Publisher and player channel quality control

When a WebRTC stream is publishing, picture quality depends on media data transfer channel between client and server, especillay for high definition streams (HD, FullHD, 4K). The ability to control channel quality and notify publisher about bandwidth decrease in time using WebSDK was added since build 5.2.398. Subscriber, in its turn, can be notified about bandwidth decrease since build 5.2.409.

The publishing and playback bitrate values on client side are periodically comparing with server side one. The steady divergence of those values means channel bandwidth decrease. The peaks and sudden changes are smoothed by Kalman filter.

Server configuration

Current server side publishing bitrate values sending to client for later comparison should be enabled with the following parameter in flashphoner.properties file

```
inbound_video_rate_stat_send_interval=1
```

Current server side playback bitrate values sending is enabled with the following parameter

```
outbound_video_rate_stat_send_interval=1
```

The settings above define bitrate values sending interval in seconds. It is recommended to send bitrate to client every second.

Channel quality displaying at client side

Let's look at channel quality and bitrate changing graphs displaying using Media Devices example.

1. A function to prepare to display graphs code

```
function createOrClearChart(chartId, bitrateComparisonChart) {
    if (!bitrateComparisonChart) {
        var canvas = document.getElementById(chartId);
        var ctx = canvas.getContext('2d');
        bitrateComparisonChart = new ComparisonChart(ctx);
    } else {
        bitrateComparisonChart.clearBitrateChart();
    }
```

```
    return bitrateComparisonChart;
}
```

function usage while publishing code

```
function publish() {
    ...
    publishConnectionQualityStat.chart =
createOrClearChart('publishBitrateChart',
publishConnectionQualityStat.chart);

    publishStream = session.createStream({
    ...
    });
    publishStream.publish();
}
```

function usage while playing code

```
function play() {
    ...
    playConnectionQualityStat.chart =
createOrClearChart('playBitrateChart', playConnectionQualityStat.chart);

    previewStream = session.createStream({
    ...
    });
    previewStream.play();
}
```

2. Channel quality and bitrate values receiving, bitrate graphs displaying `CONNECTION_QUALITY.UPDATE` event handling while publishing code

```
publishStream = session.createStream({
  ...
}).on(CONNECTION_QUALITY.UPDATE, function (quality, clientFiltered,
serverFiltered) {
  updateChart(quality, clientFiltered, serverFiltered,
publishConnectionQualityStat);
});
publishStream.publish();
```

while playing code

```
previewStream = session.createStream({
  ...
}).on(CONNECTION_QUALITY.UPDATE, function (quality, clientFiltered,
serverFiltered) {
  updateChart(quality, clientFiltered, serverFiltered,
playConnectionQualityStat);
});
previewStream.play();
```

a function to update graphs and quality code

```
function updateChart(calculatedQuality, clientFiltered, serverFiltered,
connectionQualityStat) {
    var timestamp = new Date().valueOf();
    connectionQualityStat.connectionQualityUpdateTimestamp = timestamp;
    connectionQualityStat.chart.updateChart(clientFiltered,
serverFiltered);
    connectionQualityStat.quality = calculatedQuality;
}
```

3. Set channel quality to `UNKNOWN`, if `CONNECTION_QUALITY.UPDATE` event is not received while publishing code

```
function loadStats() {
    if (publishStream) {
        ...
        if(new Date().valueOf() - CONNECTION_QUALITY_UPDATE_TIMEOUT_MS >
publishConnectionQualityStat.connectionQualityUpdateTimestamp) {
            publishConnectionQualityStat.quality =
CONNECTION_QUALITY.UNKNOWN;
        }
        ...
    }
    ...
}
```

while playing code

```
function loadStats() {
    ...
    if (previewStream) {
        ...
        if(new Date().valueOf() - CONNECTION_QUALITY_UPDATE_TIMEOUT_MS >
playConnectionQualityStat.connectionQualityUpdateTimestamp) {
            publishConnectionQualityStat.quality =
CONNECTION_QUALITY.UNKNOWN;
        }
        ...
    }
    ...
}
```

4. Channel quality displaying while publishing code

```
function loadStats() {
    if (publishStream) {
         ...
         if (publishConnectionQualityStat.quality !== undefined) {
             showStat({"quality": publishConnectionQualityStat.quality},
"outConnectionStat");
             ...
```

```
        }
        ...
    }
    ...
}
```

while playing code

```
function loadStats() {
    ...
    if (previewStream) {
          ...
          if (playConnectionQualityStat.quality !== undefined) {
              showStat({"quality": playConnectionQualityStat.quality},
"inConnectionStat");
              ...
          }
          ...
    }
    ...
}
```

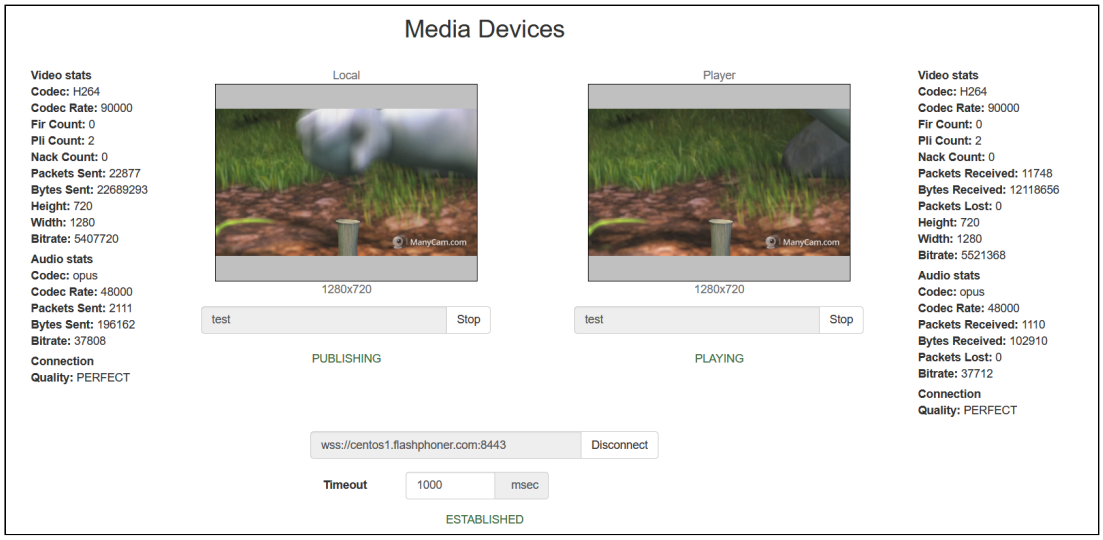
a function to display quality code

```
 function showStat(stat, type) {
     Object.keys(stat).forEach(function(key) {
         if (typeof stat[key] !== 'object') {
             let k = key.split(/(?=[A-Z])/);
             let metric = "";
             for (let i = 0; i < k.length; i++) {
                 metric += k[i][0].toUpperCase() + k[i].substring(1) + "
";
             }
             if ($("#" + key + "-" + type).length == 0) {
                 let html = "<div style='font-weight: bold'>" +
metric.trim() + ": <span id='" + key  + "-" + type + "' style='font-
weight: normal'></span>" + "</div>";
                 // $(html).insertAfter("#" + type);
                 $("#" + type).append(html);
             } else {
                 $("#" + key + "-" + type).text(stat[key]);
             }
         }
     });
 }
```

Testing

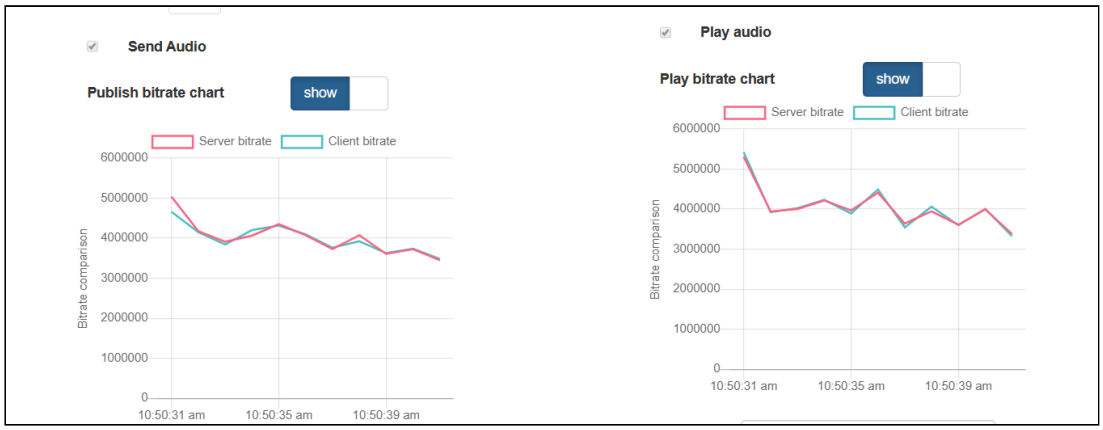
1. For the test we use:
2. WCS 5.2.409 or newer
3. Media Devices example in Chrome browser

4. publishing channel with 100 Mbps upload and download bandwidth
5. bandwidth shaping tool, winShaper on Windows or Network Link Conditioner on MacOS for example
6. Publish and play 720p stream on Media Devices page

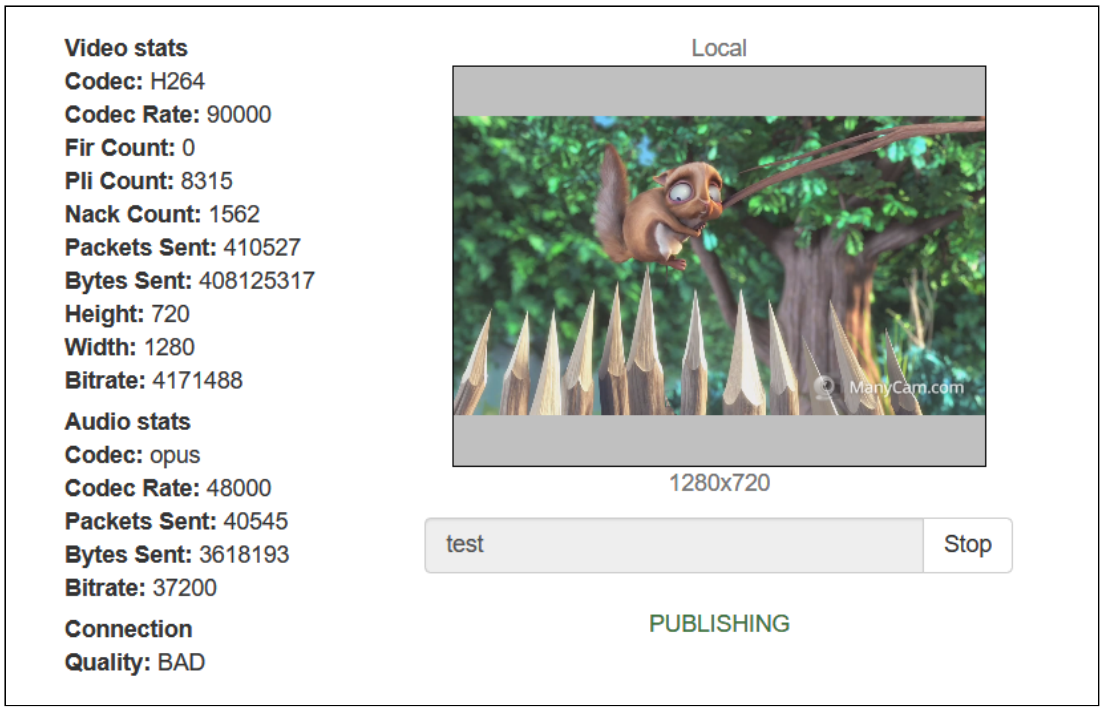


The PERFECT channel quality is displayed for publisher and player

7. Check publishing and playing bitrate graphs on perfect channel

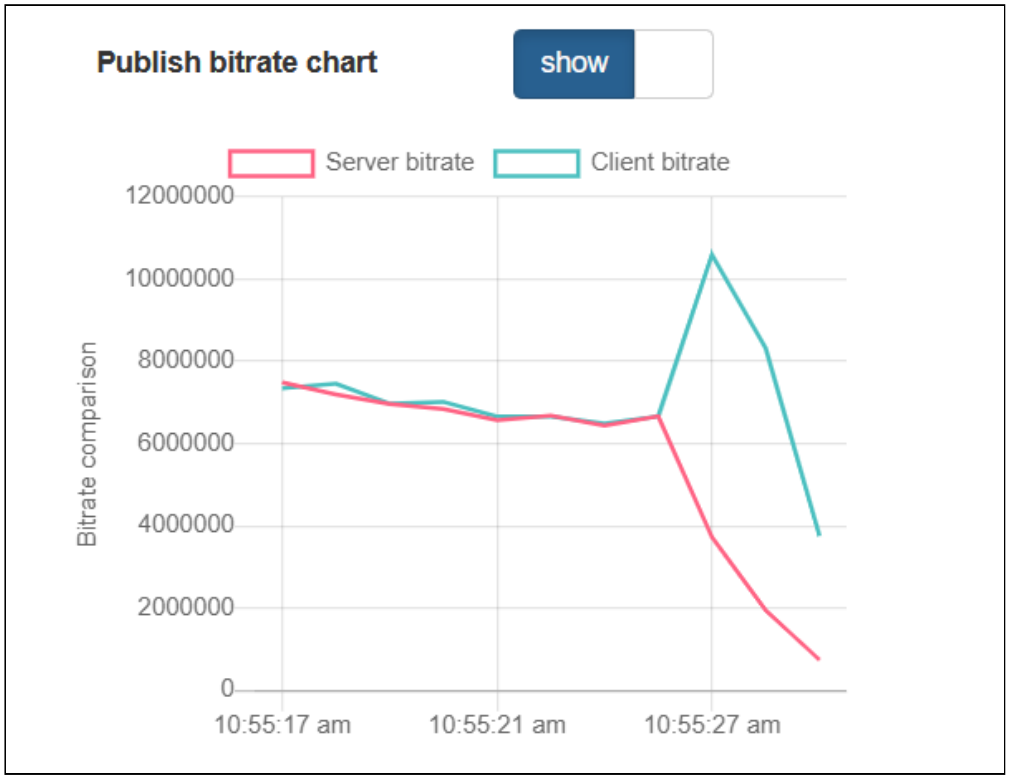


8. Shape outgoing traffic to 768 kbps, simulating a typical 3G connection

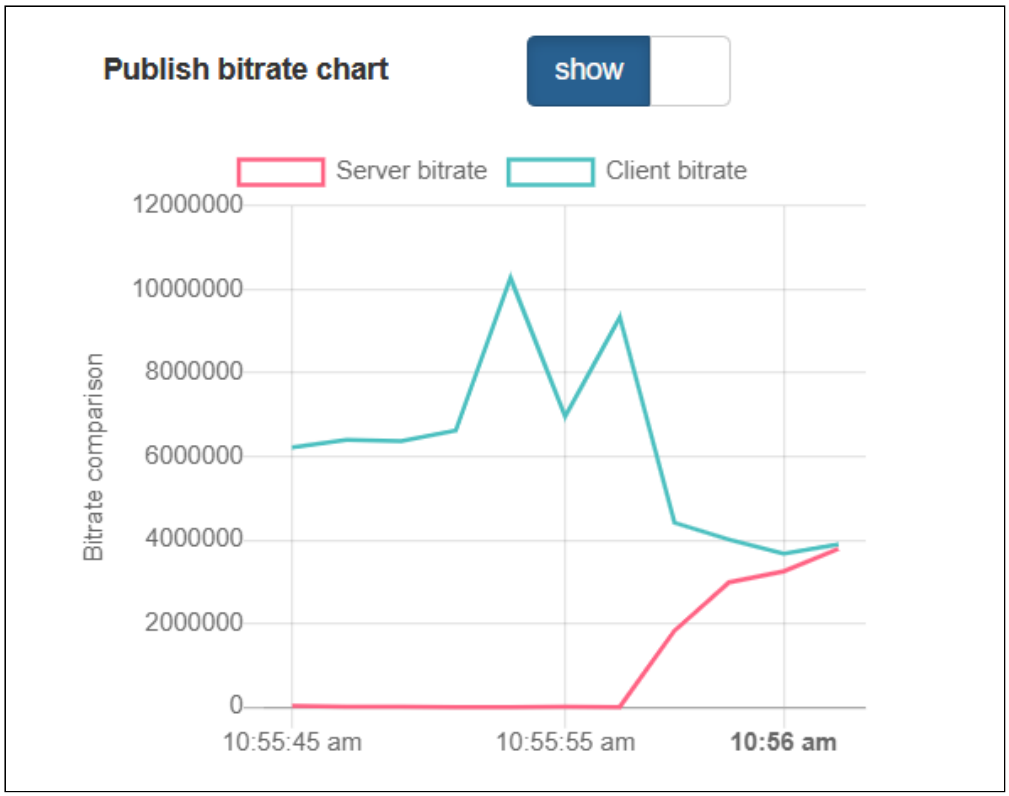


The `PERFECT` value changes to `BAD` for publisher.

Publishing bitrate graph looks as follows

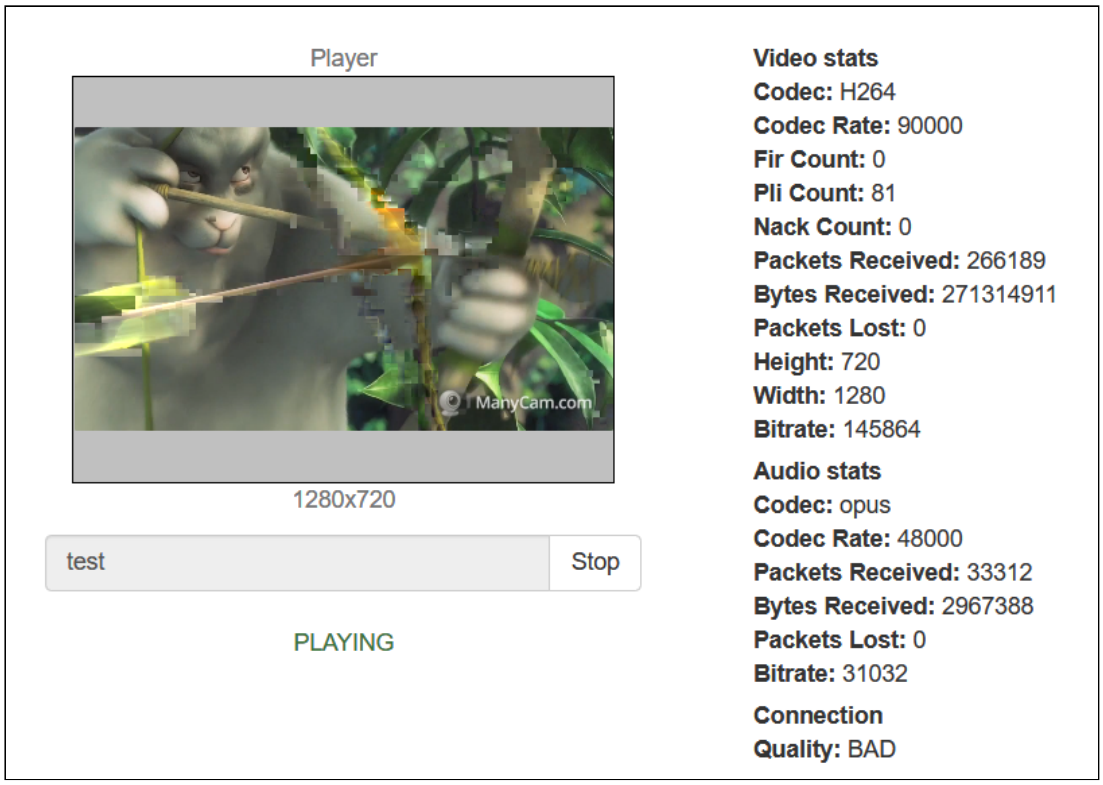


9. Stop bandwidth shaping, check publishing bitrate graphs



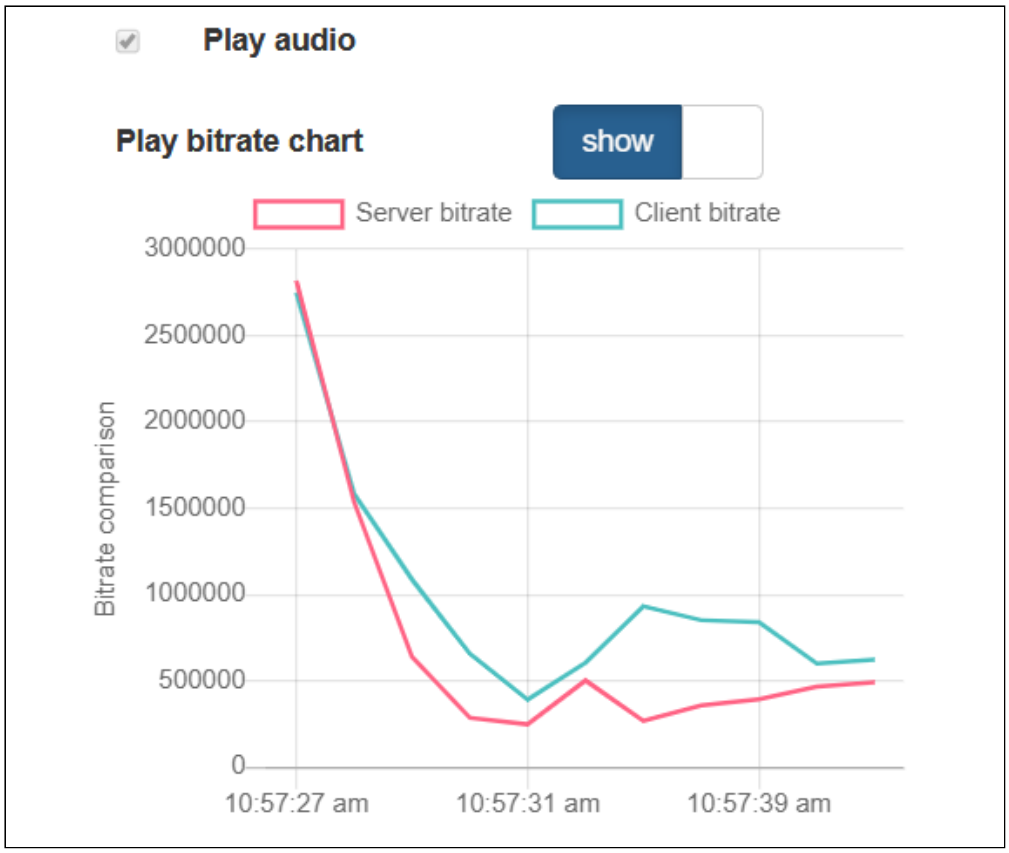
After the graphs converge again, the `PERFECT` publisher channel quality value is displayed

10. Shape incoming traffic to 768 кbps

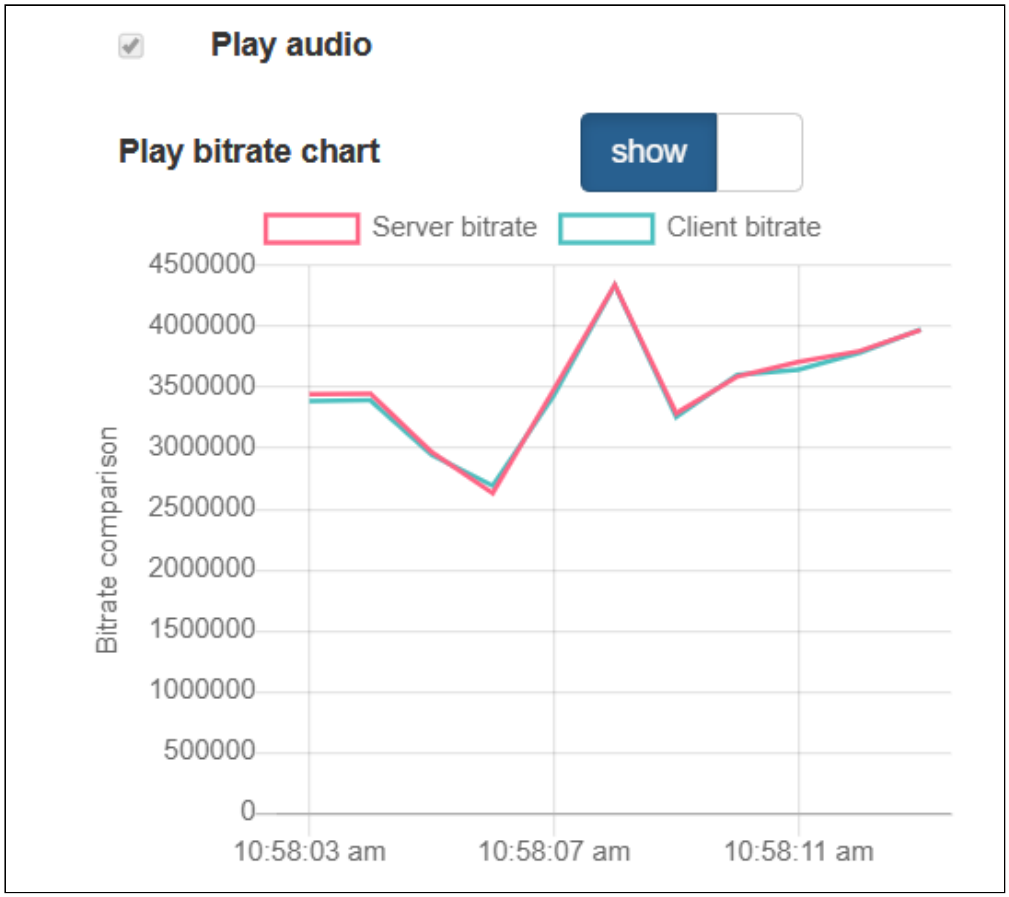


The `PERFECT` value changes to `BAD` for subscriber, picture freeze and artifacts are

observed. Playing bitrate graph looks as follows



11. Stop bandwidth shaping, check playing bitrate graphs



After the graphs converge again, the `PERFECT` subscriber channel quality value is displayed, picture is restored

Recommendations to publishers

If channel quality is displayed as `PERFECT` or `GOOD`, it means channel bandwidth is enough to publish a stream with a currrent resolution and bitrate.

If channel quality is changed steadily to `BAD`, it means channel bandwidth is not enough, and subscribers are viewing a problems. It is recommended to lower publishing bitrate and/or resolution if possible.

If channel quality is changed steadily to `UNKNOWN`, video frames can not reach the server. It is recommended to republish stream.

Recommendations to subscribers

If channel quality is displayed as `PERFECT` or `GOOD`, it means channel bandwidth is enough to play a stream with a currrent resolution and bitrate. If a problems occur while playing stream in this case, the reason of the problems is probably on publisher side.

If channel quality is changed steadily to `BAD`, it means channel bandwidth is not enough, picture freeze and artefacts are observed. It is recommended to request the stream with lower bitrate and/or resolution if possible.

If channel quality is changed steadily to `UNKNOWN`, video frames can not be received from the server. It is recommended to reconnect and restart the stream playback.