Video over IP challenges.

Mary-Luc Champel
Thomson Grass Valley
21.04.2006
Outline

> Introduction

> Packet losses issues
  > What is the issue ?
  > Pro-MPEG FEC solution.

> Timing recovery issues.
  > What is the issue ?
  > Is there any solution available ?
Introduction

> There are many advantages for transporting video feeds over IP networks in a contribution environment:
  > It is a cost effective solution,
  > IP networks may be already present in the facilities,
  > IP is a simple technology everybody understands well,...

> In some aspects transporting video feeds over IP is like transporting any other data over IP: we want to receive the exact same data that was sent in its whole!
  ⇒ PACKET LOSS ISSUE

> But video is not just like any other data, it has some real-time aspects that need to be ensured: we want to receive the video packets in the exact same pace and order they were sent!
  ⇒ TIMING RECOVERY ISSUE
Video over IP.

- IETF has defined several RFCs so as to standardize Video transport over IP networks.

<table>
<thead>
<tr>
<th>OSI layer model</th>
<th>Application</th>
<th>Presentation</th>
<th>Session</th>
<th>Transport</th>
<th>Network</th>
<th>Data Link</th>
<th>Physical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>« user interface »</td>
<td>« translation »</td>
<td>« synchronization, session »</td>
<td>« packets, flow control, error handling »</td>
<td>« addressing, routing »</td>
<td>« data frames to bits »</td>
<td>« hardware »</td>
</tr>
</tbody>
</table>

- MPEG-2 AV
- MPEG-2 TS
  - (RFC 2250)
- RTP (RFC 1889)
- UDP (RFC 768)
- IP (RFC 791)
- RTSP (RFC 2326)
- RTCP (RFC 1889)
- IGMP (RFC 2236)
- Any
Outline

> Introduction

> Packet losses issues
  > What is the issue ?
  > Pro-MPEG FEC solution.

> Timing recovery issues.
  > What is the issue ?
  > Is there any solution available ?
Origin of errors.

> Bit alteration at the physical layer are converted into IP frame drops due to higher layers CRC mechanisms.

> Ethernet switches or IP routers may also drop packets if congestions occur on their output ports. One or more packet may be discarded.
Differences with previous FEC schemes

> Typical FEC schemes are usually deployed at the network PHYSICAL layer and in the specific case of IP networks, errors that are not corrected at the PHYSICAL layer are translated into packet losses due to CRC mechanisms at the TRANSPORT layer (TCP, UDP ...).

> Typical FEC schemes usually intend to correct bit errors whereas Pro-MPEG FEC scheme intends to recover packet losses.

> Pro-MPEG FEC scheme is dedicated to Video over IP and is deployed at the RTP level (network TRANSPORT layer).

> Unlike typical FEC schemes, Pro-MPEG FEC scheme relies on very simple algorithms that can be implemented with no hardware specific parts.

> The FEC packet is a regular RTP packet with a specific payload type that identifies that the RTP packet contains FEC information.

> The RTP payload of the FEC packet is computed by XORing the payloads of associated data packets and allows to recover the payload of a lost packet.
1D FEC scheme overview.

- FEC packets are computed over the columns of a sizeable (L&D parameters) matrix of data packets.
- FEC packets computed over columns bring error bursts correction capability.
- The FEC scheme implies a latency equal to the buffering of the data matrix and its FEC packets.
**Example of correction failures.**

Here are the two main error patterns that cannot be corrected:

2 data packets on the same column

<table>
<thead>
<tr>
<th></th>
<th>FEC₀</th>
<th>FEC₁</th>
<th>FEC₂</th>
<th>FEC₃</th>
<th>FEC₄</th>
<th>FEC₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td></td>
</tr>
</tbody>
</table>

1 data packet and its associated FEC

<table>
<thead>
<tr>
<th></th>
<th>FEC₀</th>
<th>FEC₁</th>
<th>FEC₂</th>
<th>FEC₃</th>
<th>FEC₄</th>
<th>FEC₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>16</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FEC₀</td>
<td>FEC₁</td>
<td>FEC₂</td>
<td>FEC₄</td>
<td>FEC₅</td>
<td></td>
</tr>
</tbody>
</table>

2 data packets on the same column

<table>
<thead>
<tr>
<th></th>
<th>FEC₀</th>
<th>FEC₁</th>
<th>FEC₂</th>
<th>FEC₃</th>
<th>FEC₄</th>
<th>FEC₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td></td>
</tr>
</tbody>
</table>

1 data packet and its associated FEC

<table>
<thead>
<tr>
<th></th>
<th>FEC₀</th>
<th>FEC₁</th>
<th>FEC₂</th>
<th>FEC₃</th>
<th>FEC₄</th>
<th>FEC₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>16</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FEC₀</td>
<td>FEC₁</td>
<td>FEC₂</td>
<td>FEC₄</td>
<td>FEC₅</td>
<td></td>
</tr>
</tbody>
</table>

FEC is the Forward Error Correction.
2D FEC scheme overview.

- FEC packets are computed over rows and/or columns of a sizeable (L&D) matrix.
- FEC packets computed over columns bring error bursts correction capability.
- FEC packets computed over rows bring additional single errors correction capability.
- The FEC implies a latency equal to the buffering of the data matrix and its FEC packets.
Example of correction success.

6x6 data matrix with 9 data packets lost and 1 FEC packet lost

The 9 missing data packets are successfully recovered!!!
Example of correction failures.

Here are the two main error patterns that cannot be corrected:

1 data packet and its 2 associated FEC packets

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>16</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
</tr>
</tbody>
</table>

4 data packets positioned on exactly 2 rows and 2 columns

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>17</td>
<td>18</td>
<td>23</td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>22</td>
<td>23</td>
<td>27</td>
<td>29</td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>33</td>
<td>34</td>
<td>35</td>
</tr>
</tbody>
</table>

FEC'0 FEC'1 FEC'2 FEC'3 FEC'4 FEC'5
FEC packet datagram.

FEC packets are RTP packets that are easily identified thanks to their payload type (PT in RTP header).

Easy identification of data packets covered by a given FEC packet:
- Offset (L),
- NA (D),
- SN base (Sequence number of first associated data packet),
- D (Line or Row FEC packet).

FEC header and payload allow to recover both RTP header and RTP payload of a missing data packet.

The FEC can use various specific computation type (type field, XOR = 0, Reed-Solomon = 1, ...).
Configurations and performances.

The following examples are given for a typical use of video streaming over IP:
Video at 4Mb/s and transported with 7 MPEG-2 TS packets per IP packet.

<table>
<thead>
<tr>
<th>L</th>
<th>D</th>
<th>I</th>
<th>PLR</th>
<th>overhead</th>
<th>latency (ms)</th>
<th>MTBE before FEC (sec)</th>
<th>MTBE after FEC (day)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>1.E-03</td>
<td>20%</td>
<td>525</td>
<td>2.74</td>
<td>32.87</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>1.E-04</td>
<td>20%</td>
<td>525</td>
<td>27.40</td>
<td>42105.50</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>1.E-05</td>
<td>20%</td>
<td>525</td>
<td>274.00</td>
<td>42336213.68</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.E-03</td>
<td>40%</td>
<td>127</td>
<td>2.74</td>
<td>35.13</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.E-04</td>
<td>40%</td>
<td>127</td>
<td>27.40</td>
<td>35431.83</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.E-05</td>
<td>40%</td>
<td>127</td>
<td>274.00</td>
<td>33688413.06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>L</th>
<th>D</th>
<th>I</th>
<th>PLR</th>
<th>overhead</th>
<th>latency (ms)</th>
<th>MTBE before FEC (sec)</th>
<th>MTBE after FEC (day)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>1.E-05</td>
<td>10%</td>
<td>525</td>
<td>274.00</td>
<td>70.47</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>1.E-06</td>
<td>10%</td>
<td>525</td>
<td>2740.00</td>
<td>7047.13</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.E-05</td>
<td>20%</td>
<td>127</td>
<td>274.00</td>
<td>158.56</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.E-06</td>
<td>20%</td>
<td>127</td>
<td>2740.00</td>
<td>15856.19</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>1.0</td>
<td>1.E-05</td>
<td>10%</td>
<td>30</td>
<td>274.00</td>
<td>70.47</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>1.0</td>
<td>1.E-06</td>
<td>10%</td>
<td>30</td>
<td>2740.00</td>
<td>7047.13</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1.0</td>
<td>1.E-05</td>
<td>20%</td>
<td>16</td>
<td>274.00</td>
<td>158.56</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1.0</td>
<td>1.E-06</td>
<td>20%</td>
<td>16</td>
<td>2740.00</td>
<td>15856.19</td>
</tr>
</tbody>
</table>

- L, D: dimension of the matrix
- I: Interleaving depth used in FEC sequencing.
- PLR: Packet Loss Rate of the Network
- MTBE: Mean Time Between Errors
Pros of the Pro-MPEG FEC scheme.

- This FEC scheme is based on an existing standard (RFC 2733).

- This FEC proposal is very adapted to packet losses correction and can deal with single and/or burst packet losses.

- This FEC proposal can be adapted to many different needs thanks to its variety of parameters (profiles, size and shape of the matrix of media packet, sequencing of FEC packets among data packets).

- This FEC proposal can use XOR computation and thus requires no hardware specific parts and can be computed very quickly on any kind of CPU.

- This FEC proposal is totally transparent for receivers that do not handle it.
  - no changes have to be done on applications/end devices that do not use this FEC today.
Outline

> Introduction

> Packet losses issues
   > What is the issue ?
   > Pro-MPEG FEC solution.

> Timing recovery issues.
   > What is the issue ?
   > Is there any solution available ?
Timing Issues

> Receiver clock must lock to headend’s master clock
> Avoid receiver’s STC clock drifting in order not to cause overflow or underflow
> PCR clock packet is generated by broadcasters within every 100ms (or 40ms by DVB)

> Broadcaster’s headend will control the broadcast packet rate (pace) to ensure not to overflow or underflow the receiver VBV buffer for every program (channel)
Some Timing definitions

- System Clock (13818-1 Section 2.4.2.1)
  - 27 MHz
  - Accuracy: ± 30 ppm (± 810 Hz)
    - ± 5 ppm (DVB, ETR 154 Section 4.1.3)
  - Slew: .075 Hz per second (primarily to meet Chroma Subcarrier generation requirements)

- Program Clock Reference (PCR)
  - “is the time encoded...measured in units of the 27MHz system clock” (2.4.2.2)
  - 33-bit PCR_base in 90KHz units and 9-bit PCR_ext in 27MHz units:
    
    \[
    \text{PCR}(i) = \text{PCR}\_\text{base}(i) \times 300 + \text{PCR}\_\text{ext}(i) \\
    \text{PCR}\_\text{base}(i) = \left(\frac{\text{system\_clock\_frequency} \times t(i)}{300}\right) \mod 2^{33} \\
    \text{PCR}\_\text{ext}(i) = \left(\frac{\text{system\_clock\_frequency} \times t(i))}{300}\right) \mod 300
    \]
  - Accuracy of ± 500ns “due to imprecision in initial PCR value and in PCR modification during remultiplexing. *It does not include any error in packet arrival time due to network jitter.*” (2.4.2.2)
PCR relevance

- Video encoder
  - PCR is set here!
  - Measuring PCR here is relevant!

- RTP packetizer

- IP Network
  - PCR is used here!
  - Measuring PCR here is relevant!

- RTP depacketizer

- Video decoder
  - Measuring PCR here is irrelevant!
RTP timestamps allow to remove IP network jitter but
- RTP timestamps are only 90kHz
- PCR are 27Mhz
- PCR will still suffer from:
  - up to 11μs of jitter,
  - de-blocking management.

How initial stream pace before IP/RTP packetizer (at the emitter side) can be reproduced after de-packetizer (at the receiver side)?
Transport Stream types

CBR stream

Piece-wise constant VBR stream

Non piece-wise constant VBR stream

regular TS-packet  TS-packets carrying a PCR
RFC 2250 weakness.

Partial TS extraction
Packetization

Jitter inducing network

RTP timing recovery
De-Packetization

TS-packets without relevance to the transmitted programme
TS-packets with relevance to the transmitted programme
TS-packets carrying a PCR

dT1
dTA
dTB
dT2 # dT1
Service rate recovery - basic solution (1/2)

> All timing information is derived ultimately from PCR reference in AV Stream
  > PCR values are required every 40ms to 100ms depending on system specification (i.e. ATSC, DVB, etc.).
  > TS Packets are de-packetized at a CBR between two PCRs.
Service rate recovery - basic solution (2/2)

- **Pros**
  - Suitable for CBR (simply recomputes packet rate on basis of amount of data received between 2 PCR packets).
  - Compatible with RFC 2250.
  - Straight-forward approach.

- **Cons**
  - Awkwardly adapted for piece-wise constant VBR streams.
  - Unsuitable for non piece-wise constant VBR streams.
  - PCR information is used to provide a function it was not intended to.
  - At least 2 PCRs shall be received before TS packets can be unpacked.
  - Weak robustness against packet losses (especially packets with PCRs).
  - Unsuitable for Multiple Program Transport Streams (MPTS)
  - No use of the RTP timestamp (only available timing information about sending time).
Service rate recovery - advanced solutions

- All timing information is inserted in AV Stream
  - For each TS packet present in the RTP packet, a timing tags are generated.
  - Up to seven tags are either:
    - A) added in a RTP header extension (tags = timestamps, eg 27MHz/42 bits)
    - B) added at the end of the RTP payload (tags = timestamps)
    - C) added at the end of the RTP payload (tags = delta time referenced to RTP timestamp of the packet)
# Service rate recovery - advanced solution

<table>
<thead>
<tr>
<th>A</th>
<th>Timestamps in header extension</th>
</tr>
</thead>
</table>
| **Cons** | Not compatible with many legacy implementations of RFC2250.  
No use of the RTP timestamp.  
Weak robustness against packet losses (extension header is not handled by FEC). |
| **Pros** | Suitable for CBR and VBR streams.  
Compatible with RFC2250 |

<table>
<thead>
<tr>
<th>B</th>
<th>Timestamps at end of payload</th>
</tr>
</thead>
</table>
| **Cons** | No use of the RTP timestamp.  
Not strictly compatible with RFC2250. |
| **Pros** | Suitable for CBR and VBR streams.  
Compatible with legacy implementations of RFC2250.  
Strong robustness against packet losses (time info is in RTP payload) |

<table>
<thead>
<tr>
<th>C</th>
<th>Delta time tags at end of payload</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cons</strong></td>
<td>Not strictly compatible with RFC2250.</td>
</tr>
</tbody>
</table>
| **Pros** | Suitable for CBR and VBR streams.  
Compatible with legacy implementations of RFC2250.  
Strong robustness against packet losses (time info is in RTP payload).  
Makes the best use of RTP network clock recovery. |
Conclusion

- Pro-MPEG has come with an effective solution for the packet loss issue (CoP#3r2).

- VSF is currently working with SMPTE so as to have this solution become an accredited standard.

- But now VSF meets a new challenge: to define a consensual solution for the timing recovery issue...
Thank you for your attention.

Mary-Luc Champel
Thomson Corporate Research
Network & Communications Laboratory

mary-luc.champel@thomson.net