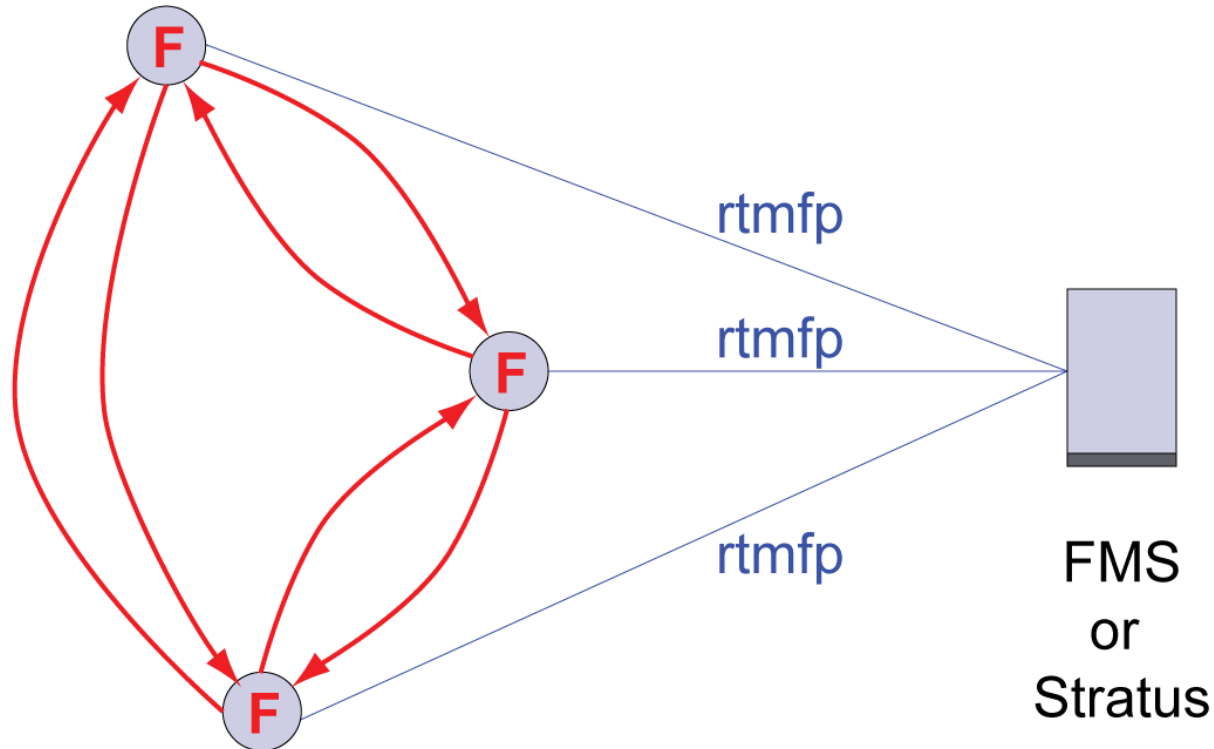


Player-to-Player Communications with RTMFP



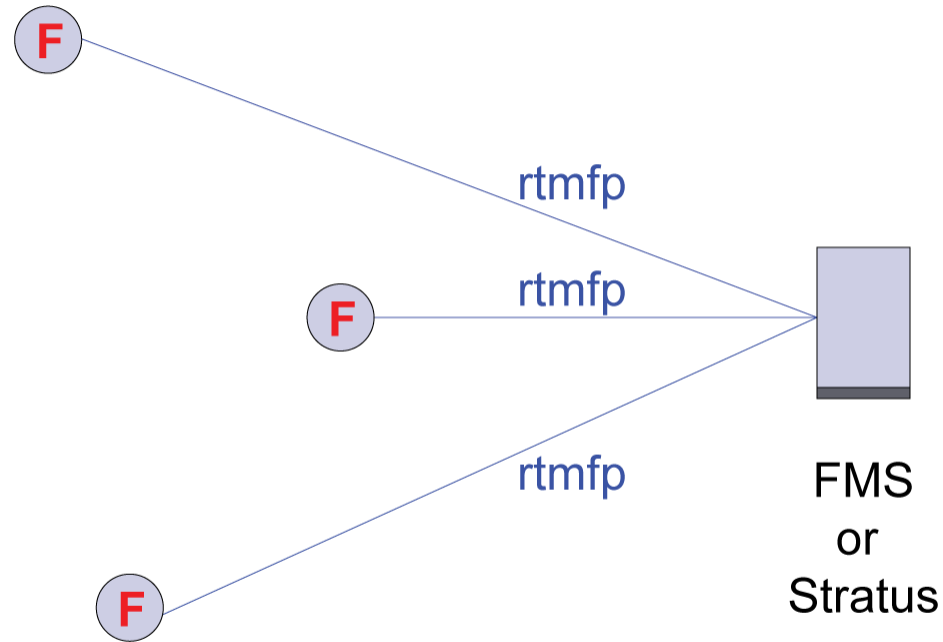
What do you need to develop/use RTMFP?

- Flash Player 10
- Flex 3.2 SDK, Flex Builder 3.0.2 or
- Flash CS4
- Stratus – Adobe’s hosted rendezvous service or
- Adobe Flash Collaboration Service (Cocomo) or
- Future release of FMS

Real Time Media Flow Protocol

- Send Audio, Video, and Data directly from Flash Player to Flash Player
- Each player must be connected to Adobe's Stratus service or a future release of FMS
- Not designed to relay audio or video or to act as a Peer-to-Peer network

Player-to-Player Communications with RTMFP



Connect to Stratus or Connect to an FMS Application Instance

Player-to-Player Communications with RTMFP

```
private function init():void{
    // Create a New NetConnection as usual:
    nc = new NetConnection();
    // Setup the usual event listeners:
    nc.addEventListener(NetStatusEvent.NET_STATUS, netStatus);
    nc.addEventListener(AsyncErrorEvent.ASYNC_ERROR, asyncError);
    nc.addEventListener(IOErrorEvent.IO_ERROR, ioError);
    nc.addEventListener(SecurityErrorEvent.SECURITY_ERROR, securityError);
}

private function connect():void{
    // Connect to FMS using rtmfp instead of rtmp:
    nc.connect("rtmfp://localhost/helloRTMFP/roomA", userInfo);
}
```

Player-to-Player Communications with RTMFP

```
private function init():void{
    // Create a New NetConnection as usual:
    nc = new NetConnection();
    // Setup the usual event listeners:
    nc.addEventListener(NetStatusEvent.NET_STATUS, netStatus);
    nc.addEventListener(AsyncErrorEvent.ASYNC_ERROR, asyncError);
    nc.addEventListener(IOErrorEvent.IO_ERROR, ioError);
    nc.addEventListener(SecurityErrorEvent.SECURITY_ERROR, securityError);
}

private function connect():void{
    // Connect to Stratus using rtmfp:
    nc.connect("rtmfp://stratus.adobe.com/" + devKey);
}
```

Player-to-Player Communications with RTMFP

```
private function netStatus(event:NetStatusEvent):void{
    if (event.info.code == "NetConnection.Connect.Success"){
        publishStream();
    }
}
```

Player-to-Player Communications with RTMFP

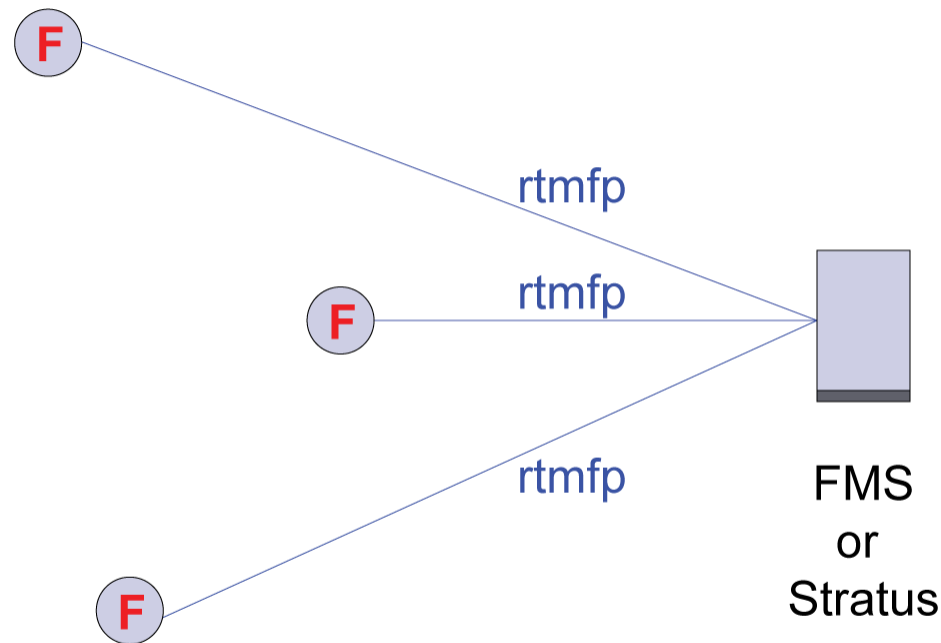
```
private function publishStream():void{
    ns = new NetStream(nc, NetStream.DIRECT_CONNECTIONS);
    ns.addEventListener(NetStatusEvent.NET_STATUS, netStatus);

    camera = Camera.getCamera();
    if (camera){
        ns.attachCamera(camera);
        video.attachCamera(camera);
    }

    microphone = Microphone.getMicrophone();
    if (microphone){
        microphone.codec = SoundCodec.SPEEX;
        ns.attachAudio(microphone);
    }

    ns.publish("streamName");
}
```

Player-to-Player Communications with RTMFP



Nothing has changed – All three clients may have “published” a direct stream but there are no subscribers.

Player-to-Player Communications with RTMFP

```
private function playStream():void{
    ns = new NetStream(nc, peerID);
    ns.addEventListener(NetStatusEvent.NET_STATUS, netStatus);
    video.attachNetStream(ns);
    ns.play("streamName");
}
```

Player-to-Player Communications with RTMFP

```
private function playStream():void{  
    ns = new NetStream(nc, peerID);  
    ns.addEventListener(NetStatusEvent.NET_STATUS, netStatus);  
    video.attachNetStream(ns);  
    ns.play("streamName");  
}
```

What's in peerID?

Player-to-Player Communications with RTMFP

```
private function playStream():void{  
    ns = new NetStream(nc, peerID);  
    ns.addEventListener(NetStatusEvent.NET_STATUS, netStatus);  
    video.attachNetStream(ns);  
    ns.play("streamName");  
}
```

What's in peerID?

c26b09e2755f3556b6e09c32ab674de1f1dw32693af3e473221c0d7dfc5218f08

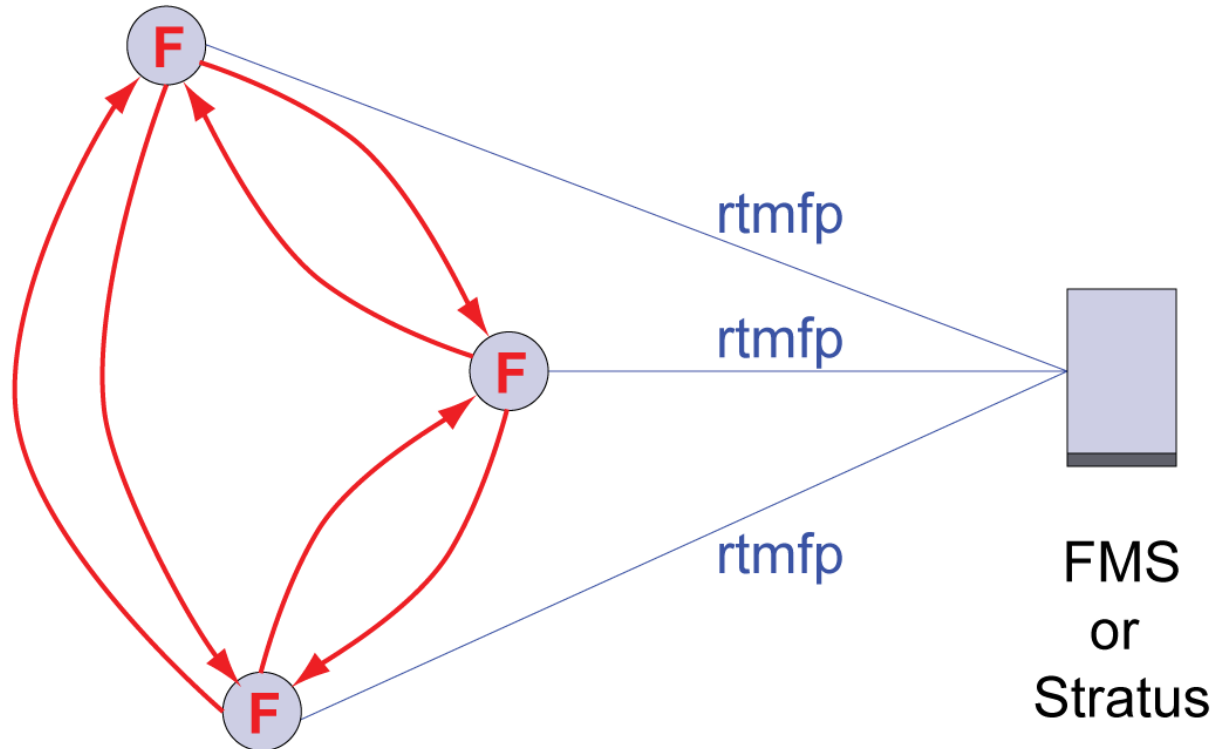
Player-to-Player Communications with RTMFP

```
private function playStream():void{
    ns = new NetStream(nc, peerID);
    ns.addEventListener(NetStatusEvent.NET_STATUS, netStatus);
    video.attachNetStream(ns);
    ns.play("streamName");
}
```

What's is it?

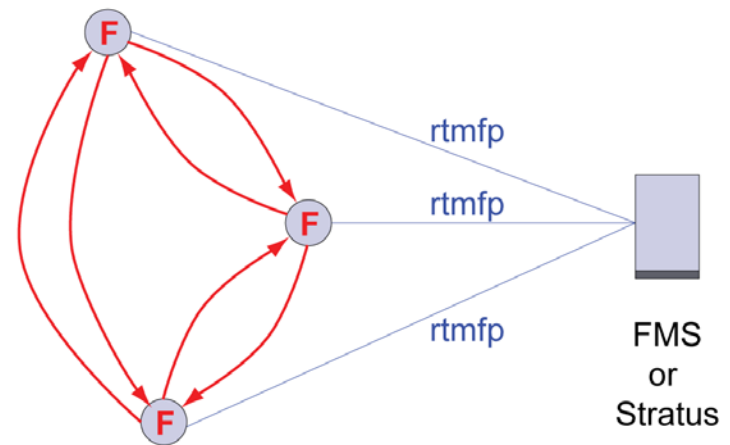
- Unique ID of a Flash Player connected to stratus or FMS
- Required to play a stream from that player
- 256 bit number
- Will never be reused
- Provided to the Player by Stratus or FMS
- Derived from strongly random values

Player-to-Player Communications with RTMFP



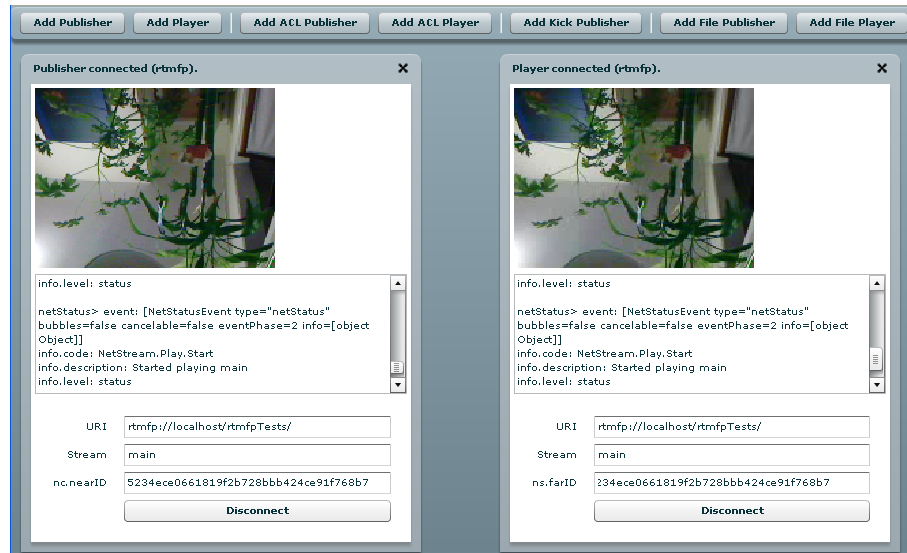
Player-to-Player Communications with RTMFP

Why is this good?



- Lower Latency UDP Streams
- Audio sent with highest priority
- Speex codec means dropped audio packets are OK
- Client-to-client latency vs. client-to-server-to-client
- Extremely Low Server Bandwidth Costs!!
- Designed to be secure

Player-to-Player Communications with RTMFP



Simple RTMFP Test Application

Player-to-Player Communications with RTMFP

Publish

1. Request a Connection
2. Wait for
NetConnection.Connect.Success
3. Create a Direct NetStream
4. Attach Microphone and Camera
to the stream
5. Publish the stream with a name

Play

1. Request a Connection
2. Wait for
NetConnection.Connect.Success
3. Create a NetStream with a
Remote Client's Peer ID
4. Attach stream to the Video object
5. Play the stream by name

How does the Subscriber Get a Publisher's Peer ID?

- In Publisher: **netConnection.nearID**
- In FMS, put them in a Remote Shared Object
- XMPP system to exchange Peer IDs
- Web Service
- Others...

Using FMIS to Share Peer IDs

- Today – FMIS + Stratus Application
- Future – FMIS

FMIS + Stratus

Many ways to approach this.

Here's a simple one:

1. Connect to Stratus
2. Get the `rtmfpConnection.nearID` – that's the client's peer id that other clients need.
3. Pass the `rtmfpConnection.nearID` into the `rtmpConnection.connect()` call so FMIS can put the peer ID in a shared object.

Requires two separate `NetConnection` objects.

FMIS + Stratus (on the client)

```
rtmpConnection.connect(rtmpAddress, rtmfpConnection.nearID);
```

FMIS + Stratus (on FMIS)

```
clients_so = SharedObject.get("public/clients", false);
```

```
application.onConnect = function(client, peerID){  
    var clientInfo = {  
        id: client.id,  
        protocol: client.protocol,  
        peerID: peerID  
    }  
    clients_so.setProperty(client.id, clientInfo);  
    application.acceptConnection(client);  
    client.call("init", null, clientInfo);  
}
```

```
application.onDisconnect = function (client){  
    clients_so.setProperty(client.id, null);  
}
```

Player-to-Player Communications with RTMFP

public/clients Remote Shared Object

Property Name	Property Value
BCAoYlcB	id: "BCAoYlcB" peerID: "4462cd6c614b9985921a514d35c356557b2e20abbcd2741c325f8959823c2471" protocol: "rtmfp"
CJAQYFNJ	id: "CJAQYFNJ" peerID: "92d93e09b2a0b194d8688980120627ed3093e593a0b97b98ff7a8618571ea94a" protocol: "rtmfp"
DIA4o9NJ	id: "DIA4o9NJ" peerID: "14f298edc010f608999dc6e839054c4fbc7cf05fa701cdd4d8ed134433bb2aab" protocol: "rtmfp"

Flash Latency Demo



rtmfp (stratus beta)



rtmp (Influxis)

Future Release of FMIS

1. Connect to FMIS
2. FMIS gets the peer id from `client.farID` and updates a shared object
3. Clients listen for changes in the shared object as clients connect and disconnect.

Future Release of FMIS

```
clients_so = SharedObject.get("public/clients");

application.onConnect = function(client, userInfo){
  clientInfo = {
    id: client.id,
    protocol: client.protocol, // in case we failed over to rtmp!
    peerID: client.farID
  }
  clients_so.setProperty(client.id, clientInfo);
  application.acceptConnection(client);
  client.call("onConnect", null, clientInfo);
}

application.onDisconnect = function(client){
  clients_so.setProperty(client.id, null);
}
```

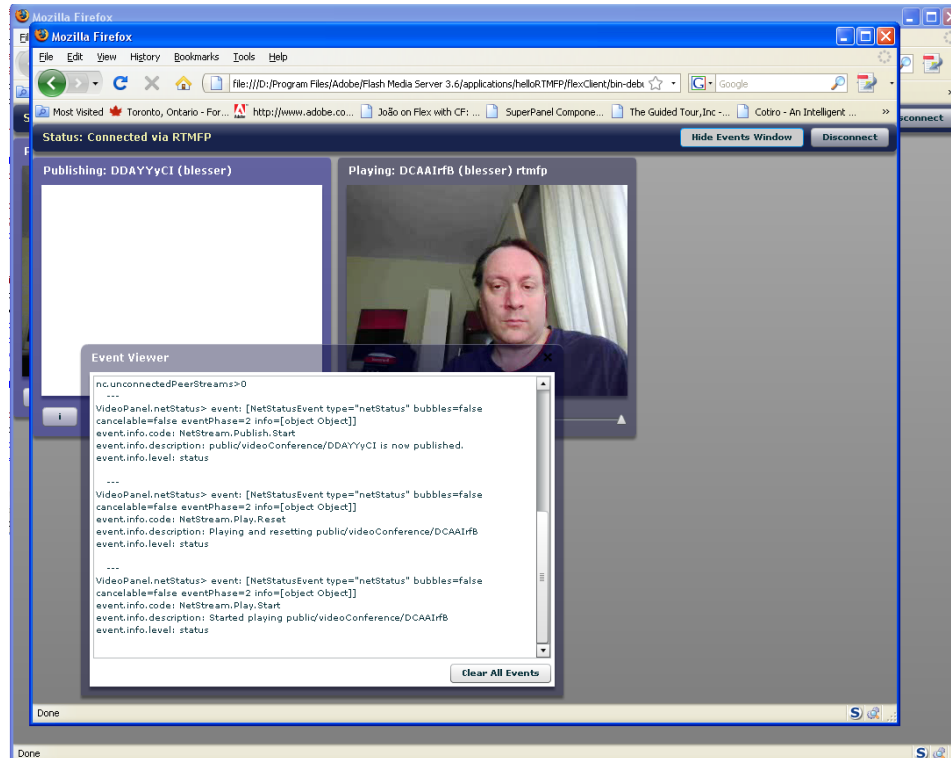
Player-to-Player Communications with RTMFP

public/clients Remote Shared Object

Property Name	Property Value
BCAoYlcB	id: "BCAoYlcB" peerID: "4462cd6c614b9985921a514d35c356557b2e20abbc2741c325f8959823c2471" protocol: "rtmfp"
CJAQYFNJ	id: "CJAQYFNJ" peerID: "92d93e09b2a0b194d8688980120627ed3093e593a0b97b98ff7a8618571ea94a" protocol: "rtmfp"
DIA4o9NJ	id: "DIA4o9NJ" peerID: "14f298edc010f608999dc6e839054c4fbc7cf05fa701cdd4d8ed134433bb2aab" protocol: "rtmfp"

Player-to-Player Communications with RTMFP

FMIS Simple Video Conference Demo



Access: Stratus vs. FMIS

Stratus (in beta)

- Today, any client can communicate with any client if they know the client's peer id and stream name.

FMIS (future release)

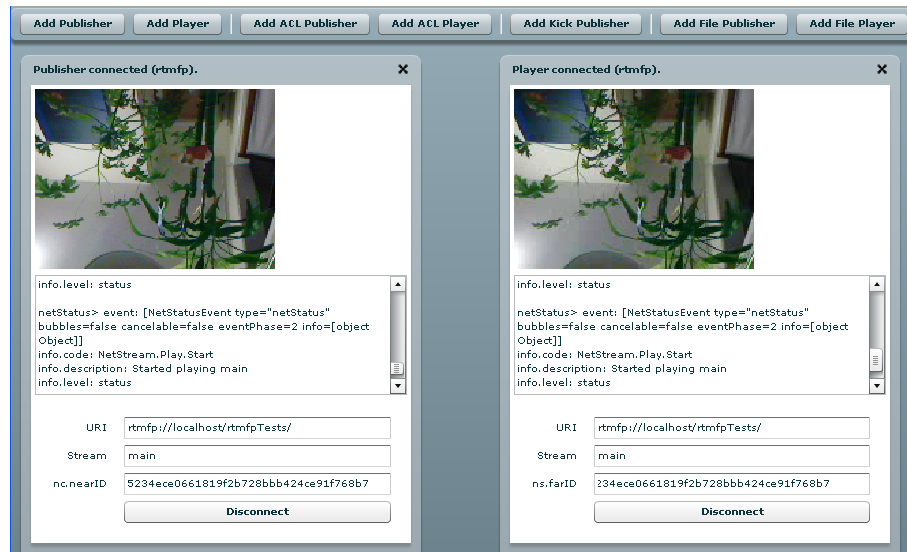
- Clients can only communicate when connected to the same application instance.

Controlling Access

- Only provide a Peer ID to an authorized user/client. (It's not feasible to guess a 256 bit Peer ID.)
- Use `netStream.client.onPeerConnect` to accept/reject subscribers.
- Get a peer stream from `netStream.peerStreams` and call `netStream.close()` on it.
- Get a peer stream from `netStream.peerStreams` and call `netStream.send()` on it. (Private message to one client.)

Player-to-Player Communications with RTMFP

Controlling Access/Demo



Sending Data

```
outgoingStream.send("receiveFile", file.name, file.data as ByteArray);
```

```
private var nsClient:Object;
```

```
var filePlayer:FilePlayer = this;
```

```
nsClient = {
```

```
    receiveFile: function(fileName:String, fileData:ByteArray):void{
```

```
        filePlayer.receiveFile(fileName, fileData);
```

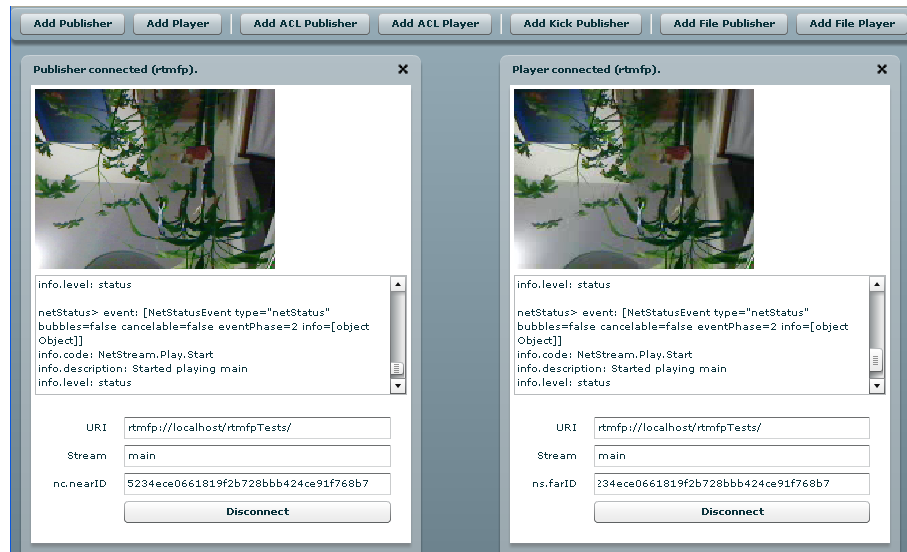
```
    }
```

```
}
```

```
incomingStream.client = nsClient;
```

Player-to-Player Communications with RTMFP

File Transfer/Demo



API Review: NetConnection

`netConnection.nearID`

`netConnection.farID`

`netConnection.protocol`

`netConnection.maxPeerConnections`

`netConnection.farNonce`

`netConnection.nearNonce`

`netConnection.unconnectedPeerStreams`

API Review: NetStream

```
new NetStream(nc, NetStream.DIRECT_CONNECTIONS);
```

```
new NetStream(nc, <peerID>);
```

```
netStream.peerStreams
```

```
netStream.client.onPeerConnect()
```

```
netStream.farID
```

```
netStream.farNonce
```

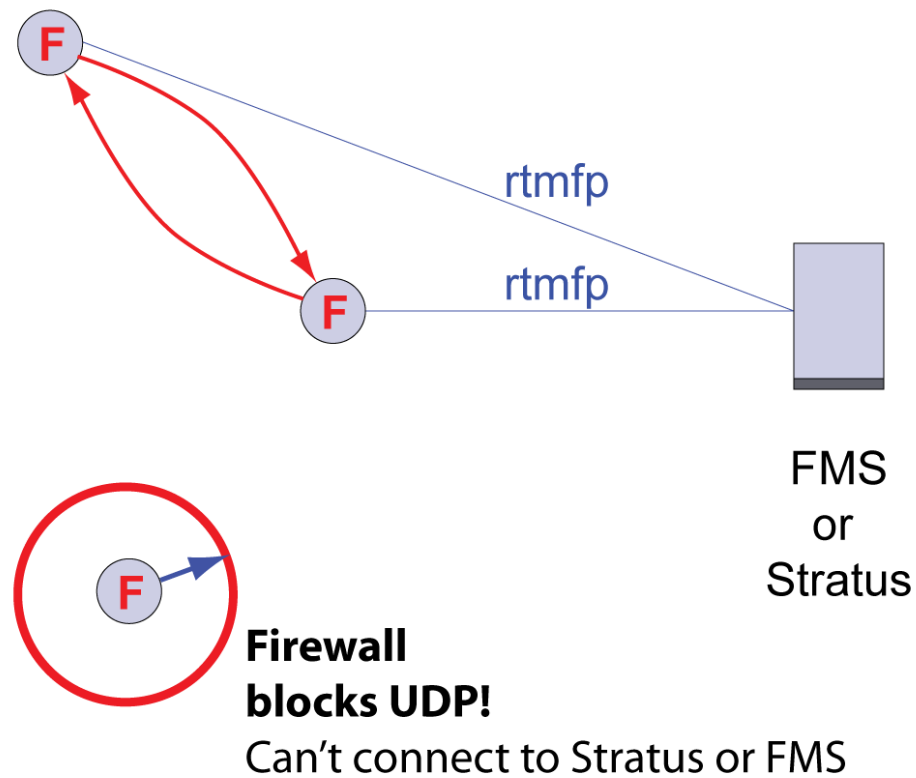
```
netStream.nearNonce
```

Connection Problems?

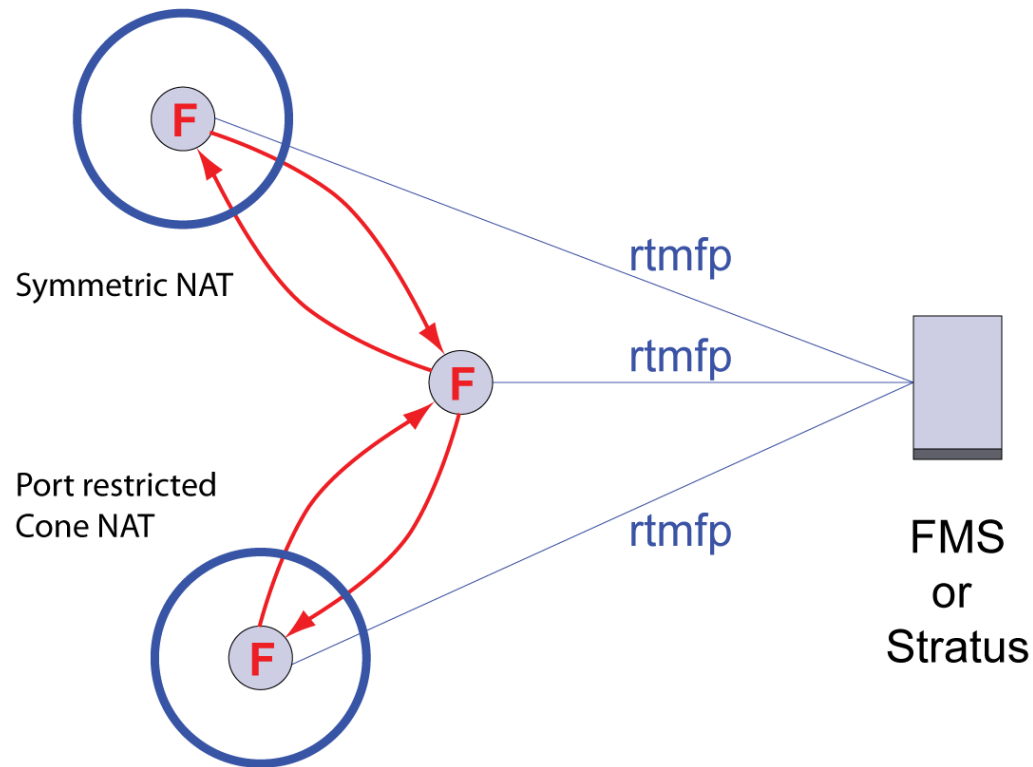
Must be able to connect to both server and clients:

- Server UDP port 1935 and one higher port
- Dynamically assigned UDP client ports
- Stratus does not provide a fail-over option
- Failover is built in to Adobe Flash Collaboration Service and can be scripted in a future release of FMS

Connection Problems?

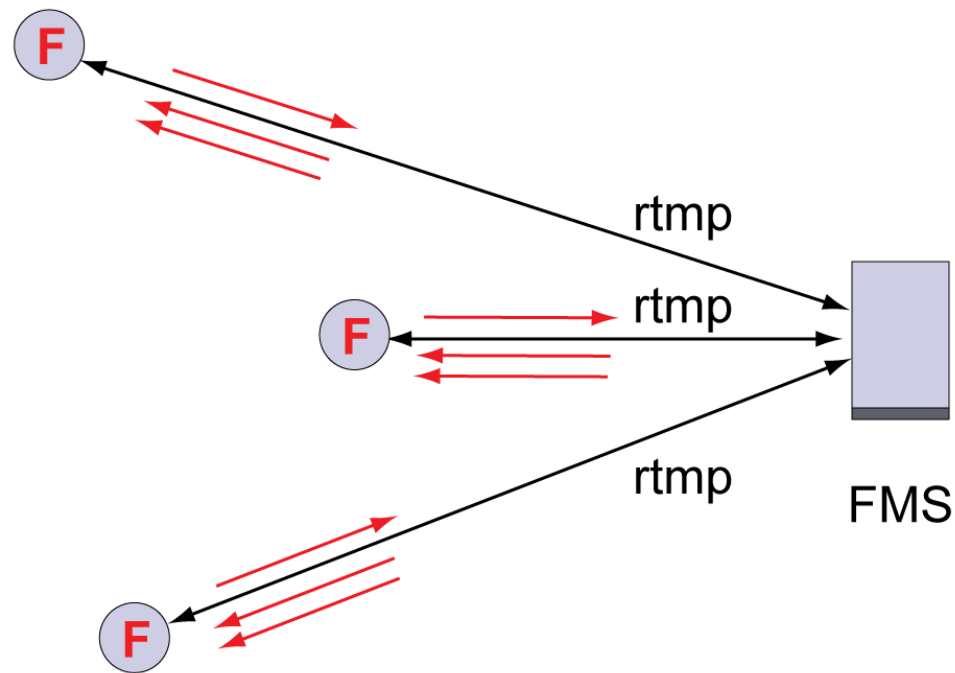


Connection Problems?



Player-to-Player Communications with RTMFP

RTMP/FMS



Connection Problems?

Property Name	Property Value
BCAoYlcb	id: "BCAoYlcb" peerID: protocol: "rtmp"
CJAQYFNJ	id: "CJAQYFNJ" peerID: "92d93e09b2a0b194d8688980120627ed3093e593a0b97b98ff7a8618571ea94a" protocol: "rtmfp"
DIA4o9NJ	id: "DIA4o9NJ" peerID: "14f298edc010f608999dc6e839054c4fbc7cf05fa701cdd4d8ed134433bb2aab" protocol: "rtmfp"

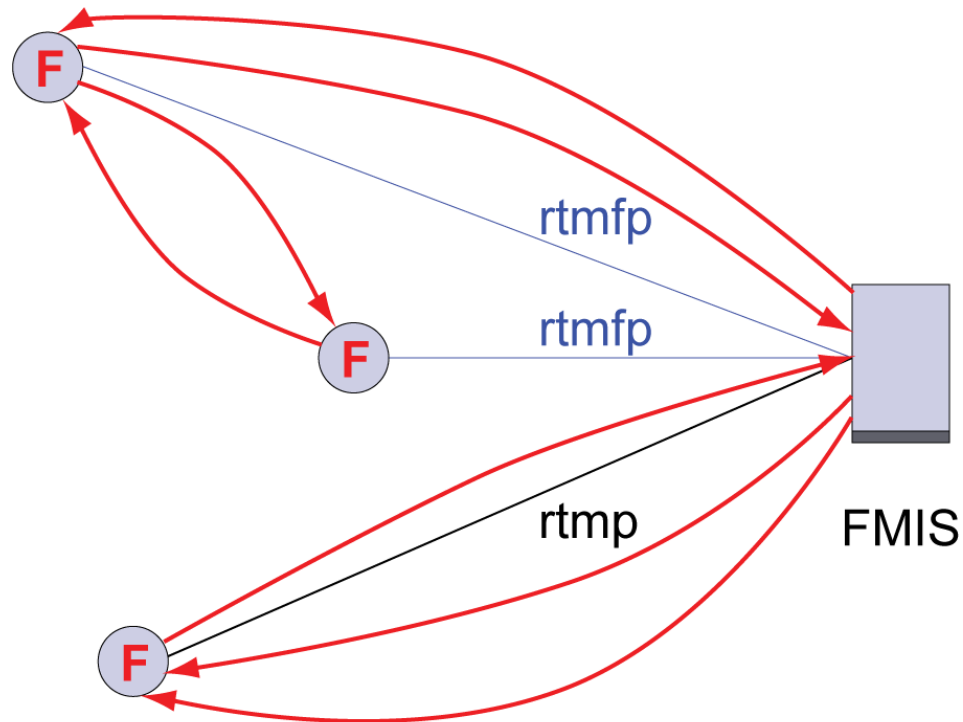
RTMFP+RTMP/FMS

Failover is not automatic:

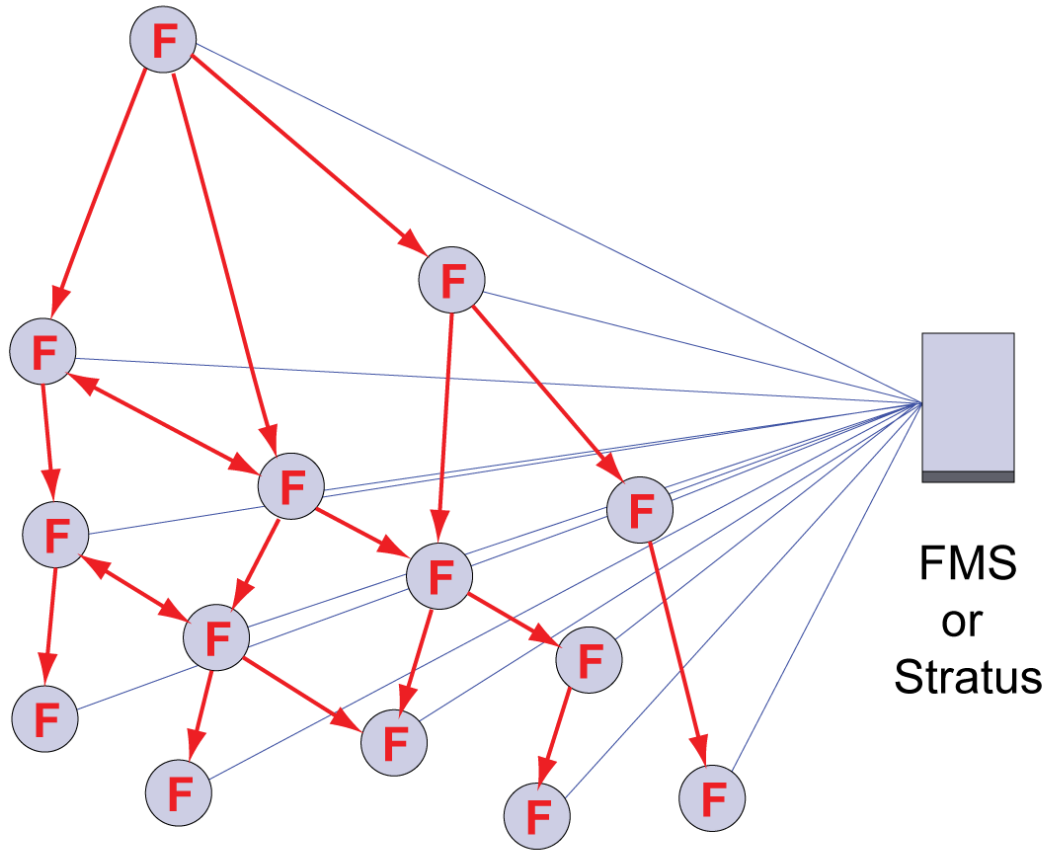
- When an RTMFP connection attempt fails call `connect()` again with `rtmp`.
- A direct stream cannot be relayed through the server, so create a second stream with the same audio, video, and data.

Player-to-Player Communications with RTMFP

RTMFP+RTMP/FMS



What's Next?



Player-to-Player Communications with RTMFP

Links and Thanks

Slides and Code will show up here eventually:

<http://flash-communications.net/technotes/fitc2009/index.html>

Long Article on RTMFP on my O'Reilly Blog:

<http://broadcast.oreilly.com/2009/04/adobes-real-time-media-flow-pr.html>

Getting started with Stratus:

<http://adobe.com/go/stratus/>

http://www.adobe.com/devnet/flashplayer/articles/rtmfp_stratus_app_print.html

Getting started with Adobe Flash Collaboration Service:

<http://labs.adobe.com/technologies/afcs/>

<http://blogs.adobe.com/collabmethods/>

Thanks to Matthew Kaufman and others at Adobe for answering so many questions.