file download (possibly progressive)



- Live Application-Level Multicast
  - Broadcast (1 to many, some latency tolerable)
  - Interactive (many to many, or 1 to many with feedback, low latency required)



- Distributed Data Storage
  - DHT-like structures to form distributed database

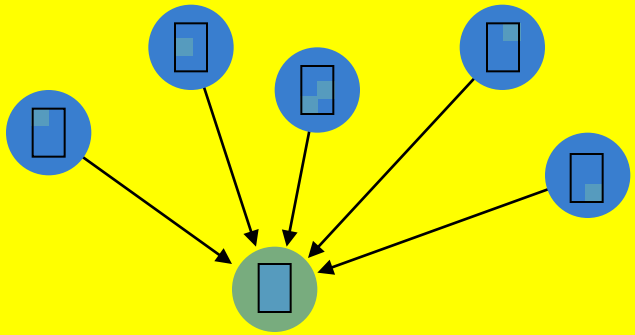


Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

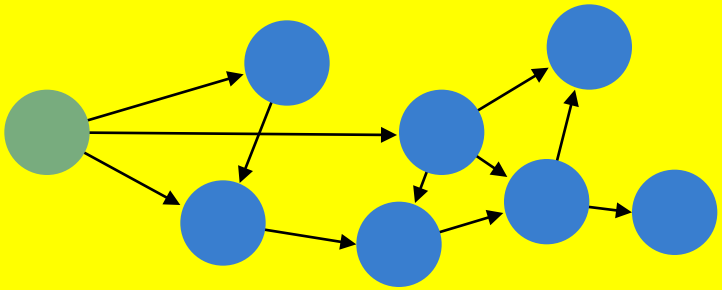
New Peer-to-Peer Capabilities in Flash Player

- Point-to-point
  - Live streaming
  - Document delivery

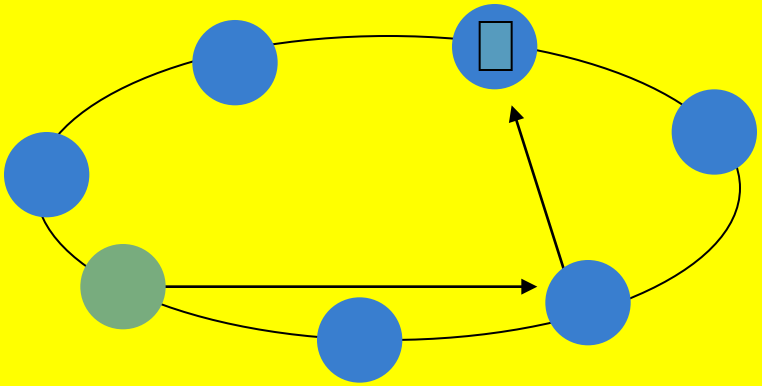
- “Swarming”
  - Large-file download (possibly progressive)



- Live Application-Level Multicast
  - Broadcast (1 to many, some latency tolerable)
  - Interactive (many to many, or 1 to many with feedback, low latency required)



- Distributed Data Storage
  - DHT-like structures to form distributed database



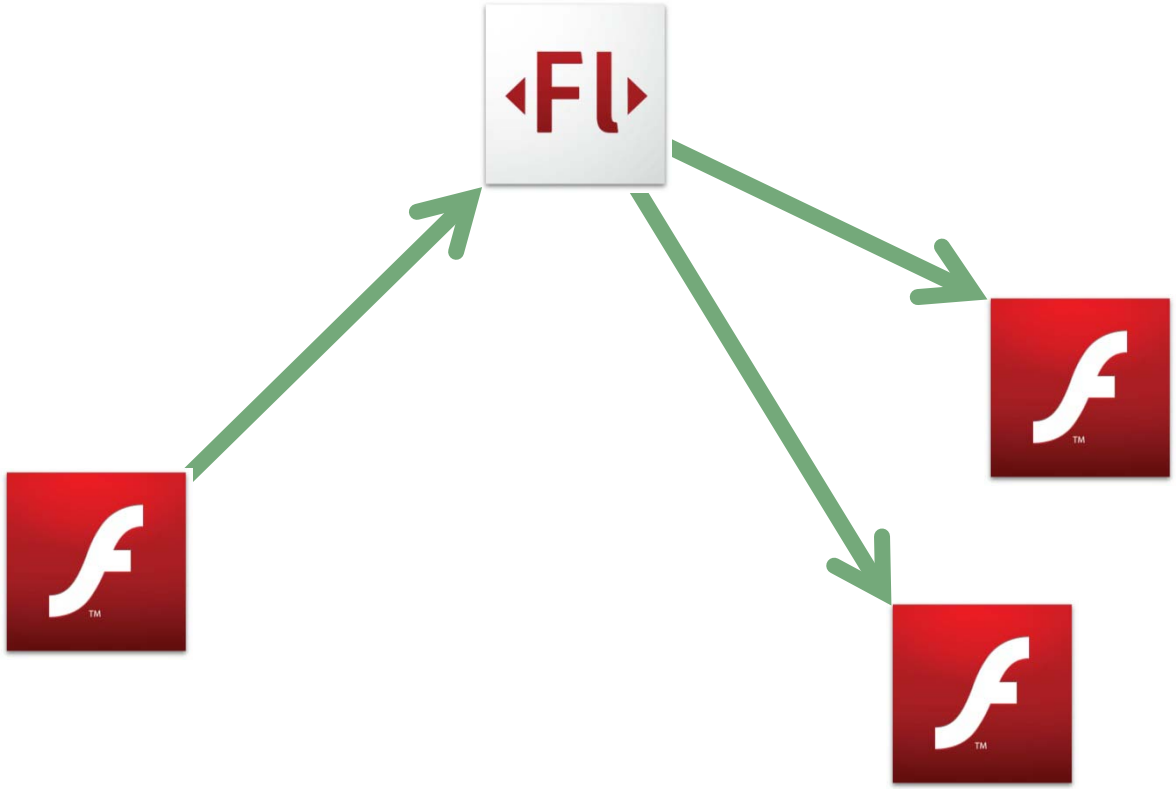
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Overview of New Peer-to-Peer Capabilities

- Join and participate in a self-organizing peer-to-peer overlay network
- Use that network for:
  - Directed Routing
  - Object Replication
  - Posting
  - Application-Level Multicast
- Use Native IP Multicast together with Application-Level Multicast
  - “Fusion”

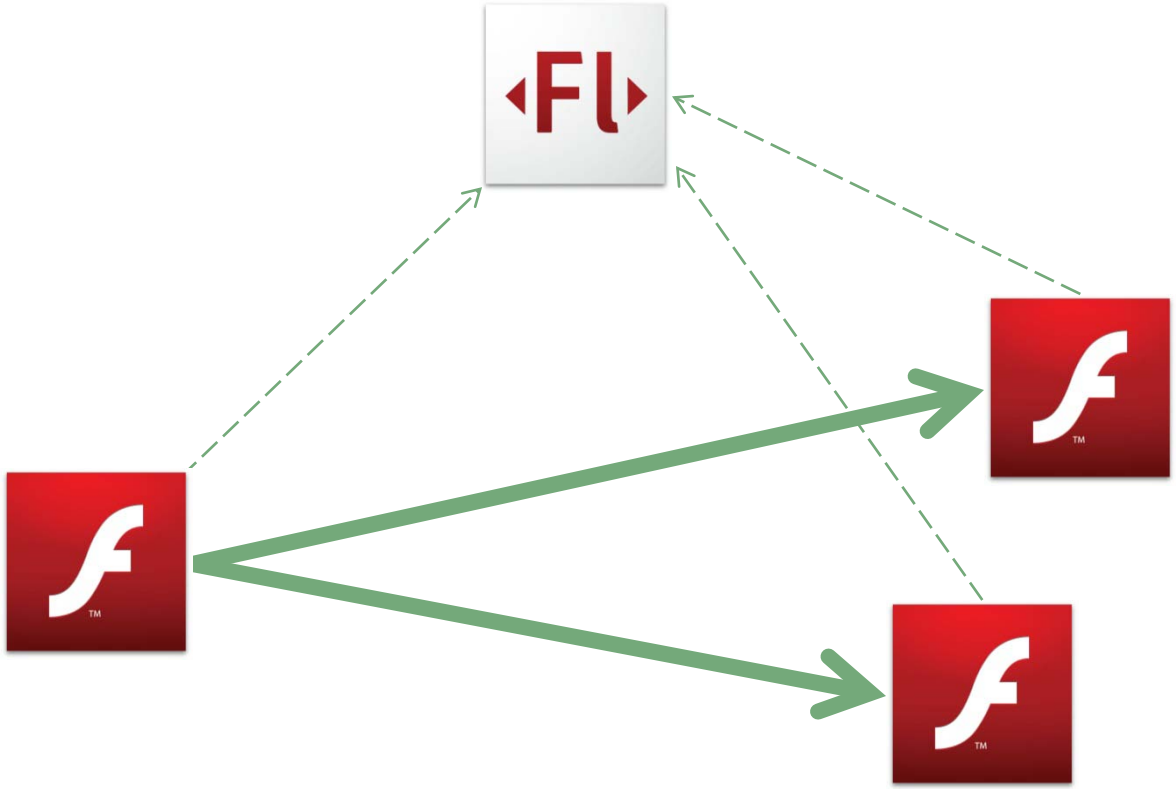
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Publishing to (a few) Subscribers through FMS



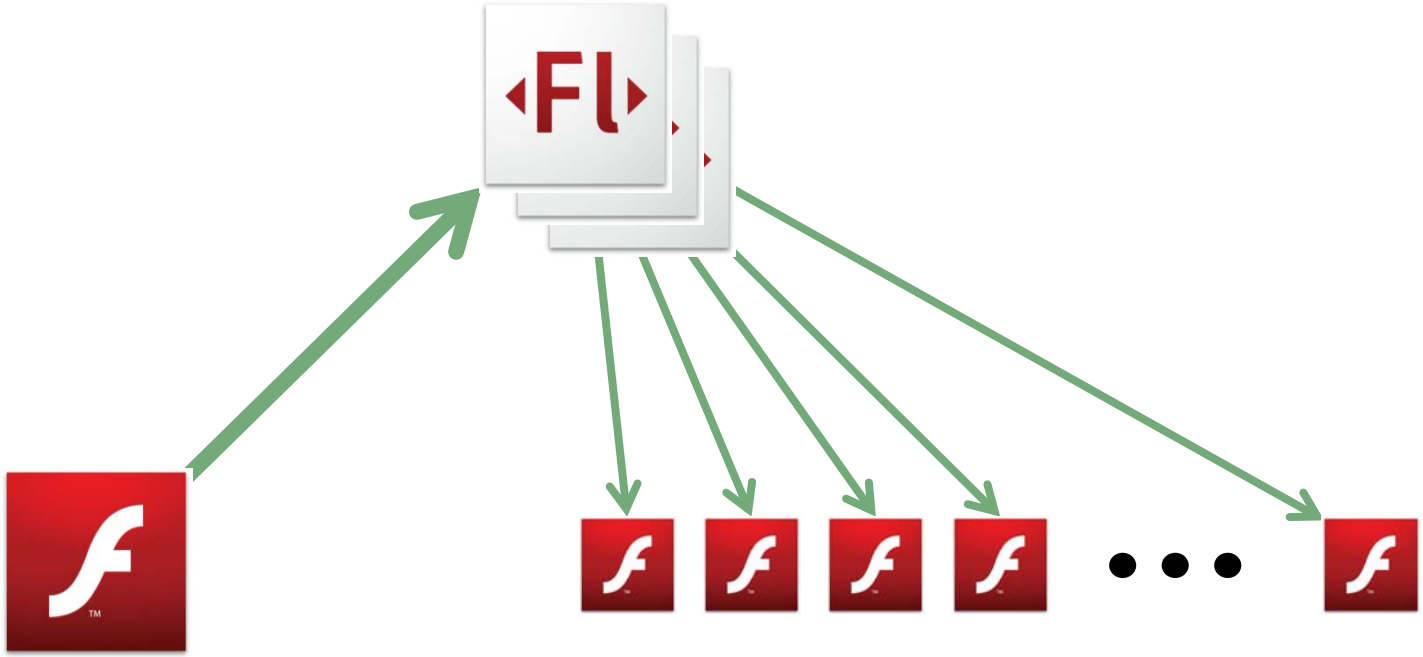
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Publishing to (a few) Subscribers Peer-to-Peer



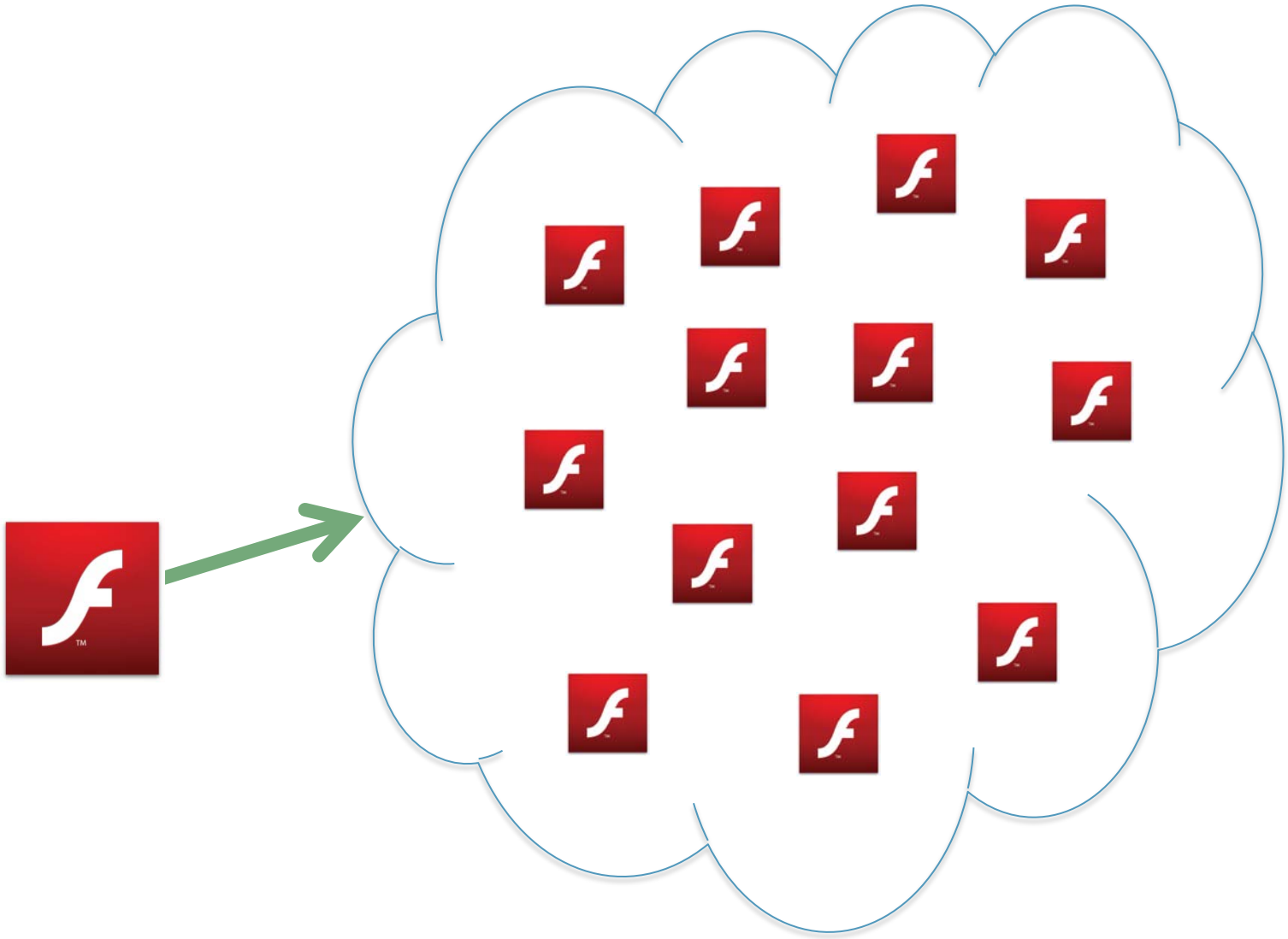
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Publishing to (many) Subscribers through FMS



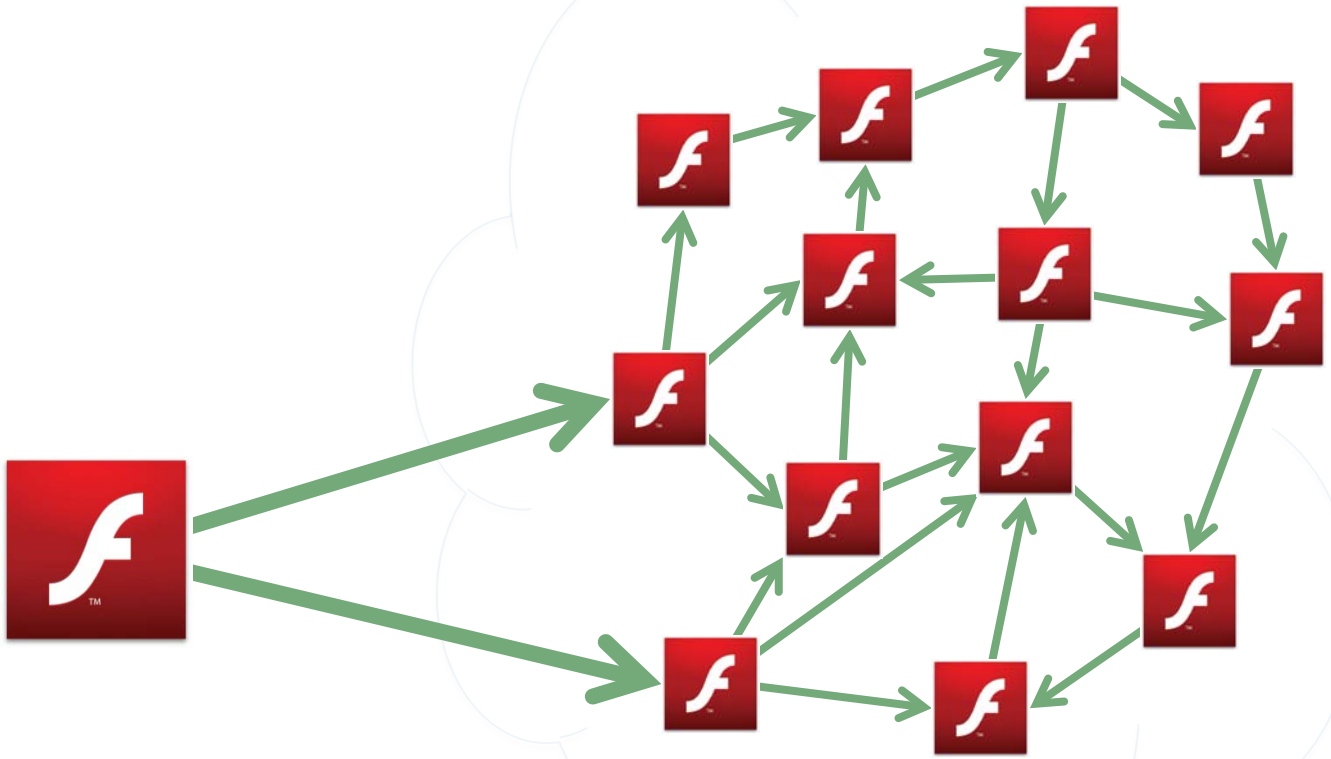
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Publishing to (many) Subscribers using RTMFP Groups



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Publishing to (many) Subscribers using RTMFP Groups



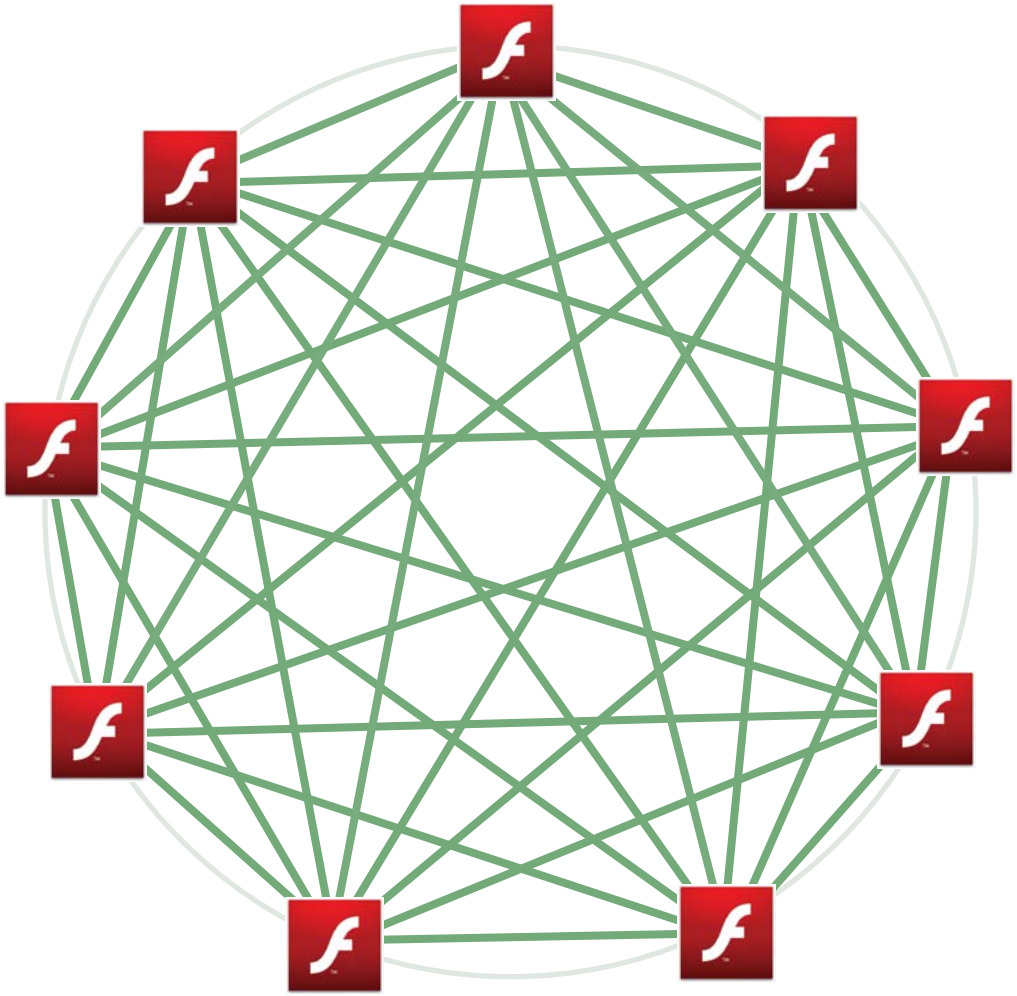
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Flash Player Communication

- 1 to 1 (1 to few)
  - Communicate via Server
    - RTMP (RTMPT, RTMPS)
    - RTMFP
  - Communicate Directly
    - RTMFP
- 1 to Many, Many to Many
  - Communicate via Server
  - Communicate using RTMFP Groups

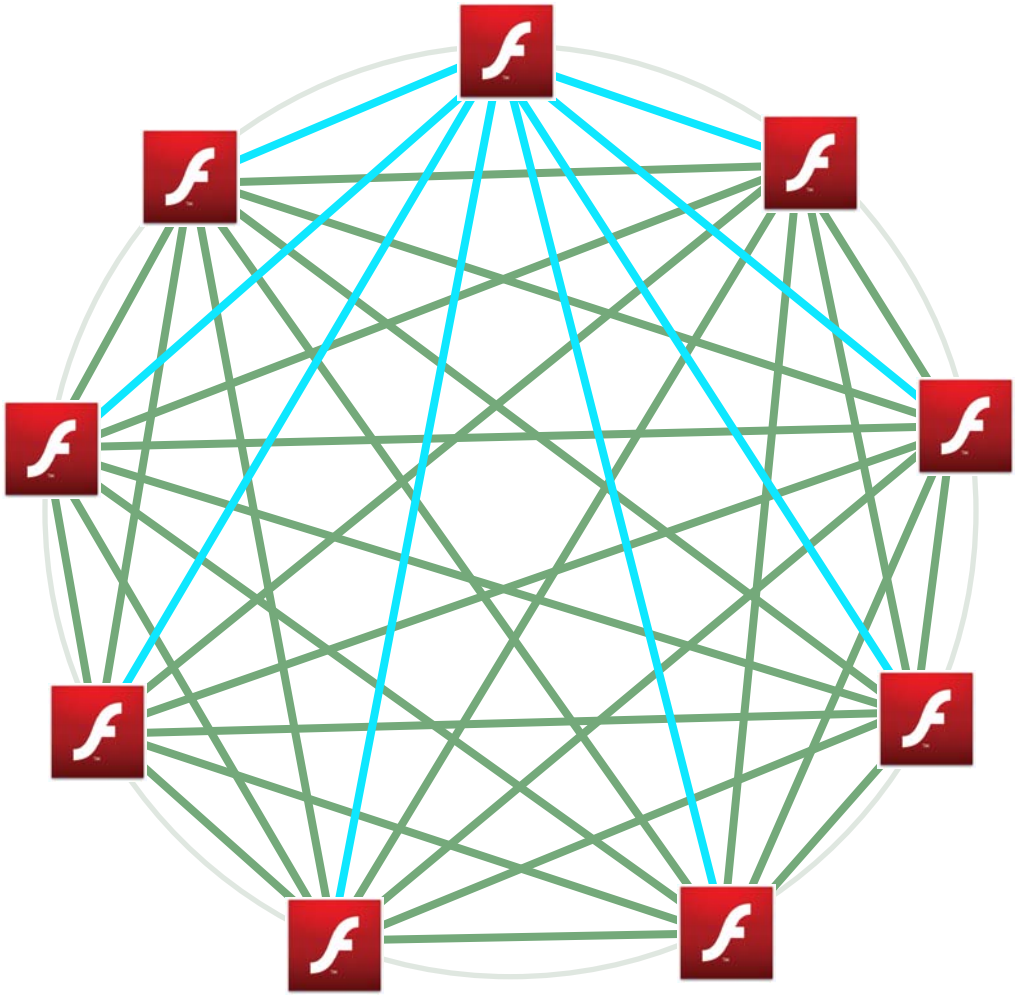
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Full Mesh of Peers



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Full Mesh of Peers



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

RTMFP Groups

- Player needs to communicate with *n* peers efficiently
  - *n* might be very large (millions)
  - O(*n*) at each peer (O(n^2) overall) doesn't scale
- Solution: Overlay network
  - Topology optimized for multiple uses

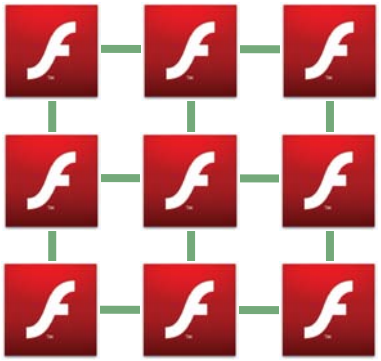
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

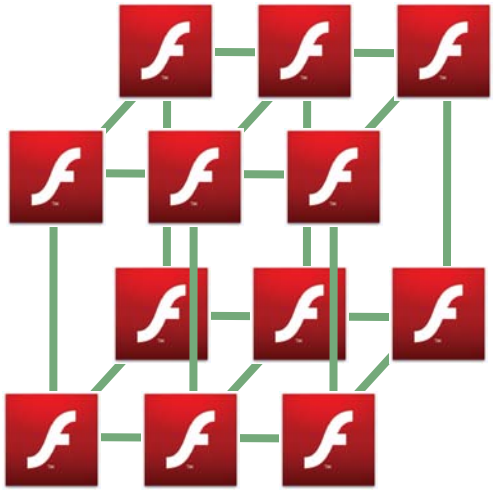
RTMFP Group Topology

- RTMFP is used as transport
  - NAT/Firewall traversal, Encryption, IP Address Mobility, Congestion Control, Partial Reliability, etc.
- Each peer has O(log *n*) connections
  - Typical: $2 \log_2 n + 13$
- Epidemic theory answers probability of connectedness
  - For $\log_2 n + c$ connections, probability is $e^{-e^{-c}}$
    - So c should be at least 10
- Topology management
  - decoupled from data transport
  - distributed to peers
- Controlled access
  - Ensures joiners have secret (the Groupspec) without exposing secret

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

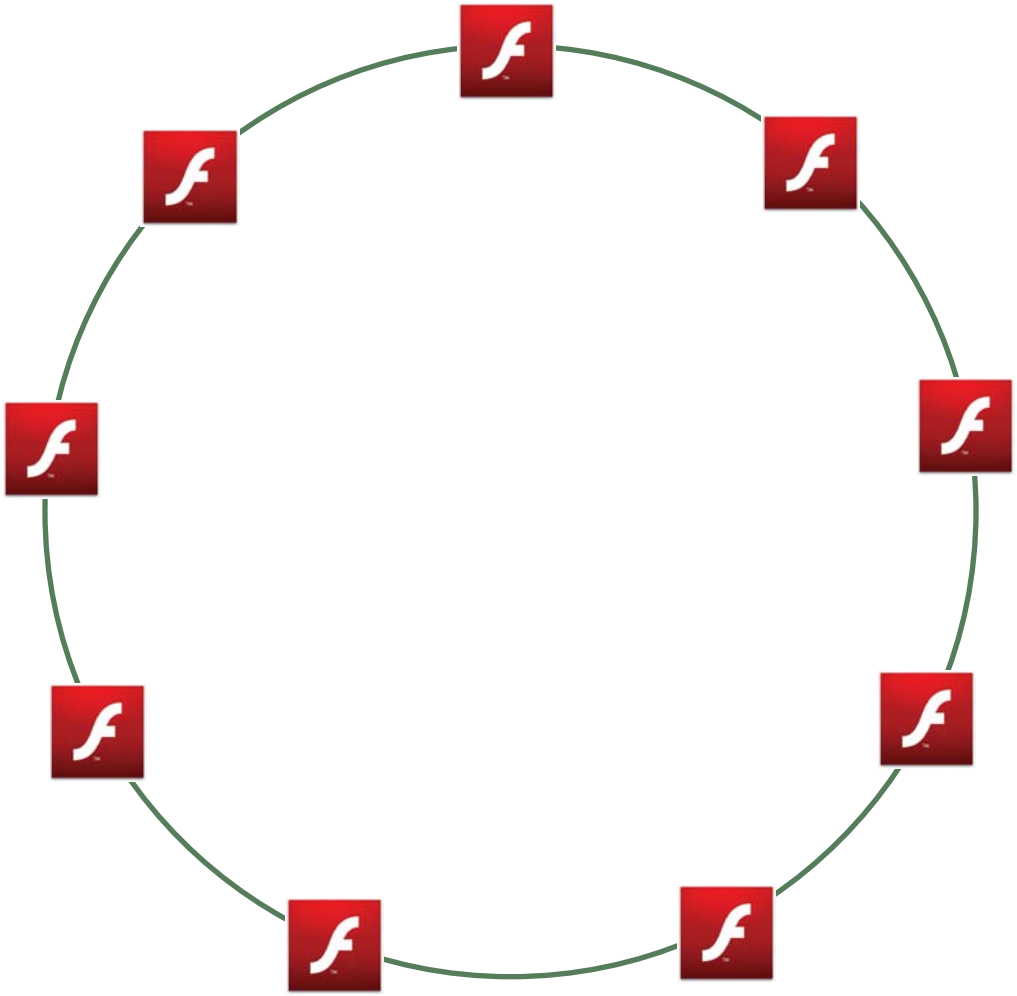
Group Topology





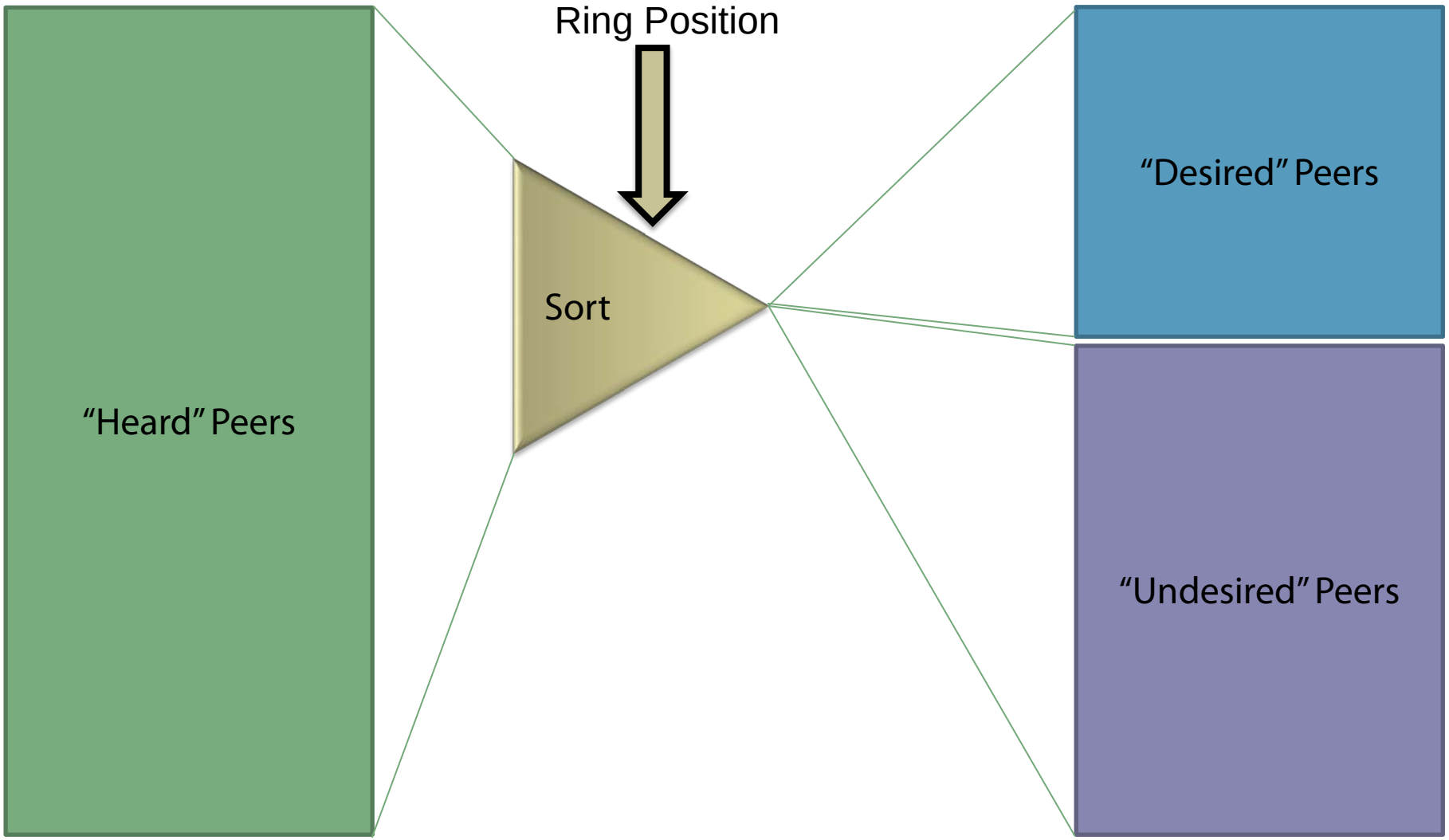
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Group Topology



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Group Topology Management



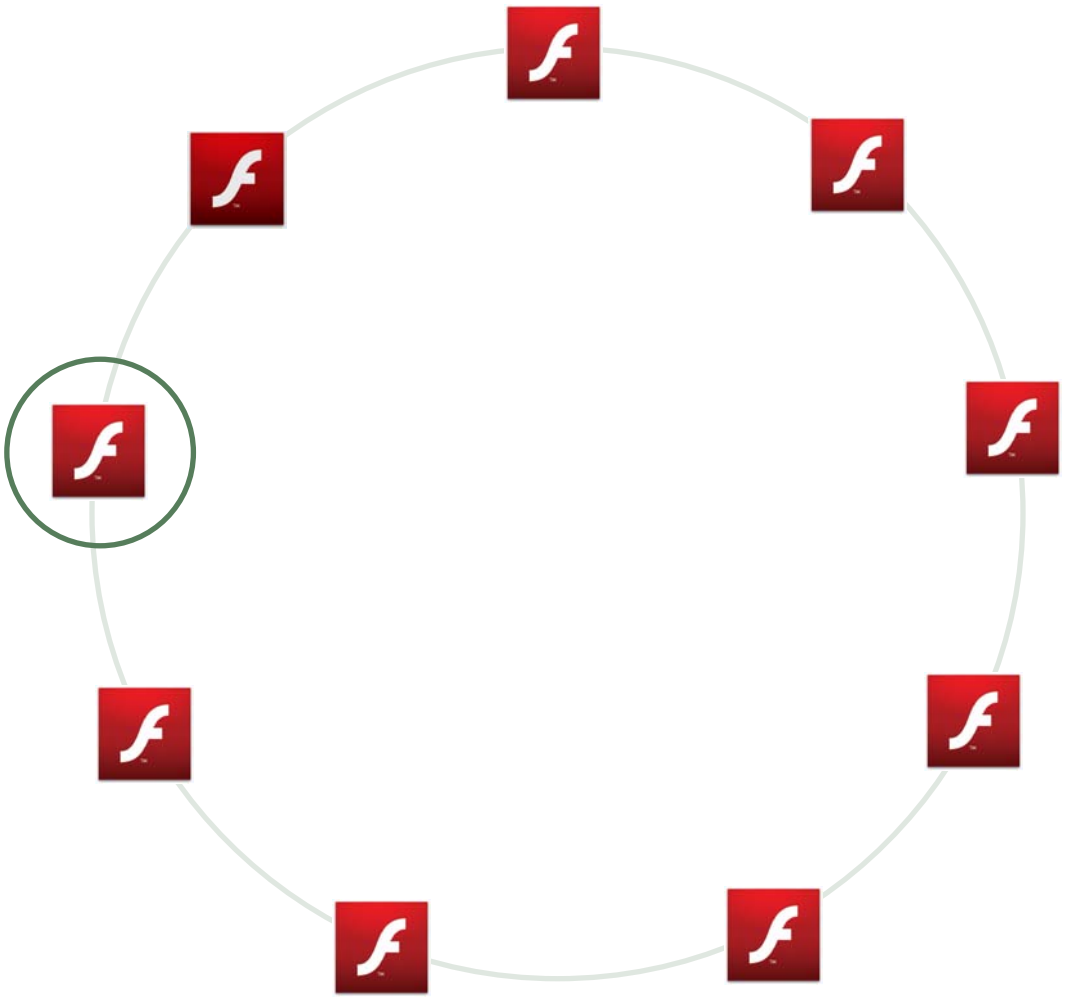
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Topology Management Algorithm

- Initialize the “heard list” from one or more bootstrap nodes
- Repeat forever:
  - Sort the “heard list” to find what is best for self making a “best” list
  - Connect (or stay connected) to the nodes on the “best” list
  - Ask connected nodes that are no longer on the “best” list to disconnect
  - If a node asks you to disconnect, do so only if not in your “best” list
  - Every interval, pick one node at random that you are talking to
    - Sort the “heard list” to find what is best for them
    - Send them the result of that sort and ask them to do the same in return (“push-pull”)
  - Every time you hear about new nodes, merge them into the “heard list”
  - Delete nodes from “heard list” if their age limit is exceeded
- Resulting self-organizing topology depends entirely on sorting algorithm
- Continuous optimization through gossip, rate is higher than natural churn

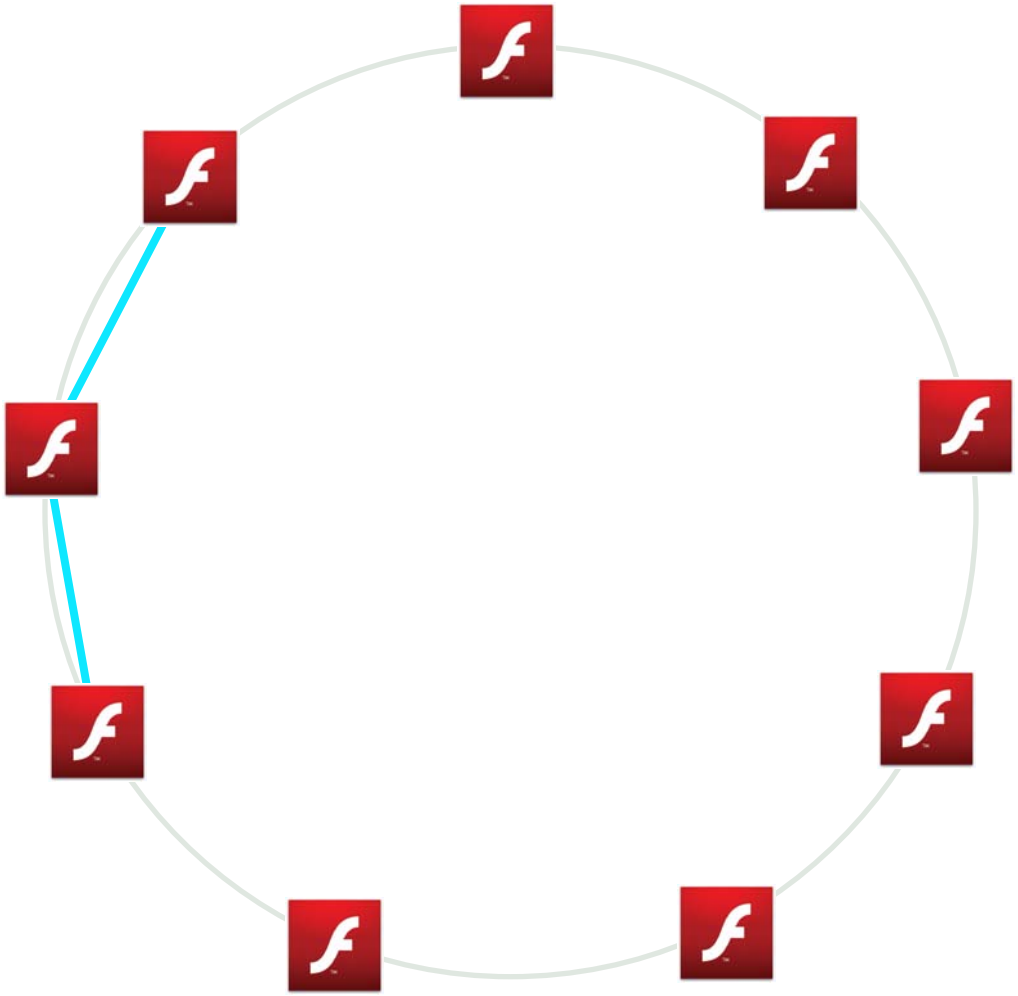
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Group Topology Details



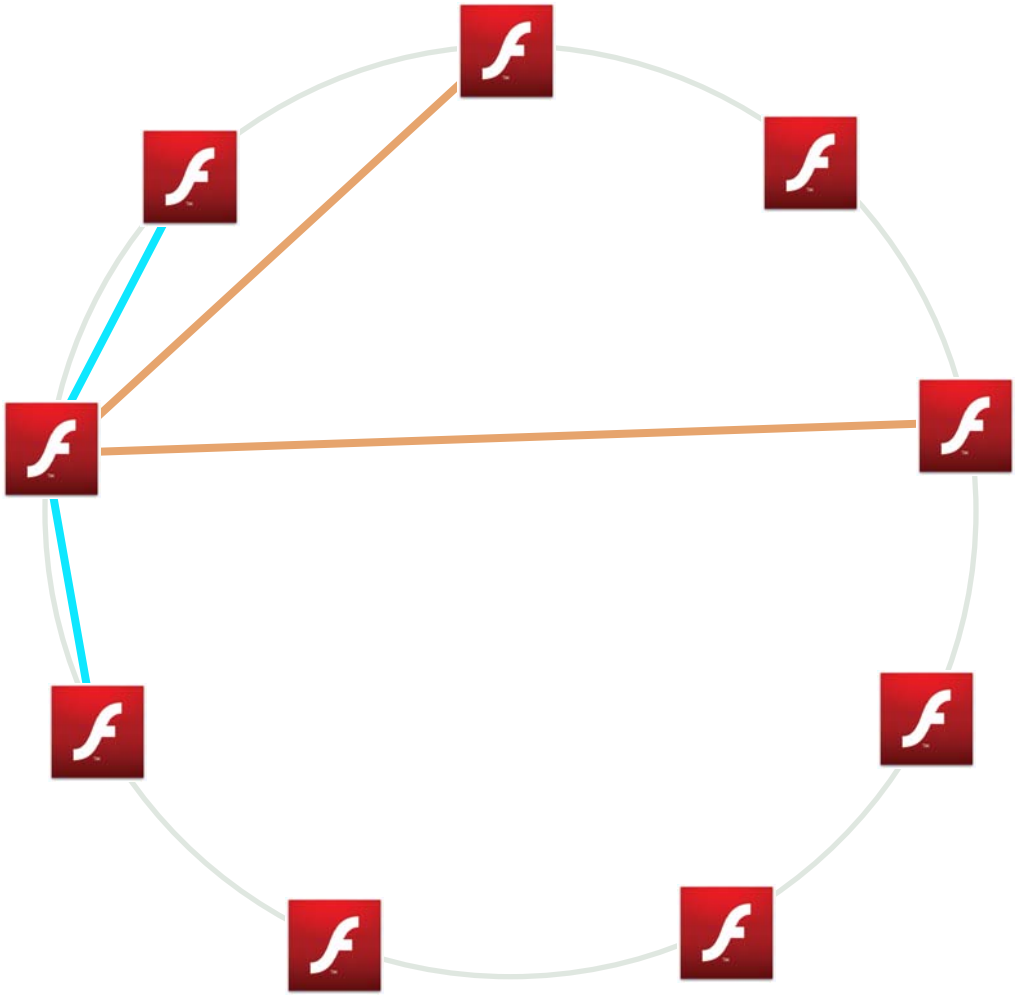
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Group Topology Details



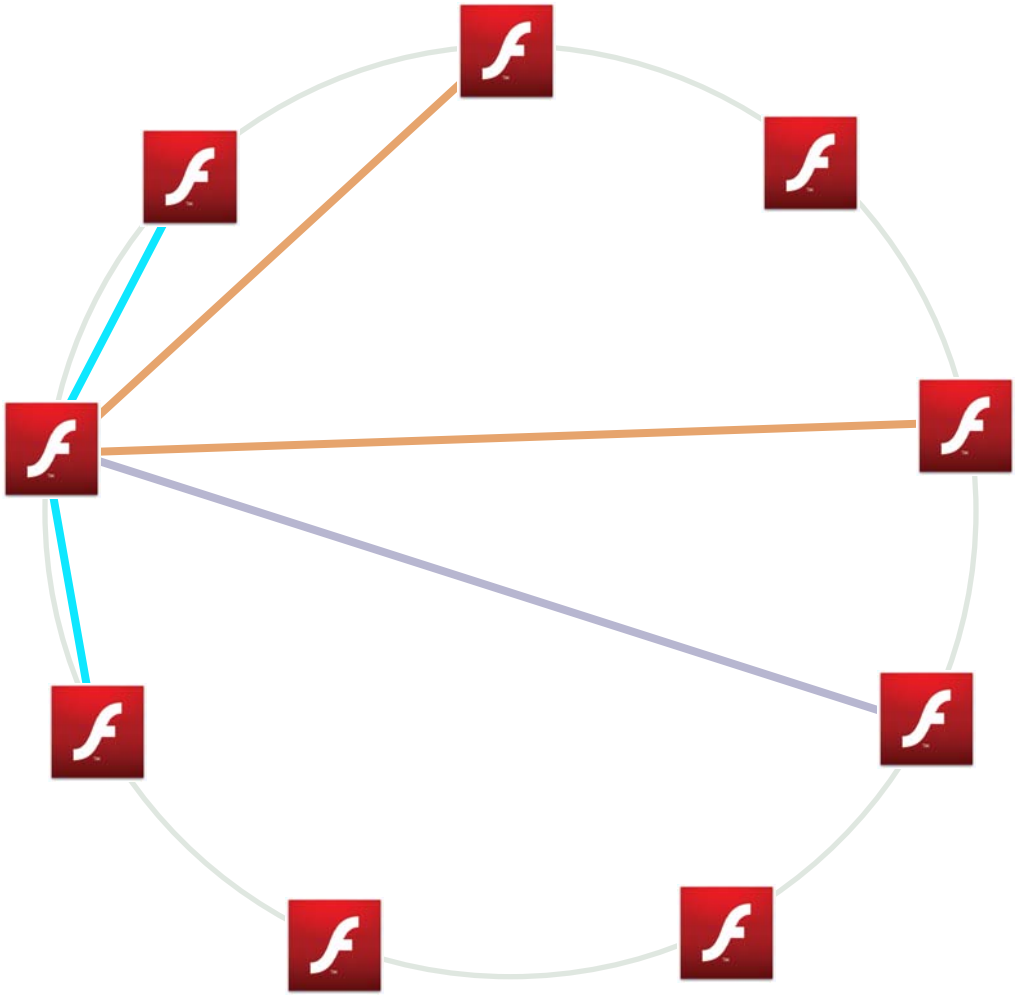
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Group Topology Details



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Group Topology Details



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Topology Management Details

- Nodes have a 256-bit numeric identifier ($0 \ldots 2^{256}-1$)
  - Number of dimensions = 1
- Sort chooses:
  - 6 closest (mod 2^{256}) in numeric space (3 closest on each side)
  - Numeric location + 1/2, 1/4, 1/8, 1/16, …
  - 6 nodes with lowest latency
  - 1 random node (per round)
- Results in a fully-connected ring, plus:
  - Redundant neighbor connections
  - Shortcuts across the ring to reduce average hop count to less than $\log_2 n$
  - Additional nodes that are nearby (based on measured latency)
  - Random nodes sufficient to continue optimizing

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Describing Groups in Flash Player

- Groupspec
  - String starting with “G:” (e.g., “G:01010b…..”)
    - First part becomes the immutable “identity” of the Group
      - Certain options (permissions) must be in this part
      - If a peer tries to change, they are in a Group, but not this one
    - Note: Case-sensitive
- GroupSpecifier class makes them

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

GroupSpecifier Class

- Properties
  - `routingEnabled`
  - `multicastEnabled`
  - `objectReplicationEnabled`
  - `postingEnabled`
  - `peerToPeerDisabled`
  - `ipMulticastMemberUpdatesEnabled`
  - `serverChannelEnabled`
- Methods
  - `GroupSpecifier(name:String)`
  - `makeUnique()`
  - `setPublishPassword()`
  - `setPostingPassword()`
  - `addBootstrapPeer()`
  - `addIPMulticastAddress()`
- Getting the Groupspec
  - `groupspecWithAuthorizations()`
  - `groupspecWithoutAuthoriziations()`

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Describing Self within Group

- Ring Position
  - 256-bit numeric identifier
  - Formed from PeerID

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Accessing Groups from ActionScript

- NetGroup
  - Management and statistics
  - All Group functionality except Multicast
- NetStream
  - Multicast
- More than one object can refer to the same Group
  - As long as in same NetConnection, overhead is shared

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

NetGroup Functionality

- Properties
  - `estimatedMemberCount`
  - `neighborCount`
  - `info`
    - `NetGroupInfo` properties
- Methods
  - `addNeighbor`
  - `addMemberHint`
  - `close`
- NetStatusEvents
  - `NetGroup.Neighbor.Connect`
  - `NetGroup.Neighbor.Disconnect`

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Using Groups

- Connect
  - Peer-Assisted Networking dialog will be displayed
    - When NetStream or NetGroup is constructed
    - Unless user remembers “allow” for your domain,
    - or `peerToPeerDisabled` is set in GroupSpecifier
- Use
  - Directed Routing
  - Object Replication
  - Posting
  - Multicast
    - Fusion

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Directed Routing

- Requires stable and correct topology to work well
- Works with ring positions
- API
  - `receiveMode`
    - `NetGroupReceiveMode.EXACT` (Initialized in this mode)
    - `NetGroupReceiveMode.NEAREST`
  - `sendToNearest`
  - `sendToNeighbor`
    - `NetGroupSendMode.NEXT_INCREASING, NetGroupSendMode.NEXT_DECREASING`
  - `sendToAllNeighbors`
- At each hop, arrives in ActionScript as event
  - `NetGroup.SendTo.Notify`

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Directed Routing

- Each hop can
  - Forward, Modify, Reply, or Discard
- Use cases:
  - Replying to specific node
  - Distributed Hash Table database
  - Research projects

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Object Replication

- Objects are addressed by numeric Index
- ActionScript provides backing store
  - Easier to use files from AIR
- API
  - `addHaveObjects / removeHaveObjects`
  - `addWantObjects / removeWantObjects`
  - `writeRequestedObject / denyRequestedObject`
  - `NetGroup.Replication.Fetch.Result`
  - `NetGroup.Replication.Request`
- Use cases:
  - Workspace replication
  - Whiteboard
  - File transfer
  - …

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Object Replication

- One Object ID space per group
- Fully reliable delivery
  - All other RTMFP Groups data transports are best-effort
- Hints for moving around large amounts of data:
  - Break up into reasonable-sized objects (64k)
  - Consider using block 0 as descriptive object block
  - Consider making Group “name” be related to hash of descriptive object block

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Posting

- Per-sender state (efficiency) tradeoff
  - Multicast is for a small # of senders @ high data rates
  - Posting is for a large # of senders @ low data rates
- API
  - `post()`
  - `NetGroup.Posting.Notify`
- Hints:
  - Messages must be unique to be considered new
    - Add a serial number
  - Keep messages small and relatively infrequent per sender
- Works without ActionScript involvement (unlike previous two modes)
- Use cases:
  - Text chat, sensor reporting, opinion polls

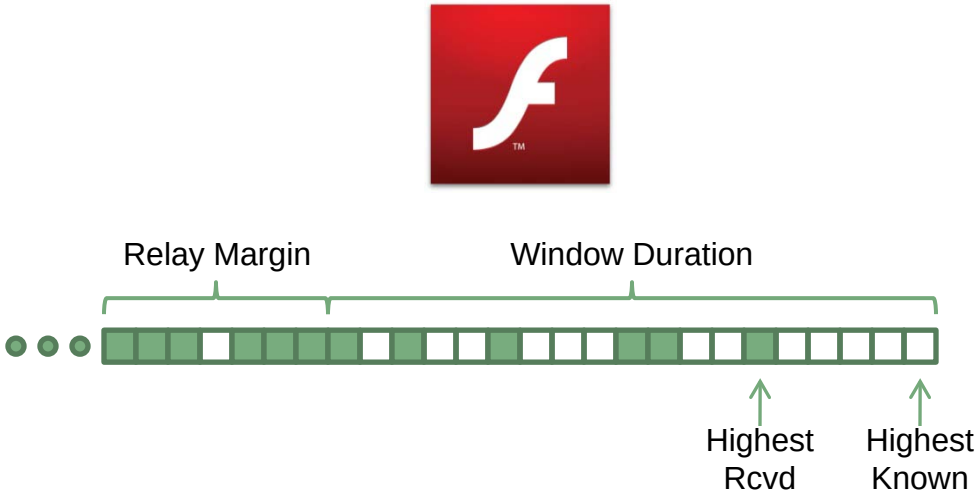
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast

- Uses NetStream API
  - Construct with Groupspec instead of PeerID
- Designed for low latency
- Again, works without ActionScript involvement

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

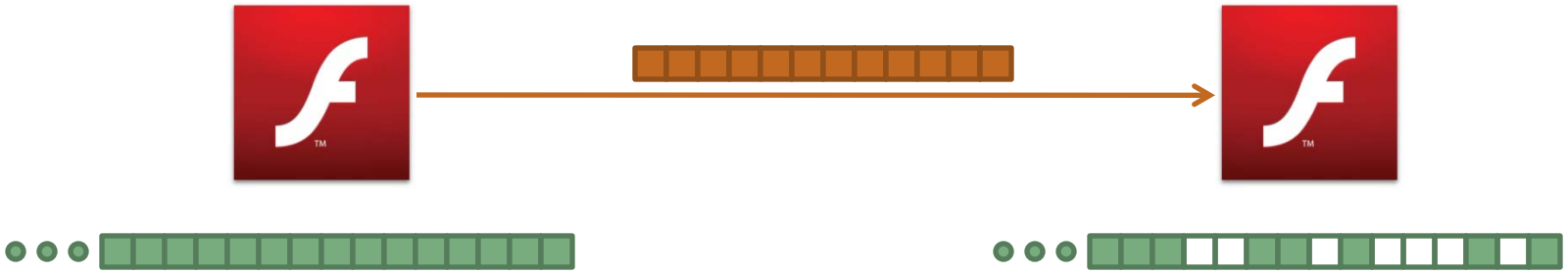
Multicast



- Multicast system attempts to get every block in the Window Duration
  - Once all blocks arrive for a message, message is processed
  - If any block of a message falls outside the Window, system no longer attempts to get any other block of that message
- Relay Margin provides extra time for in-flight requests to be fulfilled
- Both are in terms of time

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

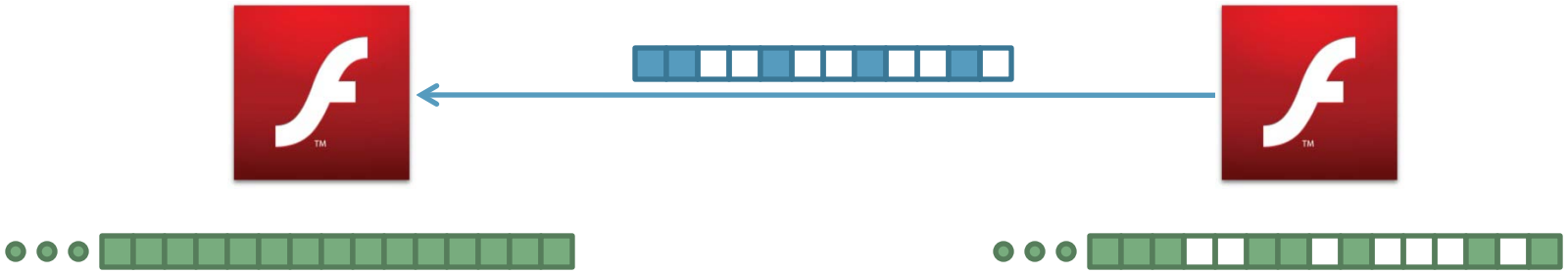
Multicast – Pull



- Send map of what blocks you have to one (or all) neighbor(s)
  - Neighbor waits a short time to see who can source each block

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



- Send map of what blocks you have to one (or all) neighbor(s)
  - Neighbor waits a short time to see who can source each block
- Neighbor picks you to send some of the blocks and sends request

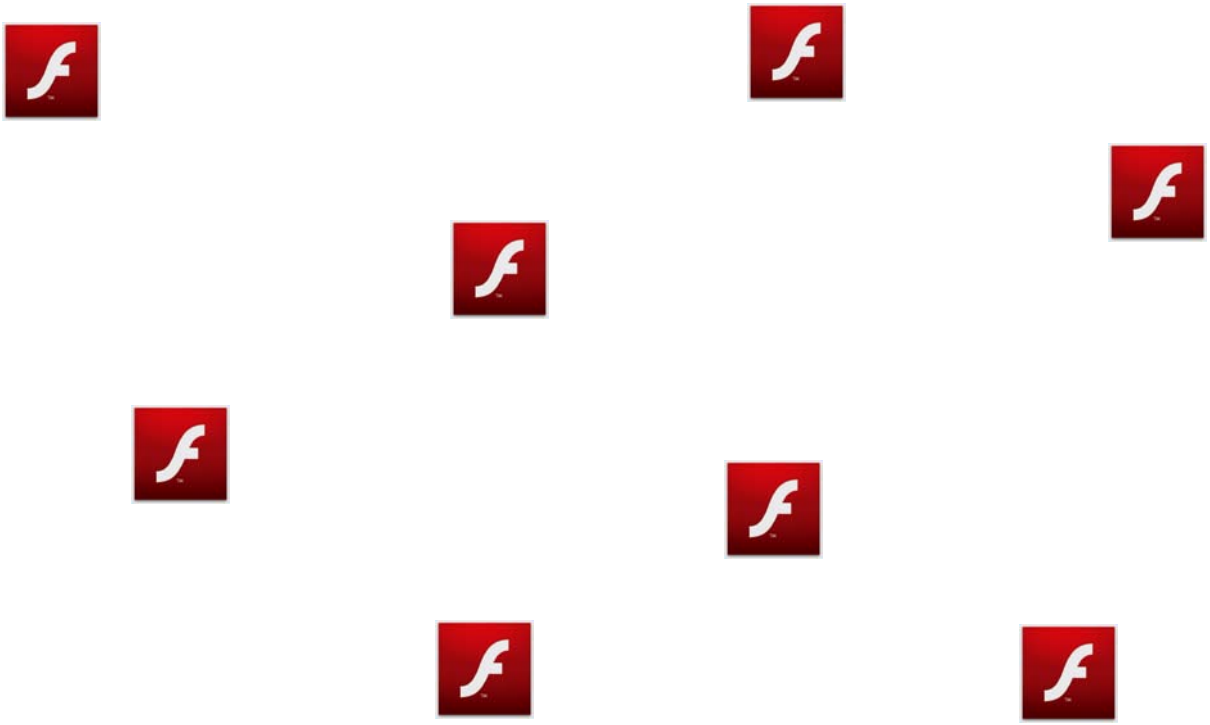
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull

- Send map of what blocks you have to one (or all) neighbor(s)
  - Neighbor waits a short time to see who can source each block
- Neighbor picks you to send some of the blocks and sends request
- Send requested blocks to neighbor
  - All transmissions are partially-reliable
  - Neighbor will ask someone else if request goes unfulfilled

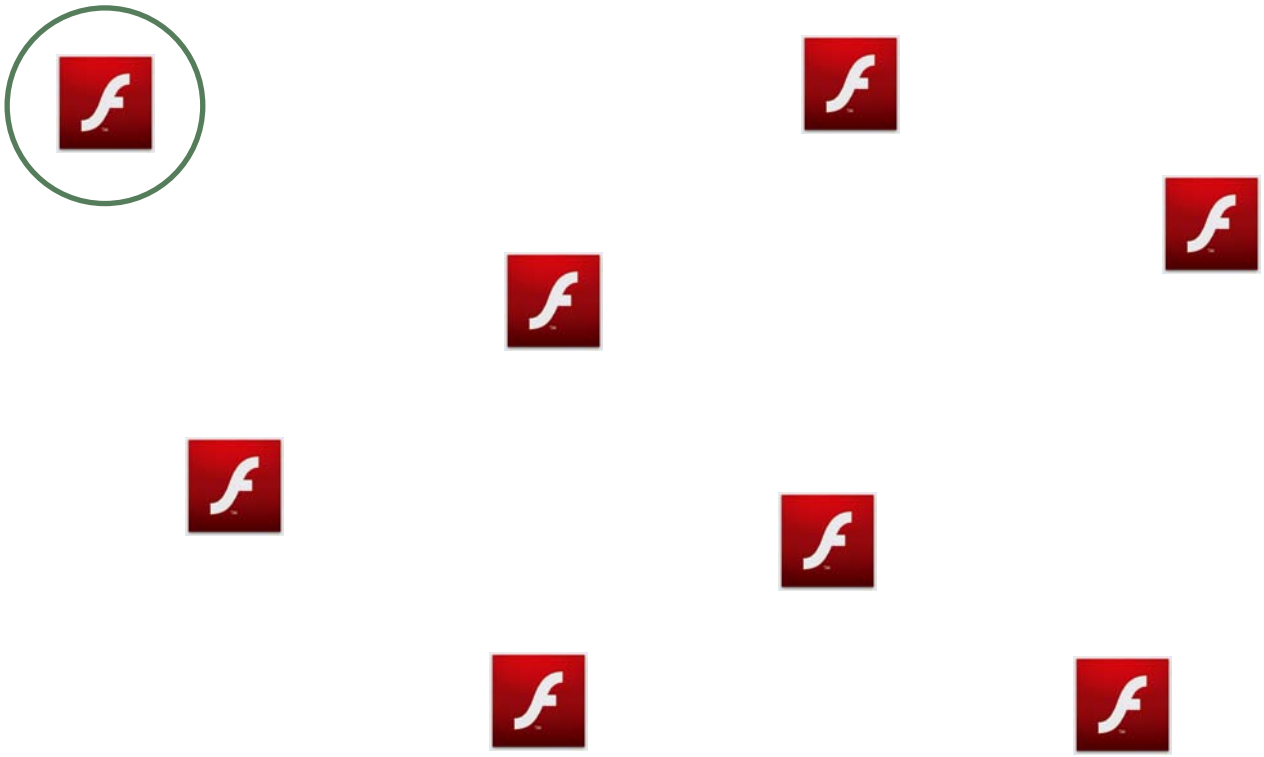
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



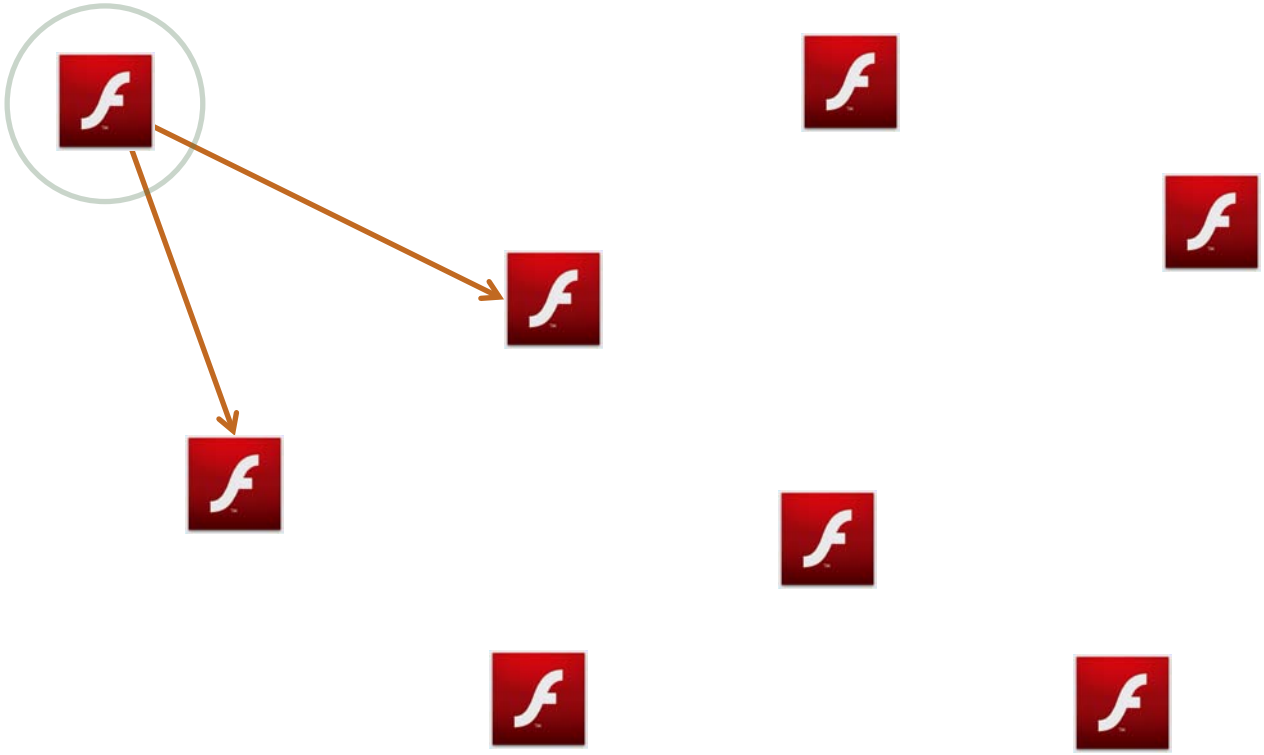
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



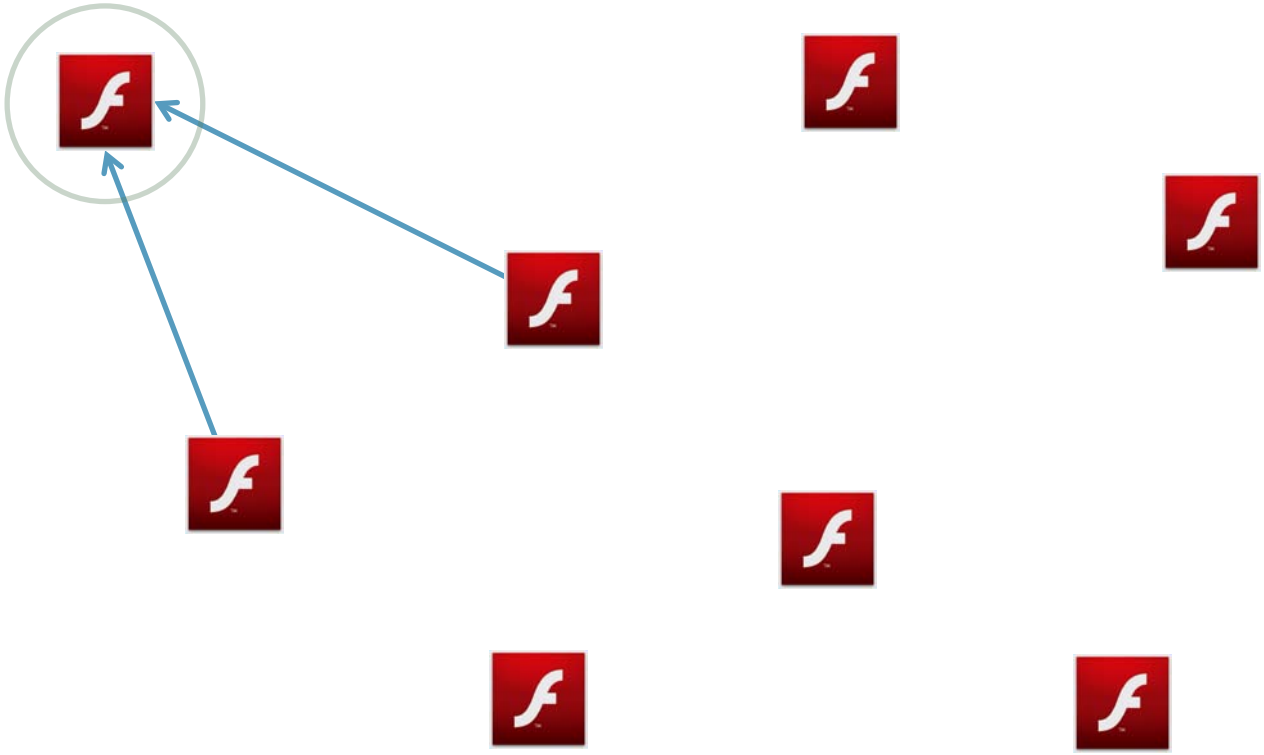
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



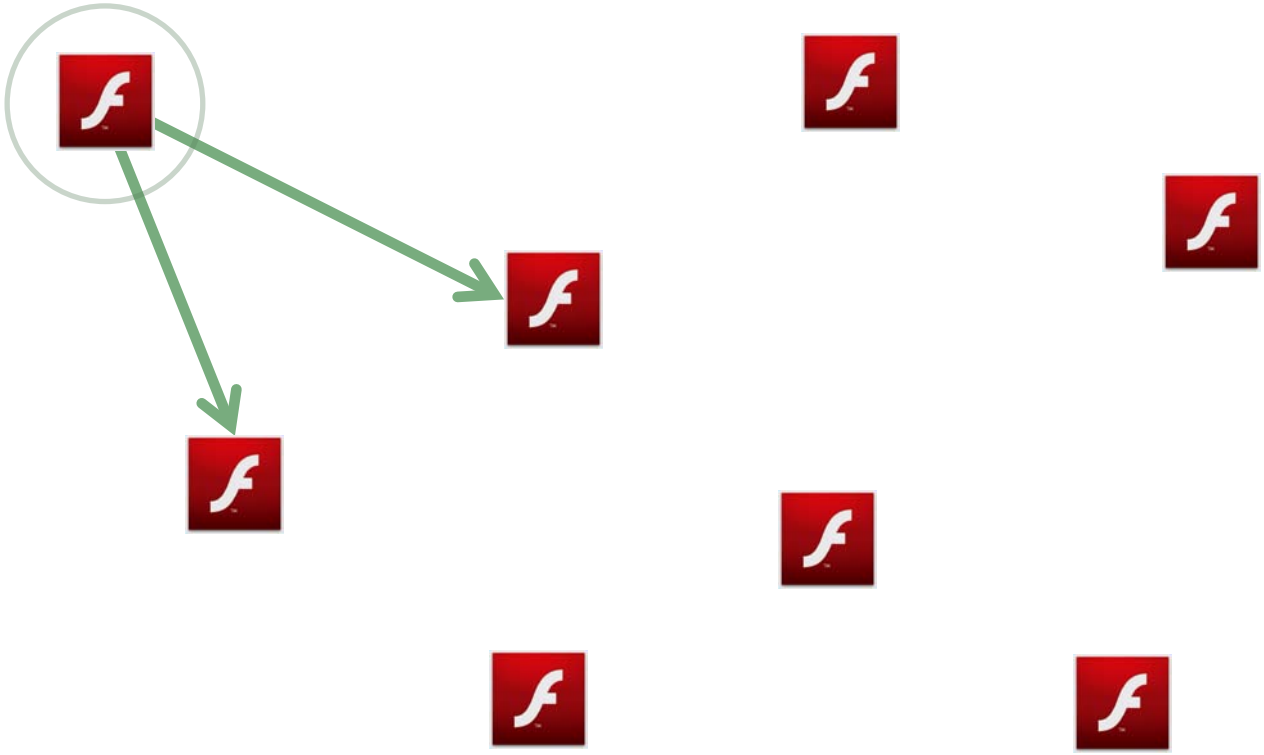
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



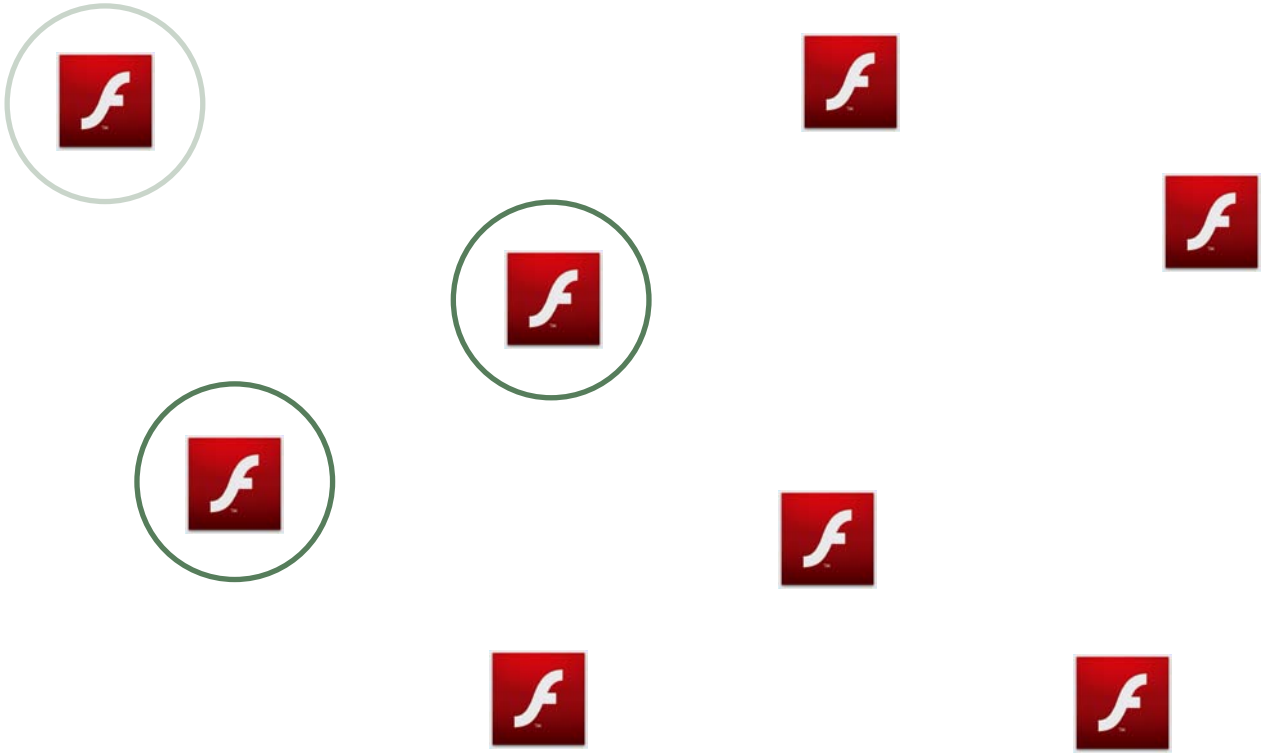
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



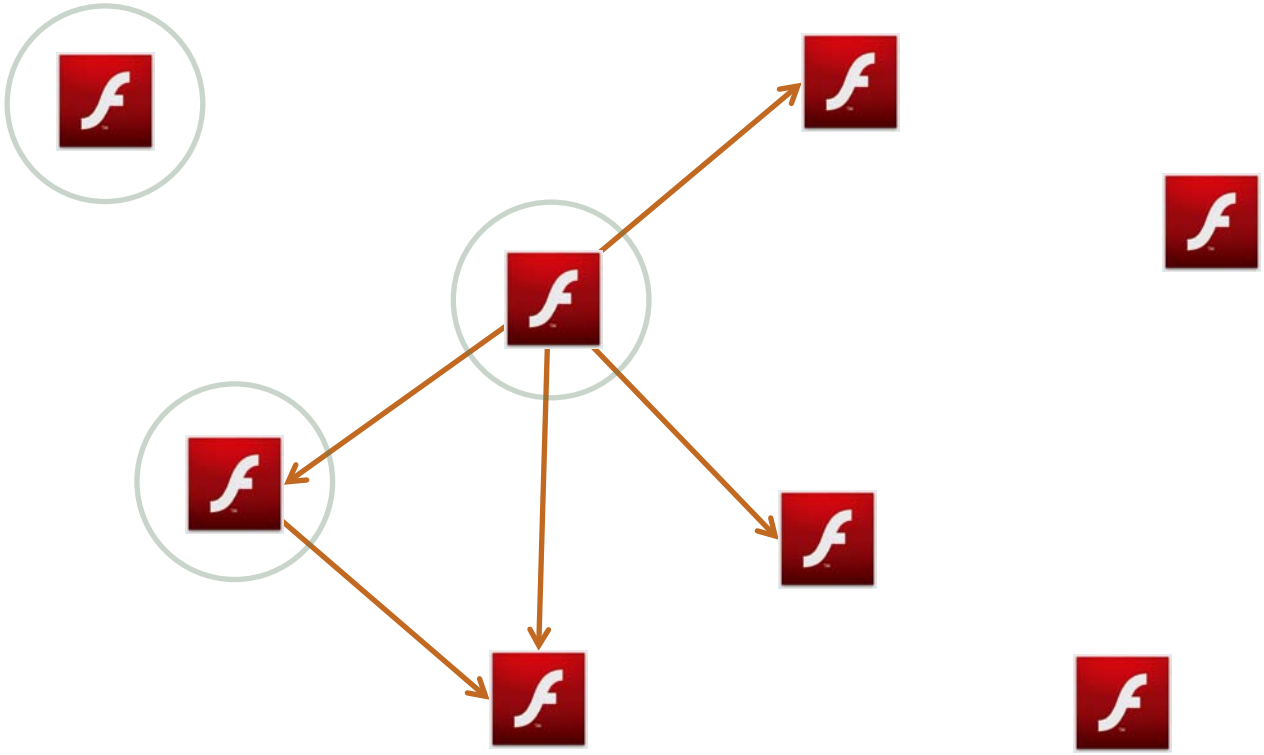
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



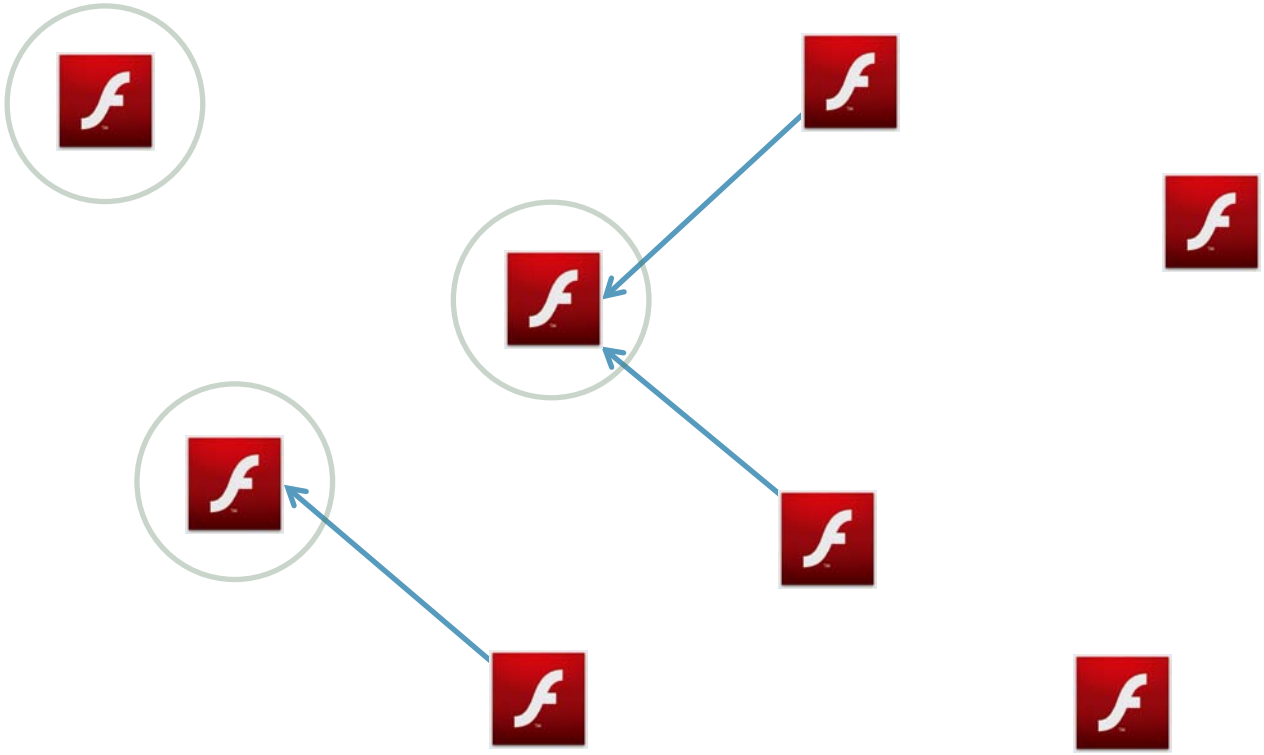
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



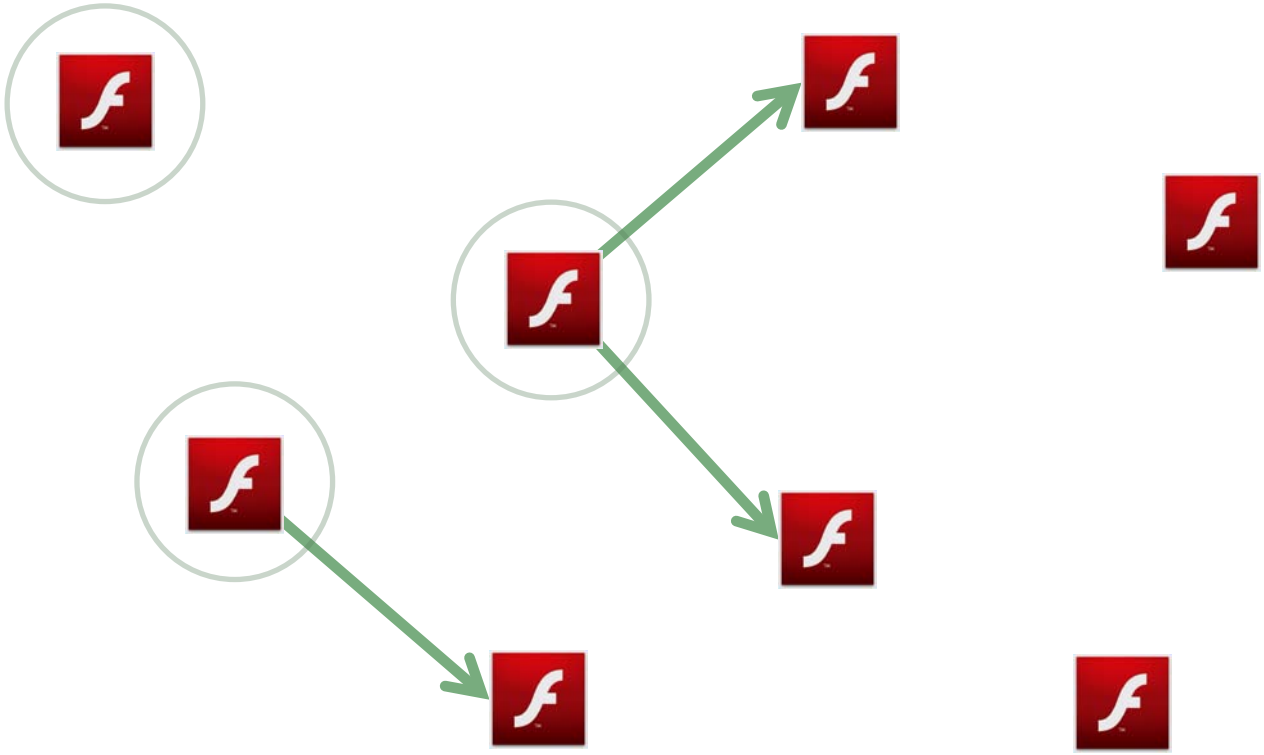
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



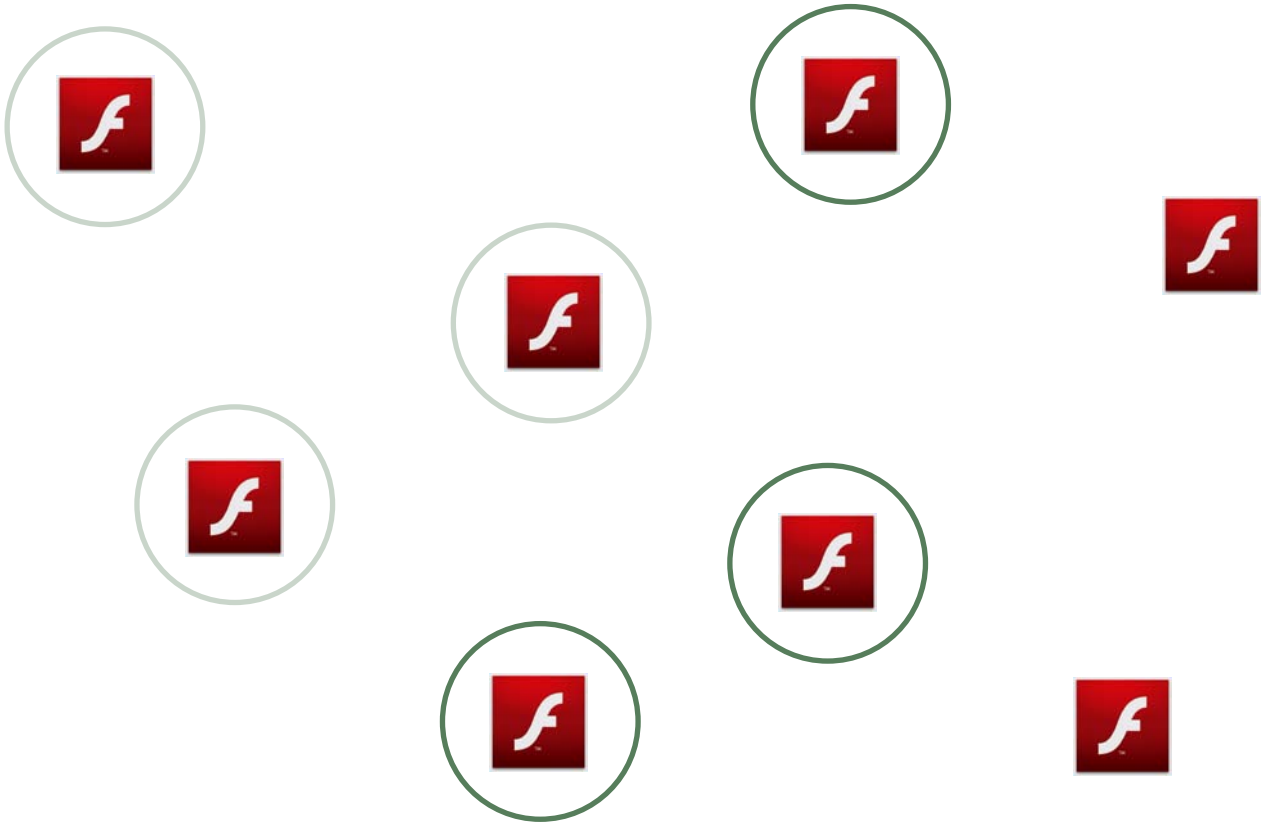
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



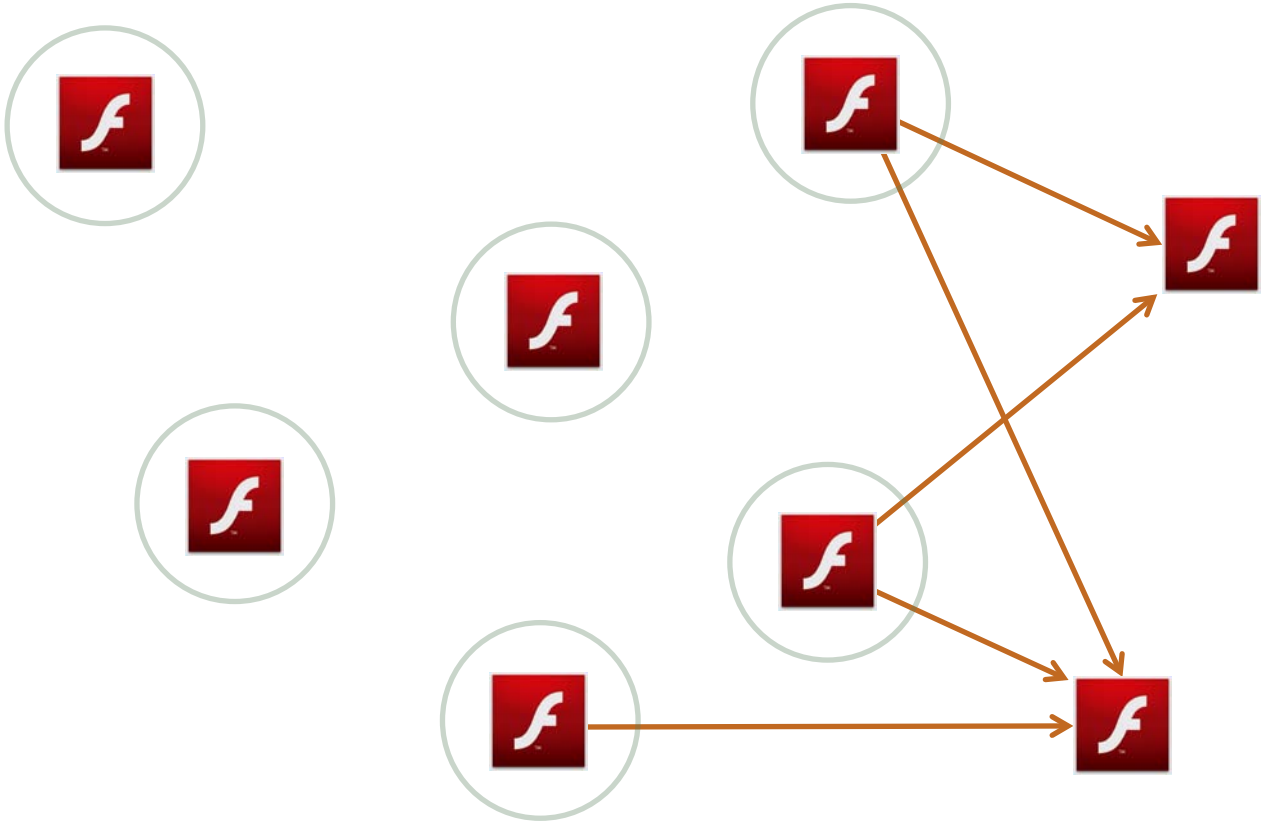
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



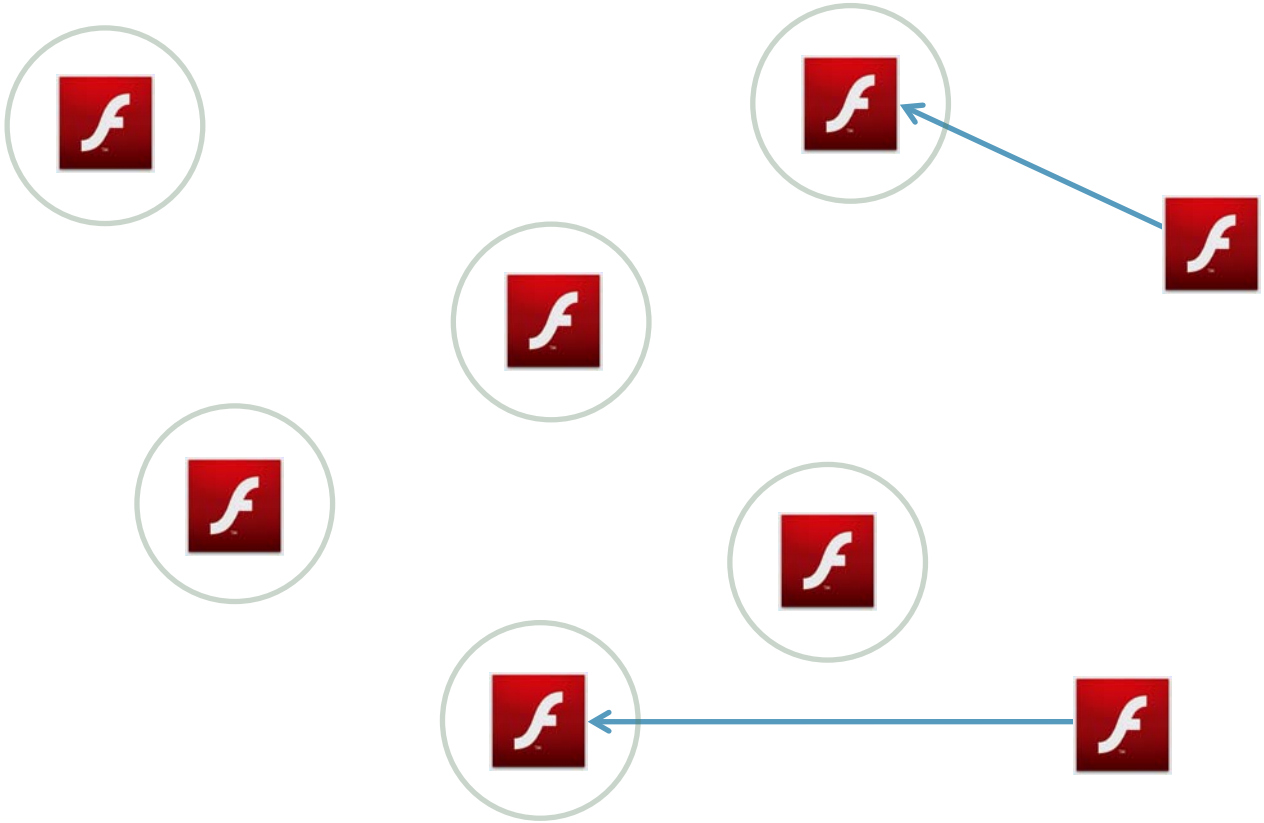
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



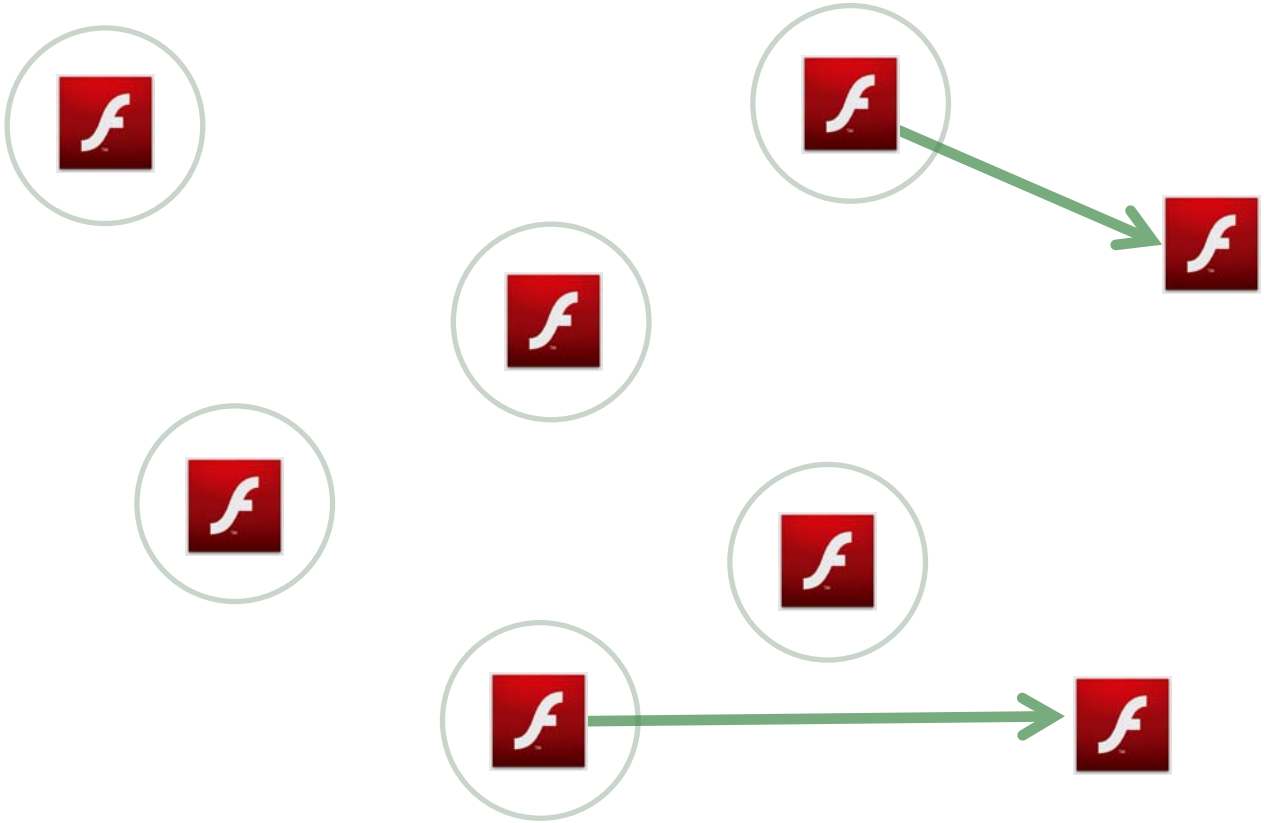
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



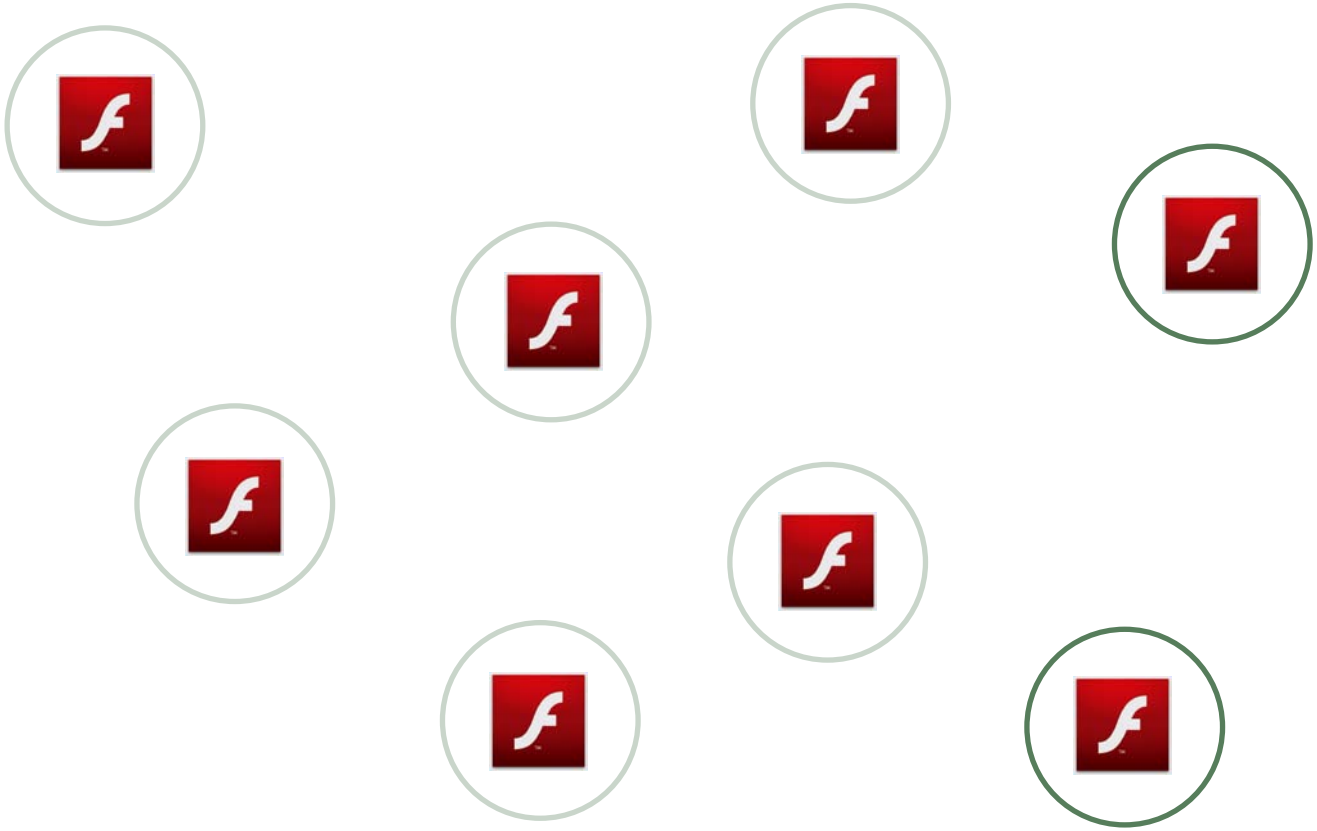
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



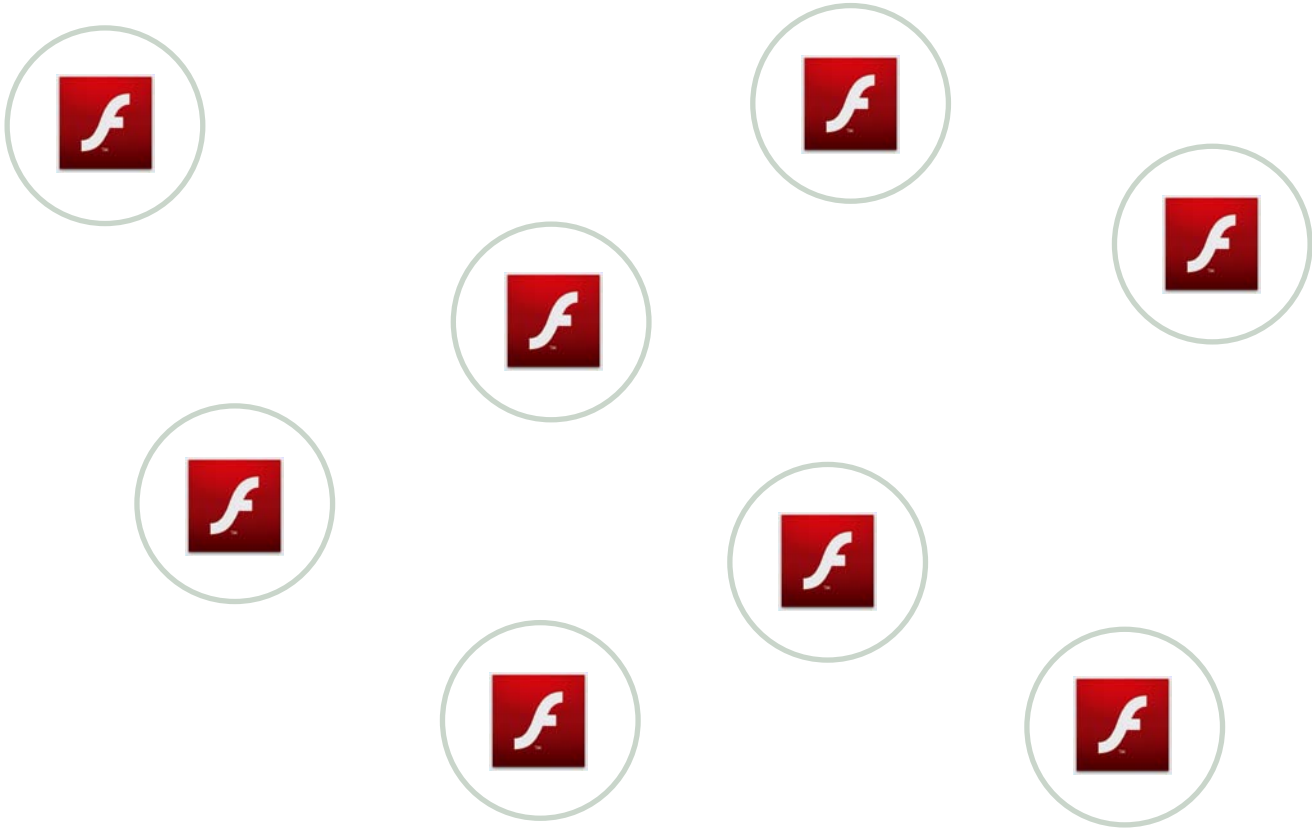
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



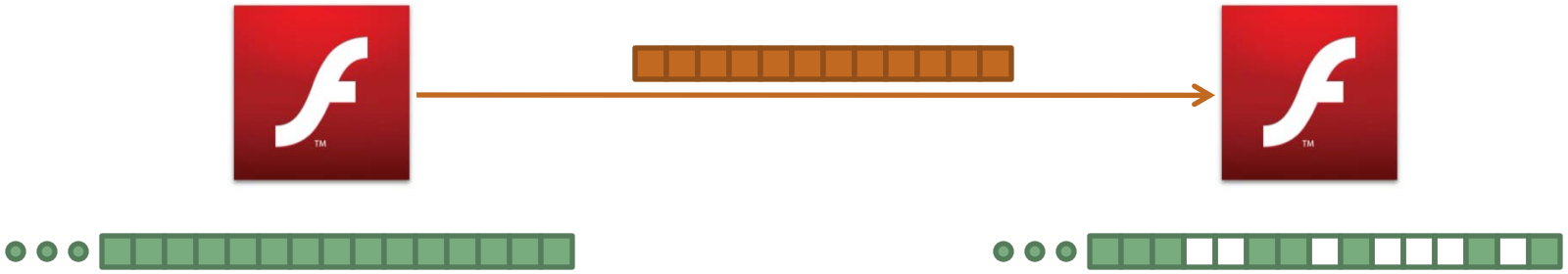
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Pull



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

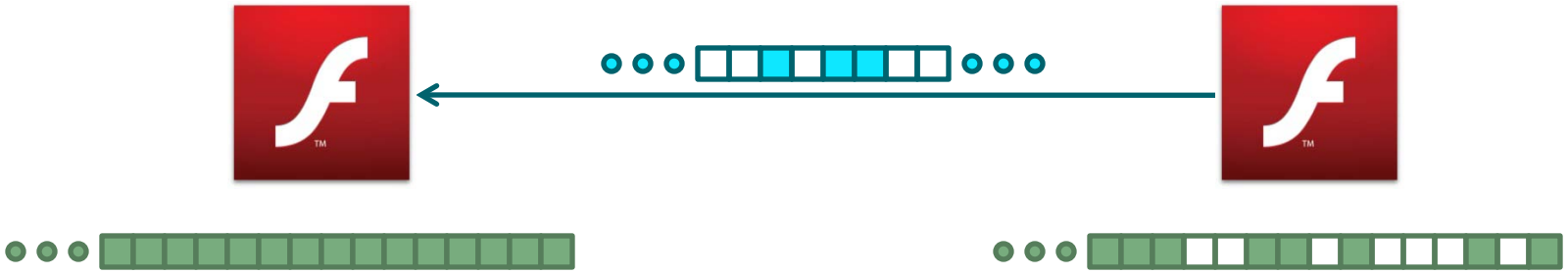
Multicast – Push



- Send map of what blocks you have to one (or all) neighbor(s)
  - Neighbor notices that data is flowing

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



- Neighbor picks candidate sources for each sequence number slice
- Neighbor sends mask describing which sequence numbers to push
  - Modulo size of mask (in above example every 3rd, 5th, and 6th)
  - Periodically tests other neighbors to check for lower latency
  - For each sequence number slice, keeps just quickest source

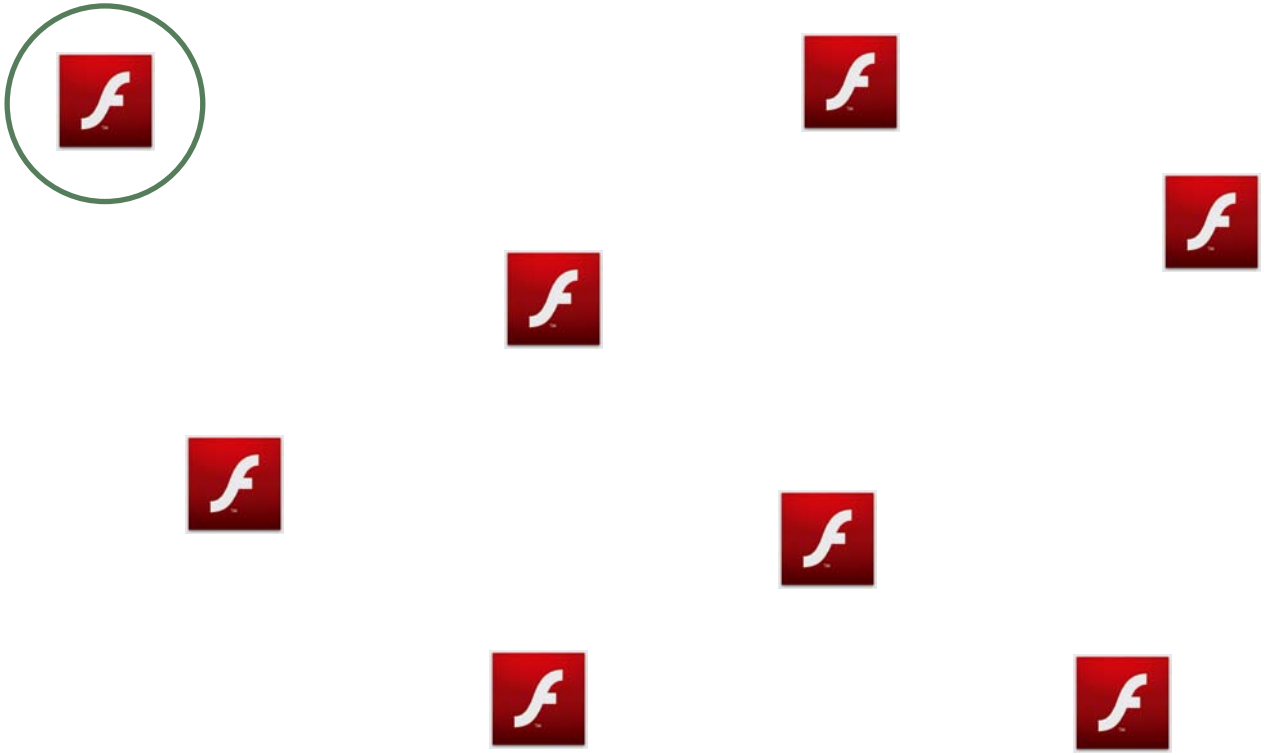
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push

- Data is pushed immediately as it arrives for requested slices
- For each slice, there is a limit to the number of push clients served

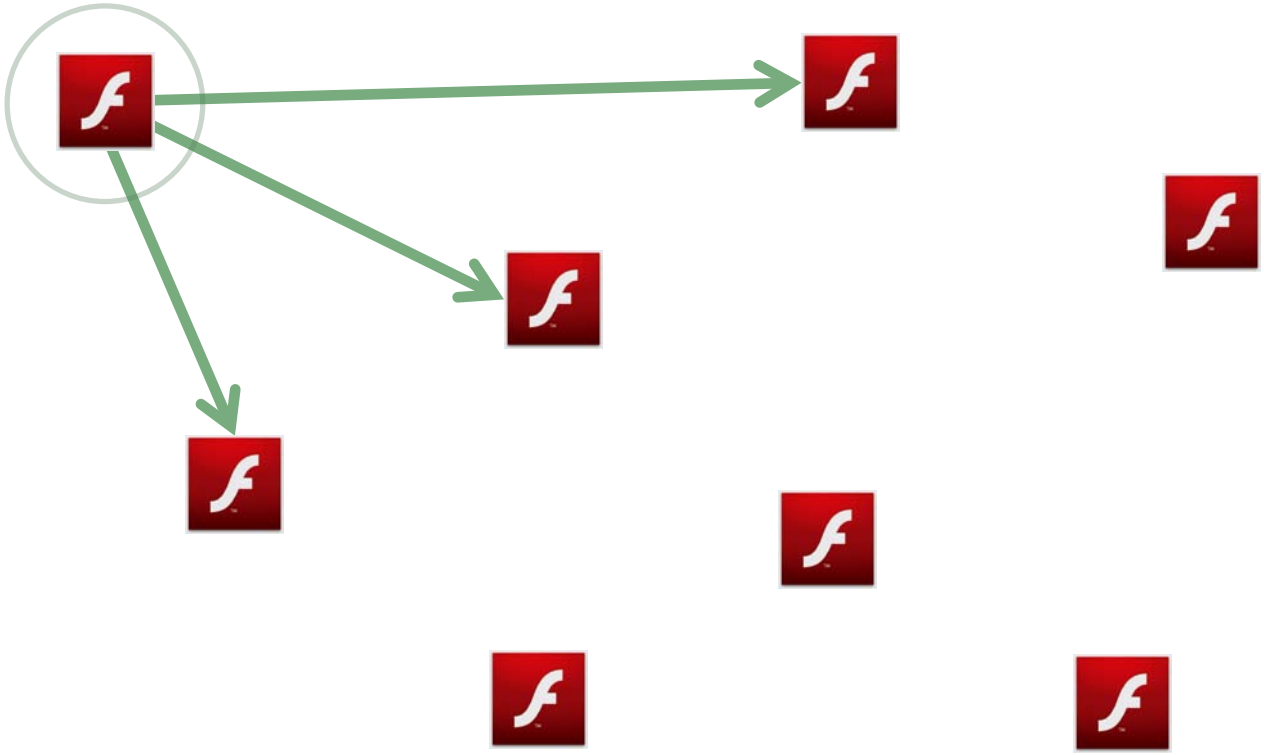
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



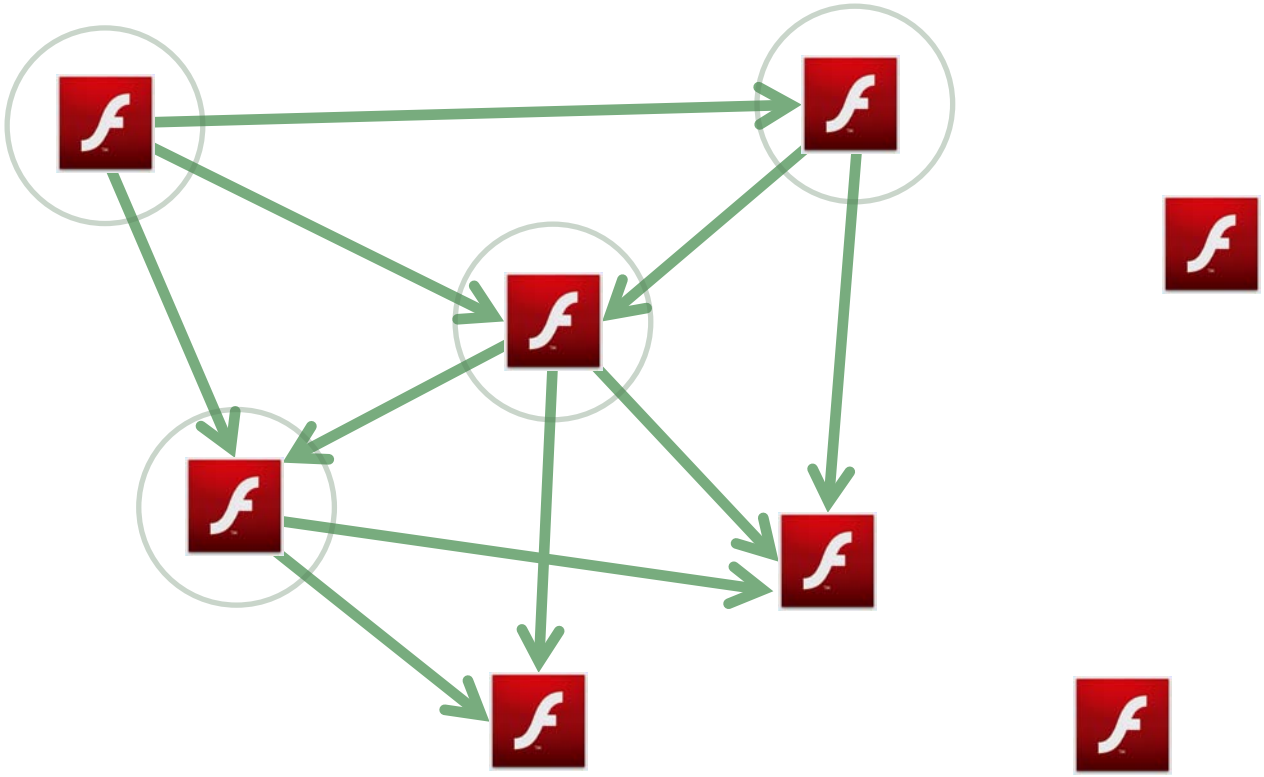
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



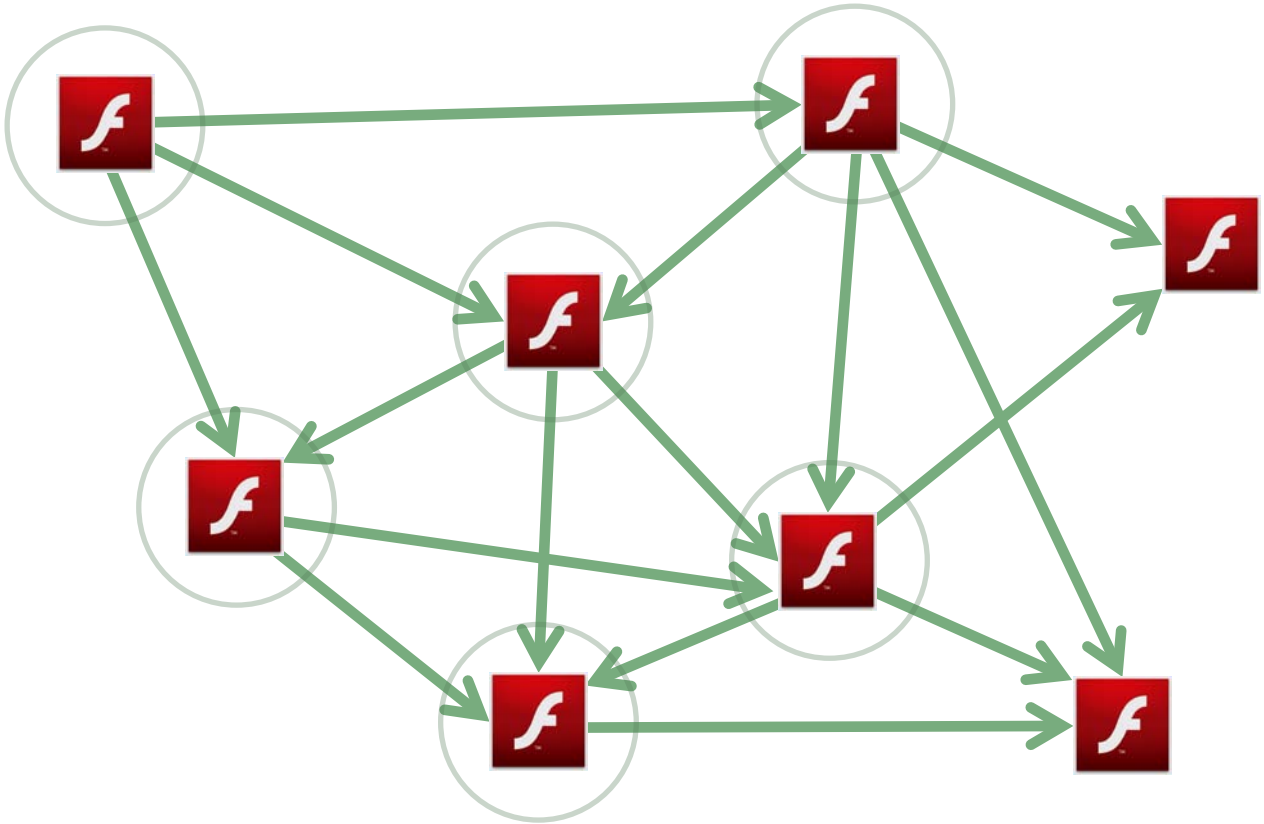
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



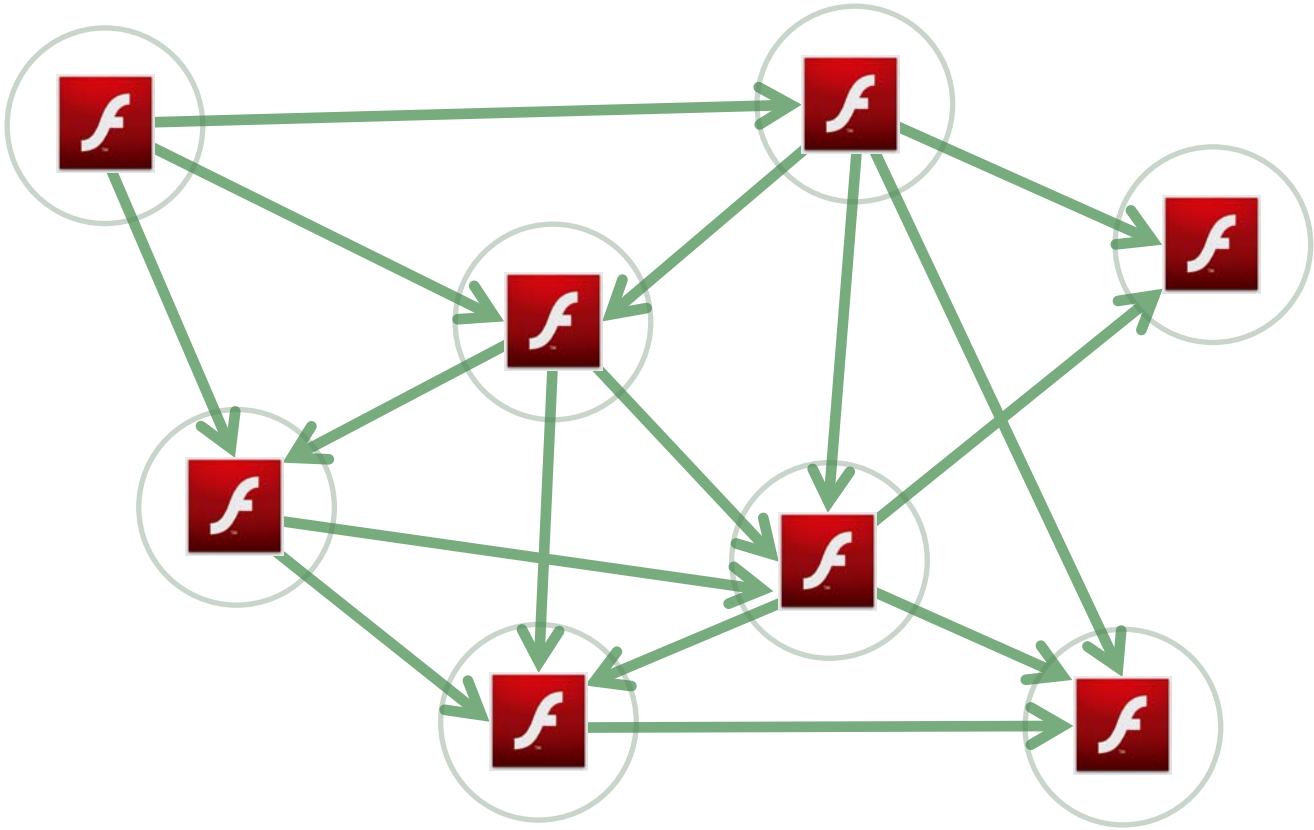
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



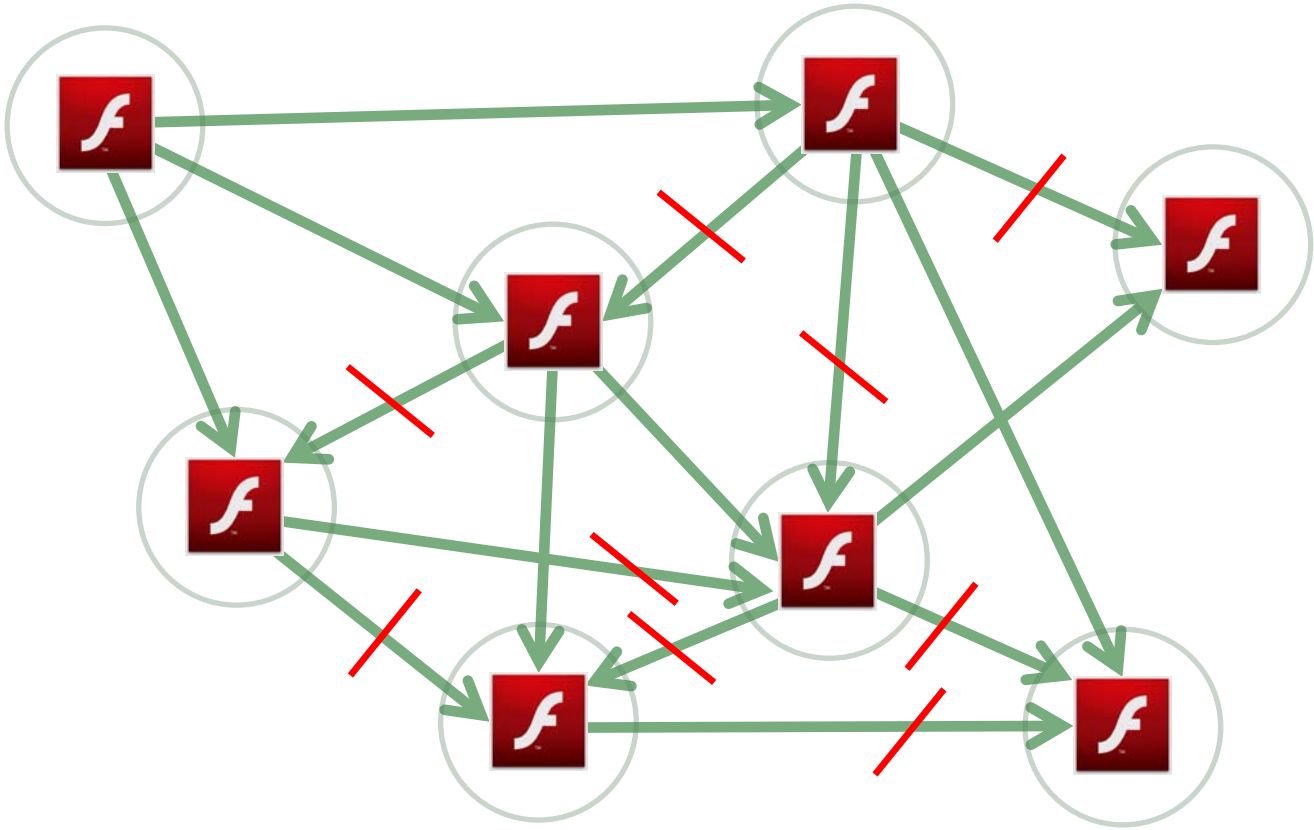
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



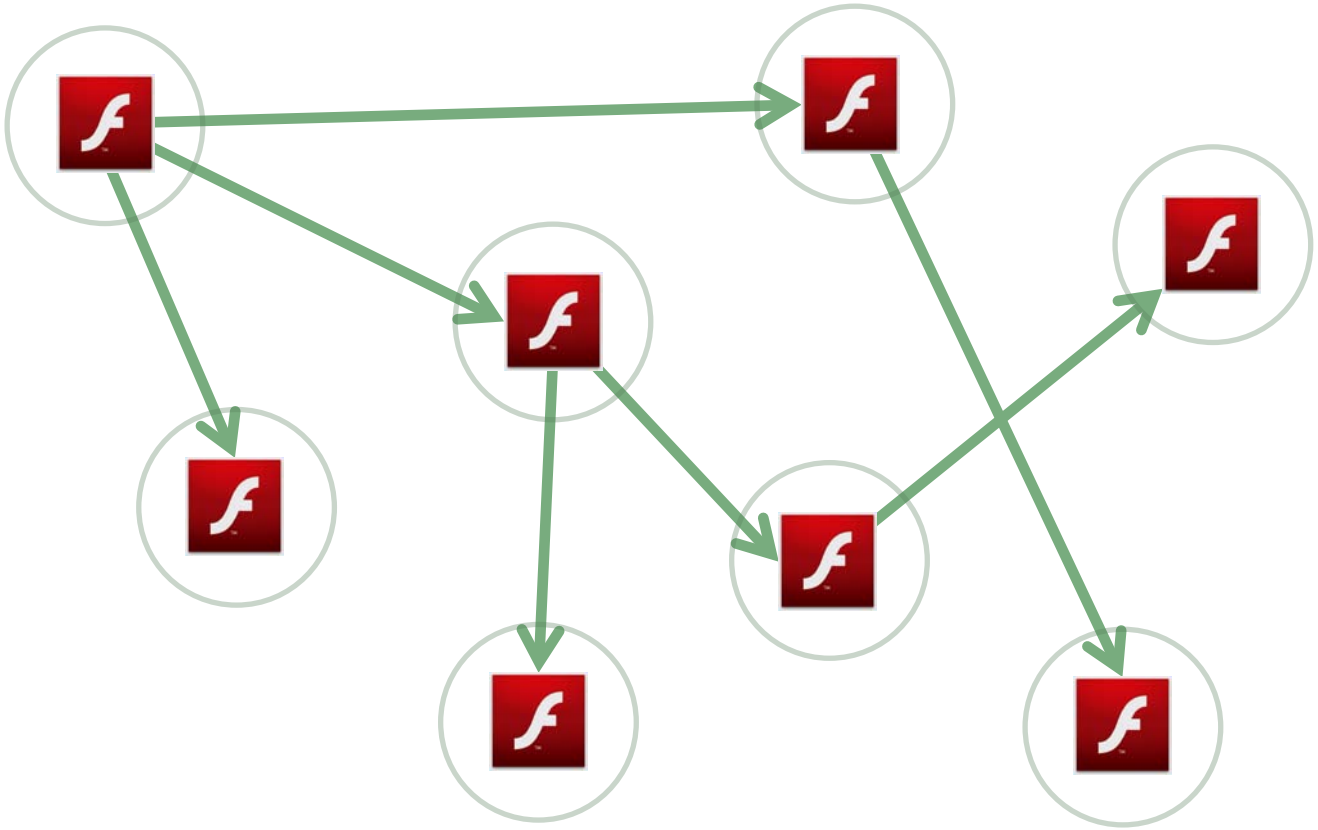
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



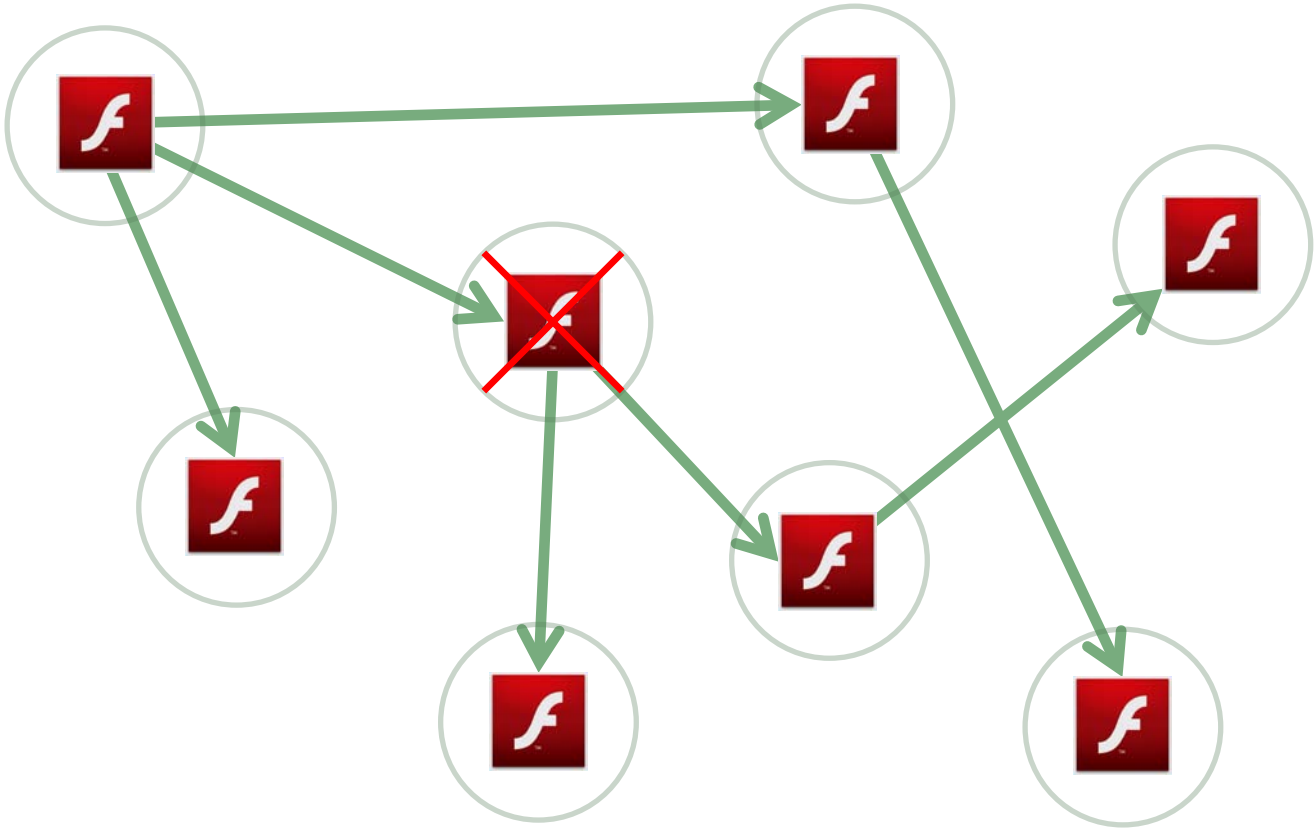
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push



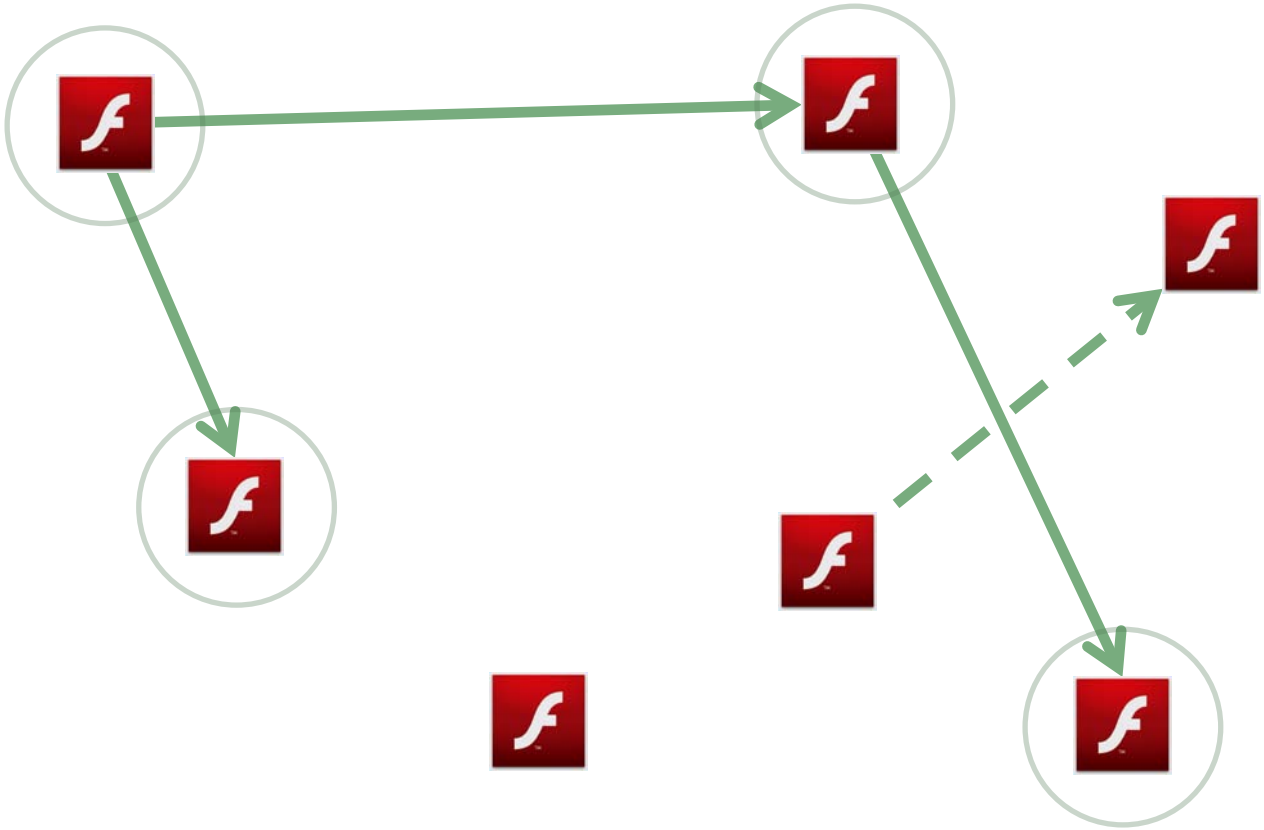
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



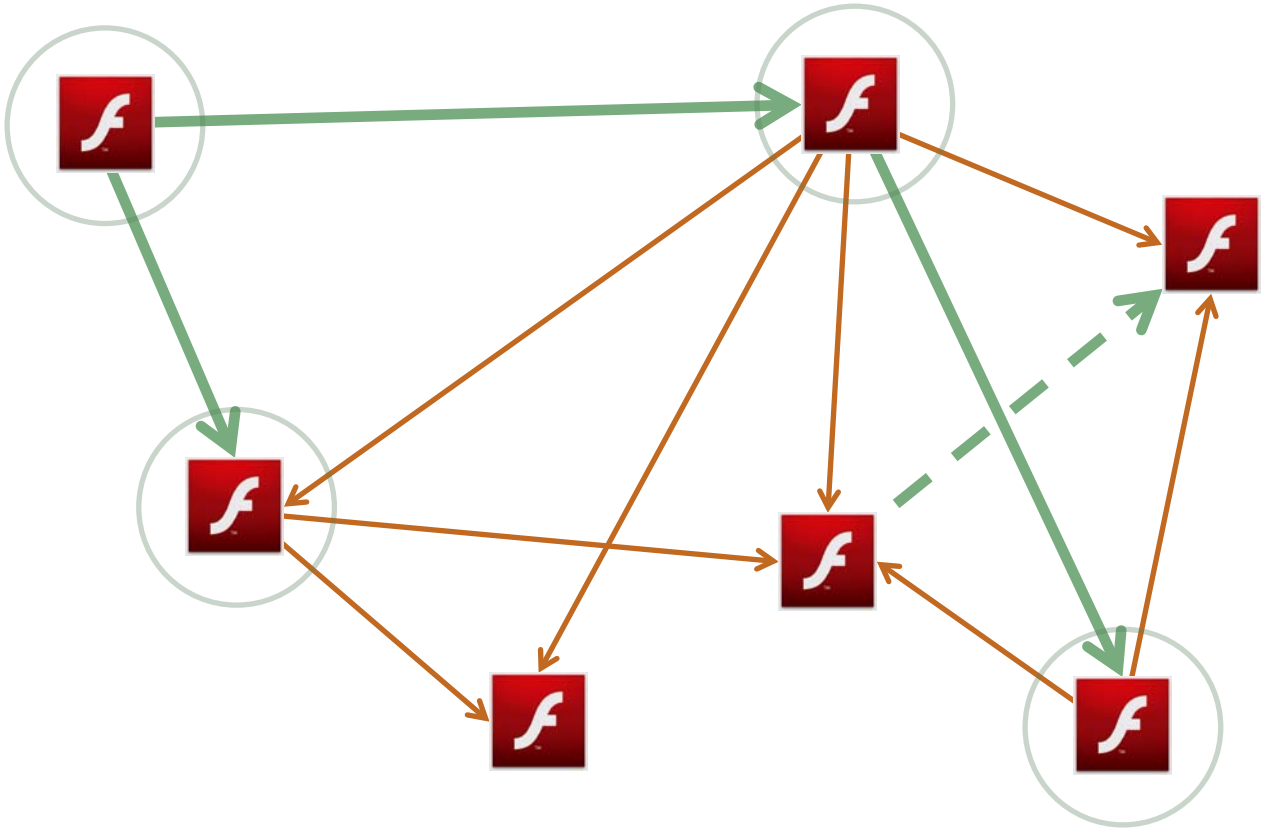
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



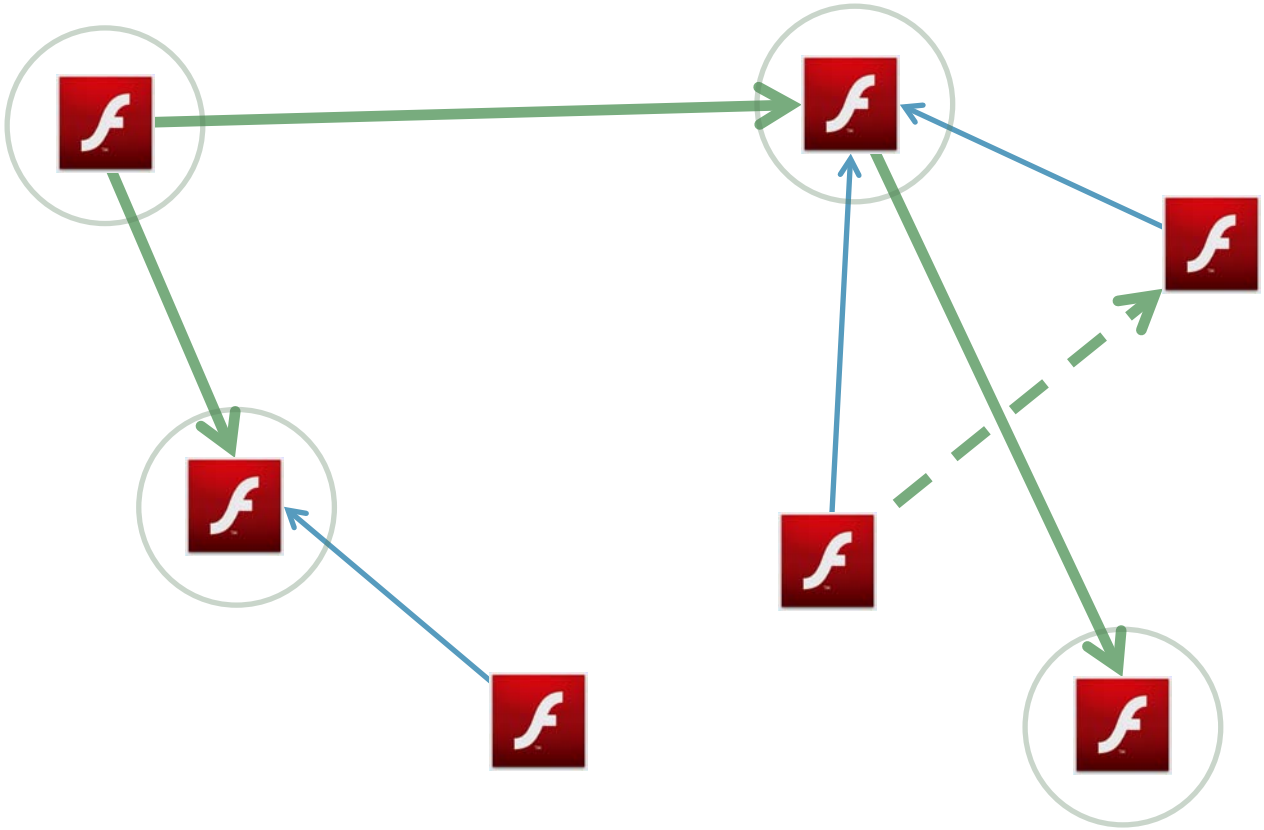
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



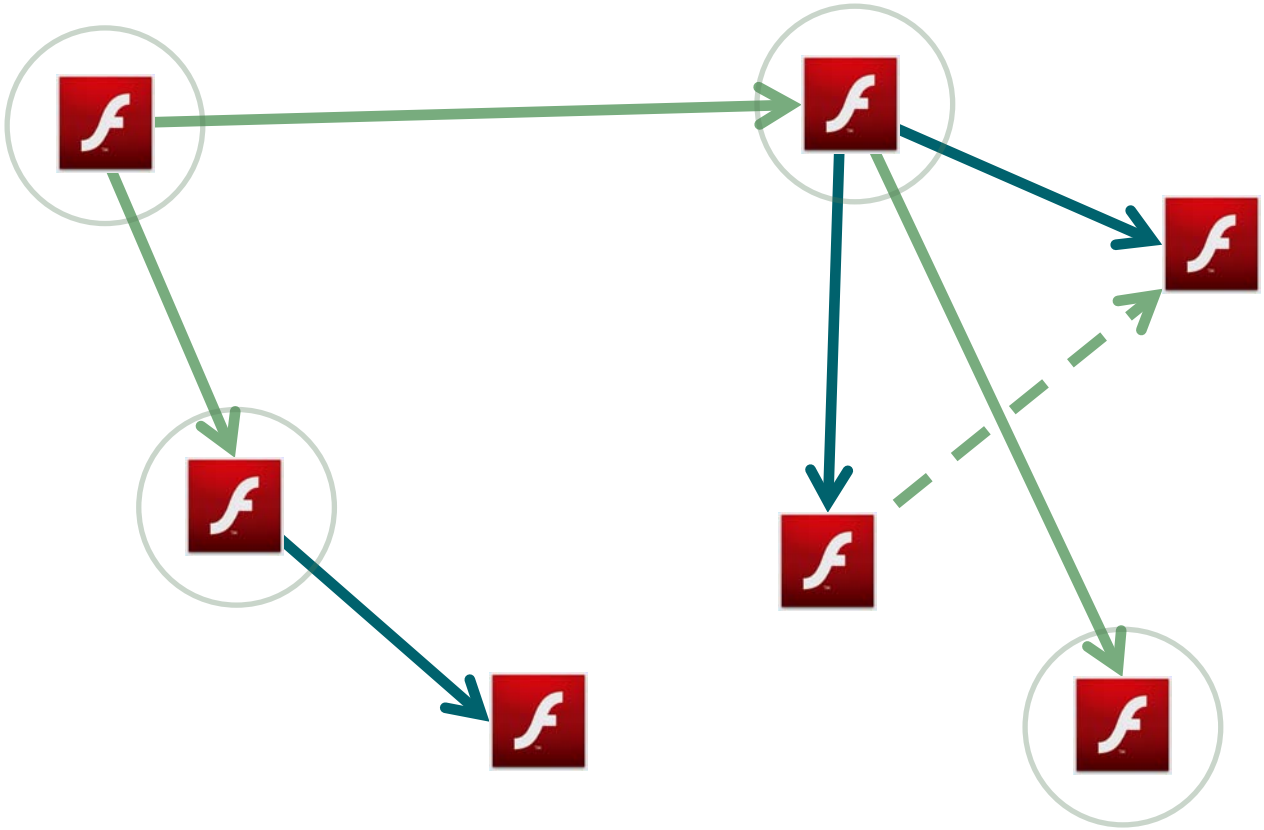
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



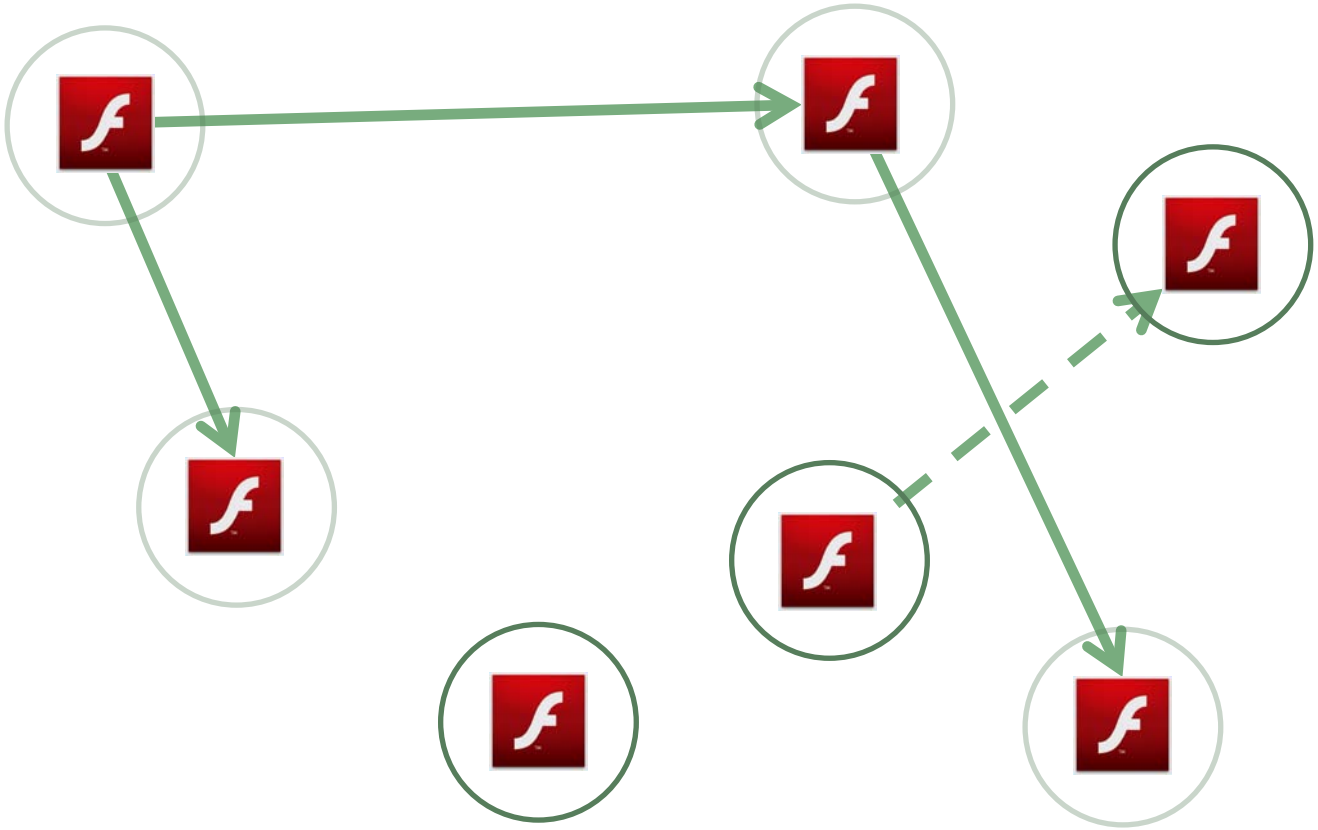
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



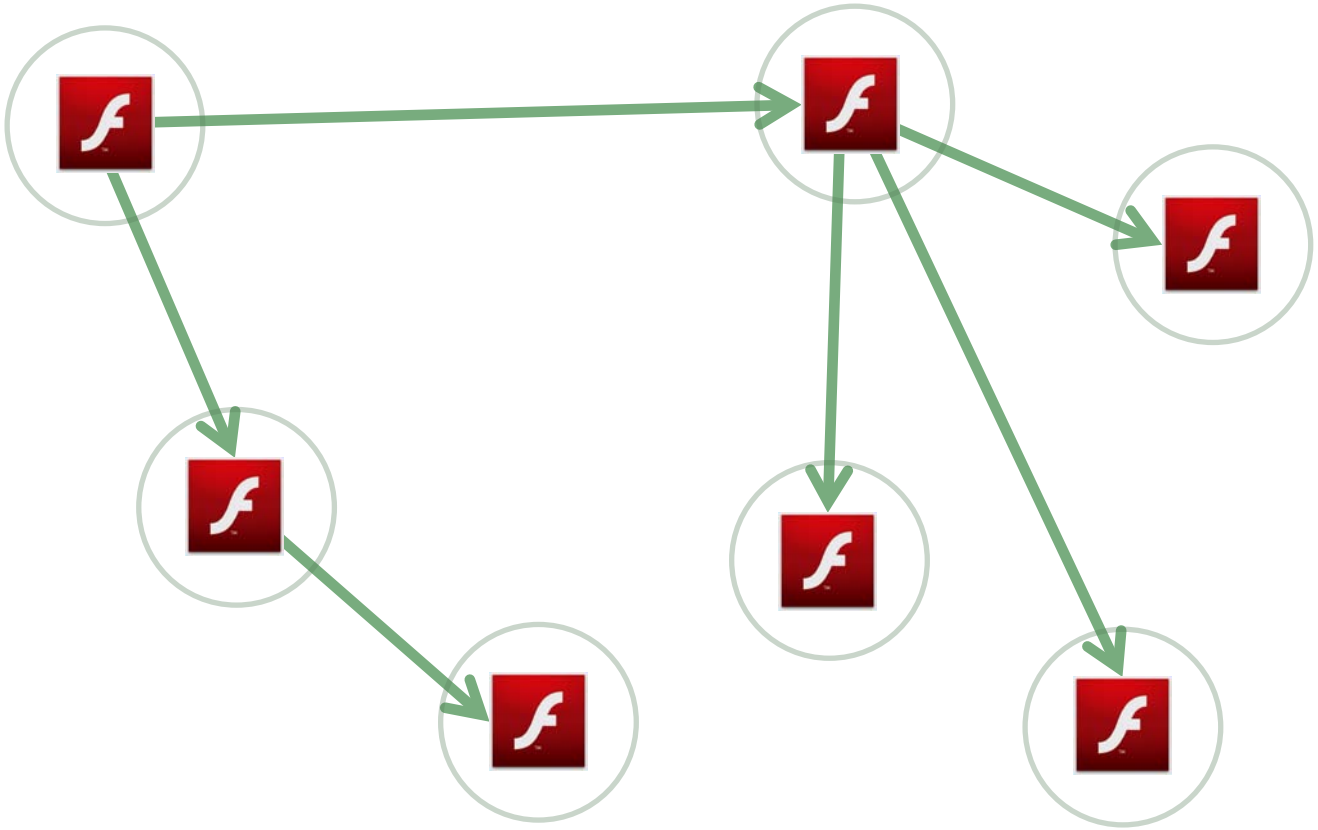
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



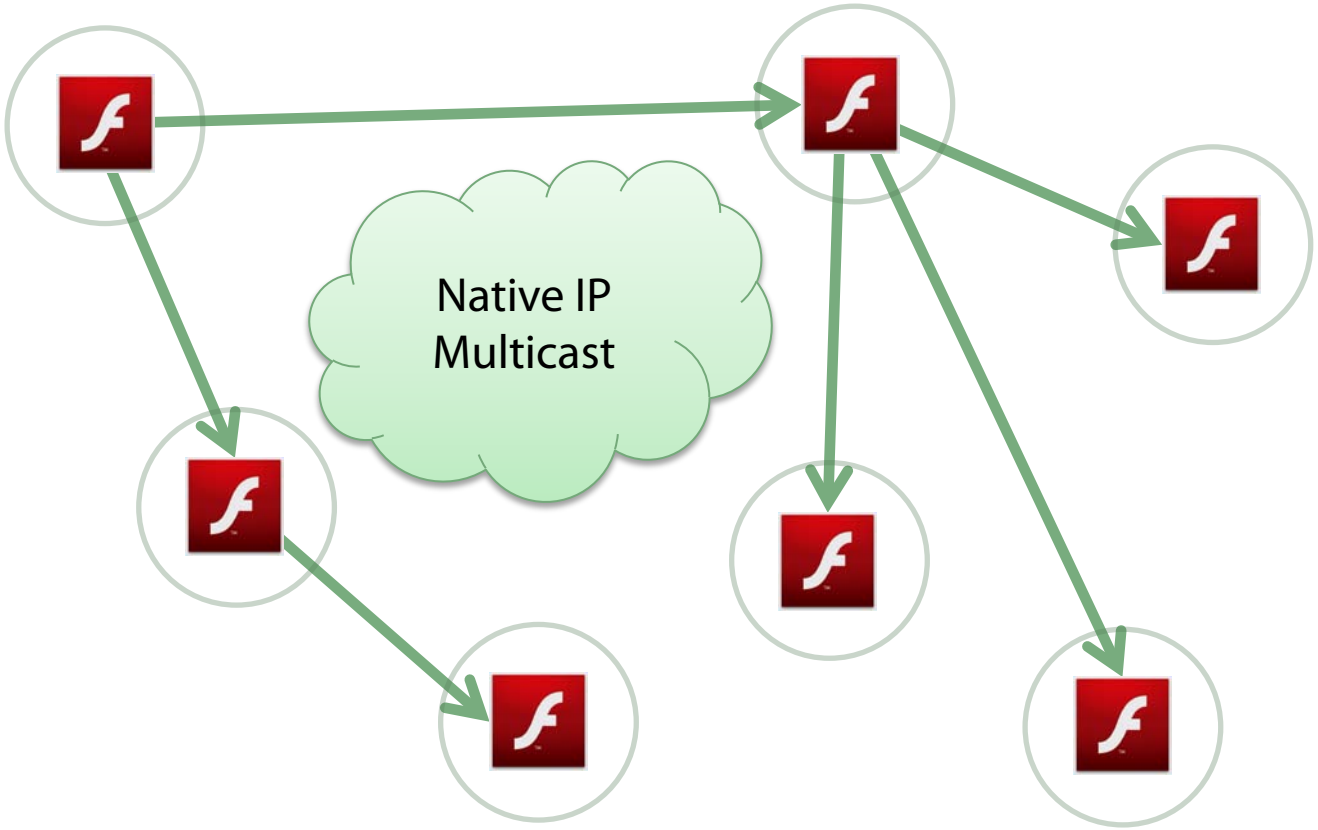
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Push-Pull



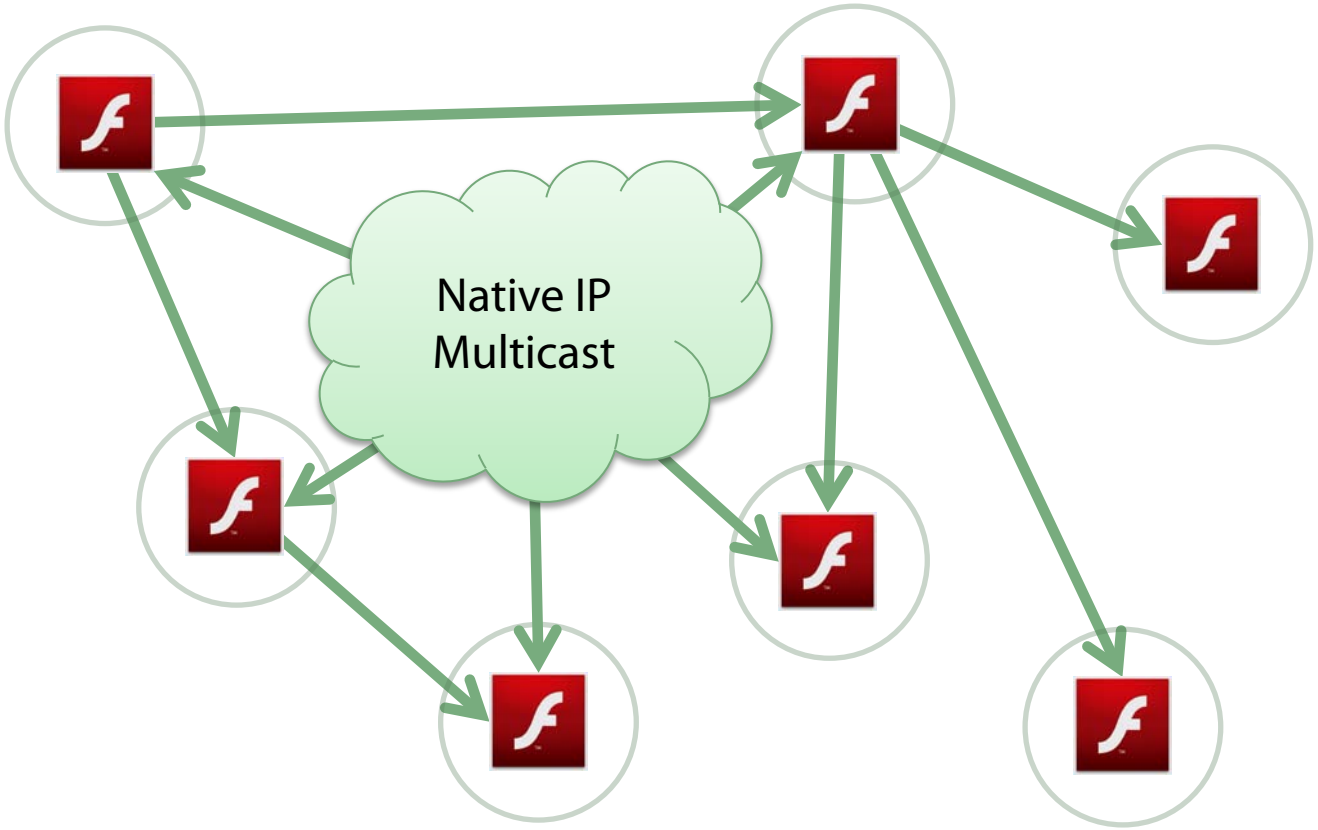
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Fusion



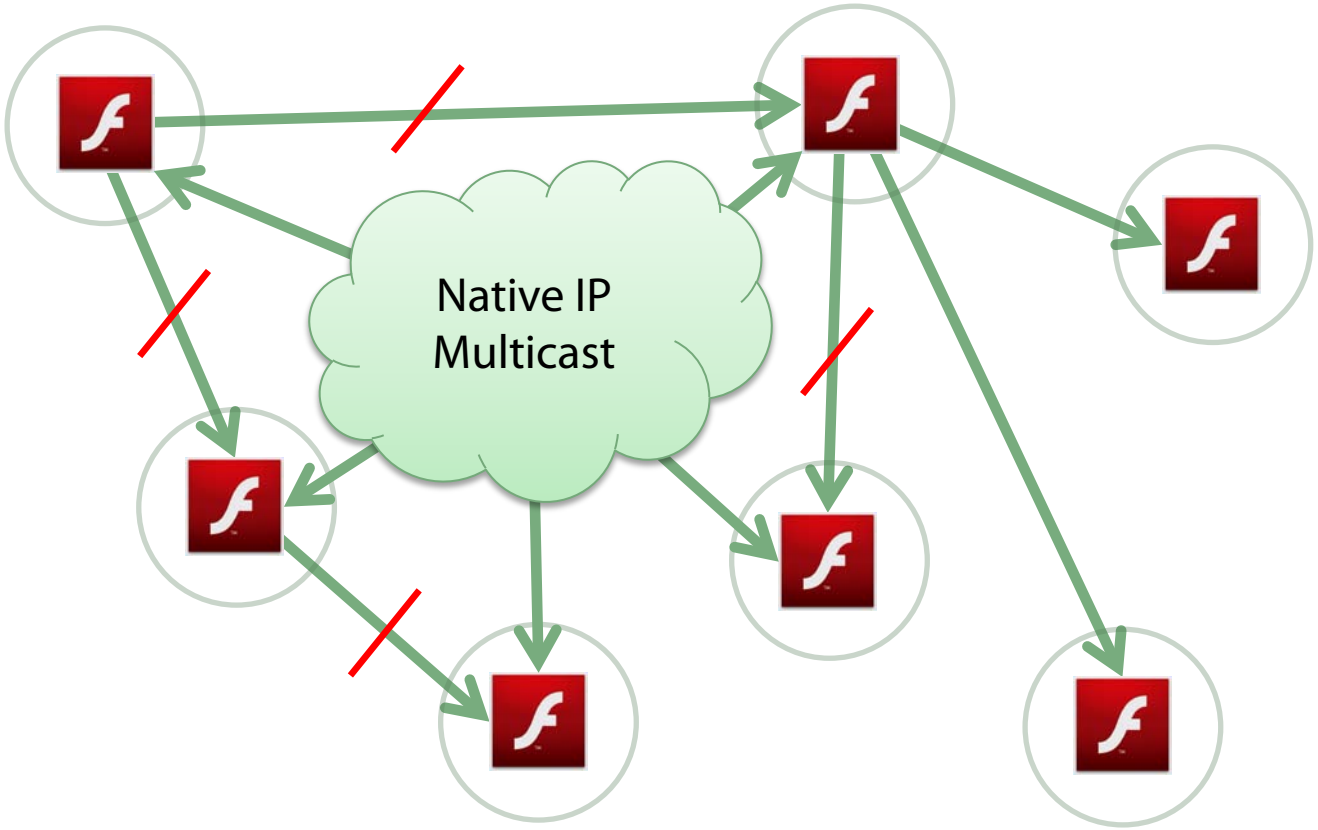
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Fusion



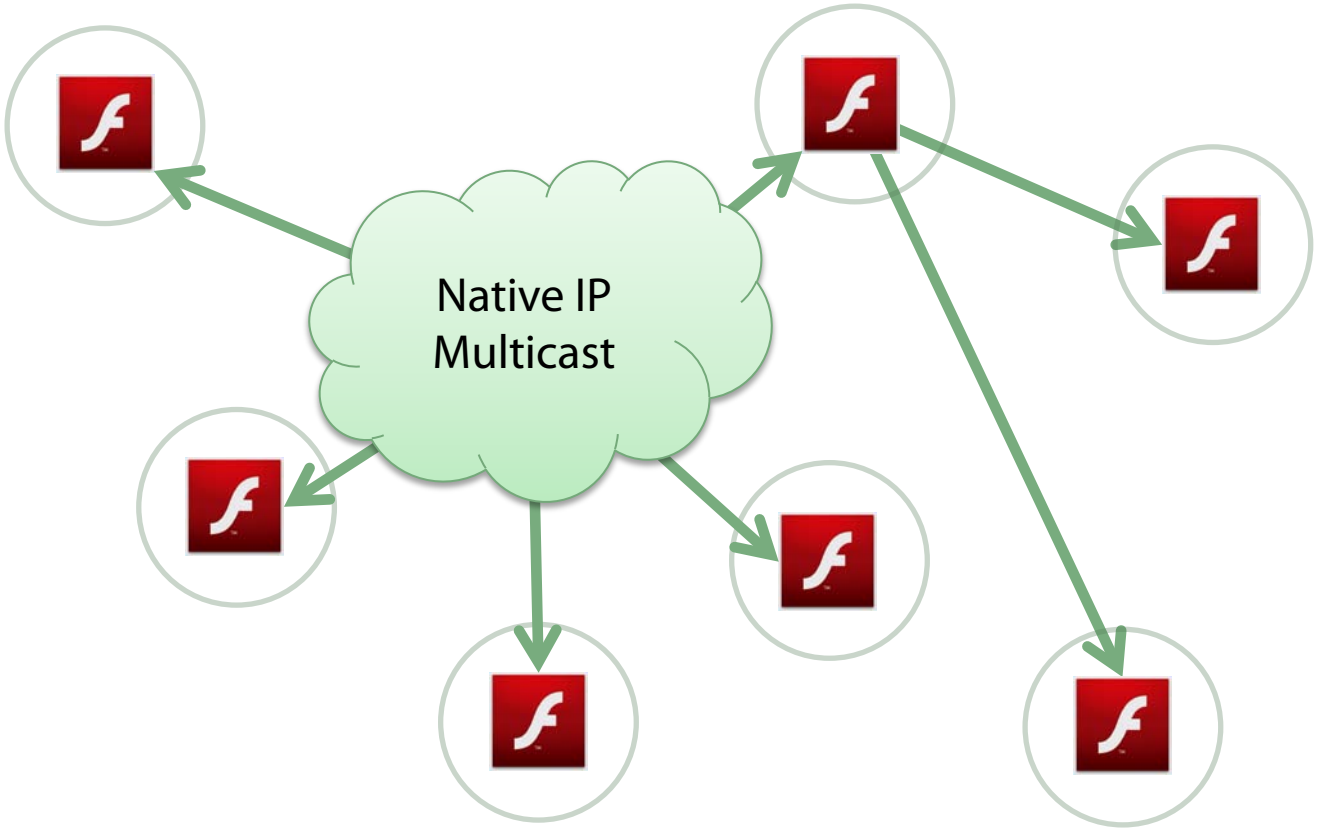
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Fusion



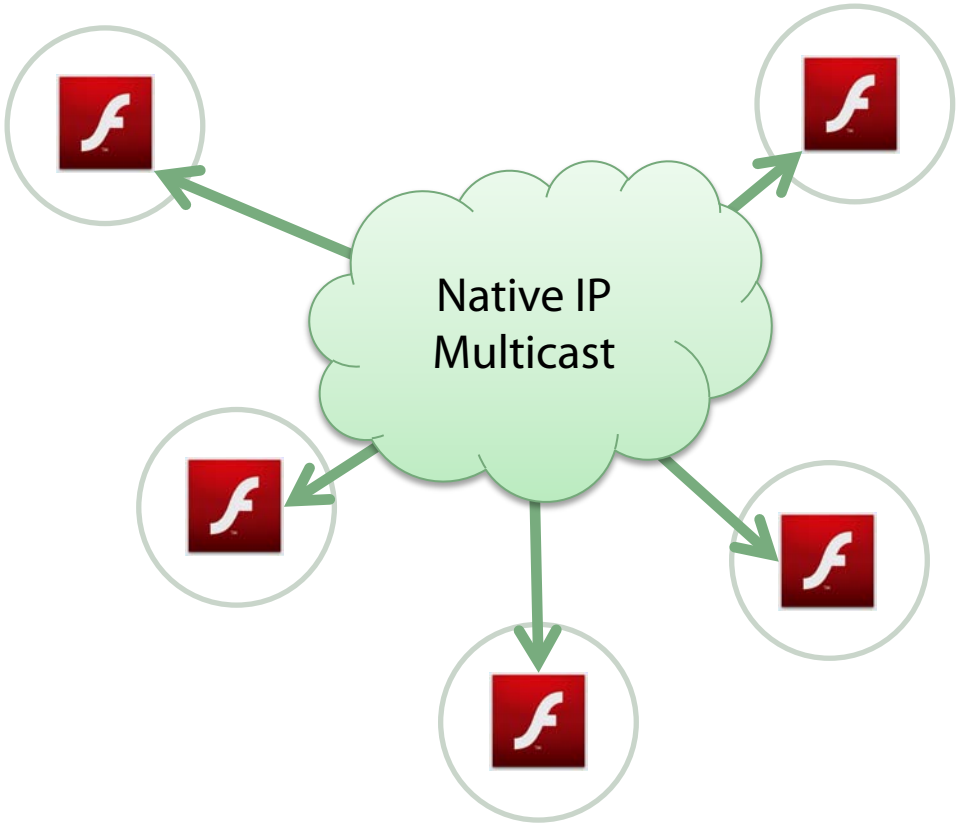
Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – Fusion



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast – IP-only



Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast

- Uses NetStream API
  - Construct with Groupspec instead of PeerID
- Low latency
  - Push/pull
    - Pull
      - Ensure nodes get the data
        - Initial source
        - Back-fill of loss (Fusion or Push)
        - Provides data when spanning tree temporarily wrong
    - Push
      - Reduce latency
        - Multiple spanning trees
- >1 Sender/group
- >1 Stream/sender (streams have names)

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast API

- Properties
  - `multicastPushNeighborLimit`
  - `multicastWindowDuration`
  - `multicastRelayMarginDuration`
  - `multicastAvailabilityUpdatePeriod`
  - `multicastFetchPeriod`
  - `multicastAvailabilitySendToAll`
  - `multicastInfo`
- Events
  - `NetStream.MulticastStream.Reset`
  - `NetGroup.MulticastStream.PublishNotify`
  - `NetGroup.MulticastStream.UnpublishNotify`

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast QOS

- sendDataBytesPerSecond
- sendControlBytesPerSecond
- receiveDataBytesPerSecond
- receiveControlBytesPerSecond
- bytesPushedToPeers
- fragmentsPushedToPeers
- bytesRequestedByPeers
- fragmentsRequestedByPeers
- bytesPushedFromPeers
- fragmentsPushedFromPeers
- bytesRequestedFromPeers
- fragmentsRequestedFromPeers
- sendControlBytesPerSecondToServer
- receiveDataBytesPerSecondFromServer
- bytesReceivedFromServer
- fragmentsReceivedFromServer
- receiveDataBytesPerSecondFromIPMulticast
- bytesReceivedFromIPMulticast
- fragmentsReceivedFromIPMulticast

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Native IP Multicast

- Provide Native IP Multicast address(es) in Groupspec
- LAN Discovery
  - Increase odds of finding nearby (low-latency) peers
  - Serverless ad-hoc RTMFP Group using new NetConnection.connect(“rtmfp:”);

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Multicast Fusion

- Fuse native IP multicast with peer-to-peer Multicast
  - Peers can use pull to fill in missing data
  - Peers with native IP can feed to peers without
- Single Groupspec (stream description) can be played in either environment
- Turn off peer-to-peer and use for Native IP-only
- (Future) FMS is publisher
  - Flash Player cannot **publish** Native IP Multicast
- Security of Native IP Multicast traffic (Fusion and LAN Discovery)
  - Protected (and separated from other traffic) with Groupspec-derived HMAC
  - Encrypted with Groupspec-derived AES key

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Server Control / Status Channel

- Enabled in Groupspec
  - `GroupSpecifier.serverChannelEnabled`
- Status monitoring
- Multicast
  - Server seed
  - Server fill-in
- Bootstrapping
  - Server can tell Flash Player about other peers to add
- New Stratus feature!
  - Group auto-bootstrapping

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Other New Things

- Partial Reliability Control at publishing peer
  - `NetStream.audioReliable`
  - `NetStream.videoReliable`
  - `NetStream.dataReliable`
- Peer-to-Peer Audio/Video Sample Access Control at publishing peer
  - `NetStream.audioSampleAccess`
  - `NetStream.videoSampleAccess`
- RTMFP NetConnection with no server
  - `NetConnection.connect("rtmfp:");`

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.

Q&A

Copyright 2009 Adobe Systems Incorporated. All rights reserved. Adobe confidential.


MAX
CONNECT. DISCOVER. INSPIRE.