





# Diagnosing Media Issues on the Fusion Platform

Version V.1.0

CaféX Communications 135 West 41st Street, Suite 05-108, New York, NY 10036 www.cafex.com

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC
			Page	1 of 74



# **Version History**

Document Control		
Version	Author	Description/Change History
1.0 November 15th 2019	CaféX Support (TH)	A guide to determine typical media issues that may be seen using CaféX Fusion Client SDK or Fusion Live Assist.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	2 of 74



## **Table of Contents**

Introduction Use of Third Party Tools Streaming Traffic Overview **Stream Metrics** Video Traffic Overview **Keyframes Delta Frames Diagnosing Media Issues** Mandatory Additional Logs **Other Useful Information** Architecture Overview **Call Flow** Understanding Fusion Media Broker Ports **Configuring Multiple Media Broker Ports Example Configuration** Advanced Configuration Understanding Media Streams WebRTC to WebRTC Calls WebRTC to SIP Calling an MCU **Understanding Media Setup** 

STUN DTLS RTP & RTCP

Setting Up Wireshark for Analyzing RTP Streams Enable Automatic Decoding of RTP Streams Save Useful Filters Initial PCAP Analysis Stream Analysis has "Timestamp incorrect" Is STUN Working?

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	3 of 74





No STUN in the pcap The DTLS Handshake Working DTLS Handshake Failed DTLS Handshakes Drawing the Call Flow Sequence SIP Side with Wireshark (from FAS) Advanced Packet Capture Analysis Measuring Bandwidth Analysis of an SIP audio stream Stream Analysis H.264 Codec H.264 Profiles: H264 Decoding Finding the Sequence Parameter Set Audio & Video Analysis Measuring Retransmissions Analyzing Streams on a Network Bridge Picture Quality Picture Loss Recovery **Finding PLIs** RFC2032 FIRs & SIP INFOs NACK RTX Fragmentation Adaptive Bitrate SIP-side considerations Finding TMMBR Finding REMB Analyzing Lip Sync Captures taken from an iPAD **RVI Captures Introduce Erroneous Skew Measurements** 

<u>Capture taken from a Mavericks Mac</u> iPAD capture shows Packet Loss on Outbound Stream

Document Title:	Diagnosing Media Issues on the Fusion Platfor	'n	Classification	PUBLIC
			Page	4 of 74



Captures from a Browser Bandwidth Estimates

Local Firewall Configuration

Testing the Local Firewall Ports

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC
			Page	5 of 74



# Introduction

This document gives a very basic overview of how RTP traffic traverses a typical network and will describe how to:

- Analyze a typical SIP call flow through Fusion Client SDK
- Understand the media paths established between FCSDK and other network devices
- Understand common reasons for media establishment failure
- Quantify Packet Loss for a call from an iPAD Client
- Quantify Packet Loss for a call from a Chrome Browser Client
- Quantify Packet Loss at the Fusion Media Broker

# **Use of Third Party Tools**

This document uses a number of industry standard third party tools, such as Wireshark. It may be that the user interfaces in this document will change in these tools. As a reader, it is more important you understand why the tools are being used and prepare for differences in the step by step process.

Document Title:	Diagnosing Media Issues on the Fusion Plat	form	Classification	PUBLIC
			Page	6 of 74



# **Streaming Traffic Overview**

Here will introduce common terminology for describing all forms of Audio and Video Streams.

Generally speaking a stream is a sequence of packetized media samples. Typically, audio and video codecs have an associated sample rate. A codec also describes the clock rate; which is the rate the audio was sampled.

For example:

- The G711 audio codec suggests (but does not mandate) that 20ms of audio are transmitted per packet.
- G711 has a clock rate of 8000Hz; (eg 8000 samples per second).
- Each packet will contain 160 samples.

**Note:** G711 doesn't mandate 20ms of audio, so a packet may contain more of less than 160 samples; as a result the receiver must handle these variations to construct an audible stream.

The RTP packets contain information to help the far end reconstruct the stream, below is an example of an RTP packet:

```
Real-Time Transport Protocol
10.... = Version: RFC 1889 Version (2)
..0... = Padding: False
...0... = Extension: False
...0000 = Contributing source identifiers count: 0
0... = Marker: False
Payload type: ITU-T G.711 PCMU (0)
sequence number: 41853
Timestamp: 1973995500
Synchronization Source identifier: 0xabfbc9d8 (2885405144)
Payload: b4c35e2998a047d3d52e5112c9102adac61609592d3fe0ab...
```

The Payload contains the data the was sent in this packet.

The *Sequence Number* identifies a packet's position in a stream. The first packet of a stream will assign a random number and every subsequent packet will be incremented

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	7 of 74



by one. The Sequence Number's primary objective is to identify lost packets.

The *Timestamp* is used to construct the sample with the correct timing. In this stream, the previous packet had a Timestamp of: 1973995340. Thus, the receiver knows that this sample contains 160 samples(ie: 1973995500 - 1973995340 =160). Multiple packets in a stream will have identical Timestamps if: they have been retransmitted or there are one of multiple packets used to create a sample (such as a video frame spread over many packets).

## **Stream Metrics**

Assuming a well behaved sender, generally three metrics are used to measure the quality of the stream at the receivers end. This metrics can be used by receivers when reconstructing streams. They are:

- Latency: The time it takes to get a packet from sender to receiver.
- **Skew**: If the stream is a G711 steam, one typically expects samples to be arriving every 20ms. Skew measures cumulative lateness of a given packet relative to the previous packet and since the start of the stream.
  - In theory, the 100th packet should arrive 2000ms from the start of the stream; if it arrives at 2020ms and the 99th packet arrived at 1980ms, the packet is considered 20ms late and the skew will be measured as -20ms.
  - Small amounts of skew fluctuation should be managed by a receiver.
- Jitter: Generally it is a measure of the "Packet Delay Variation".
  - It is a good way of comparing one point of a given call with another.

Page 8 of 74	Document Title:	Diagnosing Media Is	ssues on the Fusion Platfo	orm	Classification	PUBLIC
					Page	8 of 74



# Video Traffic Overview

When a Video stream is established between two video enabled devices there are independent RTP/UDP streams which flow in either direction between the devices. The video stream is broken up into packets and these packets make up two fundamental components that a video devices needs in order to construct a coherent image:

- Keyframes
- δ-frames (delta-frames)

You should remember that video codecs and compression can drastically change how these frames are sent and received. For now, we will assume there are no Error Correction algorithms such as: PLI or NACK which can be used to trigger a new keyframe if required or resend a packet if it is missing. Similarly, Forward Error correction algorithms provide information in the stream to verify and correct a stream if something is lost.

## Keyframes

A key frame is used to construct an entire image which can be displayed. It contains all of the necessary information to render an image and is not dependent on any other parts of the stream.

A key frame will span multiple UDP packets (depending on the resolution of the video).

Keyframes are an important way of allowing a decoder to refresh and start again if things are going badly.

## **Delta Frames**

 $\delta$ -frames are collections of packets which only contain parts of the previous image. They only contain information which has changed since the previous frame. A video device will interpret these delta-frames extrapolate an image. Below is an example showing 2-keyframes (numbered 1 and 5) and 3  $\delta$ -frames (numbered: 2,3,4); the bottom of the image shows what the video device renders.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	9 of 74





Image from: <u>http://nickyguides.digital-digest.com/keyframes.htm</u> There are many reasons for packet-loss on a network. This could be due to: weak wifi signals, high network contention, high network throughput and Quality of Service guarantees implemented by networks. It must be understood that:

- If there are UDP packets lost for δ-frames then a video device will not be able to extrapolate a coherent image until a keyframe arrives. This is displayed as a partially corrupted or pixelated image. When a keyframe arrives it allows the video device to render a fresh image.
- Keyframes are constructed from multiple packets; If one of these packets is lost then the video device will not be able to render the new keyframe and will attempt to continue extrapolating images using δ-frames until the next suitable keyframe arrives.

**Note:** Video is more sensitive to lost or corrupt data compared to audio, this is for a number of reasons:

- Spoken audio is can be mostly silence, so you don't notice
- Our eyes are very good at detect subtle changes
- Corruption is often cumulative

Document Title:	Diagnosing Media I	ssues on the Fusion Platfo	orm	Classification	PUBLIC
				Page	10 of 74



# **Diagnosing Media Issues**

Diagnosing media issues has some absolute requirements which you cannot work without. There are also nice to haves which will make life easier but are not necessary in most situations. If you have the **Mandatory** items it may not be necessary to collect the **Additional Logs**.

## Mandatory

- Understanding of the call flow and Architecture
- A Media Broker packet capture of a complete single call displaying the issue:
  - Media Broker Logs (DEBUG by default) of a single call with the issue
  - Media Broker pcap of a single call with the issue
- Calls.log from FAS (if WebRTC to SIP)

# **Additional Logs**

- Gateway Config XML or DEBUG server.log from Gateway
- Media Broker Logs (DEBUG by default) of a working call
- Media Broker pcap of a working call
- iOS/Android console logs
- Web Console Logs
- DEBUG FAS Logs

# **Other Useful Information**

Information such as version numbers etc, listed above, can normally be found from the Mandatory logs. Steps to collect logs can be found in the product troubleshooting guides found at: https://support.fusion.cafex.com.

Document Title: D	Diagnosing Media Issue	on the Fusion Platform	Classification	PUBLIC
			Page	11 of 74



# **Architecture Overview**



All communications between WebRTC Clients and FCSDK are secure. The Media DTLS handshake will be covered in more detail, but this prevents packet inspection; however, some important details can be gleaned. By default SIP side transactions and media are not encrypted, so more inspection can be performed.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	12 of 74





## Call Flow

A typical scenario for FCSDK is for the FCSDK client to callout to a SIP endpoint via the Fusion Gateway and through a PBX or contact center.



Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	13 of 74



The FCSDK Gateway allocates a Media Broker to direct a call when a new request is received. Media Broker allocates a process and ports for the traversal of media. For a given call, codec and video resolutions are fixed. Initial information about the streams can be retrieved the transactional SDP at the start of the call.

Before initiating any detailed packet analysis, it is worth trying to understand the SDP negotiations, so you can hypothesis, what you expect each client and the Media Broker to be doing. This may simply be checking codecs and ports, but also understanding if any other transactions may result in a misunderstanding of each other's protocols.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	14 of 74





# **Understanding Fusion Media Broker Ports**

This diagram explains what values should be specified when adding Media Broker

configuration to a FCSDK installation:



#### SIP Network

 Local Address CIDR - is the address range the Media Broker will bind to for RTP communications on the SIP Network.

**Note**: If you have 2 network interfaces on the box don't use 'all' as the CIDR but target the internal interface only For example X.X.X/32

#### WebRTC Client

• Source CIDR Address - is the address range on which the Gateway will

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	15 of 74



### receive WebRTC traffic from clients

**Note**: If all external traffic comes via a single ReverseProxy you can create a rule with the ReverseProxy internal address as the Source CIDR (X.X.X.X/32) with the Public address as the external firewall address. Then to allow internal clients to connect directly to the Gateway you can create a 2nd rule with 'all' as the Source CIDR with the Public address as the internal Media Broker address.

- *Public Address* is the address the client must send RTP traffic to; typically the front of a firewall.
- Local Address is the address the Media Broker will bind to in order to receive RTP traffic

Configuring Multiple Media Broker Ports

Media Brokers of FCSDK 2.1.31 introduces simultaneous rtp-proxy processes for managing calls. This impacts how ports are allocated between these processes.

- SIP port Range These ports are distributed across the rtp-proxy instances, in groups of 4.
  - Number of SIP-Ports to allocated = (4 ports for every WEB-RTC Client per call)x( Maximum Number of Concurrent calls on a Media Broker) + (a small contingency [eg: 10%]).
  - Ports are not reallocated immediately when a call is ended, so on smaller systems the contingency should be a larger percentage.
- WEB-RTC Port Range:
  - Number of WEB-RTC Ports to Allocated = (Number of rtp-proxy processes [default is 5]).
  - It is necessary to allocate the same number of ports to each Source CIDR Address to ensure that each rtp-proxy process can assign the correct interface/port pair to a call.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC
			Page	16 of 74





## **Example Configuration**

	Sou	urce Address CIDR					
g.	all						
	RT	P Public and Lo	cal Port				
		Public Address	Public Port	Local Address	Local P	ort	
		81.144.171.73	16000	172.31.252.111	16000		
		81.144.171.73	16001	172.31.252.111	16001		
		81.144.171.73	16002	172.31.252.111	16002		
		81.144.171.73	16003	172.31.252.111	16003		
	0	81 144 171 73	16004	172.31.252.111	16004		
	172 RT	2.31.253.0/24	cal Port			Add	Delete
	172 RT	2.31.253.0/24 P Public and Lo Public Address	cal Port	Local Address	Local P	Add	Delete
	172 RT	Public Address	Cal Port Public Port 16000	Local Address 172.31.252.111	Local P 16000	Add	Delete
	172 RT	Public and Lo Public Address 172.31.252.111 172.31.252.111	cal Port Public Port 16000 16001	Local Address 172.31.252.111 172.31.252.111	Local P 16000 16001	Add	Delete
	172 RT	Public Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	cal Port Public Port 16000 16001 16002	Local Address 172.31.252.111 172.31.252.111 172.31.252.111	Local P 16000 16001 16002	Add	Delete
	172 RT	Public and Lo Public Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	cal Port Public Port 16000 16001 16002 16003	Local Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	Local P 16000 16001 16002 16003	Add	Delete
	172 RT	Public and Lo Public Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	Cal Port Public Port 16000 16001 16002 16003 16004	Local Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	Local P 16000 16001 16002 16003 16004	Add	Delete
	1772 RT	Public Address 172.31.253.0/24 Public Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	cal Port Public Port 16000 16001 16002 16003 16004	Local Address 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111 172.31.252.111	Local P 16000 16001 16002 16003 16004	Add ort Add	Delete

Above is an example configuration for 1 Media Broker with 5 rtp-proxy processes. It is intended to be used with Live Assist<sup>™</sup>; where a consumer's media is sent to a public IP address, but Agent media is sent to an internal interface.

It is possible to allocate the same local port and interface (eg: 172.31.252.111:16000) against each CIDR.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	17 of 74





### Warning

When modifications to the ports are made all rtp-proxy processes will restart to acquire their new config. This will terminate any calls in progress.

## Advanced Configuration

In case of a public Internet or ISP outage, some installations require that a Media Broker supports more than a single public Media Broker Address. This can be done by:

- Defining 2 (or more) Source Addresses for each ISP.
- Allocating disjoint sets of public addresses against each source address.

If an ISP becomes unavailable the Media Broker will stop receiving requests from the ISP; as a result the set of public addresses will never be allocated.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	18 of 74



# **Understanding Media Streams**

Depending on your call setup numerous media streams will be established between clients, endpoints and the Media Broker. If you are troubleshooting, it is often worth creating a diagram of your expectations so each stream can be identified in any packet captures.

While the above images and notes give a good indication of a normal call setup, each customer may have their own individual setups with small to significant differences.

The following are some common example scenarios:

Document Title:	Diagnosing Media Issues on the Fusion Plat	orm Classification	PUBLIC
		Page	19 of 74





# WebRTC to WebRTC Calls

For each Client (WebRTC Side) you will see 8 Streams:

- One RTP stream for sending Video
- One RTP stream for receiving Video
- One RTP stream for sending Audio
- One RTP stream for receiving Audio

You will also see, on the sip side 4 Streams:

- Media Broker send itself one video stream per client (2 Streams total)
- Media Broker send itself one audio stream per client (2 Streams total)



Document Inte. Diagn	Diagnosing Media Issues on the Fusion Platform		PUBLIC
		Page	20 of 74





# WebRTC to SIP

For the WebRTC Client you will see 4 Streams:

- One RTP stream for sending Video
- One RTP stream for receiving Video
- One RTP stream for sending Audio
- One RTP stream for receiving Audio

For the SIP Client you will see 4 Streams:

- One RTP stream for sending Video
- One RTP stream for receiving Video
- One RTP stream for sending Audio
- One RTP stream for receiving Audio



Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	21 of 74



# Calling an MCU

Three way calling is not overly common but does complicate the number of streams that will be seen.

For each Client (WebRTC Side) you will see 12 Streams:

- One RTP stream for sending Video
- One RTP stream for receiving Video
- One RTP stream for sending Audio
- One RTP stream for receiving Audio

You will also see, 12 streams on the sip side:

- Media Broker send the MCU one video stream per client (3 Streams total)
- Media Broker send the MCU one audio stream per client (3 Streams total)
- Media Broker receive from the MCU one video stream per client (3 Streams total)
- Media Broker receive from the MCU one audio stream per client (3 Streams total)



Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	22 of 74



## **Understanding Media Setup**

The following diagram expands on the interaction between Web-RTC clients and the Media Broker during a call set up.

STUN Binding Requests are the first UDP messages sent between clients.

The Media Broker must be able to send UDP outbound towards all Web-RTC clients. Web Client Media Broker SIP client



## STUN

STUN's primary purpose is to open paths through firewalls and authenticating source ports. Typically, there is a request and a corresponding success response from each client, resulting in a four packet exchange. If there is a failure at this stage, such a firewall block, no media will establish.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	23 of 74



Once the WebRTC client has processed the answer SDP it will start sending STUN binding request packets to the public interface port (default: 16000) on MediaBroker. One exception is Firefox, where ICE is sent to the client for it to select a candidate.

Media Broker waits to receive a STUN request then generates a response, before sending its STUN requests back to the client. STUN will be seen throughout the call, it is used to check paths are media paths are still valid. They are typically send every 0.5s from the majority of Google WebRTC clients.

Interactions with clients running the Google Chrome webRTC library behave a little differently. Chrome clients only send a STUN request after it finishes sending a success response to a received request. As in the previous diagram, you may see 6 packets during the STUN setup. If you're analyzing the STUN, the second request and response pair from Chrome will take precedence and contain the media candidates.

Once STUN us successful the client and Media Broker can continue to establish the RTP stream.

## DTLS

Next DTLS exchanges happen in order to get keys for encrypting RTP. The client initiating the call takes on the role of DTLS client and sends a client hello to MB, which responds with a DTLS packet containing a server hello and a number of other details. The client then responds with a certificate and other information, to which MB then sends a change cipher specification and an encrypted handshake message. If there are packet losses then lost ones are automatically resent after delays specified in the DTLS RFC.

The diagram earlier shows the format you will see the DTLS packets, any deviation from that sequence (ignoring retransmissions) means something has gone wrong. A failure may result in an error packet, or it may fail silently. However a failure happens it is likely packets will continue to be resent without answer.

Often, there are two encrypted alerts sent at the end of the call, one from each side. They can be ignored, but can be a useful way of determining when one side thinks the call has ended.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	24 of 74



Once STUN and DTLS are complete the media can begin to flow.

## **RTP & RTCP**

While the webRTC client is establishing a media path, the SIP client is sending RTP and RTCP to the MB to the internal ports (default: 17000+). These packets will be discarded by Media Broker because it could not send them to the client until the DTLS handshake completes. Once complete, the Media Broker can start to send received media in both directions.

For a given call leg, there are four ports allocated for SIP side media on Media Broker. The even numbered ports being used to receive RTP audio and video, and the port one higher for the corresponding RTCP. The Media Broker's webrtc side only uses one port and uses the SSRCs to distinguish the audio from the video media.

**Note**: The ports in the SDP of SIP clients are the ports a client wants to receive media on. It can send media from a different port, though this rarely happens.

Media on the web side is encrypted, without decrypting a PCAP, only the main RTP header is readable. For an RTCP packet, only the headers up to and including the first SSRC header is readable. The encryption also adds 10 (RTP) or 20 (RTCP) bytes at the end of the packet.

**Note**:Extended headers exist but you will probably never see them, but special codecs use similar extensions for various details which will be encrypted.

When MB receives packets from the web side they are held in the Media Broker before sending out on the SIP side in a jitter buffer. The jitter buffer's purpose is to reduce jitter. The buffer is initially ~300ms, but will increase if network conditions are poor, giving more time for packets to reorder before processing and sending downstream. In passthrough calls, packets received from the SIP have SSRCs, timestamps and sequence numbers changed in RTP packets so the client only ever sees one continuous stream when being transferred. The RTCP packets are effectively discarded and created within MB, although PLIs will result in immediate creation of a PLI to send to the client.

Document Title:	Diagnosing Media Issues on the Fusion Platfo	Classification	PUBLIC	
			Page	25 of 74



Some web client support the RED codec to send most RTP packets for the video stream which wraps the original packet. If you need to tell what packets are inside, to determine which codec was picked, or the content of ULPFEC, you will need to decrypt the pcap.

When either the webRTC client or the SIP side end the call, the Gateway sends an HTTP DELETE to Media Broker, which then tears the call down down. You may see some ICMP destination unreachable messages from one endpoint briefly, ignore these. Media broker will respond with details of the call's statistics. At the same time the Gateway sends a SIP BYE or an end message to the client as necessary.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	26 of 74



# Setting Up Wireshark for Analyzing RTP Streams

# **Enable Automatic Decoding of RTP Streams**

To reduce the amount of streams you will have to manually decode, on the toolbar go to *Analyze* | *Enabled Protocols...* 



Scroll down the list to find *RTP* and check *rtp\_udp* to automatically decode RTP over UDP

🔺 👿 RTP	Real-Time Transport Protocol
rtp_rtsp	RTP over RTSP
rtp_stun	RTP over TURN
✓ rtp_udp	RTP over UDP

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	27 of 74



# **Save Useful Filters**

Saving filters can save you having to manually type in useful filters each time you are analyzing a pcap. Click the small + on the right hand side of the filter bar:



Enter a label describing the filter, and the filter required then click OK:

Label: Example	Filter: crop && Istun && Irto && Irto && Isip && Isip && Idns	ОК

You will now have a button on the right hand side of the filter bar that will automatically apply the filter you have save

			0	
Expression	+	MbRTF	Example	
		2.2		

## **Initial PCAP Analysis**

**1.** First step is to decode the rtp streams for analysis. Open the pcap in wireshark, filter on:

udp && !stun && !rtp && !rtcp && !icmp && !sip && !dns

This shows any UDP packets that are not stun, rtp, rtcp, icmp sip or dns.

**2.** Now look for any packets going from or to a known Media Broker media port (16000-16005, 17000-17099) e.g.

759 33.130881 128.136.166.108	16004 166.172.189.6	35687 UDP	142 16004	35687 Len=98
1037 34.131719 128.136.166.108	16004 156.172.189.6	35687 UDP	14 <mark>2</mark> 16004 →	35687 Len=98
1394 35.130807 128.136.166.108	16004 166.172.189.6	35687 UDP	142 16004	35687 Len=98

3. Right click the packet and select Decode As...

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	28 of 74





142 16004 → 35687		
142 16004 → 35687	Mark/Unmark Packet	Ctrl+M
142 16004 → 35687	Ignore/Unignore Packet	Ctrl+D
488 Certificate	Set/Unset Time Reference	Ctrl+T
496 Client Key Exch	Time Shift	Ctrl+Shift+T
121 Encrypted Hands		Con Sinte 1
142 16004 → 35687	Packet Comment	
118 16004 → 35687	Edit Resolved Name	
142 16004 → 35687	Eartheoonea Hanne	
142 16004 → 35687	Apply as Filter	+
142 10004 → 35687	Prepare a Filter	•
142 16004 → 35687	Conversion Filter	
118 16004 → 35687	Conversation Filter	
142 16004 → 35687	Colorize Conversation	•
134 16004 → 35687	SCTP	+
134 16004 → 35687	Follow	•
134 16004 → 35687		
110 16004 → 35687	Сору	۰.
142 16004 → 35687	Protocol Preferences	۲
C	Decode As	
	Show Packet in New Window	v

**4.** In the next window set Current to RTP and click ok

Field	Value		Туре	Default	Current	
UDP port	▼ 16004	-	Integer, base 10	(none)	(none)	•
					RSVP	^

**5.** Repeat these steps until no more packets can be seen from/to the known Media Broker ports.

6. Clear your filters by clicking the cross:

udp && Istun &&Irtp &&Irtcp&&Icmp &&Isip&&Idns

7. Now go to the Telephony menu and select RTP > RTP Stream

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	29 of 74



VoIP Calls				
GSM	•	Dst Prt	Protocol	Len
IAX2 Stream Analysis		35687	DTLSv1	1
ISUP Messages		16004	DTLSv1	e
LTE	•	35687 35687	DTLSv1	12
МТРЗ	⇒	16004	DTLSv1	e
RTP	•	CRTP S	streams	>
RTSP	•	Stream	m Analysis	

**8.** You will now see all of the RTP streams going to or from the Media Broker (it may be useful to order these by Payload as below).

Source Address	Source Port	Destination Address	Destination Port	SSRC	Payload	Packets	Lost	Max Delta (ms)	Max Jitter	Mean Jitter
192.168.9.18	16001	192.168.30.84	58150	0x73461d14	RTPType-97	146	0 (0.0%)	0.000	0.000	0.000
192.168.9.18	17088	192.168.30.84	25426	0xdb93dd8c	H264	235	0 (0.0%)	299.823	8.707	3.134
192.168.30.84	58150	192.168.9.18	16001	0xdb93dd8c	RTPType-97	250	0 (0.0%)	0.000	0.000	0.000
192.168.30.84	25426	192.168.9.18	17088	0xfed43bcf	H264	154	0 (0.0%)	30.418	3.149	0.990
192.168.9.18	16001	192.168.30.84	58150	0xee4aff77	RTPType-109	288	0 (0.0%)	0.000	0.000	0.000
192.168.30.84	20788	192.168.9.18	17040	0x1ffbec9c	opus	357	0 (0.0%)	23.902	0.850	0.419
192.168.30.84	58150	192.168.9.18	16001	0xc621d975	RTPType-109	288	0 (0.0%)	0.000	0.000	0.000
192.168.9.18	17040	192.168.30.84	20788	0xc621d975	opus	273	0 (0.0%)	43.820	3.449	0.544

**9.** The Above RTP streams show a working call, with audio and video in both directions (4 video streams, 4 audio streams). These can be used to draw an I/O graph showing the bitrates.

**10.** To draw the graph go to Statistics > I/O Graph



Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	30 of 74



Name	Display filter	Colo	Style	Y Axis	Y Field
🗸 sip in	(ip.src==10.154.176.2		Line	Bits/s	
🗸 rtc out	(ip.src==10.104.42.30		Line	Bits/s	
rtc in	(ip.src==10.154.176.2		Line	Bits/s	
sip out	(ip.src==10.104.42.30		Line	Bits/s	

11. Remove any existing filters by selecting the filter and clicking the minus::

**12.** On the RTP Streams window highlight one stream (you can highlight more than one at a time so make sure!) and click prepare filter:

192.168.9.18	17040	192.168.30.84	20788	0xc621d975 opus	273	0 (0.0%) 43.820	3.449	0.544		
8 streams, 1 selecte	ed, 273 total pac	kets. Right-click for more op	tions.				ſ	Close Find Payare		Export
							L	Close Find Revers	e Prepare Filter	Export

#### **13.** Copy the filter from the main WireShark window:

📕 tcj	odump	_cs-gclai	bon1.	рсар				-		
File	Edit	View	Go	Capture	Analyze	Statistics	Telephony	Wireless	Tools	Help
		0	0.0		۹ 👄 🖻	2 7	₺ 🗔 🗐		Q. 🎹	
	o.src==	192.168.	9.18 8	k& udp.srcp	ort==17040	&& ip.dst=	=192.168.30.84	4 && udp.ds	tport==2	20788 && rtp.ssrc==0xc621d975)

**14.** On the I/O Graph window add a new filter using the + button. Paste the copied filter into the display filter area and set the Y Axis to Bits/s. Change the Name to an appropriate description of the stream

Name	Display filter	Colo Style	Y Axis	Y Field	Smoothing	3
Audio Out SIP	(ip.src==192.168.9.18 && ud	p.srcport==17040 & Line	Bits/s		None	
+ - •	Mouse 🔘 drags 🍥 zooms	Interval 1 sec 💌	Time of day	🔲 Log s	cale	Reset
			Save As	Сору	Close	Help

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	31 of 74



## Stream Analysis has "Timestamp incorrect"

It is possible to see many "Incorrect timestamp" messages when performing a Stream Analysis on a Video Stream:

Forward (	Direction	Reversed Direc	tion					
		Analysi	ng stream from 10.10	.12.1 port 50944 to 1	0.10.13.136 port	16000 SSF	RC = 0x32164EFF	
acket 🗕	Sequence	Delta(ms)	Filtered Jitter(ms)	Skew(ms)	IP BW(kbps	Marker 4	Status	
.544	31	0.00	0.00	0.00	176.37	SET	Incorrect timestamp	
.556	32	0.00	0.00	0.00	186.37		[ Ok ]	
557	33	0.00	0.00	0.00	191.35	SET	Incorrect timestamp	
.567	34	0.00	0.00	0.00	161.35		[Ok]	
.579	36	0.00	0.00	0.00	122.62		[Ok]	
.583	37	0.00	0.00	0.00	130.29	SET	Incorrect timestamp	
.593	38	0.00	0.00	0.00	136.02		[Ok]	
.594	39	0.00	0.00	0.00	142.72	SET	Incorrect timestamp	
.607	40	0.00	0.00	0.00	147.14		[ Ok ]	
609	41	0.00	0.00	0.00	150.36	SET	Incorrect timestamp	
617	42	0.00	0.00	0.00	154.62		[ Ok ]	
.619	43	0.00	0.00	0.00	160.06	SET	Incorrect timestamp	
627	44	0.00	0.00	0.00	160.16		[ Ok ]	
628	45	0.00	0.00	0.00	163.58	SET	Incorrect timestamp	
		Max del Max jitte Max ske Total RT Duratior	ta = 0.00 ms at packet er = 0.00 ms. Mean jitti w = 0.00 ms. P packets = 5946 (exp n 214.45 s (0 ms clock o	no. 0 er = 0.00 ms. pected 5946) Lost Rī drift, corresponding t	P packets = 0 ( o 1 Hz (+0.00%)	0.00%) Sec	juence errors = 0	

Video Streams use the "Mark" attribute to indicate when a Frame ends. In a video stream, groups of packets with the same timestamp indicate that the packets belong to the same Frame. In the example below there is a sequence of 3 packets with a timestamp *10220400* and the last packet has the "Mark" attribute set. It is safe to assume that the three packets belong to the same frame.

Document Title:	Diagnosing Media I	ssues on the Fusion Platf	orm	Classification	PUBLIC
				Page	32 of 74



10	32013_235.pcap	[Wireshark 1.6.4 (SVN Re	ev 39941 from /trunk-1.6)]	* 1980	+ Jugita		* 148				_ 0 _>
Eile	Edit View Go	<u>Capture</u> <u>Analyze</u> <u>Sta</u>	atistics Telephony <u>T</u> ools Int	ernals <u>H</u> elp							
			0 0 0 0 7 2		Q Q Q	🖭   🙀 🗹 🔜 ;	% I 0	<b>B</b>			
Filter:	(ip.src==10.10.1	2.1 && udp.srcport==509	44 && ip.dst==10.10.13.136 { 💌	Expression	Clear Apply						
No.	Time	Source	Destination	Protocol Le	ngth Info						
173	87 115.04633	2 10.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3006,	Time=10205100	
173	90 115.05406	1 10.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3007,	Time=10205100	
173	91 115.05408	310.10.12.1	10.10.13.136	RTP	360 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3008,	Time=10205100,	Mark
174	02 115.13350	810.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3009,	Time=10213200	
174	03 115.13865	310.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3010,	Time=10213200	
174	04 115.13867	510.10.12.1	10.10.13.136	RTP	328 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3011,	Time=10213200,	Mark
174	16 115.21550	4 10.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3012,	Time=10220400	
174	18 115.22051	810.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3013,	Time=10220400	
174	19 115.22073	510.10.12.1	10.10.13.136		669 PT=Dy	namicRTP-Type-1	100, S	SRC=0x32164EFF,	Seq=3014,	Time=10220400,	Mark
174	30 115.28724	7 10.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3015,	Time=10227600	
174	32 115.29131	510.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3016,	Time=10227600	
174	33 115.29141	010.10.12.1	10.10.13.136	RTP	576 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3017,	Time=10227600,	Mark
174	46 115.37810	610.10.12.1	10.10.13.136	RTP 1	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3018,	Time=10235700	
174	49 115. 37857	1 10.10.12.1	10.10.13.136	RTP :	L264 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3019,	Time=10235700	
174	50 115.37860	4 10.10.12.1	10.10.13.136	RTP	272 PT=Dy	namicRTP-Type-1	100, 5	SRC=0x32164EFF,	Seq=3020,	Time=10235700,	Mark
×											- F
I Er	ame 17419: 6	69 bytes on wire	(5352 bits), 669 byte	s captured	(5352 bit	5)					
ET ET	hernet II. S	rc: MS-NLB-PhysSe	erver-01 d7:01:01:05 (0	02:01:d7:01	:01:05).	Dst: Vmware 88:4	:46:3a	(00:50:56:88:4	6:3a)		
I In	ternet Proto	col Version 4. Sr	c: 10.10.12.1 (10.10.	12.1). Dst:	10.10.13	.136 (10.10.13.)	136)	•			
I US	er Datagram	Protocol, Src Por	t: 50944 (50944), Dst	Port: 1600	0 (16000)						
E Re	al-Time Tran	sport Protocol									
	10 =	Version: RFC 1889	Version (2)								
	=	Padding: False									
	0 =	Extension: False									
	0000 =	Contributing sour	ce identifiers count:	0							
	1 =	Marker: True									
	Payload type	: DynamicRTP-Type	-100 (100)								
	Sequence num	ber: 3014									
0000	00 50 56 8	8 46 3a 02 01 d7	01 01 05 08 00 45 00	.PV.F:	E.						
0010	02 8f 7f b	d 40 00 ff 11 cc	03 0a 0a 0c 01 0a 0a	@							
0020	and the second se										
	0d 88 c7 0	0 3e 80 02 7b 66	a0 80 e4 0b c6 00 9b		T						
0030	0d 88 c7 0 f3 70 32 1 81 ba 6e f	0 3e 80 02 7b 66 5 4e ff d1 33 e3 3 2e 17 04 9c 49	a0 80 e4 0b c6 00 9b 56 74 ef c8 21 04 a9 62 ab 62 81 9e 9b 03	.p2.N3	T .Vt! Th.h						
0030	0d 88 c7 0 f3 70 32 1 81 ba 6e f 8c 21 7a 3	0 3e 80 02 7b 66 5 4e ff d1 33 e3 3 2e 17 04 9c 49 9 bb 89 34 4d 5f	a0 80 e4 0b c6 00 9b 56 74 ef c8 21 04 a9 62 ab 62 81 9e 9b 03 c7 35 71 c6 0d cc 42	.p2.N3	T Vt! Ib.b 5qB						
0030 0040 0050 0060	0d 88 c7 0 f3 70 32 1 81 ba 6e f 8c 21 7a 3 f1 e9 fa c	0 3e 80 02 7b 66 5 4e ff d1 33 e3 3 2e 17 04 9c 49 9 bb 89 34 4d 5f 4 c3 7a 27 0f 95	a0 80 e4 0b c6 00 9b 56 74 ef c8 21 04 a9 62 ab 62 81 9e 9b 03 c7 35 71 c6 0d cc 42 56 80 f2 9b 9d bb 26	.p2.N3 .n .!z94M	T Vt! Ib.b 5qB .V&						

Typically, audio codecs indicate the start of a 'audio-burst' with a "Mark" attribute. The last packet of the video frame is "Marked" and wireshark has assumed that this packet is the beginning of an audio-burst. Wireshark assumes that the timestamp is incorrect because the last packet has the same timestamp; it is not possible to start a new audio-burst with a previously used timestamp.

Essentially, these warnings can be ignored on video streams.

Document Title:	Diagnosing Media Iss	ues on the Fusion Platform	Classification	PUBLIC	
				Page	33 of 74



# Is STUN Working?

No STUN = No Media

If STUN is working correctly this rules out an issue with the media path on the WebRTC side. This can be checked inside the pcap. To start open the pcap file and set the filer to *stun* 

4		Ø	۲	010	×	C	9	۲	2	Ŷ		Ð,	Q	Ŧ	
	stun	\$													

For each WebRTC Client you should then see the following:

No.	Time	Source	Src Prt	Destination	Dst Prt	Protocol	ol Length Info	
Г	400 6.653167	192.168.30.84	58150	192.168.9.18	16001	STUN	144 Binding Request user: mN/o2Uhj:baffaa39	
	401 6.653871	192.168.9.18	16001	192.168.30.84	58150	STUN	140 Binding Request user: baffaa39:mN/o2Uhj	
	402 6.656769	192.168.30.84	58150	192.168.9.18	16001	STUN	108 Binding Success Response XOR-MAPPED-ADDRESS: 192.168.9.18:1600	2
	406 6.674098	192.168.30.84	58150	192.168.9.18	16001	STUN	144 Binding Request user: mN/o2Uhj:baffaa39	
	407 6.674862	192.168.9.18	16001	192.168.30.84	58150	STUN	108 Binding Success Response XOR-MAPPED-ADDRESS: 192.168.30.84:581	0
	408 6.674909	192.168.9.18	16001	192.168.30.84	58150	STUN	140 Binding Request user: baffaa39:mN/o2Uhj	
	410 6.683493	192.168.30.84	58150	192.168.9.18	16001	STUN	108 Binding Success Response XOR-MAPPED-ADDRESS: 192.168.9.18:1600	2
	1963 11.487112	192.168.30.84	58150	192.168.9.18	16001	STUN	140 Binding Request user: mN/o2Uhj:baffaa39	
	1964 11.488196	192.168.9.18	16001	192.168.30.84	58150	STUN	108 Binding Success Response XOR-MAPPED-ADDRESS: 192.168.30.84:581	0
	1965 11.488239	192.168.9.18	16001	192.168.30.84	58150	STUN	140 Binding Request user: baffaa39:mN/o2Uhj	
	1966 11.490476	192.168.30.84	58150	192.168.9.18	16001	STUN	108 Binding Success Response XOR-MAPPED-ADDRESS: 192.168.9.18:1600	5

Where more than one WebRTC Client is being used you may need to be more specific with filtering, by setting the source and destination port specific to each client:

stun && (udp.srcport == 58150 || udp.dstport == 58150) Where 51850 is the udp port of the current client

## No STUN in the pcap

A failed STUN setup may return no results, in this case either:

- 1. Media Broker has not received the STUN Request
- 2. Media Broker is not listening for STUN Requests

To resolve this you will need to ensure that

- 1. Media broker service is started and listening correctly
- 2. The Media Broker configuration is correct, specifically the Public and Local IPs/Ports
- 3. Firewalls on the media path have the correct ports opened and forwarding

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	34 of 74





- 4. A local firewall is not preventing communication (**See**: Local Firewall Configuration)
- 5. selinux is disabled

**Important:** Media Broker needs to send outbound STUN towards web clients. See: **Testing the Local Firewall Ports** 

Document Title:	Diagnosing Media	Issues on the Fusion Platfo	orm	Classification	PUBLIC
				Page	35 of 74



# The DTLS Handshake

As part of setting up the media paths there is a DTLS handshake. If this DTLS Handshake fails, for whatever reason, media on the WebRTC side will not work. To check this open the pcap file and set the filer to *dtls* 



Where more than one WebRTC Client is being used you may need to be more specific with filtering, by setting the source and destination port specific to each client, e.g.

*dtls* && (*udp.srcport* == 58150 || *udp.dstport* == 58150) **Note:**Where 51850 is the udp port of the current client

## Working DTLS Handshake

An example of a working DTLS handshake is:

No.	Time	Source	Src Prt	Destination	Dst Prt	Protocol	Length Info
T	403 6.658606	192.168.9.18	16001	192.168.30.84	58150	DTLSv1	. 197 Client Hello
	493 7.658398	192.168.9.18	16001	192.168.30.84	58150	DTLSv1	. 197 Client Hello
	495 7.664068	192.168.30.84	58150	192.168.9.18	16001	DTLSv1	. 717 Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
	497 7.673989	192.168.9.18	16001	192.168.30.84	58150	DTLSv1	. 614 Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Me
	499 7.686059	192.168.30.84	58150	192.168.9.18	16001	DTLSv1	. 119 Change Cipher Spec, Encrypted Handshake Message
	2757 13.464151	192.168.9.18	16001	192.168.30.84	58150	DTLSv1	. 83 Encrypted Alert

## Failed DTLS Handshakes

Examples of a failed DTLS handshake are below. This particular failure was seen, when a very old version of FCSDK was used with a newer browsers (Chrome 56+, FF 51+).

1308         6.258009         159.45.93.148         16000         50.150.80.192         59679         DTLSv1         197         Client Hello           1362         6.378998         50.150.80.192         59679         159.45.93.148         16000         DTLSv1         717         Server Hello, Certificate, Server Key Exchange, Certificate Request, Server           1364         6.381839         159.45.93.148         16000         DTLSv1         59         Alert (Level: Fatal, Description: Internal Error)           1385         6.39284         F158.98         102         59679         DTLSv1         59         Alert (Level: Fatal, Description: Internal Error)	No.	Time	Source	Src Prt	Destination	Dst Prt	Protocol	Length Info
1362         6.378998         50.150.80.192         59679         159.45.93.148         16000 DTLSvL         717 Server Hello, Certificate, Server Key Exchange, Certificate Request, Server           1364         6.381839         159.45.93.148         16000 DTLSvL         59 Alert (Level: Fatal, Description: Internal Error)           1384         6.381839         159.45.93.148         16000 DTLSvL         59 Alert (Level: Fatal, Description: Internal Error)           1384         6.381839         159.45.93.148         16000 DTLSvL         59 Alert (Level: Fatal, Description: Internal Error)	1308	6.258009	159.45.93.148	16000	50.150.80.192	59679	DTLSv1	197 Client Hello
1364 6.381839 159.45.93.148 16000 50.150.80.192 59679 DTLSv1 59 Alert (Level: Fatal, Description: Internal Error)	1362	2 6.378998	50.150.80.192	59679	159.45.93.148	16000	DTLSv1	717 Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
1295 6 430294 EA 150 90 103 E0670 150 45 02 149 16000 DTLCu1 717 Server Halls Cartificate Server Key Exchange Cartificate Request Server	1364	4 6.381839	159.45.93.148	16000	50.150.80.192	59679	DTLSv1	59 Alert (Level: Fatal, Description: Internal Error)
10000 DILSVI 717 Server herro, cercificate, server key cachange, cercificate kequest, server	1385	6.429384	50.150.80.192	59679	159.45.93.148	16000	DTLSv1	717 Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done

Or you may see the handshake enter a loop:

Document Title:	Diagnosing Media Issues on the Fusion Platfo	Classification	PUBLIC	
			Page	36 of 74


No. Time Source	Port Destination	Port Protocol	▲ Length Info
76700 255.043550 0.0.0.0	60 255,255,255,255	67 DUCP	344 DEC DISCUEL - Halsaction 10 ALSESS
70/00 204. 549555 0.0.0.0	16003 66 87 140 0	11774 pm p.d. 0	105 received a lost
2008 4.725978 108.105.1.108	10003 00.87.149.0	11774 DILSV1.0	to the ypted ster
2599 0.59/14/ 108.103.1.108	16000 108.33.234.44	05205 DTL5V1.0	105 cherypted Alert
2000 12.5234/7 100.105.1.100	10004 00.87.149.0	16003 pt pd 0	145 Chert Hello
2888 12.4121/2 108.33.234.44	51440 108.163.1.108	16002 DILSVI.0	145 Chert Hello
2890 12.412834 108.103.1.108	10002 108. 33. 234. 44	16003 pt od 0	and server metro, cerchicate, cerchicate sequest, server metro bone
2909 12.430101 100.33.234.44	16002 108 22 224 44	10002 DTLSV1.0	1016 Certificate, Cherit Key Exchange, Certificate Veriny, Change Cipher Spec, Cherit Herio
2910 12.408521 108.105.1.108	10002 108.33.234.44	31440 011391.0	15) change cipitel spec, encrypted natustake Message
3040 13.323023 108.103.1.108	10004 00.87.149.0	26240 DTLSV1.0	for community wills contificate contificate Demont Community will Demo
30/3 13.403204 00.0/.149.0	16004 66 87 140 0	26240 ptt ptd 0	135 contributes and the second s
3002 13.493334 108.103.1.108	10004 00.87.149.0	20240 011591.0	12/0 cercificate, chelc key Exchange, cercificate verify, change cripier spec, encrypted ha
3090 13.3/2/09 00.8/.149.0	16004 66 87 140 0	16004 DTLSV1.0	105 Change Ciprier Spec, Encrypted Handshake Message
3336/ 32.710/6/ 108.103.1.108	10004 00.87.149.0	20240 DILSVI.0	to sharped alert
33594 52.713767 108.163.1.108	10002 108.33.234.44	51440 DILSVI.0	tos encrypted Alert
34499 97.401000 108.33.234.44	16002 108 22 224 44	10002 DTLSV1.0	145 CHERCHEHO
54500 97.401399 108.105.1.108	10002 108.55.254.44	01467 DTL5V1.0	ado se ver nerio, cerciricate, cerciricate kequest, se ver nerio done
34503 97.441023 108.33.234.44	6148/ 108.163.1.108	10002 DTLSV1.0	1018 Certificate, Chent Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Har
34534 97.430781 108.103.1.108	16002 108.33.234.44	55000 pt od 0	100 change ciprier spec, Encrypted nariosnake message
54321 97.745253 108.105.1.108	10004 108.55.254.44	53099 DILSVI.0	195 Client mello
34/8/ 98./43364 108.163.1.108	10004 108.33.234.44	55099 DTLSV1.0	195 Crient Hello
34833 98.93/842 108.33.234.44	55099 108.105.1.108	10004 DTLSV1.0	oss server herro, cercificate, cercificate Request, server herro bone
34840 98.948/19 108.163.1.108	16004 108.33.234.44	55099 DILSVI.0	12/6 Certificate, Client Key Exchange, Certificate Verity, Change Cipner Spec, Encrypted Har
54882 99.109257 108.55.254.44	35099 108.103.1.108	10004 DTLSV1.0	135 Change Ciprier Spec, Encrypted Handshake Message
58850 130 273202 108 162 1 108	16002 108.33.234.44	55000 ptt pd 0	105 Encrypted Alert
58850 130.2/2303 108.163.1.108	10004 108.33.234.44	55099 DILSVI.0	tos encrypted Alert
60008 207.293847 47.199.109.238	16002 47 100 160 230	10002 DTLSV1.0	145 Crient Mello
60099 207.294401 108.165.1.108	10002 47.199.109.258	55517 DILSVI.0	ado server nerio, cercificate, cercificate Request, server nerio bune
60101 207.345540 47.199.169.238	5551/ 108.163.1.108	10002 DILSVI.0	1018 Certificate, Chent Key Exchange, Certificate Verity, Change Cipner Spec, Encrypted Har
60157 208.032061 108.163.1.108	10004 100.1/2.190.55	16003 pt o.d. 0	195 Crient Merro
00100 208.539134 47.199.109.238	3331/ 108.103.1.108	10002 011591.0	Tota cercificate, cirent key Exchange, cercificate verify, change cipiter spec, Encrypted ha
60183 209.032194 108.163.1.108	10004 100.1/2.190.55	3025 DILSVI.0	195 Chent Hello
60185 209.104911 108.1/2.190.55	3023 108.103.1.108	10004 DTLSV1.0	135 Server Merto, Cercificate, Cercificate Request, Server Merto Done
00100 209.11498/ 108.105.1.108	10004 100.1/2.190.33	5025 DILSV1.0	1270 Certificate, Chent Key Exchange, Certificate verify, Change Cipher Spec, Encrypted ha
60188 209.192218 166.172.190.55	3025 108.163.1.108	16004 DILSVI.0	135 Change Cipher Spec, Encrypted Handshake Message
00430 210.3342/647.199.109.238	55517 108.165.1.108	10002 DTLSV1.0	1018 Cercificate, crient key Exchange, cercificate verify, change cipner spec, Encrypted har
01923 214.339127 47.199.109.238	55517 108.105.1.108	10002 011591.0	409 Cer Chinate
01920 214. 35924/ 47. 199. 169. 238	55517 108.163.1.108	16002 DTLSV1.0	490 Chient Key Exchange, Certificate Verify, Change Cipher Spec
65004 222 261256 47 100 160 228	55517 108.163.1.108	16002 DTLSV1.0	121 ENCTYPLED TATUSTAKE MESSAGE
65004 222.361/36 47.199.169.238	5551/ 108.163.1.108	16002 DILSVI.0	489 Certificate
65005 222.30167147.199.109.238	55517 108.103.1.108	10002 DTLSV1.0	10 Crenc key exchange, contribute verify, Change Cipher Spec
21633 338 364556 47, 199, 169, 238	55517 108.163.1.108	16002 DTLSV1.0	121 ENCTYPLED TATUSTAKE MESSage
/1025 256.504330 47.199.109.238	55517 106.163.1.108	10002 DILSVI.0	409 Cer Ci i Cate
/1024 238.304/13 4/.199.169.238	55517 108.163.1.108	16002 DTLSV1.0	490 Chient Key Exchange, Certificate Verify, Change Cipner Spec
71020 200.004/04 47.199.109.208	16004 166 173 100 55	2002 DILSVI.0	121 Encrypted nanosnake message
70471 249.044932 108.103.1.108	10004 100.1/2.190.55	5025 DILSVI.0	to the pred ster.
/04/9 249.851525 108.163.1.108	10002 47.199.169.238	3331/ UTLSV1.0	105 ERCHOPLEG ALERC
51 0.119110 127.0.0.1	91120 127.0.0.1	57065 HTTP	200 GET /Statistics mit/1.1
32 0.1135/0 12/.0.0.1	5/063 12/.0.0.1	41150 HITP	794 HTTP/1.1 200 OK (CERC/PTATI)

# **Drawing the Call Flow Sequence**

Drawing out the call flow sequence can be useful when analysing complicated calls. In the call flow you can record the relevant pieces of the SIP message and SDP.

### SIP Side with Wireshark (from FAS)

The SIP call flow must be taken from the FAS. The Media Broker pcaps do not contain SIP messages. However single box installs will contain both RTP and SIP dialogs. Open the pcap in wireshark and decode all the rtp streams. Then on the toolbar select *Telephony* | *Sip Flows:* 

Document Title:	Diagnosing Media Issues on the Fusion Platform			Classification	PUBLIC
				Page	37 of 74







You will need to find the correct call. Complete calls that were ended will show a state of COMPLETED:



Highlight the appropriate call then click *Flow Sequence* in the bottom right corner:



This will draw the complete call flow as seen from FAS.

The image below is taken from a single box install so shows both SIP messages and the RTP streams, note that the SIP messages are between 192.168.9.18 (FAS) and 10.10.30 (a PBX).

Note:

- You cannot see what PBX has sent to a SIP client side from here.
- Nor can you see the SDP sent to the webRTC client

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	38 of 74

# cafex.com



	192.16	58.9.18 10.10	. 10. 30	192.168.30.84	
4.420895	58675	INVITE SDP (o.	5060		SIP INVITE From: "User1" <sip:1001@cs-gclabon< th=""></sip:1001@cs-gclabon<>
4.425983	58675	100 Trying	5060		SIP Status 100 Trying
4.458003	58675	180 Ringing	5060		SIP Status 180 Ringing
6.290687	58675	200 OK SDP (o	5060		SIP Status 200 OK
6.318604	58675	ACK	5060		SIP Request INVITE ACK 200 CSeq:1
6.423659	17040	RTP	(opus)	20788	RTP; 357 packets. Duration: 4294960.873s SSRC:
8.001930	17040	RTP	(opus)	20788	RTP, 273 packets. Duration: 4294959.295s SSRC:
8.339938	17088	RTP (	H264)	25426	RTP, 154 packets. Duration: 4294958.957s SSRC:
8.351523	17088	RTP (	H264)	25426	RTP, 235 packets, Duration: 4294958.945s SSRC:
9.395154	58675	INFO	5060		SIP INFO From: "User1" <sip:1001@cs-gclabon1< td=""></sip:1001@cs-gclabon1<>
9.396891	58675	200 OK	5060		SIP Status 200 OK
13.467908	58675	BYE	5060		SIP Request BYE CSeq:3
13.470637	58675	200 OK	5060		SIP Status 200 OK

As most installs are not single box you will need to click the relevant SIP message, this will move wireshark's display to the correct packet, where you can view the SIP SDP (SIP INVITE in the case below:

File Edit View Go Capture Analyze Statistics lelephony Wireless	loois Help	102.152.0.10	
∡ ■ ⊴ ⊛ 🌗 🖹 🗙 🖬 ۹ ⇔ ⇔ 🕾 🖲 💂 🚍 🔍 ۹	Q. 11	192.168.9.18 192.168.30.84 10.10.10.30	
Apply a display filter <ctri-></ctri->	Expression + MbRTP Example	4.420895 58675 ENVITE SDP.( 5060	SIP INVITE From: "User1" <sip:1001@cs-gclabon< th=""></sip:1001@cs-gclabon<>
Source Src Prt Destination Dst Prt Protocol Length Info	Frame 213: 2053 bytes on wire (16424 bits), 2053 b	4,425983 58675 5060	SIP Status 100 Trying
127.0 43368 127.0.0.1 48197 TCP 348 [TCP segment.	Einux cooked capture	4.458003 58675 100 Ringing 5060	SIP Status 180 Ringing
42:1e: ARP 62 Who has 192	Internet Protocol Version 4, Src: 192.168.9.18, Ds	6,290687 58675 200 OK SDP ( 5060	SIP Status 200 OK
a2:b5: ARP 62 Who has 192	Transmission Control Protocol, Src Port: 58675 (58)	6 318604 secre ACK seco	SID Request INVITE ACK 200 CSet 1
127.0 48197 127.0.0.1 43368 TCP 68 48197 → 4336.	<ul> <li>Session Initiation Protocol (INVITE)</li> </ul>	5 473650 13040 RTP (bpus) 20399	RTD 357 nariute Duration: 4294960.873e 558C
127.0 43368 127.0.0.1 48197 HTTP 2851 POST /sdp/52.	Request-Line: INVITE sip:1054@10.10.10.30 SIP/2	RTP (sous)	DTD 3TD endow Devolute 100000 305 CC00
127.0 48197 127.0.0.1 43368 TCP 68 48197 → 4336.	Message Header	8.001930 1/040 PTD (4/364)	RTR 275 peckets. Duration: 4294959.2556 55RC2
127.0 48197 127.0.0.1 43368 HTTP 1816 HTTP/1.1 201.	Message Body	8.339938 17088	RTB 154 packets. Duration: 4294958.957s 55RC:
127.0 43368 127.0.0.1 48197 TCP 68 43368 → 4819.	Session Description Protocol	8.351523 17088 KTP (8264) 25426	RTB 235 packets. Duration: 4294958-945s SSRC:
192.16 8092 192.168.9 54687 HTTP 1816 HTTP/1.1 201.	Session Description Protocol Version (v):	9.395154 58675 INFO = 5060	SIP INFO From: "User1" <sip:1001@cs-gclabon1< th=""></sip:1001@cs-gclabon1<>
192.16 54687 192.168.9 8092 TCP 68 54687 → 8092	Dwner/Creator, Session Id (o): - 910821064	9.396891 58675 200 OK 5060	SIP Status 200 OK
192.16 41972 192.168.2 53 DNS 83 Standard que.	Session Name (s): -	13.467908 \$8675 BYE \$960	SIP Request BYE CSeg:3
192.16 53 192.168.9 41972 DNS 99 Standard que.	Time Description, active time (t): 0 0	13 470/27 Parts 200 OK	FID Forte 200 OK
192.16 44129 192.168.2 53 DNS 87 Standard que.	Media Description, name and address (m): a	13.470037 306/3 4 5060	SIP SURVEY ON OK
192.16 53 192.168.9 44129 DNS 122 Standard que.	Connection Information (c): IN IP4 192.168		
192.16 34075 192.168.2 53 DNS 83 Standard que.	Media Attribute (a): rtpmap:109 opus/48000		
192.16 53 192.168.9 34075 DNS 99 Standard que.	Media Attribute (a): fmtp:109 maxplayback		
192 16 45700 230 0 0 4 45700 UDP 176 45700 + 4570	Media Attribute (a): rtpmap:0 PCMU/8000		
192.16 58675 10.10.10 5060 SIP/SDP 2053 Request: INV.	Media Attribute (a): rtpmap:8 PCMA/8000		
10.10 5000 192.108.9 580/5 ILP 08 5000 - 500/5	Media Attribute (a): rtpmap:18 G729/8000		
10.10 5060 192.168.9 58675 SIP 502 Status: 100	Media Attribute (a): rtpmap:96 telephone-c		
192 16 58675 10 10 10 5060 TCP 68 58675 + 5060	nette neetbace (a), senireer		

When you are more familiar with the call flow, you can read the inbound Offers and Answers reading the Media Broker sdp.log

Document Title:	Diagnosing Media Issues on the Fusion Platfo	Classification	PUBLIC	
			Page	39 of 74



# **Advanced Packet Capture Analysis**

## **Measuring Bandwidth**

Currently, the easiest way to measure actual Bandwidth usage is to use Wireshark's **IO Graph** which can be found under *Statistics*.

### 1) Find the appropriate Filter for a stream:

	Dete	cted 19 RTP stream	ns. Choose one f	or forward and rev	erse direction for	analysis	
Src IP addr	Src port	Dst IP addr	Dst port	<ul> <li>SSRC</li> </ul>	Payload	<ul> <li>Packets</li> </ul>	Lost
10.20.12.27	16810	10.16.116.53	16728	0x529B3C65	RTPType-97	19364	0 (0.0%)
10.16.116.53	16728	10.20.12.27	16810	0x55304119	RTPType-97	12211	10 (0.1%)
10.10.13.136	16000	10.10.12.1	62910	0x4FF7B2F5	RTPType-100	13891	-55 (-0.4%)
10.10.12.1	62910	10.10.13.136	16000	0xFF77B2DC	RTPType-100	13496	-112 (-0.8%)
∢ [							۲
		Forward: 10.20.1 Select a	2.27:16810 -> 10 reverse stream v	0.16.116.53:16728, with Ctrl + left mo	SSRC=0x529B3C65 use button		
Unselect	Find Revers	e Save As	Mark Packet	s Prepare Filter	Copy	Analyze	Close

2) Copy the Filter from the Main Wireshark window:

Document Title:	Diagnosing Media Issues on the Fusion Platfo	Classification	PUBLIC	
			Page	40 of 74



tsh	ark.pcap [Wireshark 1.6.4 (SVN Rev 39941 from /trunk-1.6)]	
<u>F</u> ile	<u>Edit V</u> iew <u>G</u> o <u>C</u> apture <u>A</u> nalyze <u>S</u> tatistics Telephony <u>T</u> ools <u>I</u> nternals <u>H</u> elp	
	H & M M   E 🖬 X 2 L   Q + + A 7 L   E E   Q Q 0 E   H M M M S	
Filter	: dst==10.16.116.53 && udp.dstport==16728 && rtp.ssrc==0x529B3C65) 🗨 Expression Clear Apply	
No. Fr Li US Ne	Time         Source         Destination         Protocol         Length         srcport         destport         New Column           1         1389212646.981133000         10.10.12.225         10.10.13.255         NBNS         94         netbio netbios           2         1389212646.288029000         F5Networ_ec:ae:06         ARP         62           4         1389212646.499974000         F5Networ_ec:ae:06         ARP         62           4         1389212646.705177000         F5Networ_ec:ae:06         ARP         62           ""           ame 1: 94 bytes on wire (752 bits), 94 bytes captured (752 bits)           nux cooked capture           ternet Protocol Version 4, Src: 10.10.12.225 (10.10.12.225), Dst: 10.10.13.255 (10.10.13.255)           er Datagram Protocol, Src Port: netbios-ns (137), Dst Port: netbios-ns (137)           tBIOS Name Service	Info Anatoria and

### 3) Paste the Filter into the IO Graph and set the Y Axis to Bits per Second



We can see the stream is running at approximately 500kbits<sup>-1</sup>

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	41 of 74



## Analysis of an SIP audio stream

				Detected 10 KT	streams. Choose	one for forwar	d and reverse dir	ection for analysis			
Src IP addr	<ul> <li>Src port</li> </ul>	Dst IP addr	<ul> <li>Dst port</li> </ul>	<ul> <li>SSRC</li> </ul>	Payload	<ul> <li>Packets</li> </ul>	Lost	Max Delta (ms)	<ul> <li>Max Jitter (ms)</li> </ul>	<ul> <li>Mean Jitter (ms)</li> </ul>	Pb?
0.10.12.1	57840	10.10.13.136	16000	0xB39E333E	g711U	4181	41 (1.0%)	745.68	57.86	25.63	х
0.10.13.136	16000	10.10.12.1	57840	0x516F4E43	g711U	4114	10 (0.2%)	56.99	11.54	7.61	X
0.16.116.53	16840	10.20.12.27	16624	0x98F1A409	g711U	4142	0 (0.0%)	21.78	0.39	0.13	
0.20.12.27	16624	10.16.116.53	16822	0xC3916941	g711U	4	0 (0.0%)	0.02	3.52	14.99	
0.20.12.27	16624	10.16.116.53	16840	0xCF139E0A	g711U	1918	10 (0.5%)	159.79	276.67	7.80	х
0.10.12.1	57840	10.10.13.136	16000	0xD94ACE43	Reserved for RT	CP 1	0 (0.0%)	0.00	0.00	0.00	
0.10.12.1	57840	10.10.13.136	16000	0x3B6155CA	RTPType-100	13359	-463 (-3.6%)	0.00	0.00	0.00	х
0.10.13.136	16000	10.10.12.1	57840	0x11E585B9	RTPType-100	16835	-11088 (-192.9	%) 0.00	0.00	0.00	Х
0.16.116.53	16866	10.20.12.27	16590	0x17FEA6DD	RTPType-97	5009	1 (0.0%)	0.00	0.00	0.00	х
0.20.12.27	16590	10.16.116.53	16866	0x7E30B814	RTPType-97	7588	0 (0.0%)	0.00	0.00	0.00	x
				Se	lect a forward strea Select a reverse str	am with left mo eam with Ctrl +	ouse button, and • left mouse butt	then on			
					Unselect	Find Reverse	Save As	Mark Packets	Prepare Filter C	opy Analyze	Close

The audio from Media Broker to the SIP phone is not encrypted and can be heard using wireshark's RTP Player, by Selecting the appropriate Stream and pressing *Analyze* then selecting *Player*. The stream needs to be decoded, for now select a large Jitter Buffer (200ms).

*	P	Þ		w w w w vb	w wear ware	a of opposite and	
+++				<b>1</b>			
5 s	36 5	37 5	38 s	39 5	40 s	41 5	42 5
From 10.	20.12.27:16624 to :	10.16.116.53:16840	Duration:86.41 Dr	op by Jitter Buff:103(	5.4%) Out of Sec	: 5(0.3%) Wrong	Timestamp: 179(9.3
			🔄 Vie	w as time of day			
tter buffer (	[ms] 200 📮 🛽	Use RTP timestam	Decode	Play	P <u>a</u> use	Stop	Close

The Stream is clearly disrupted with sequence errors from 36s onwards.

## **Stream Analysis**

Wireshark provides some Stream Analysis, which is helpful for audio diagnosis.

In this example, wireshark shows clumping on the inbound g711 stream. Bursts of

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	42 of 74





inbound packets arrive in large groups and later than expected:

orward	Direction	Reversed Dir	ection				
		Analysin	g stream from 10.1	0.12.1 port 57840 to	10.10.13.136 port	16000 SSRC = 0xB39E333E	
acket •	Sequence	Delta(ms)	Filtered Jitter(n     25.01	ns] • Skew(ms) -209.02	IP BW(kbps	Marker Status	•
8206	20598	0.00	23.57	-249.82	78.96	[ Ok ]	
8207	20599	0.00	23.35	-229.83	80.64	[Ok]	
8208	20600	4.46	22.86	-214.29	82.32	[ Ok ]	
8211	20601	0.07	22.68	-194.36	84.00	[Ok]	
8222	20602	25.93	21.63	-200.29	85.68	[Ok]	
8224	20603	7.88	21.04	-188.17	87.36	[Ok]	
8231	20604	17.71	19.87	-185.88	89.04	[Ok]	
8300	20605	154.60	27.04	-320.48	73.92	[ Ok ]	
8301	20606	0.02	26.60	-300.50	75.60	[ Ok ]	
8302	20607	0.00	26.18	-280.50	77.28	[Ok]	
8303	20608	0.00	25.80	-260.50	78.96	[Ok]	
8304	20609	0.00	25.43	-240.51	80.64	[Ok]	
8305	20610	0.01	25.09	-220.52	82.32	[Ok]	
8306	20611	0.07	24.77	-200.59	84.00	[Ok]	
8307	20612	0.02	24.47	-180.61	85.68	[Ok]	
8313	20613	23.36	23.15	-183.97	85.68	[Ok]	
8322	20614	19.81	21.72	-183.78	84.00	[Ok]	
8335	20615	20.69	20.40	-184.46	85.68	[Ok]	
8360	20616	19.46	19.16	-183.92	85.68	[ Ok ]	
8461	20617	121.24	24.29	-285.17	75.60	[Ok]	
8510	20618	37.72	23.88	-302.89	73.92	[Ok]	
8512	20619	0.12	23.63	-283.01	75.60	[Ok]	
8518	20620	4.05	23.15	-267.05	77.28	[Ok]	
8519	20621	0.02	22.95	-247.07	78.96	[Ok]	
8520	20622	0.00	22.77	-227.08	80.64	[Ok]	
8521	20623	0.00	22.59	-207.08	82.32	[Ok]	
8523	20624	0.06	22.43	-187.14	84.00	[ Ok ]	
8540	20625	15.87	21.28	-183.01	85.68	[Ok]	
8547	20626	28.66	20.50	-191.68	84.00	[Ok]	
		Max delta Max jitter : Max skew Total RTP Duration 8	= 745.68 ms at pac = 57.86 ms. Mean ji = -1106.34 ms. packets = 4222 (e: 8.50 s (-778 ms clo	ket no. 39012 tter = 25.63 ms. (pected 4222) Lost F ck drift, correspondir	RTP packets = 41 (0 ng to 7930 Hz (-0.8	197%) Sequence errors = 4 8%)	

G711 packets should arrive every 20ms. The Delta is the time between packets and the example above shows that 18301 to 18313 have all arrived at the same time at packet 18300 which recorded a Delta of 154ms. Also, the skew is large, this indicates that the audio packet has arrived ~180-290ms late, relative to their expected packet arrival time.

As a comparison, the inbound stream from the SIP phone; which is considered good; has a consistent 20ms Delta, a Jitter value of almost 0 and a Skew of only 3ms.

## H.264 Codec

The first keyframe contains will typically contain the Sequence Parameter Set which contains information common to all the pictures in the H264 stream.

This will contain information like the H.264 profile being used that should match the SDP negotiation:

Document Title:	Diagnosing Media Issues on the Fusi	Classification	PUBLIC	
			Page	43 of 74



### H.264 Profiles:

profile-level-id define the properties of the incoming H.264 stream. It indicates the profile level the decoder must comply to in order to decode the incoming NAL unit stream.

It is a Base16 representation of 3 bytes in the SPS of NAL unit. 1 byte - profile\_idc 1 byte - profile\_iop 1 byte - level\_idc

for example, profile-level-id=42E015 imply profile\_idc = 42 imply Baseline profile profile\_iop = E0 imply only common subset of profile is supported level\_idc = 15 imply level 2.1

In general, profile-level-id and packetization-mode identify the media format configuration for H.264

See RFC 3984 Section 8.1 for details

Check <u>http://en.wikipedia.org/wiki/H.264/MPEG-4\_AVC#Levels</u> for more details on specific levels.

The SPS can be used to verify what an H264 stream contains: <u>https://cardinalpeak.com/blog/the-h-264-sequence-parameter-set/</u>

### H264 Decoding

If you know an unencrypted/decrypted stream is H264, but wireshark isn't showing it as such, go to preferences->protocols->h264 and set the payload type to the one for the h264 stream. If multiple are needed then it can take a comma separated list (no spaces).

Document rule. Diagno	ang Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	44 of 74





## Finding the Sequence Parameter Set

- 🗖	8095	4839280649	3501_sip_se	nds_H264_	to_mb_35	86981564.pc	ap [Wiresha	rk 1.6.4 (SVN	Rev 39941 fr	om /trunk-1.6)]								
Eile	Edit	t <u>V</u> iew <u>G</u>	<u>Capture</u>	Analyze	Statistic	s Telephor	y Iools I	ternals <u>H</u> e	р									
		94 94 9	1 🖻 🖥	1 × 2	81	2, 4 4	• 7 1		] @ Q	0, 🖸   🌌	1 🗹 🕵 %	Ħ						
Filte	n 🗌							Expression	Clear Ap	iply								
No.	Tin	ne	Source		srcport	Destination	destp	ort Protoco	Length N	[PTimestamp	Info							
	0.	00000000	0 150.11	0.114.2	1 58016	153.40.	8.101 170	16 H264	67	3325	8 PT=Dynamic	RTP-Туре-96	, SSRC=0xD5CCFEBC	, seq=33258	Time=341742664	NAL uni	t - Sequence	parameter set
	3 0	00394300	150.11	0 114 2	1 58016	153.40.	8 101 170	6 H264	1370	3326	0 PT=Dynamic	PTP-Type-96	SSPC=0xDSCCFEBC	, Seq=33260	Time=341742004	NAL UDI	t - Coded sli	ce of an TDP n
	1 0.	01373300	150.11	0.114.2	1 58016	153.40.	8.101 170	6 H264	1280	3326	1 PT=Dynamic	RTP-Type-96	, SSRC=0xD5CCFEBC	, Seq=33261.	Time=341742664	NAL uni	t - Coded sli	ce of an IDR p
	5 0.	03901800	150.11	0.114.2	1 58016	153.40.	8.101 170	6 H264	599	3326	2 PT-Dynamic	<b>RTP-Туре-96</b>	, SSRC=0xD5CCFEBC	, seq-33262	Time=341742664	Mark N	AL unit - Coo	led slice of an
	5 0.	06831300	0 150.11	0.114.2	1 58016	153.40.	8.101 170	6 H264	212	3326	3 PT=Dynamic	RTP-Туре-96	, SSRC=0xD5CCFEBC	, Seq=33263	Time=341748670	Mark N	AL unit - Coo	led slice of a
	0.	10145600	0 150.11	0.114.2	1 58016	153.40.	8.101170	6 H264	64	3326	4 PT=Dynamic	RTP-Type-96	, SSRC=0xD5CCFEBC	, Seq=33264,	Time=341751673	Mark N	AL unit - Coo	led slice of a
	30.	16984700	150.11	0.114.2	1 58016	153.40.	8.101 170	6 H264	64	3320	6 PT=Dynamic	RTP-Type-96	SSRC=0xDSCCFEBC	, Seq=33205, Seq=33266	Time=341757679	Mark N	AL UNIT - COC	ied slice of a
1	0.	20306200	150.11	0.114.2	1 58016	153.40.	8,101 170	6 H264	64	3326	7 PT=Dynamic	RTP-Type-96	SSRC=0xD5CCFEBC	, Seg=33267	Time=341760682	Mark N	AL unit - Coo	led slice of a
1	L 0.	23626400	0 150.11	0.114.2	1 58016	153.40.	8.101 170	6 H264	64	3326	8 PT=Dynamic	RTP-Type-96	, SSRC=0xD5CCFEBC	, Seq=33268	Time=341763685	Mark N	AL unit - Coo	led slice of a
1	2 0.	26948400	0 150.11	0.114.2	1 58016	153.40.	8.101 1703	6 H264	64	3326	9 PT-Dynamic	RTP-Туре-96	, SSRC=0xD5CCFEBC	, seq=33269,	Time=341766688	Mark N	AL unit - Coo	led slice of a
1	3 0.	30265400	0 150.11	0.114.2	1 58016	153.40.	8.101 170	6 H264	64	3327	0 PT=Dynamic	RTP-Туре-96	, SSRC=0xD5CCFEBC	, Seq=33270	Time=341769691	Mark N	AL unit - Coo	led slice of a
1	10.	33589000	0 150.11	0.114.2	1 58016	153.40.	8.101170	16 H264	64	3327	1 PT=Dynamic	RTP-Туре-96	, SSRC=0xD5CCFEBC	, Seq=33271,	T1me=341772694	Mark N	AL unit - Coo	ied slice of a
< L	-																	
€ F	ame	e 1: 67 b	ytes on	wire (5	536 bit:	s), 67 by	tes capti	ired (536	bits)									
E E	inux	cooked	capture															
	ter	Datagram	Protocc	sion 4,	Port:	58016 (5)	R016) DS1	Port: 1	7036 (170)	1: 153.40. 36)	8.101 (153.4	0.8.101)						
E R	pal-	Time Tra	nsport P	Protocol	FOILT	JUOTO (J	010), 03	. FOIL. 1	(050 (1/0.	,0,								
E H	264																	
0000	04	0 00 00 5 00 00	01 00 06	00 1a	e2 65 37 11	d1 00 00 0b 77 96	00 08 00 6e 72 19	E. 3.		i.								1
0020	9	9 28 08	55 e2 a0	42 8c	00 1f	00 00 80	60 81 ea	. (.e.	B									E
0030	1 6	4 5e 94	18 d5 cc	fe bc	27 42	a0 14 95	a0 20 09	i^. H	'В									-
0040	. 0	C 04 02																
U F	P	ackets: 2830	Displayed: 2	830 Marke	d: 0 Load t	ime: 0:00.162	2											

### Audio & Video Analysis

Typically all the packets that create a frame are sent around the same time, so the deltas in video streams are not comparable to those of audio streams.

- 1) Select the Stream to Analyze and Prepare a Filter
- 2) Alter the filter to only include Marked Packets append: && rtp.marker==true
- 3) Apply the filter
- 4) Save the Displayed Packets as a CSV and Open in Excel (or equivalent).

Document Title:	Diagnosing Media I	ssues on the Fusion Plat	orm	Classification	PUBLIC
				Page	45 of 74





Save in:	Desktop			• • • •	<b></b>	
Recent Places	Librario System	<b>is</b> Folder			Â	
Desktop	thill System	Folder			E	
Libraries	Comp System	u <b>ter</b> Folder				
Computer	Netwo System	<b>rk</b> I Folder				
Network	iperf File fol	der				
	no? fil	ec			-	
Fi	e name:	test.csv		-	Save	
Sa Packet Range	ave as type:	CSV (Comma	Separated Valu	Packet Format	Cancel Help	
All nackate		102477	6018	M Packet sur	mary line	
C Selected packet	*	1	1	IM. Packet det	ans.	
C Marked packel	8		0	As display	au 💌	
C First to last mar	ked	0	0	Packet Byt	es	
C Range:		0	0	Each pack	et on a new page	
Remove Ignore	ed packets	0	0			

5) Add a New Column "Frame Deltas"

6) The Delta is equal to the Difference in Time\*1000 from the previous packet and the current packet

7) This can be used to identify unusual 'frame' behaviour.

Document Title:	Diagnosing Media	ssues on the Fusion Platfo	Classification	PUBLIC	
				Page	46 of 74



	19-0	u +   <del>-</del>	(Constant)	100 10	1000		100 m	MCU_to_MB	KEYFRAME	S - Microso	oft Exc	el	-	10-1		( interior			X
File	Ho	me Insei	rt Page	Layout	Formulas D	ata F	Review 1	√iew										ے 🚷 ہ	- @ 23
	*	Calibri	* 11	· A A	- = = -		📑 Wra	p Text	General		•					<b>X</b>	Σ	A7 A	
Dasta	<b>•</b>			. A A			_				00	Candi	S I		Incod		. <b>.</b> .	Zu uu	
- Paste	<b>V</b>	<b>B</b> 7 <u>N</u>	•	A - A		12 3	Mer	ge & Center 🕚		• • .00	.00	Forma	tting * as Tab	le * Styles *	, v	y y	at Q*	Filter * Select *	
Clipbo	ard 🗔		Font		6	Align	nment	r	ā Nu	umber	15		Styles			Cells		Editing	
	R3		. (=	<i>f</i> <sub>x</sub> =B3*	*1000-B2*100	D													*
4	D	E	F	G	н	1	J	K	L	M		N	0	P	Q	R	S	Т	U
1 sr	cport	Destinatic	destport	Protocol	Length N	TPTimes	Info									1			
2	58016	153.40.8.1	17036	RTP	599	33262	PT=Dynam	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3262, Time=	341742664	4, Mark	0			
3	58016	153.40.8.1	17036	RTP	212	33263	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3263, Time=	341748670	), Mark	29.295			
4	58016	153.40.8.1	17036	RTP	64	33264	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3264, Time=	341751673	3, Mark	33.143			
5	58016	153.40.8.1	17036	RTP	64	33265	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3265, Time=	341754676	5, Mark	35.183			
6	58016	153.40.8.1	17036	RTP	64	33266	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3266, Time=	341757679	, Mark	33.208			
7	58016	153.40.8.1	17036	RTP	64	33267	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3267, Time=	341760682	2, Mark	33.215			
8	58016	153.40.8.1	17036	RTP	64	33268	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3268, Time=	341763685	5, Mark	33.202			
9	58016	153.40.8.1	17036	RTP	64	33269	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3269, Time=	341766688	B, Mark	33.22			
10	58016	153.40.8.1	17036	RTP	64	33270	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3270, Time=	341769691	I, Mark	33.17			
11	58016	153.40.8.1	17036	RTP	64	33271	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3271, Time=	341772694	4, Mark	33.236			
12	58016	153.40.8.1	17036	RTP	200	-										3			
13	58016	153.40.8.1	17036	RTP	100											5			
14	58016	153.40.8.1	17036	RTP	180	*										5			
15	58016	153.40.8.1	17036	RTP	160	0										- 7			
16	58016	153.40.8.1	17036	RTP	140	<b>.</b>										L			
17	58016	153.40.8.1	17036	RTP	140	12										9			
18	58016	153.40.8.1	17036	RTP	120	0										- 3			
19	58016	153.40.8.1	17036	RTP	100		-									5			
20	58016	153.40.8.1	17036	RTP	100	-				+		+				5			
21	58016	153.40.8.1	17036	RTP	80		and and					1		6		1			
22	58016	153.40.8.1	17036	RTP	60	- <b>-</b>		4 At 1 1						. =		- 1			
23	58016	153.40.8.1	17036	RTP	196		16 A A		1 × 1 × 1			1. N		×6		2			
24	58016	153.40.8.1	17036	RTP	40			+								3			
25	58016	153.40.8.1	17036	RTP	20		0.00	28								_ 2			
26	58016	153.40.8.1	17036	RTP	-											L			
27	58016	153.40.8.1	17036	RTP	0	0	200	<b>.</b>	400	600			800	1000	1	200 2			
28	58016	153.40.8.1	17036	RTP	04	33288	PT=Dynan	псктр-тура	2-90, SSKC	=UXDSCCFI	ЕВС, З	seq=3	5288, rime=	341823743	, iviark	33.172			
29	58016	153.40.8.1	17036	RTP	64	33289	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3289, Time=	341826748	8, Mark	35.175			
30	58016	153.40.8.1	17036	RTP	64	33290	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3290, Time=	341829751	1, Mark	33.227			
31	58016	153.40.8.1	17036	RTP	64	33291	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3291, Time=	341832754	1, Mark	31.235			
32	58016	153.40.8.1	17036	RTP	64	33292	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCFI	EBC, S	Seq=33	3292, Time=	341835757	7, Mark	35.177			
33	58016	153.40.8.1	17036	RTP	64	33293	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3293, Time=	341838760	D, Mark	33.182			
34	58016	153.40.8.1	17036	RTP	64	33294	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3294, Time=	341841763	3, Mark	33.228			
35	58016	153.40.8.1	17036	RTP	64	33295	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3295, Time=	341844766	5, Mark	33.179			
36	58016	153.40.8.1	17036	RTP	64	33296	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3296, Time=	341847769	9, Mark	33.238			
37	58016	153.40.8.1	17036	RTP	64	33297	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3297, Time=	341850772	2, Mark	33.212			
38	58016	153.40.8.1	17036	RTP	654	33303	PT=Dynan	nicRTP-Type	e-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3303, Time=	341856778	B, Mark	173.895			
39	58016	153.40.8.1	17036	RTP	670	33304	PT=Dynan	nicRTP-Type	-96, SSRC	=0xD5CCF	EBC, S	Seq=33	3304, Time=	341859781	1, Mark	13.674			
14 4 3	MO	U_to_MB_	KEYFRAME	s / 🔁 /										-					
Ready										Average: 1	1/360.	25343	Count: 1915	Sum: 332	210164.81		100% (-)-	0	+

You can perform similar analysis with audio streams.

Generally, this helps us determine how well behaved a stream is; areas with high-deltas help indicate where in the call problems are occurring. Reasons for high-deltas may include:

Areas of High Packet Loss Queued traffic in the network Delayed Packets in the network

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	47 of 74





### **Measuring Retransmissions**

It is often useful to separate retransmissions from the original stream.

This is a many step process:

- 1. Take an appropriate capture in this example we've taken it from Media Broker
- 2. Separate the Stream using an appropriate filter
- 3. Perform Stream Analysis:
- 4. Save the Stream as a CSV

Wirest	hark: RTP Str	eam Analysis	61.00		mag fast	-	· B B west Md Made South () - 2	
Forward	d Direction	Reversed Dire	ection					
					Analysin	ng stream fro	rom 172.31.252.29 port 16000 to 172.31.252.109 port 64293 SSRC = 0xE05A8FC8	
acket	<ul> <li>Sequence</li> </ul>	• Delta(ms)	<ul> <li>Filtered Jitte</li> </ul>	r(ms) 4 Skew(ms)	IP BW(kbp	s • Marker	r 🔸 Status	•
i1	30422	0.00	0.00	0.00	6.82	SET	[ 0k ]	
i2	30423	0.00	0.00	0.00	13.26	SET	[ Ok ]	
3	30424	0.00	0.00	0.00	20.24	SET	[ Ok ]	
4	30425	0.00	0.00	0.00	24.99	SET	[ Ok ]	
5	30426	0.00	0.00	0.00	34.25	SET	[ 0k ]	
6	30427	0.00	0.00	0.00	36.86	SET	[Ok]	
1	30427	0.00	0.00	0.00	39.46	SET	Wrong sequence nr.	
11	30906	0.00	0.00	0.00	10.22		Wrong sequence nr.	
12	30907	0.00	0.00	0.00	20.45		[ Ok ]	
i13	30908	0.00	0.00	0.00	25.75	SET	Incorrect timestamp	
14	30909	0.00	0.00	0.00	35.98		[ Ok ]	
i15	30910	0.00	0.00	0.00	46.20		[ Ok ]	
16	30911	0.00	0.00	0.00	51.25	SET	Incorrect timestamp	
17	30912	0.00	0.00	0.00	61.47		[Ok]	
18	30913	0.00	0.00	0.00	71.70		[Ok]	
19	30914	0.00	0.00	0.00	81.92		[ Ok ]	
i20	30915	0.00	0.00	0.00	92.14		[ Ok ]	
21	30916	0.00	0.00	0.00	102.37		[ Ok ]	
22	30917	0.00	0.00	0.00	112.59		[ Ok ]	
23	30918	0.00	0.00	0.00	114.96	SET	Incorrect timestamp	
24	30919	0.00	0.00	0.00	125.18		[Ok]	
25	30920	0.00	0.00	0.00	135.41		[Ok]	
26	30921	0.00	0.00	0.00	140.77	SET	Incorrect timestamp	
27	30922	0.00	0.00	0.00	150.99		[ Ok ]	
28	30923	0.00	0.00	0.00	161.22		[ Ok ]	
529	30924	0.00	0.00	0.00	171.44		[ Ok ]	
~	20025				Max delta Max jitter Max skew Total RTP Duration 6	= 0.00 ms at = 0.00 ms. M = 0.00 ms. packets = 32 50.60 s (0 ms	at packet no.0 Mean jitter = 0.00 ms. 2280 (expected 3280) Lost RTP packets = -704 (-21.46%) Sequence errors = 708 s clock drift, coresponding to 1 Hz (-0.00%)	
Save p	ayload		Save as CSV		<u>R</u> efresh		Jump to Graph Player Next non-Ok	<u>C</u> lose

- 5. Open the Steam in Excel:
- 6. Create a New Column for Identifying Wrong Sequence Number:
  - a. Something like this: =IF(I3="Wrong sequence nr.",A3,"")
  - A Better way is to look for recurring sequence numbers:
     =IF(ISNA(VLOOKUP(B3,\$B\$2:B2,1,FALSE)=B3),A3,"")
- 7. Have a Column for creating the appropriate filter:
  - a. Something like : =IF(M3="", "", CONCATENATE("frame.number==",M3))

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	48 of 74



- b. or: use ISNA if #N/A comes up
- 8. Have a Column that concatenates a big filter:
  - a. Something like: =IF(N3="",O2,CONCATENATE(O2,"||",N3))
- 9. Copy the Filter in Wireshark, it should look like:

||frame.number==61||frame.number==62||frame.number==63||frame.number==64

. . . . . .

- a. You'll need to remove the first || and the last character
- 10. Now you can use wireshark's filter to differentiate between the original stream and retransmissions:



This graph shows the results from a wireshark of an isolated video stream:

- Original stream in black
- Transmissions in red
- Total in Green

Document Title:	Diagnosing Media I	ssues on the Fusion Platform	Classification	PUBLIC
			Page	49 of 74





### **Double Packets**

When a tcpdump is performed using the *-i any* option the capture is taken above the interface layer. We have seen some capture taken at Media Broker that contain duplicates for packets being sent by Media Brother. These should have only been targeted at a single interface, but at the level the capture is taken the packet is presented to both interfaces.

Duplicate packets can disrupt analysis, but they can be removed using the following wireshark utility:

editcap -d orig.pcap noDups.pcap

## Analyzing Streams on a Network Bridge

In this example zeroshell is being used to limit bandwidth across as a network bridge. This section is not a tutorial for setting up Zeroshell or capturing packets from the zeroshell machine. Instead, it contains some useful wireshark filters for determining which interface packets are flowing through on the transparent network bridge.

On the bridge each packet is displayed twice: on the way in, and on the way out. Assuming that a capture on any-interface was performed; they can be distinguished by the "Linux cooked capture information".

The following filters can be used:

Filter	Packets
(sll.pkttype==3)	Unicast to another host
(sll.pkttype==4)	Sent by us

These can be used to filter an srrc stream in the capture:

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	50 of 74





Wiresh	ark IO	Graphs:	BadC_AT_ZS.pcap									X
			/	~		~	4					- 250
		0s	10s	20s	30s	1.1.1	40s		50s	1 1 1 1 1	60s	
٠.					III							۱.
Graphs										X Axis		-
Graph 1	Color	Filter:	==60994 && rtp.ssrc==	0x98934E1F) &&	(sll.pkttype==3)	Style:	Line	-	Smooth	Tick inter	val: 1 sec	-
Graph 2	Color	Filter:	:==60994 && rtp.ssrc==	0x98934E1F) &&	(sll.pkttype==4)	Style:	Line	-	Smooth	Pixels per	tick:	.0 💌
Graph 3	Color	Filter:				Style:	Line	-	Smooth	View	as time of d	ay
Graph 4	Color	Filter:				Style:	Line	-	Smooth	V Axis	De electe (Tie	
Graph 5	Color	Filter				Style:	Line	-	Z Smooth	Scale:	Auto	K V
[]						21,12				Smooth:	No filter	•
Help		<u>C</u> o	ру							<u>S</u> ave		ose

We can see above that Zero Shell has flattened or limited the bandwidth available.

Document Title:	Diagnosing Media	ssues on the Fusion Platf	orm	Classification	PUBLIC
				Page	51 of 74



cafex.com



# **Picture Loss Recovery**

When a video steam cannot be rendered a new keyframe is required. These are requested by a client in a number of ways:

- PLIs,
- RFC2032 FIRs
- SIP INFO.

PLIs are the only mechanism on the web side, so MB injects PLIs into a stream when it receives FIRs or INFOs. Media Broker can translate between PLIs and FIRs on the SIP side meaning any FIRs going to MB are sent as PLIs to the web client and any PLIs sent by the web client will be translated to FIRs on the SIP side.

PLIs can be found in the RTCP stream:

9105 21.983580 192.168.19.29	17089 192.168.18.38	20235 RTCP	76 Receiver Report	
9112 21.993169 192.168.19.29	17089 192.168.18.38	20235 RTCP	76 Receiver Report	
9113 22.003537 192.168.19.29	17089 192.168.18.38	20235 RTCP	88 Receiver Report Payload-specific Feedback P	LI
9132 22.033741 192.168.19.29	17089 192.168.18.38	20235 RTCP	88 Receiver Report Payload-specific Feedback P	'LI
9165 22.073769 192.168.19.29	17089 192.168.18.38	20235 RTCP	88 Receiver Report Payload-specific Feedback P	'LI
9172 22.084397 192.168.19.29	17089 192.168.18.38	20235 RTCP	100 Sender Report Source description	

If you want to filter, auto-completion can help to write a wireshark filter for sub-types:

### rtcp.psfb.fmt==1

rtcp is the packet type, psfb for payload specific feedback which is the section type for PLIs, and fmt is the feedback message type).

The corresponding RTP stream will hopefully contain a keyframe shortly afterwards. The filter: *h264.nal\_unit\_hdr* can help you find them:

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	52 of 74

	(h264.nal_unit	_hdr == 7 && ud	p.dstport == 17088)    sip	0	Expression + UDP
No	. Time	Source	Src Por Destination	Dst Por Protocol	Length Info
	18 0.762344	192.168.19.29	38897 192.168.8.225	5060 SIP/SDP	835 Request: INVITE sip:1018@ubiquity.net
	19 0.773350	192.168.8.225	5060 192.168.19.29	38897 SIP	441 Status: 100 Trying
	20 0.930351	192.168.8.225	5060 192.168.19.29	38897 SIP	1116 Status: 180 Ringing
	33 2.010359	192.168.8.225	5060 192.168.19.29	38897 SIP/SDP	1874 Status: 200 OK
-	39 2.038092	192.168.19.29	5080 10.10.10.30	5060 SIP	555 Request: ACK sip:1018@10.10.10.30:5060
	111 2.547425	192.168.18.38	20234 192.168.19.29	17088 H264	66 PT=H264, SSRC=0x81FBB483, Seq=39492, Time=3777664813 SPS
	3382 9.817722	192.168.19.29	5080 10.10.10.30	5060 SIP/SDP	766 Request: UPDATE sip:1018@10.10.10.30:5060
	3446 10.013870	10.10.10.30	5060 192.168.19.29	5080 SIP/SDP	313 Status: 200 OK
	3577 12.363306	192.168.19.29	5080 10.10.10.30	5060 SIP/SDP	766 Request: INVITE sip:1018@10.10.10.30:5060, in-dialog
	3578 12.367760	10.10.10.30	5060 192.168.19.29	5080 SIP	552 Status: 100 Trying
	3612 12.960572	2 10.10.10.30	5060 192.168.19.29	5080 SIP/SDP	399 Status: 200 OK
	3617 12.975322	192.168.19.29	5080 10.10.10.30	5060 SIP	555 Request: ACK sip:1018@10.10.10.30:5060
	3623 13.010355	5 192.168.18.38	20234 192.168.19.29	17088 H264	66 PT=H264, SSRC=0x73D0C0D2, Seq=5808, Time=1279686953 SPS
	7007 18,452729	192.168.19.29	5080 192,168,8,225	5060 SIP	431 Request: OPTIONS sip:192.168.8.225
	7009 18.456616	5 192.168.8.225	5060 192.168.19.29	5080 SIP	478 Status: 200 OK
	9136 22.045440	192.168.18.38	20234 192.168.19.29	17088 H264	66 PT=H264, SSRC=0x73D0C0D2, Seq=6278, Time=1280499955 SPS
	9716 23.379776	5 192.168.18.38	20234 192.168.19.29	17088 H264	66 PT=H264, SSRC=0x73D0C0D2, Seq=6331, Time=1280619956 SPS
	102 24.581245	5 192.168.19.29	5080 10,10,10,30	5060 SIP	555 Request: BYE sip:1018@10.10.10.30:5060
L	102 24.584546	10.10.10.30	5060 192.168.19.29	5080 SIP	515 Status: 200 OK

## **Finding PLIs**

This filter will help find PLIs and was useful in diagnosing an issue with ICMP errors:

### icmp || sip || rtcp.rtpfb.fmt

eth0_diagnostic_logging_tcpdump.pcap [Wireshark 1.12.0-rc2 (v1.12.0-rc2-0-gfd01	[7ee from master-1.12)]				- 0 - X
<u>File Edit View Go Capture Analyze Statistics Telephony Tools Internals</u>	: <u>H</u> elp				
● ● ◢ ■ ◢ ⊨ ≞ ೫ ₴ ! ٩, ∻ ∻ ₽ ₮ ऺ ! 🔲 ⊑	🛯 0, 0, 0, 🖺 📓 🖉 🎭	12			
Filter: icmp    sip    rtcp.rtpfb.fmt 💌 Expre	ression Clear Apply Save				
No.         Source         stoppet         Source         stoppet         Stoppet           2046         2014-07-18         14:00:12.5:6672000         17:3:1.252.15         stp         17.           2046         2014-07-18         14:00:12.5:6672000         17:3:1.252.15         stp         17.           2014         2014         2014         14:00:12.5:6672000         17:3:1.252.15         stp         17.           2014         2014-07-18         14:00:12.5:841000         17:3:1.252.16         stp         17.           2117         2014-07-18         14:00:12.7:892000         17:3:1.252.10         stp         17.           2177         2014-07-18         14:00:12.7:892000         17:3:1.252.10         10:1-tw17.           2252         2014-07-18         14:00:12.7:89000         17:3:1.252.10         10:1-tw17.           2263         2014-07-18         14:00:13.1:8158000         17:3:1.252.10         10:01:1.1:10:11           2329         2014-07-18         14:00:16.95742000         17:3:1.252.10         10:00:71           2140         2014-07-18         14:00:16.95742000         17:3:1.252.10         10:01:1-tw17.           2140         2014-07-18         14:00:16.957932000         17:3:1.252.10         10:00:1-tw17.	Initial         despet Peterol Le           2.31.252.310         1C1-WWSTP/SDF           2.31.252.35         s1p         STP           2.31.252.35         s1p         STP           2.31.252.35         s1p         STP           2.31.252.35         s1p         STP           2.31.252.10         30763         STP/SDF           2.31.252.10         1C1-WKSTP/SDF         31753           2.31.252.10         1C1-WKSTP         313.252.10           2.31.252.10         5773         RTCP           2.31.252.10         5773         RTCP           2.31.252.10         50003         RTCP           2.31	nth NTPTimestamp 1322 764 447 683 616 638 651 1322 764 202 62 62 62 62 62 62 62 62 62 6	Ide         Status: 200 oK             Status: 200 oK           Status: 200 oK             Request: ACK \$[p:172.31.252.55:5060]transport=tcp             Request: ACK \$[p:172.31.252.55:5060]transport=tcp             Request: ACK of p:172.31.252.55:5060]transport=tcp             Status: 200 oK             Request: ACK \$[p:172.31.252.55:5060]transport=tcp             Status: 200 oK             Request: ACK \$[p:172.31.252.55:5060]transport=tcp             Status: 200 oK             Request: ACK \$[p:172.31.252.55:5060]transport=tcp             Request: OTIONS \$[p:172.31.252.75:5060]transport=tcp             Request: OTIONS \$[p:172.31.252.75:5060]transport=tcp             Status: 200 oK             Sender Report           Sender Report </td <td>user:epid:WAAumLpmLV21a0vmC8XvqwAA;gruu r:epid:WAAumLpmLV21a0vmC8XvqwAA;gruu in-dialog   Generic RTP Feedback   Generic RTP Feedback ack</td> <td>∙uu, in-dia I</td>	user:epid:WAAumLpmLV21a0vmC8XvqwAA;gruu r:epid:WAAumLpmLV21a0vmC8XvqwAA;gruu in-dialog   Generic RTP Feedback   Generic RTP Feedback ack	∙uu, in-dia I
2032 2014-07-18 14:00:32.068944000 172.31.252.110 30007 17 2106 2014-07-18 14:00:32.0673100 172.31.252.110 3007 17 2106 2014-07-18 14:00:32.028356000 172.31.252.110 10704 17 2110 2014-07-18 14:00:32.068703000 172.31.252.110 17074 17 2112 2014-07-18 14:00:32.068703000 172.31.252.110 17074 17 2112 2014-07-18 14:00:32.069032000 172.31.252.110 17076 17	2.31.253.194 5773 RTCP 2.31.253.194 5773 RTCP 2.31.252.55 56000 ICMP 2.31.252.55 56000 ICMP 2.31.252.55 56000 ICMP 2.31.252.55 56000 ICMP 2.31.252.55 56002 ICMP	62 62 242 38529 242 38530 242 38531 84 14201	Generic KIP Feedback Generic KIP Feedback Destination unreachable (Port unreachable) Destination unreachable (Port unreachable) Destination unreachable (Port unreachable) Destination unreachable (Port unreachable)		
<ul> <li>□ Frame 2349: 764 bytes on wire (6112 bits), 764 bytes cap □ Ethernet II, Src: 00:00:29:50:83:e6 (00:00:29:50:83:e6),</li> <li>□ Internet Protocol version 4, Src: 172.31.252.110 (172.31. □ Transmission Control Protocol, Src Port: icl-twobase7 (2): □ Session Initiation Protocol (AcK)</li> </ul>	tured (6112 bits) Dst: 00:0c:29:74:de:8f (00:0c: .252.110), Dst: 172.31.252.55 ( 5006), Dst Port: sip (5060), Se	29:74:de:8f) 172.31.252.55) q: 4768, Ack: 36	83, Len: 698		
	SIP/2.0				
■ message measure    via: S1P/2.0/TCP 172.31.252.110:5060;egress-zone-Def    via: S1P/2.0/TCP 172.31.252.110:65070;tranch=29fe64Kb call-10: 5283da5da65076287063c2753742508127.31.252.    CSeg: 101 ACK    From: csip:edna9/pc.cdflab.cafex.com>;tag=c68cd3ble(    Tor: <sip:edna9 pc.cdflab.cafex.com="">;tag=c68cd3ble(    tor: <sip:edna9 pc.cdfl<="" td=""><td>au TtZone; branch-29h64bK27deff79 49153775f90a14852aca93ecFa00aa 55 c8b58a6</td><td>bac4e84d62eb7c32 616;received=172</td><td>f9c46f9024;proxy-call-fd=d3fb2074-8948-4216-92bc-1b692 .31.252.110;rport=34751;ingress-zone=TaMicrosoftLyncser</td><td>961ae2;rport verviae280A</td><td></td></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9></sip:edna9>	au TtZone; branch-29h64bK27deff79 49153775f90a14852aca93ecFa00aa 55 c8b58a6	bac4e84d62eb7c32 616;received=172	f9c46f9024;proxy-call-fd=d3fb2074-8948-4216-92bc-1b692 .31.252.110;rport=34751;ingress-zone=TaMicrosoftLyncser	961ae2;rport verviae280A	
Content-Length: 0					
D000         00         02         29         74         de         8f         00         0c         29         50         83         e6         08         00         45         00   .	.)t)PE. x0.@. @n. 7az. 9v. /S gACK si p:172.31				-
	D.C. D.C. b				

The Screenshot shows:

• The call finishes setting up at 14:00:13 with receipt on an ACK.

Document Title:	Diagnosing Media Issue	s on the Fusion Platform	Classification	PUBLIC
			Page	53 of 74





- There is a PLI at 14:00:21 Marked with Payload Specific
- The first ICMP event happens at 14:00:32.

This was caused by a third party application crashing on receipt of a PLI.

## **RFC2032 FIRs & SIP INFOs**

If PLIs do not appear to be working and you can't find any PLIs coming from a SIP device it may be it uses RFC2032 FIRs and SIP INFOs (also known as PFUs or FPUs). SIP INFOs received by FAS will normally result in the web client sending a keyframe as MB translates them to PLIs in that direction, and FCSDK can be configured to have FAS send SIP INFOs when the web client sends a PLI.

	udp.port == 17021    sip					
No.	Abs Time	Time	Source	Src Por Destination	Dst Por Protocol	Length Info
+	7 09:37:36.262264	0.108020	192.168.19.29	38937 192.168.8.225	5060 SIP/SDP	835 Request: INVITE sip:5298@ubiquity.net
	8 09:37:36.272108	0.117864	192.168.8.225	5060 192.168.19.29	38937 SIP	441 Status: 100 Trying
	9 09:37:36.404362	0.250118	192.168.8.225	5060 192.168.19.29	38937 SIP	1019 Status: 180 Ringing
μ.	18 09:37:37.736245	1.582001	192.168.8.225	5060 192.168.19.29	38937 SIP/SDP	1537 Status: 200 OK
	28 09:37:37.757640	1.603396	192.168.19.29	5080 192.168.8.225	5060 SIP	715 Request: ACK sip:5298@192.168.17.23:5060
	52 09:37:38.043654	1.889410	192.168.8.225	5080 192.168.19.29	5060 SIP/XML	1124 Request: INFO sip:1002@engtest166.cafex.com:5060
	54 09:37:38.047532	1.893288	192.168.19.29	5060 192.168.19.29	5080 SIP/XML	1346 Request: INFO sip:1002@engtest166.cafex.com:5060
	59 09:37:38.058654	1.904410	192.168.19.29	5080 192.168.19.29	5060 SIP	925 Status: 200 OK
	60 09:37:38.061449	1.907205	192.168.19.29	5060 192.168.8.225	5080 SIP	767 Status: 200 OK
	294 09:37:38.701006	2.546762	192.168.19.29	17021 192.168.17.172	25929 RTCP	100 Sender Report Source description
	318 09:37:38.734401	2.580157	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	339 09:37:38.765135	2.610891	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	361 09:37:38.793917	2.639673	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	380 09:37:38.834036	2.679792	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	395 09:37:38.856268	2.702024	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	398 09:37:38.864752	2.710508	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	417 09:37:38.905462	2.751218	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	434 09:37:38.936365	2.782121	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	452 09:37:38.965387	2.811143	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	469 09:37:38.997329	2.843085	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	490 09:37:39.027621	2.873377	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	492 09:37:39.037558	2.883314	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	511 09:37:39.068647	2.914403	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI
	528 09:37:39.099476	2.945232	192.168.19.29	17021 192.168.17.172	25929 RTCP	88 Receiver Report Payload-specific Feedback PLI

In this example we can see lots of PLIs going from Media Broker to a SIP phone, but none in the other direction; INFOs are being received however. Often the INFOs will repeat until the phone gets a keyframe, but this device appears not to do that.

Looking at the DTLS, we can see the INFOs came in to the gateway, prior to the web client finishes establishing its media path. Unlike standard RTCP these are sent on the RTP ports and are effectively an RTCP packet containing just the FIR. With newer versions of Wireshark it seems mostly it will automatically decode them as RTCP, but it can't be relied upon, so there are two ways to filter for them.

	5.port == 17020 &	& !(rtp.p_ty	oe==102)					
Vo.	Abs Time	Time	Source	Src Por Destination	Dst Por Protocol	Length Info		
283	3 09:37:38.694401	2.540157	192.168.17.172	25928 192.168.19.29	17020 RTCP	62 Full	Intra-frame	Request (H.2
Doc	ument Title:	Diagnosing	Media Issues on	the Fusion Platform		Classification	PUBLIC	



### Note:

If you want to check if PLIs or FIRs have raced in before the DTLS exchange has completed and not repeated then you can create a filter that shows dtls or video RTP/RTCP ports with rtcp packets.

dtls || ((udp.port==17020 || udp.port=17021) && rtcp)

# NACK

When packet loss occurs in the video stream on the WebRTC leg of a call the main recovery method used is Negative ACKnowledgement, a.k.a NACK. This is request for retransmission of some RTP packets sent via a section in an RTCP packet, which should result in the other end resending them if it still has a copy. If too much packet loss occurs, NACKs will not always be used, instead PLIs will be sent and old data given up on, or if the round trip time is high (over 50ms) the stream will switch on ULPFEC which will result in far fewer, if any, NACKs being sent.

As a result of encryption on the web side, any packet capture where you need to confirm which NACKs are present in RTCP will need to be decrypted. It is possible to tell if retransmissions of the requested RTP packets happens without decrypting, and if RTX is not used then you can simply look at the sequence numbers to see which; however, when RTX is used you will still need to decrypt to find out which are being resent.

# RTX

Modern webRTC clients, support RTX, where NACKs are not simple retransmissions. You can tell if this is in use as both the offer and answer on the web side will include the RTX codec, and have two SSRCs for the video media line which share the same MSID.

When in use responses to NACKs will be sent with the payload type for RTX and using a separate set of sequence number; to tell the sequence number it was sent for you need to look at the first two bytes of the payload. The extra sequence number is a special feature of RTX packets that is removed once processing of it is done, so you will not see it on the SIP side.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	55 of 74





**Note**: RTX appears multiple times with different payload types in more recent versions of Chrome, you may see it in its FMTP attribute, but only one will be used and if there are multiple in both the offer and answer negotiations the RTX will be the one for RED.

## Fragmentation

UDP packets that are fragmented are not understood by any web clients, but if a packets comes from the SIP side that is at or above the maximum MTU of the network for the path between Media Broker and the FCSDK client then it will get fragmented as encryption adds a few bytes to the packet.

To check quickly if fragmentation is present in a capture use the filter:

*ip.flags.mf* ==1 *or ip.frag\_offset gt 0.* 

If this results in "Fragmented IP Protocol" frames being show for a stream then fragmentation will be a problem.

# **Adaptive Bitrate**

Video endpoints distributed over the internet cannot guarantee a stable bitrate required for real time communications. It is important that the stream of packets which construct a video call arrive at their destination in a timely fashion; depending on the network pathway between an client and Media Broker, network buffering or QoS restrictions may limit or severely impact video performance.

Media Broker implements REMB (for WEB-RTC Clients) and TMMBR (for SIP Clients) specifications so that Media Broker can monitor the RTCP Reports and react to the ever changing environment clients may face.

These protocols will request more or less bandwidth from clients if the network conditions change. If there is no change the protocol will maintain the current bitrate.

Media Broker allows the administrator to configure constraints on the bandwidth to maintain video quality parameters can be met by the business requirement.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	56 of 74



In its current implementation Media Broker does not support Dynamic Video Resolution Changes; this means if a call is established at 720HD this resolution will be maintained throughout the call.

The *Maximum* and *Minimum Adaptive Bitrate* values, give bounds to the threshold that the Media Broker will go to when rendering video

This value effects Media Broker in two ways:

- 1. Media Broker will request no-more or less than these value from Clients.
- Media Broker will not use a value outside of this range to encode video. Media Broker will ensure that the Video it generates is of an appropriate quality and not render a 720HD video with insufficient bitrate, nor render video with an inappropriately high bitrate with a diminished return.

The *Initial Adaptive Bitrate* value is used by Media Broker when sending video at the beginning of a call before there is enough data collected from the RTCP to behave appropriately.

In most consumer cases it is appropriate to set this value equal to the *Minimum Adaptive Bitrate*; if the network is sufficient the bitrate of the call will improve shortly after the call starts; however, some video solutions may prefer the video starts at a higher bitrate, in which case clients on an insufficient networks will have a worse experience until the bitrate falls.

The following diagram demonstrates how data is received at a client with an actual physical bandwidth which is capped at 350kbps. Such a consumer will never be able to receive more than 350kbps due to a limitation in their network capabilities.

Page 57 of 74	Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC
				Page	57 of 74







The graphs on the left hand side show what the user's experience will be when the *Initial Value* is around 512kbps (more than the client's maximum). The graphs on the right hand side show what happens for this user when the Initial value is set to 300kbps (below the client's minimum).

The graph on the left shows the user will receive a poor video experience for the first few seconds of the call (and data may be lost); the user will see degraded video and audio may be effected.

The graph on the right shows the user's maximum bandwidth is reached very quickly and there is very little interruption to the user's experience.

### SIP-side considerations

In solutions where clients are establishing video calls to traditional SIP-based video devices, such as deskphones, soft-UAs or MCUs the Media Broker can utilize the network architecture for a better video experience.

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC
			Page	58 of 74



If SIP-endpoints are based in a dedicated network the Media Broker can apply a fixed-bitrate to a video negotiation. This is advantageous because typically dedicated video-networks can guarantee the network stability required for communications and thus both endpoints can agree an optimal bitrate when the call is established. The value of the fixed bitrate can be found in the Media Broker's *proxy.properties* file. The property is *sip.bitrate.override.main=4000000*. This will set an AS parameter equal to 4mbps in the SDP sent in requests on the SIP side.

Page 59 of 74	Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
					Page	59 of 74



## Finding TMMBR

In wireshark TMMBR appears inside a *Generic RTP Feedback* block of the RTCP:

tcpdump.pcap [Wireshark 1.12.11 (v1.12.11-0-gc74c83c from master-1.12)]					
Eile Edit View Go Capture Analyze Statistics Telephony Tools Internals Help					
● ● ∡ ■ ∡ ⊨ 🗎 X 2 ⊂ ⇔ ⇔ ⇒ 7 ½   □ 🖬 0 ⊂ ⊂ ⊡   ¥ ⊠ 🖪 %   X					
Filter: udp && !rtp   Expression Clear Apply Save					
Destination Dest Port Protocol Length Info					
0 192.168.0.64 51696 RTCP 138 Sender Report					
6 192.168.0.29 16000 RTCP 114 Sender Report					
5 172.31.252.29 17001 RTCP 76 Application specific (Flux) subtype=1					
5 172.31.252.29 17001 RTCP 76 Application specific (Flux) subtype=1					
5 172.31.252.29 17001 RTCP 76 Application specific (Flux) subtype=1					
5 1/2.31.252.105 2058/ RTCP 120 Sender Report Source description Generic RTP Feedback					
/ 1/2.31.252.29 1/005 RICP 80 Application Specific (Flux) Subtype=1					
1/2.31.252.29 1/005 RICP 80 Application specific (Flux) subtype=1					
172.31.252.65 1/005 KTCP OV Application Specific (First) Subtype=1					
1 102 168 0 64 S1696 PTCP 162 Sender Report					
5 172.31.252.29 17001 RTCP 76 Application specific (Flux) subtype=1					
<pre>Elinux cooked capture Internet Protocol Version 4, Src: 172.31.252.29 (172.31.252.29), Dst: 172.31.252.105 (172.31.252.105) User Datagram Protocol, Src Port: 17005 (17005), Dst Port: 20587 (20587) Real-time Transport Control Protocol (Sender Report) Real-time Transport Control Protocol (Source description) Real-time Transport Control Protocol (Generic RTP Feedback): TMMBR: 262144 10 = Version: RFC 1889 Version (2)0 = Padding: False0 0011 = RTCP Feedback message type (FMT): Temporary Maximum Media Stream Bit Rate Request (TMMBR) (3) Packet type: Generic RTP Feedback (205) Length: 4 (20 bytes) Sender SSRC: 0x73952025 (1939152933) Media source SSRC: 0x492c579d (1227642781) TMMBR 1 [RTCP frame length check: OK - 76 bytes]</pre>					
0000       00       04       00       10       06       52       54       00       12       a4       6b       00       08       00      RT      RT					
● Markets: 29002 · Displayed: 1046 (3.6%) · Load time: 0:00.314					

## Finding REMB

This is a lot harder to see in a wireshark, because the RTCP packets are encrypted. You can see a close approximation of what is being sent by Media broker when it is received at your client, if you are using Chrome. This will be covered in the Browser topic when looking at bandwidth estimates.

Document Title: D	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	60 of 74



## **Analyzing Lip Sync**

Once the audio and video RTP streams leave the sender, they are independent of each other and it's the job of the decoder to cope for any fluctuations caused by network conditions when reconstituting the streams.

Lip Sync issues can be seen if the audio and video streams are vastly apart from each other and clients should be using the RTCP to keep packets in sync.

The RTCP sender report contains information that the decoder can use to keep packets in sync:

Example Audio RTCP Sender Report:

```
Real-time Transport Control Protocol (Sender Report)
10..... = Version: RFC 1889 Version (2)
..0. ... = Padding: False
...0 0000 = Reception report count: 0
Packet type: Sender Report (200)
Length: 6 (28 bytes)
Sender SSRC: 0x98171833 (2551650355)
Timestamp, MSW: 3613377647 (0xd75fc46f)
Timestamp, LSW: 3687079100 (0xdbc45cbc)
[MSW and LSW as NTP timestamp: Jul 3, 2014 12:00:47.858465000 UTC]
RTP timestamp: 1891280655
Sender's packet count: 4
Sender's octet count: 640
```

Example Video RTCP Sender Report:

```
□ Real-time Transport Control Protocol (Sender Report)
10..... = Version: RFC 1889 Version (2)
..0.... = Padding: False
...0 0001 = Reception report count: 1
Packet type: Sender Report (200)
Length: 12 (52 bytes)
Sender SSRC: 0x3fe44e59 (1071926873)
Timestamp, MSW: 3613377649 (0xd75fc471)
Timestamp, LSW: 1719039185 (0x667674d1)
[MSW and LSW as NTP timestamp: Jul 3, 2014 12:00:49.400244000 UTC]
RTP timestamp: 3383869149
Sender's packet count: 100
Sender's octet count: 63585
```

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	61 of 74



In both cases the Sender Report contains an RTP timestamp and a 'wall clock' time. Each RTP packet contains its own RTP timestamp, with all this information the receiver can use this information to playback the audio and video streams relative to one another. It's not so straightforward though, as the decoder will be making assumptions on the sender's clock drift, clock skew to resynchronize the steams. Essentially, there are 3 ways of synchronizing the streams:

- Add silence
- Just ahead (remove excess)
- Alter playback feed of other stream to speed one up or slow one down

Neither of these things are instantaneous as the decoder needs to take time to sample enough data to detect a suitable bound.

Identifying a lip synchronization issue based on the packet arrival time is not trivial. Components like the Jitter Buffers may add delay allowing the encoder to keep the streams in synch.

You can calculate the relative 'wall time' for a given RTP packet by using its RTP timestamp and the RTP timestamp with the 'Wall Time' from a RTCP sender-report as a reference point. Comparing this time against the received time can provide latency for a given packet.

Performing these calculations for both the audio and video streams will allow you to compare how far apart the given streams may be arriving.

Codecs like G711 use a fixed sample rate, meaning each packet contains 160 samples at 8000MHz, or 20ms of audio. So it's quite easy to determine if the RTP Time stamp is ahead or behind it's expected arrival time.

Codecs with variable sample rates (like video) can be harder to estimate without the negotiated clock speeds; however, you can estimate it.

The follow table are values taken from RTCP packets

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	62 of 74



wall time	rtp time	received time	delta wall time	delta rtp-time	delta received time
00:47.8	3383724609	59:33.5			
00:49.0	3383836659	59:34.7	00:01.245	112050	00:01.245
00:49.1	3383838459	59:34.7	00:00.020	1800	00:00.019
00:49.1	3383842149	59:34.8	00:00.041	3690	00:00.041
00:49.1	3383845029	59:34.8	00:00.032	2880	00:00.032

We can see that a data of RTP timestamps equal to 1800 takes approximately 20ms and that's consistent, you can use this as a clock rate. You can estimate the expected arrival time for an RTP packet by:

- 1. calculating the difference between the RTP packet's timestamp and the last received RTCP packet's timestamp
- 2. Using the clock rate determine the number of ms since the RTCP packet arrived
- 3. Add this time to the wall clock time of the RTCP packet

With this information you can:

- Compare this value with the actual arrival time
- Compare the delay against another stream

Document Title:	Diagnosing Media Issues on the Fusion Platform		Classification	PUBLIC	
				Page	63 of 74



# Captures taken from an iPAD

In order to measure the packet loss of the RTP stream from the Fusion Media Broker to a Fusion Enabled iOS Application the iPAD will need to be connected to a Macintosh with wireshark installed. The following commands are taken from The Mac Developer Library (https://developer.apple.com/library/mac/qa/qa1176/\_index.html#//apple\_ref/doc/uid/DTS 10001707-CH1-SECRVI) and they describe how the mac can be used used listen to wireless network traffic received by the ipad:

### rvictl -s IPAD\_UUID

Where IPAD\_UUID is the *Identifier* for the IPAD, which can be obtained from XCode's Organizer view:



#### 1. Taking a Capture using RVI

Using Wireshark there is now an additional Interface that can be monitored. Make a call using the IPAD and allow the Call to establish and run for a few minutes, before stopping the capture and processing the results.

This capture will determine how much packet loss there is from the Fusion Media Broker to the Ipad. This is because the UDP packets will either be missing or out of sequence.

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	64 of 74



00	X	Wireshark: Capture Interfaces		
Device	Description	IP	Packets	Packets/s
🗆 🗩 en0		none	0	0
🗆 🗩 fw0		none	0	0
🗹 🗩 rvi0		none	16	2
🗆 👳 en 1		fe80::5e96:9dff:fe80:3f7d	0	0
🗆 🗩 p2p0		none	0	0
🗆 🗗 lo0		fe80::1	0	0
<u>₿</u> Help	<u>í</u> Star	rt Stop Opt	ions	<u>XC</u> lose

### 2. Filtering Traffic

There will be four UDP streams in this capture: two inbound video and audio streams from the Fusion Media Broker to the ipad and another two outbound streams from the ipad to the Fusion Media Broker. It is necessary to hunt for a packet that is suspected to be from the video stream.

Apply the following filter: *udp* && !rtp

290 13.763712000	192.168.1.107	81.144.171.73	UDP	210 Source port: 65219	Destination port: 16000
291 13.775137000	192.168.1.107	81.144.171.73	UDP	210 Source port: 65219	Destination port: 16000
292 13.784997000	81.144.171.73	192.168.1.107	UDP	1288 Source port: 16000	Destination port: 65219

The Ipad will send to Destination Port: 16000, so it is likely that the first two packets belongs to the outbound stream. The third packet is likely to belong to the inbound stream.

#### 3. Encoding the UDP Stream

Right Click a UDP packet from the Fusion Media Broker and Select: Decode As

Document Title:	Diagnosing Media Issues on the Fusi	on Platform	Classification	PUBLIC
			Page	65 of 74



00	🔀 Wireshark: Decode As	
● Decode	Network Transport	
O Do not decode	UDP_Both (16000↔65219) 🔽 port(s) as	RIPng RMCP RPC RSIP RSP
Lear Show Current		RSVP RTCP RTP
🔀 <u>H</u> elp	4 <u>4</u>	OK <u>Apply</u> <u>Close</u>

Select RTP and press Apply. Wireshark will now interpret the UDP stream as a full RTP stream. It should be noted that in this case a G.711 audio-packet was selected, but wireshark has also distinguished between inbound and outbound audio and video streams:

288 13./03100000	192.108.1.107	81.144.1/1./3	RIP EVENI	1250 Payload lype=RFP Event, Line lockoul lone
289 13.763643000	192.168.1.107	81.144.171.73	RTP EVENT	244 Payload type=RTP Event, Unknown (28) (end)
290 13.763712000	192.168.1.107	81.144.171.73	RTP	210 PT=ITU-T G.711 PCMU, SSRC=0xDCF9CE07, Seq=81, Time=18160
291 13.775137000	192.168.1.107	81.144.171.73	RTP	210 PT=ITU-T G.711 PCMU, SSRC=0xDCF9CE07, Seq=82, Time=18320
293 13.793385000	192.168.1.107	81.144.171.73	RTP	210 PT=ITU-T G.711 PCMU, SSRC=0xDCF9CE07, Seq=83, Time=18480
295 13.825741000	192.168.1.107	81.144.171.73	RTP	210 PT=ITU-T G.711 PCMU, SSRC=0xDCF9CE07, Seq=84, Time=18640

### 4. Analyze the Stream

Now that the streams have been formatted Select: Telephony->RTP->Show All Streams

Document Title:	Diagnosing Media Issues on	the Fusion Platform	Classification	PUBLIC
			Page	66 of 74



Telephony <u>T</u> ools <u>I</u> nter	nals <u>H</u> elp
<u>A</u> NSI ►	
<u>G</u> SM ►	
<u>H</u> .225	ar Apply Save
IA <u>X</u> 2 ►	Protocol Length Info
ISUP Messages	RTP 210 PT=ITU
<u>L</u> TE ▶	RTP 210 PT=ITU
<u>M</u> TP3 ►	RTP 210 PT=ITU
<u>R</u> ТР 🕨	Show All Streams
RTSP 🕨	Stream Analysis
SCTP 🕨	
SIP	RIP EVENI 1288 Payloa
SMPPOparations	BTP 210 PT=ITU
SIM <u>F</u> FOPERations	RTP 210 PT=ITU
<u>U</u> CP Messages	RTP 210 PT=ITU
🕻 <u>V</u> oIP Calls	RTP EVENT 1250 Payloa
WAP-WSP	RTP EVENT 1029 Payloa
1972 188 1 1972	DTD EVENT 1012 Dayloa

## The following will be displayed:

00				X W	'ireshark: RTP Strear	ns				
		Detected	4 RTP stre	eams. Choose oi	ne for forward a	nd revers	e direction f	or analysis		
Src addr	Src por	Dst addr	Dst poi	SSRC	Payload	Packe	Lost	Max Delta (m	Max Jitter (r	Mean Jitter
192.168.1.107	65219	81.144.171.73	16000	0xDCF9CE07	g711U	3944	0 (0.0%)	51.37	9.66	6.09
192.168.1.107	65219	81.144.171.73	16000	0x6B6570ED	RTPType-100	2114	118 (5.3%)	0.00	0.00	0.00
81.144.171.73	16000	192.168.1.107	65219	0xA5458BCD	g711U	2387	0 (0.0%)	101.77	14.06	7.32

Here it can be seen that the inbound and outbound video and audio streams. In this example 37% of packets were lost on the inbound stream to the ipad.

Document Title:	Diagnosing Media Issu	ues on the Fusion Platfo	orm	Classification	PUBLIC
				Page	67 of 74





### **RVI Captures Introduce Erroneous Skew Measurements**

It must be remembered that the capture being taken using the RVI is not at the IPAD but the Mac which it is tethered to. We have witnessed significant discrepancies in timings values taken using the RVI when compared to a network router. Notably the skew values can be seconds out within a few minutes. It is preferable to capture IPAD traces at a wifi access point or network router.

### Capture taken from a Mavericks Mac

Wireshark cannot understand the packets captured from Mavericks because Apple chose to use an unknown packet format. This can be resolved by Opening  $Edit \rightarrow Preferences$  then *Protocols-->DLT\_USER* and editing the Encapsulations Table to add the following Entry:

DLT = User 2 (DLT=149) Payload Protocol = eth Header Size = 108 Header Protocol = <leave blank=""> Trailer Size = 0 Trailer Protocol = <leave blank=""></leave></leave>		
[ ULT_Innext_packrizepapage Windhall 2.84 (DNI Re: SPARL Non Arount 2.0) Exe Exe : Son See Cartern: Andres Santis: Tanahary: Tanahary: Han 문화 남성 성상 에는 전 정 경우는 이는 수 수 속 주 중 査 (回) 역 약 약 약 중 중 요. [ 문화 전 등 상 : 문화 Exe:		
Des         Tem         Same         Destination         Petrocol         Logght         Mr         2           1         0.0000000         1322         1         0.0000000         1324         1	User DLTs Table	e: Edit - Pr 🗖 🗐 💌
11 - 6.995800 - 71 - 71 - 71 - 71 - 71 - 71 - 71 -	Payload protocol:	eth
a Frame 21:211 bytes on wire (268 bit), 211 bytes ceptered (168 bits) biser occupation of hundled; RT-148, thet, your meterances-sentecolts-structer a Suff (211 bytes)	Header size:	108
	Header protocol:	
	Trailer size:	0
	Trailer protocol:	
00000         60000         000000         000000         00000000         000000         000000		OK <u>C</u> ancel

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	68 of 74



	meoul_packet	cap.pcaping [	Mreshark 1.6	A (SVN Rev 39941 from /b			
Ed	t Yaw Go	Capture An	adyze Statis	tics Telephony Icols	Internals Help		
ild.				0.000.00.00.00.00.00.00.00.00.00.00.00.			
-							
£					· Expression	Clear Apply	
	Time	Source		Destination	Protocol	Length Info	
16	0.9293010	0 192.193.	183.171	10.0.1.139	TCP	174 https > 50164 [ACK] seq=885 Ack=4955 win=251 Len=0 T5val=1613326980 T5ecr=1054112207	
17	1.6655120	010.0.1.1	39	192.193.183.171	TCP	1522 [TCP segment of a reassembled PDU]	
18	1.6658400	010.0.1.1	39	192.193.183.171	TLSv1	1279 Application Data	
19	1.7471140	0 192.193.	183.171	10.0.1.139	TCP	174 https > 50168 [ACK] seq=1 Ack=1349 win=152 Len=0 TSval=1613327185 TSecr=1054112986	
20	1.7538/90	0 192.193.	185.1/1	10.0.1.139	TCP	1/4 https > 50168 [ACK] Seq=1 ACK=2454 Win=163 Len=0 TSval=161332/186 TSecr=1054112986	
21	2.2292040	0 192.193.	183.171	10.0.1.139	TLSV1	499 Application Data	
22	2.2296100	0 192. 193.	183.171	10.0.1.139	TLSV1	659 Application Data	
23	2.2300030	0 192.193.	185.1/1	10.0.1.139	TLSVI	211 Application bata	
24	2.2304030	010.0.1.1	59	192.193.183.1/1	TCP	1/4 50168 > https:[ACK] Seq=2454 ACK=326 Whr=81/1 Len=0 T5Val=105411353/ T5ecr=161332/262	
23	2.2512570	010.0.1.1	59	192.193.185.171	ICP	1/4 SOLOB S HELPS [ALK] Sectores Accesses withesiss Letter ISVALEDSel1353/ ISUCTEDE1532/202	
20	2.2317420	0 192.193.	10	10.0.1.139	TESVI	211 Application bata	
5	2.2522020	010.0.1.1	10	192.193.103.171	TCP	1732 10106 2 Hittps [Ack] Scherebe Ackeds Willesis Leiko (Svaleto)411333/ (Scherebist)	
20	2.4795500	010.0.1.1	20	102 102 102 102 171	TURID	1322 (LCP segment of a reassembled Pool)	
21	2 5625260	0 102 102	92 171	10.0.1.120	TCD	174 brons > 0166 fars1 ron_005 arb_2003 v(n=172 i and trus1=1612237200 trons=1054112702	
	1. 3033300	······································		10.0.1.1.55	is.r	Trainchi > Joros (Acc) adjust Account and the factority as factority as	
tan tata	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto	bytes on w oad: eth ( is) irc: Apple_ icol Versio	ire (168) Ethernet f8:1e:71 n 4, Src	(f8:1e:df:f8:1e:7)	captured (1) (), DST: 10:- (192.193.183)	585 bits) 40:f3:d2:12:157 (10:40:f3:d2:12:157) 17.13, b0:t 10.0.1.139 (10.0.1.139)	
an T: ta te te	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto smission C	bytes on w oad: eth ( is) inc: Apple_ icol Versio ontrol Pro	ire (168) Ethernet f8:1e:71 n 4, Src tocol, Si	8 bits), 211 bytes (f8:1e:df:f8:1e:7) : 192.193.183.171 c Port: https (44)	captured (1) (), Dst: 10:- (192.193.183. 8), Dst Port	688 bits) Auffild:12:1b7 (10:40:f1:d2:12:1b7) 177), ost: 10.0.1.139 (10.0.1.139) 10:006 (01008), 5018 (11.4 Act 2244, Len: 37	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple_ icol Versio ontrol Pro Layer	ire (168) Ethernet] f8:1e:71 n 4, Src tocol, Sn	8 bits), 211 bytes (f8:1e:df:f8:1e:7; : 192.193.183.171 c Port: https (44)	captured (1) 1), Dst: 10:4 (192.193.183. 8), Dst Port	685 bfs) 4071362121b7 (0046971542121b7) 1773, 5817 10.0.0.1.19 (00.0.1.19) 53088 (39089), 58qt 811, Act: 2454, Lem: 37	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) irc: Apple_ icol Versio ontrol Pro : Layer	ire (168 Ethernet) f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7) : 192.193.183.171 :c Port: https (44)	captured (1) 1), Dst: 10:- (192.193.183. 8), Dst Port	685 bits) 0:011:0212:020 (0:01:0212:02) 0:011:0212:020 (0:01:0212:02) 0:0208 (0:0089) Seq: Bil, Ack: 2354, Lem: 37	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple_ icol Versio ontrol Pro : Layer	ire (168 Ethernet) f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7) : 192.193.183.171 :c Port: https (44)	captured (1) 1), Dst: 10:+ (192.193.183. 8), Dst Port	685 bfs) 4073/02/2107 4073/02/2107 17), 68f: 10.04.1.19 (10.04.1.19) 5 5088 (30168), 5eq: 811, Ack: 2454, Len: 37	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II. 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple_ icol Versio ontrol Pro : Layer	ire (168 Ethernet) f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7) : 192.193.183.171 -c Port: https (44)	captured (1) (), Dst: 10:4 (192.193.183 8), Dst Port	685 bfts) 40-ff1:01:21:057 (00-40:ff1:00:12:057) 373, 08-ff 10-6.11:19 (05:0-0.1109) 3-50306 (50106), Seq: 411, Ack: 2154, Len: 37	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) icc: Apple_ iccl Versio control Pro : Layer	ire (168 Ethernet f8:1e:71 n 4, Src tocol, Si	8 bits), 211 bytes (f8:1e:df:f8:1e:7 : 192.193.183.171 ) c Port: https (44)	captured (1) (), Dst: 10:- (192.193.183 (), Dst Port	685 bits) 60:ff:00:21:07 (00:60:f1:00:12:05) 71:70:07:08:08:11:77 (00:5:1:11:79 7:30:86 (20:09): 660: 81:, eds: 23:54, Len: 37	
	e 23: 211 149, Payl (108 byte rnet II, 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple_ icol Versio Control Pro : Layer	ire (168 Ethernet f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7; 192.193.183.171 i rc Port: https (44)	captured (10 (1), DST: 10:- (192.193.183, (19), DST Port	685 bfs) 40713/21/21/07 (004/07/15/021/21/07) 17/70, Betr 10.01.11/9 (00.01.11/9) 1 50888 (508.68), Seqt 811, Act: 2454, Len: 37	
	e 23: 211 149, Payl (108 byte rnet II. S rnet Proto smission C re Sockets	bytes on w oad: eth ( is) irc: Apple_ icol Versio ontrol Pro : Layer	ire (168 Ethernet f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7; 192.193.183.171 i c Port: https (44)	captured (14 1), Dst: 10:4 (192.193.183 8), Dst Port	685 bfts) 40-ff1:01:21:07 (00-40:ff1:02:12:07) 57:05 (19:05, 1:19 (05:05, 1:19) 50:06 (99:06), 50; 811, 445; 23:54, Len: 37	
	e 23: 211 149, payl (108 byte rnet II. 5 rnet Proto smission re Sockets	bytes on w oad: eth ( is) inc: Apple_ incl Versio ontrol Pro : Layer	ire (168 Ethernet f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7) 192.193.183.171 'c Port: https (44:	captured (1) ), DST: 10:: (192,193,183, (192,193,183), DST PORT	685 bfs) 607f3-0212-07 10719-0212-07 10719-072-07 10709-07-07 10709-07-07 10709-07 100000-07 10000000000000000000000000	
e e e e e e e e e e e e e e e e e e e	e 23: 211 149, payl (108 byte rnet II. S rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple icol Versio ontrol Pro : Layer	ire (168 Ethernet) f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7; 192.193.183.171 'c Port: https (44)	captured (1), Dst: 10:- (192.193.183. 3), Dst Port	685 bfts) 40-ff12012/bf (00-60-ff12012/bP) 2720, 581: 00-60-1.199 (00-00-1.199) 50088 (300.69), Seq: 811, Act: 2454, Len: 27	
in a le e in u	e 23: 211 149, Pay1 (108 byte rnet II, S rnet Proto smission re Sockets	bytes on w oad: eth ( is) inc: Apple_ inc: Apple_ icol Versio ontrol Pro	ire (168 Ethernet) f8:1e:71 n 4, Src tocol, Sr	§ bits), 211 bytes (78:1e:df:f8:1e:7; : 192.193.183.171 ( :c Port: https (44)	captured (1) ), DST: 10:- (192.193.183, (192.193.183), DST PORT	685 bfts) 40-ff1:0212:057 (00-40:ff1:0212:057) 570;05 (19:06,1,19:06,0,1,19) 500;05 (9906), 500; 811, 440; 2554, Len; 37	
in a le en u	e 23: 211 149, Payl (108 byte rnet II. 5 rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple_ icol Versio iontrol Pro : Layer	ire (168 Ethernet f8:1e:71 n 4, Src tocol, Sr	s bits), 211 bytes (78:1e:df:f8:1e:7; 192.193.183.171 ( - Port: https (44)	captured (1) (), DST: 10:: 192:193.183 (), DST Port	685 bits) 60751-002-2007 50751-002-1507-51.07 50768 (2008): 660: 811, eds: 2354, Len: 37 50888 (2008): 660: 811, eds: 2354, Len: 37	
aleenu	e 23: 211 149, Payl (108 byte rnet II. s rnet Proto smission C re Sockets	bytes on w oad: eth ( s) is: inc: Apple_ col Versio control Pro : Layer	ire (168 Ethernet) f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes / / (08:10:01":f8:10:77 : 192.199.183.171 : 192.199.183.171 : c Port: https (44)	captured (1) (), DST: 10:- (192.193.18) (), DST Port	685 bfts) def1;d:1;2;b27 def1;d:1;2;b27 def1;d:1;2;b27 def1;d:1;2;b27 def2;0;1;0;1;0;1;0;1;0;1;1;0;1;1;0;1;1;1;1;	
	e 23: 211 149, Payl (108 byte rnet II., S rnet Proto smission C re Sockets	bytes on w oad: eth ( is) in: Apple. col Versio iontrol Pro Layer	ire (168) Ethernet f8:1e:71 n 4, Src tocol, Sr	5 bits), 211 bytes (f8:1e:df:f6:1e:7; 192,193,183,171 (44:	captured (1) 1), DST: 10:: 192:193.183 3), DST Port	685 bfts) 40-ff1:0212:057 (00-f61:0212:057 (1910), 69(1:06.01.19) 59086 (9906), 69(1:011, 440: 2044, Len: 37	
	e 23: 211 149, Payl (108 byte rnet II., S rnet Proto smission C re Sockets	bytes on w oad: eth ( is) rrc: Apple_ col Versio control Pro i Layer	ire (168) Ethernet; f8:1e:71 n 4, Src tocol, Sr	8 bits), 211 bytes (f8:1e:df:f8:1e:7; 192,193,183,171 c Port: https (44)	captured (1) 1), Dst: 10:4 (192.193.183 3), Dst Port	688 bits) 6073:02:230 5073:00:230 5008 (2008): 800:319 (05.6.139) 5008 (2008): 800:31, edi:3254, Len:37	
	e 23: 211 149, Payl (108 byte rnet II., S rnet Proto smission C re Sockets	bytes on w oad: eth ( is) inc: Apple_ col Version i Layer	ire (168) Ethernet; f8:1e:71 n 4, Src tocol, Sr	5 bits), 211 bytes (f8:1e:df:f8:1e:7; 192,193,183,121 t c Pert: https (44)	captured (1), DSt: 10:(122,139.188)	685 bfts) (do:40:f1:2:12:b7 (do:40:f1:2:12:b7) (do:40:f1:2:12:b7) (do:40:f1:2:12:b7) (do:40:f1:2:12:12:b7) (do:40:f1:2:12:12:12:12:12:12:12:12:12:12:12:12:	
ant T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II., s rnet Proto mission C re Sockets	bytes on w oad: eth ( is) rrC: Apple_ col Versio ontrol Pro : Layer	ire (168) Ethernet f8:1e:71 n 4, Src tocol, Si	<pre>0 bits), 211 bytes (f8:1e:df:f8:1e:7; 192.193.183.171 ( c Port: https (44) 0 00 00 65 6e 30 0</pre>	captured (1) (), DSt: 10:: (192,193,183) (), DSt Port	68 5H3) (613:02):25:03 (19:05) (19:05): 05:13:03 (19:05) 2008 (2000), 50: 81, 44:234, 140:27 2008 (2000), 50: 81, 44:234, 140:27	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II, s rnet Prote smission C e Sockets	bytes on w oad: eth ( is) inc: Apple_ col Versio control Pro Layer	00 01 0 00 02 0	<pre>0 bits), 211 bytes (f8:1e:df:f8:1e:7; 192.193.183.121 i 192.193.183.121 i c Pert: https (44) 0 00 00 65 6e 10 0</pre>	captured (1 (), Dit: 10: (192.19),183 (), Dit Port 0 1 0 1	68 5/15) 6073/02/15/02/15/02/15/05/ 17/10/05/15/02/05/05/ 58088 (2008): 660 84: 2354, Len: 37 58088 (2008): 660 84: 454: 2354, Len: 37	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II., S rnet Proto smission C re Sockets	bytes on w oad: eth ( is) ir: Apple. ic: App	00 01 0 00 00 0 00 00 0 00 00 0 00 00 0	0 00 00 05 06 10 0 0 00 00 05 06 10 0 0 00 00 05 06 10 0 0 00 00 00 00 00 00 00	captured (1 1), Dst: 10:: (192.193.183. 1), Dst Port 1), Dst Port 0 1 0	ese bris) #01713021217b70 (0940971302121b7) #01713021217b70 (0940971302121b7) * 50206 (09060), Seg: All, Ack: 2054, Len: 27 *	
an T: ta he te an cu	e 23: 211 149, Payl (108 byte rnet II, s rnet Proto smission C e Sockets	bytes on w oad: eth ( is) inc: Apple. col Versio control Pro Layer Layer 0 01 00 00 0 00 00 0 00 00 0 00 00 0 00 00	00 01 0 00 02 0 00 00 0 00 00 0 00 00 0 00 00 0 00 00	0 00 00 05 56 30 0 0 00 00 05 56 30 0 0 00 00 05 66 30 0 0 00 00 00 00 00 0 0 0 00 00 00 00 0 0 0 00 00 00 00 0 0 0 00 00 0 0 0 00 00 0 0 0 00 00 0 0 0 00 0 0 0 00 0 0 0 0 0	captured (1 ), Dit: 10: (192.19),183 ), Dit Port 0 1 0 1 0 0	665 bfts) 667 bfts):125 (10-467 fid:121b7) 175 bft: 150 (10-467 fid:121b7) 180 (1000), isq: 811, 441: 234, Len: 37 190 (1000), isq: 811, 441: 234, Len: 37	

## iPAD capture shows Packet Loss on Outbound Stream

The capture used previously shows a 5.3% packet loss for the outbound stream. It is not possible to measure outbound UDP packet loss (because the protocol is connectionless).

			10.8)						
	Detected	4 RTP stre	ams. Choose or	ne for forward a	nd revers	e direction fo	or analysis		
Src por	Dst addr	Dst poi	SSRC	Payload	Packe	Lost	Max Delta (m	Max Jitter (r	Mean Jitter
65219	81.144.171.73	16000	0xDCF9CE07	g711U	3944	0 (0.0%)	51.37	9.66	6.09
65219	81.144.171.73	16000	0x6B6570ED	RTPType-100	2114	118 (5.3%)	0.00	0.00	0.00
16000	192.168.1.107	65219	0xA5458BCD	g711U	2387	0 (0.0%)	101.77	14.06	7.32
J	Src por 65219 65219 16000 16000	Detected Src por Dst addr 65219 81.144.171.73 65219 81.144.171.73 16000 192.168.1.107 16000 192.168.1.107	Detected         4 RTP street           Src por         Dst addr         Dst poi           65219         81.144.171.73         16000           65219         81.144.171.73         16000           16000         192.168.1.107         65219           16000         192.168.1.107         65219	Detected 4 RTP streams. Choose or           Src por         Dst addr         Dst pol         SSRC           65219         81.144.171.73         16000         0xDCF9CE07           65219         81.144.171.73         16000         0x686570ED           16000         192.168.1.107         65219         0xCR6E82EF           16000         192.168.1.107         65219         0xA54588CD	Detected 4 RTP streams. Choose one for forward at           Src por         Dst addr         Dst por         SSRC         Payload           65219         81.144.171.73         16000         0xDCF9CE07         g711U           65219         81.144.171.73         16000         0x6B6570ED         RTPType-100           16000         192.165.1.107         65219         0xCB6EB2EF         RTPType-100           16000         192.168.1.107         65219         0xA5458BCD         g711U	Detected 4 RTP streams. Choose one for forward and revers           Src por         Dst addr         Dst poi         SSRC         Payload         Packe           65219         81.144.171.73         16000         0xDCF9CE07         g711U         3944           65219         81.144.171.73         16000         0xDCF9CE07         g711U         3944           65219         81.144.171.73         16000         0x686570ED         RTPType-100         2114           16000         192.168.1.107         65219         0xA54588CD         g711U         2387	Src por         Dst addr         Dst poi         SSRC         Payload         Packe         Lost           65219         81.144.171.73         16000         0xDCF9CE07         g711U         3944         0 (0.0%)           65219         81.144.171.73         16000         0x0EF9CE07         g711U         3944         0 (0.0%)           65219         81.144.171.73         16000         0x686570ED         RTPType-100         2114         118 (5.3%)           16000         192.168.1.107         65219         0xCR6E82EF         RTPType-100         1577         932 (37.1%)           16000         192.168.1.107         65219         0xA54588CD         g711U         2387         0 (0.0%)	Detected 4 RTP streams. Choose or forward and reverse direction forward and reverse direction of analysis           Src por         Dst addr         Dst por         SSRC         Payload         Packe         Lost         Max Delta (m           65219         81.144.171.73         16000         0xDCF9CE07         g7110         3944         0 (0.0%)         51.37           65219         81.144.171.73         16000         0x6B6570ED         RTPType-100         2114         118 (S.3%)         0.00           16000         192.165.1.107         65219         0xCB6E2EF         RTPType-100         1577         952 (37.1%)         0.00           16000         192.168.1.107         65219         0xA5458BCD         g7110         2387         0 (0.0%)         101.77	Detected 4 RTP streams. Choose or forward and reverse direction forwards.           Src por         Dst addr         Dst poi         SSRC         Payload         Packe         Lost         Max Delta (m)         Max Jitter (r)           65219         81.144.171.73         16000         0xDCF9CE07         g711U         3944         0 (0.0%)         51.37         9.66           65219         81.144.171.73         16000         0xB6570ED         RTPType-100         2114         118 (5.3%)         0.00         0.00           16000         192.168.1.107         65219         0xA5458BCD         g711U         2387         0 (0.0%)         101.77         14.06

This implies that there was packet loss between the IPAD and the MAC performing the packet capture! It is suspected this could be related to CPU load on the MAC but it implies that this could skew results because there could be similar levels of inbound packet loss.

Capturing at the local network's router has shown some packets appear as Comfort Noise initiating from the iPAD; this seems related to the wireshark decoding.

Document Title:	Diagnosing Media	ssues on the Fusion Platf	Classification	PUBLIC	
				Page	69 of 74



## **Captures from a Browser**

The analysis of the wireshark capture is identical to the previous section and will not be covered again. In addition, Chrome has some additional metrics which can be used:

When a call is active point the Chrome Browser to:

chrome://webrtc-internals/

When a call is active you can view the number of dropped packets:

#### Statistics ssrc\_447873670

```
cname:rRI9FQ7YIbid7KWN
msid:RUijbqmNT0IKkj9MicB1cb7nAISTSKfilhaq a0
mslabel:RUijbqmNT0IKkj9MicB1cb7nAISTSKfilhaq
label:RUijbqmNT0IKkj9MicB1cb7nAISTSKfilhaqa0
```

timestamp	Wed Oct 23 2013 13:40:06 GMT+0100 (BST)
ssrc	447873670
googTrackId	aO
transportId	Channel-audio-1
audioOutputLevel	32
bytesReceived	671520
googJitterReceived	5
packetsReceived	4197
packetsLost	1

Web-rtc-internals-parameters from TestRTC, is a good tutorial for understanding the metrics in webrtc-internals:

https://testrtc.com/webrtc-internals-parameters/

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	70 of 74



# **Bandwidth Estimates**

From webrtc-internals the bandwidth estimates can tell you how the adaptive bitrate is behaving.

▼ Stats graphs for bweforvideo (VideoBwe)

bweCompound						goog	Available	Receive	Bandwid	dth		goog	BucketD	elay			
googAvailableSendBandwidth					1.5 M				1 111		500 k					1	20
✓ qoogTargetEncBitrateCorrected											400 k						15
googActualEncBitrate				press	Hanfun 18M					-	300 <u>k</u>						10
googRetransmitBitrate				A	0.6.14					3	200 k				1	1.11.1	
goog I ransmitBitrate				1	0.0 1						100 k						\$
					0.0 M	L					0 k	L					<b>NLARI</b>
	30:00	11:31:00	11:32:00	11:33:00	11:34:00	30:00	11:31:00	11:32:00	11:33:00	11:34:00		30:00	11:31:00	11:32:00	11:33:00	11:34:00	

A video encoder has to make a lot of calculations based on how it perceives the network conditions and how the receiver is reporting theirs.

The following attributes are present:

- googAvailableReceiveBandwidth
  - the bandwidth that is available for receiving video data
- googAvailableSendBandwidth
  - $\circ$   $\;$  the bandwidth that is available for sending video data
- googTargetEncBitrate
  - the target bitrate of the the video encoder, it will try and fill the available bandwidth

Document Title:	Diagnosing Media	Issues on the Fusion Platfe	Classification	PUBLIC	
				Page	71 of 74





# **Local Firewall Configuration**

The local firewall configuration for Media broker must allow inbound UDP packets and initial outbound UDP packets for the media to establish.

The following is a sample setup for configuring the *firewalld* service to allow inbound traffic:

```
1. Install Firewalld:
         yum install firewalld
2. Add an interface to a public zone:
         sudo firewall-cmd --zone=public --permanent
   --change-interface=eno16777984
3. This can also be set in:
         vi /etc/sysconfig/network-scripts/ifcfg-eno16777984 ZONE=public
Configure the Media Broker XML script:
         vi /etc/firewalld/services/csdk-mb.xml
   <?xml version="1.0" encoding="utf-8"?>
   <service>
     <short>MB</short>
   <description>Service Description for Media Broker Service</description>
     <port protocol="udp" port="16000"/>
     <port protocol="udp" port="17000-17999"/>
     <port protocol="tcp" port="8092"/>
   </service>
```

5. Reload to see new services:

```
sudo firewall-cmd --reload
```

6. Apply Services to Zones:

sudo firewall-cmd --zone=public --permanent --add-service=csdk-mb

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	72 of 74


## **Testing the Local Firewall Ports**

You can test if packets can be sent from a public network through to the Media Broker by using a packet sending tool.

Tools like *Packet Sender* (<u>https://packetsender.com</u>) allow you to craft UDP packets and direct them to an IP and Port.

Using *tcpdump* on the Media Broker port you can listen for inbound traffic and determine if your sent packets arrive.

e To									12					
	ools Help													
Name	test													
ASCIL	tect							-						
JEV	74 65 72 74													
	74037374	1			- n	-				L.				
Addres	ss 1/2.31.2	30.45				Port	16000		Resend Dela	ау О		S UDP 🔹	Send	Save
earch S	Saved Packet	s										Delete	Saved Packet	Persistent T
	Send N	ame Res	end (sec) Te	Address	To Port	Meth	bd		ASCII				Hex	
	Send te	st 0	17	2.31.250.45 1	L6000	UDP	test			7	4 65 73 74			
Class										ली । T 66-	Canala	-	Tracffer David and	Court Chi
Clear	Log								[	Log Traffic	Save Lo	g Save	Traffic Packet	Copy to Clipbo
Clear	Log Time	From	IP From Por	t To IP	' To	o Port	Method	Error	[	Uog Traffic	Save Lo	g Save	Traffic Packet Hex	Copy to Clipbo
Clear	Log Time 2:41:35.396	From Im You	IP From Por 8888	rt To IP 172.31.250	7 Tc 0.45 160	<b>5 Port</b> 000	Method UDP	Error	test	✓ Log Traffic ASCII	Save Lo	g Save	Traffic Packet Hex 4	Copy to Clipbo

The following filter can be used to only show UDP packets arriving at port 16000: **tcpdump -i any udp port 16000** 

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode listening on any, link-type LINUX\_SLL (Linux cooked), capture size 262144 bytes 14:28:32.936669 IP 172.31.253.96.ddi-udp-1 > centos71.16000: UDP, length 4

Document Title:	Diagnosing Media Issues on the Fusion Platform	Classification	PUBLIC
		Page	73 of 74



A very simply way of sending UDP traffic from the Media Broker machine is to use the following command, specifying the destination address:

## echo "This is my data" > /dev/udp/{{DESTINATION ADDRESS}}/3000

Cit.	d *Wireless Network Connection								
Luc	Eile Edit <u>V</u> iew <u>G</u> o <u>C</u> apture <u>A</u> nalyze <u>S</u> tatistics Telephony <u>Wi</u> reless <u>T</u> ools <u>H</u> elp								
4 =	6 💿 👔	🗅 🕅 🖸 🔍 👄 👄	2 T & 🗐 🗐 (	Ð, Q, Q, 🎹					
Udp.dstport == 3000									
No.	Time	Source	Src Port Destination	Dest Port	Protocol	Length Info			
45.	24.512584	172.31.250.45	45690 172.31.2	53.96 30	00 DIS	60 PDUType	: Unknown		
45.	24.512648	172.31.253.96	45690 172.31.2	50.45 30	00 ICMP	86 Destina	tion unreachable	e (Port unreachable)	
34.	53.000917	172.31.250.45	34713 172.31.2	53.96 30	00 DIS	60 PDUType	: Unknown		
34.	53.000988	172.31.253.96	34713 172.31.2	50.45 30	00 ICMP	86 Destina	tion unreachable	e (Port unreachable)	
<u> </u>	54.598270	172.31.250.45	40258 172.31.2	53.96 30	00 DIS	60 PDUType	: Unknow		
L 40.	54.598355	172.31.253.96	40258 172.31.2	50.45 30	00 ICMP	86 Destina	tion unrèachable	e (Port unreachable)	
P Etr	hernet II, Sr	<pre>rc: HewlettP_71:c7:0</pre>	0 (e4:11:5b:71:c7:00	), Dst: IntelCor	20:11:10	(44:85:00:2d:11:fd)	)		
<ul> <li>Eth</li> <li>Int</li> <li>Use</li> <li>Dis</li> </ul>	hernet II, Sr ternet Protoc er Datagram F stributed Int	c: HewlettP_71:c7:0 col Version 4, Src: Protocol, Src Port: ceractive Simulation	0 (e4:11:5b:71:c7:00 172.31.250.45, Dst: 40258 (40258), Dst F	), Dst: IntelCor 172.31.253.96 Port: 3000 (3000)	_20:11:10	(44:85:00:2d:11:fd)	)		
<ul> <li>Etr</li> <li>Int</li> <li>Use</li> <li>Dis</li> </ul>	hernet II, Sr ternet Protoc er Datagram P stributed Int	<pre>c: HewlettP_71:c7:6 col Version 4, Src: Protocol, Src Port: ceractive Simulation</pre>	00 (e4:11:5b:71:c7:00 172.31.250.45, Dst: 40258 (40258), Dst F	), DST: IntelCor 172.31.253.96 Port: 3000 (3000)	_2d:11:+d	(44:85:00:2d:11:fd)	)		
<ul> <li>Etr</li> <li>Int</li> <li>Use</li> <li>Dis</li> </ul>	hernet II, Sr ternet Protoc er Datagram F stributed Int 44 85 00 2d	<pre>rc: HewlettP_71:c7:0 col Version 4, Src: Protocol, Src Port: ceractive Simulation 111 fd e4 11 5b 71</pre>	0 (e4:11:5b:71:c7:00 172.31.250.45, Dst: 40258 (40258), Dst F c7 00 08 00 45 00	D [q	_2d:11:td	(44:85:00:2d:11:fd)	)		
<ul> <li>Etr</li> <li>Int</li> <li>Use</li> <li>Dis</li> <li>0000</li> <li>0010</li> </ul>	At 85 00 2d 00 2c 33 05	<pre>c: HewlettP_71:c7:6 col Version 4, Src: Protocol, Src Port: ceractive Simulation   11 fd e4 11 5b 71 40 00 3f 11 b8 ee</pre>	0 (e4:11:5b:71:c7:00 172.31.250.45, Dst: 40258 (40258), Dst F c7 00 08 00 45 00 ac 1f fa 2d ac 1f	), UST: IntelCop 172.31.253.96 ort: 3000 (3000) D [q .,3.@.?	_2d:11:Td	(44:85:00:2d:11:fd)	)		
<ul> <li>Etr</li> <li>Int</li> <li>Use</li> <li>Dis</li> <li>0000</li> <li>0010</li> <li>0020</li> </ul>	At 85 00 2d 44 85 00 2d 00 2c 33 05 fd 60 9d 42 72 00 42	<pre>c: HewlettP_71:c7:6 c: lewlettP_71:c7:6 protocol, Src Porti- ceractive Simulation 111 fd e4 11 5b 71 40 00 3f 11 b8 ce 20 b0 80 01 86 53 34 00 cf 11 d5 ce 20 b1 21 cf 21 cf 20 cf 21 cf 20 cf 21 cf 20 cf 20</pre>	0 (e4:11:5b:71:c7:00 172.31.250.45, D5t: 40258 (40258), Dst F c7 00 08 00 45 00 c7 00 08 00 45 00 c3 c1 ff a 2d ac 1f 54 68 69 73 20 69	), JST: IntelCor 172.31.253.96 ort: 3000 (3000) D [q .3.@.?e4This	_2d:11:Td	(44:85:00:2d:11:fd)	)		
<ul> <li>Etr</li> <li>Int</li> <li>Use</li> <li>Dis</li> </ul>	eenet II, Sr ternet Protoco er Datagram F stributed Int 44 85 00 2d 00 2c 33 05 fd 60 9d 42 73 20 6d 79	<pre>C: HewlettP_71:c7:6 col Version 4, Src: Protocol, Src Port: ceractive Simulation 1 11 fd e4 11 5b 71 40 00 3f 11 b8 ee 0 bb 80 01 8 65 34 20 64 61 74 61 0a</pre>	0 (e4:11:5b:71:c7:00 172:31:250:45, Dst: 40258 (40258), Dst F c7 00 08 00 45 00 ac 1f fa 2d ac 1f 54 68 69 73 20 69 00 00	D [q ,3.@.? , B e4This s my dat a	_2d:11:Td	(44:85:00:2d:11:fd)	)		

Obviously, the destination address will need to be publicly routable, so if your test machine is behind a NAT, this technique will not work. Suitable knowledge of your network's routing rules are required. It may be easier to verify simply the inbound and outbound STUN by establishing a call.

Document Title:	Diagnosing Media I	ssues on the Fusion Platf	Classification	PUBLIC	
				Page	74 of 74