ストリーミングシステム（I）
プロトコルとオーサリング

圧縮技術・伝送技術・サーバ構築実践

森出 茂樹　moride@portside.net

アジェンダ

- 圧縮技術
  - MPEGを中心に音声コーデック、画像コーデック解説
- 伝送技術
  - RTP/RTSP を実際のフローもまじえて解説
- サーバ構築実践
  - カーネルチューニング、トラブルシューティング

ストリーミングの歴史・技術の軌跡

- 1990年 はじめてのRFC=1190 現在もExperimental
- 1990年代前半はMBONEで発達
- 1996年 RTP RFC1889
- 1994年 StreamWorks 1.0 はじめての商用アプリとして
- 1995年 RealAudio1.0
- 1996年 NTT SoftwareVision
- 1997年 Microsoft NetShow2.0
- 1999年頃からRFC規格に基づく動作をするような製品
- ストリーミングが技術として成熟してきて、相互運用の要求
  RealServer8 では QuickTime クライアントに向けて配信可能

ストリーミングの規格

- ストリーミングシステムの要素技術
  - 圧縮技術
  - 伝送技術
  - 制御技術
- 標準化機関

標準化団体	主な規格
IETF	プロトコル関係
W3C	SMIL
ITU-T	リアルタイム通信系コーデックとプロトコル
ISO/IEC	MPEGに代表される圧縮コーデック

圧縮技術

MPEGを中心に音声コーデック、画像コーデック解説

圧縮技術

- 圧縮を行う仕組み＝コーデック（CODEC）
  - COder-DECoderの略
- 圧縮技術には2つのルーツ
  - 旧CCITT、ITU-Tによる通信系の圧縮
    - デジタル携帯電話などが代表例
  - MPEGに代表される蓄積、放送系の規格
    - DVD、デジタル放送が旬

MPEG Family

- Moving Picture Experts Group
- MPEG-1
  - ビデオCDなどが対象
  - 1.5Mbpsまで
- MPEG-2
  - DVD,デジタル放送
  - 4～10Mbps程度、HDTVでは数十Mbps
- MPEG-4
  - インターネット、モバイル、マルチメディア志向
  - 高圧縮率、低ビットレート
- MPEG-7
  - マルチメディアコンテントの記述インターフェース
  - 検索するための標準、圧縮や伝送の規格ではない
- MPEG-21
  - コンテンツ配信、著作権保護、コンテンツID

Audio CODEC

音声、オーディオコーデック

Audio CODEC

- 代表的な標準

規格	特徴・用途
G.711	電話音声用 64kbps 無圧縮
G.723.1	A-CELP 5.3kbps, MP-MLQ 6.3kbps
G.729	CS-ASELP 8kbps
MPEG1 Audio Layer 3	MP3
MPEG2 AAC	DVD、デジタル放送

μ-law、A-law系 Audio CODEC

- G.711に使われる
- 1972年標準化
- 小さな音は高分解能で、大きな音は低分解能で扱う
- 広いダイナミックレンジを実現
- 圧縮率が低い（もしくは無いとも言える）
- 音質はそこそこ
- 64kbpsを前提（8bit×8KHz）
- 互換性を要求される場合以外はあまり使われない

CELP系 Audio CODEC

- CELP: code excited linear prediction
- G.723.1やG.729で使われる
- 人間の声を出す仕組みに特化して高圧縮率を実現
- あらかじめ用意した振幅データ(音声ベクトル)を組み合わせる
- 母音と子音に音を分解
- 喉や気道の共鳴などをモデル化
- 音声以外の音楽や自然の音などの圧縮には向かない

MPEG系 Audio CODEC

- 画像音声の統合型圧縮方法
- 標準化委員会の名前がそのまま規格名称になった
- 人間の聴覚特性を利用
- 小さな音や大きな音の前後の音を省略(聴覚心理的圧縮)
  - 原音を細かく周波数ごとに分解
  - レベルを比較し小さい音を省略
  - 時間変化も追跡
  - 大きな音に時間的に近い音も省略
- ステレオの場合には左右別々に処理せずに左右の差分を計算
- コーデックで処理するビットレートと音質が比例関係
- MP3のステレオ128kbpsがCD音質並といわれる

MPEG-1 MPEG1 Audio

- MPEG1オーディオの正式名称はISO/IEC IS11172-3
  - 1992年に標準化
- 帯域分割符号化(32バンド)
- 変形離散コサイン変換(MDCT)
- 聴覚心理的圧縮
- Layer Ⅰ
  - 通常256kStereo、384サンプル/フレーム
- Layer Ⅱ
  - 通常192kStereo、3個組1152サンプル/フレーム
- LayerⅢ(MP3)
  - 通常128kStereo、1152サンプル/フレーム、ハフマン符号化、MDCT

MPEG2 Audio

- MPEG1 Audio に以下の追加機能
  - マルチチャネル化5ch+LFE(Low Frequency Effect)
  - 他言語対応:7ヶ国語のサブオーディオ
  - 2つのモード
    - BC(Backward Compatible) MPEG1後方互換
    - AAC(Advanced Audio Coding)
- AACには3つのプロファイル
  - Main
  - LC(Low Complexity)
  - SSR(Scalable Sampling Rate)

MPEG4 Audio

- MPEG1,2に比べて高圧縮率、低ビットレートを志向
  - MIDI, 音声合成なども標準化
- 4つのプロファイル
  - スピーチ・プロファイル（音声専用）
  - 合成オーディオ・プロファイル（MIDI）
  - スケーラブル・プロファイル
    - スピーチ・プロファイルのスーパーセット
    - AAC、Twin-VQをエンハンスメントレイヤとして使用
  - メイン・プロファイル
    - 上記3つのプロファイルのスーパーセット
- 非常に複雑

Video CODEC

画像コーデック

Videoコーデック

- コーデックの共通化
- MPEGとITU-T H.系コーデックに対応関係

規格	特徴・用途
MPEG1/H.261	1.5Mビット/秒程度、主にCD-ROMなどの用途
MPEG2/H.262	4M～60Mビット/秒、DV,DVD,デジタルBSなどもこれ
MPEG4/H.263	主に移動通信での利用を想定したもの、低速に強く圧縮率大

MPEG系画像CODEC

- いろいろな圧縮方法を組みあわるところに特徴
  - 逆にいうと非常に込み入った内容
- MPEG系ビデオコーデック三つの原理
  - 1枚の画面中の圧縮（空間的相関関係を利用）
  - 画面間の差分を元にした圧縮（時間的相関関係を利用）
  - 純粋な符号圧縮（符号の出現確立を利用）

1枚の画面中の圧縮 MPEG画像CODEC

- 空間的相関関係を利用
  - 離散コサイン変換（DCT: Discrete Cosine Transform）
  - 細かい変化よりも大きな変化を優先
    - 物の表面のでこぼこより物の形
  - 細かいところの情報を故意に捨てる
  - 空間周波数の低周波成分の抜き出し

画面間の差分で圧縮 MPEG画像CODEC

- 時間的相関関係を利用
  - 動いている部分だけを伝送
  - 単純に以前の画面と重ね合わせて違うところだけを抜き出す
  - カメラを振った時、移動方向と移動量だけを送る
  - 「動きベクトルの検出」

純粋な符号圧縮 MPEG画像CODEC

- 符号の出現確立を利用
  - 可変長符号化
  - エントロピー符号化
  - ZIP,LZHと同じ種類
  - 純粋なデータ圧縮

MPEG4

- MPEG1,2に比べて高圧縮率、低ビットレートを志向
- 3つの特徴
  - 符号化効率の改善
    - 各種予測処理、スプライトなど
  - 任意形状画像への対応
  - エラー耐性ツール
- 各種アルゴリズムの組み合わせをプロファイルとして定義
  - 9種類
- MPEG4 Ver2ではさらに拡張

MPEGシステムとは

- HTML のように個々のコンテンツの統合を行う
  - 時系列化
  - メディアの同期
  - スクランブル機能
- MEPG2-PS Program Stream
  - DVD や PC はこっち
  - PES(packetized Elementary Stream) をグループ化する
  - 最大64Kbyte
- MPEG2-TS Transport Stream
  - デジタルBSなどはこの方式
  - PESを再分割して細切れに
  - ATMでの伝送に最適化
  - 固定長188バイト(47x4)
    - 53バイトのATMセルに47バイトずつ載せる
- PS,TSともにPESは共通

伝送技術

RTP/RTSP を実際のフローもまじえて解説

擬似ストリーミング

- ストリーミングは専用のプロトコルではなくHTTPやFTPでも可能
- 擬似ストリーミングと呼ぶ
  - コーデックで圧縮したコンテンツをファイルとして置くだけ
  - 専用のストリーミングサーバを必要としない
  - 運用が簡単、アクセスの少ないコンテンツ、短いコンテンツに利用
- 擬似ストリーミングの問題
  - ライブ放送が出来ない。
  - 送出速度が制御されない
    - 常に最大スピードでコンテンツが送られ他の通信に悪影響を与える場合がある
  - 基本的にはコンテンツの最初からしか再生出来ない
    - サーバのコンテンツを丸ごと転送するだけ、途中からの再生や頭出しが出来ない
  - コンテンツが簡単にコピーされてしまう
  - 予期しないキャッシング
    - ネットワーク途中のキャッシュの影響で最新のコンテンツが見られない場合

RTP Real-time Transport Protocol RFC1889

- ストリーミング用の伝送を行う標準プロトコル
  - 主に多人数での電子会議を行うために作られた
  - 現在ではストリーミングのためのプロトコルとして利用
- RTPの役割は画像や音を運ぶトラック
  - データを識別する共通の方法とパケットのタイムスタンプのつけ方を定義
  - 荷札や送り状の書き方
  - 通常、音と画像は別々に梱包
  - 制御をするためのプロトコルRTCPも含まれる
- 荷物(パケット)の梱包方法にいては別規約(RFC)
  - オーディオやビデオのデータの圧縮方法ごとにペイロードフォーマットがある
  - シュミレーションデータなどのペイロードフォーマットもある
- RTP自身は通信に必要なリソースの予約やQoSは保証しない
  - アプリケーションで実現する必要がある
- 送出側へのフィードバック方法RTCPもRTP文書内で規定

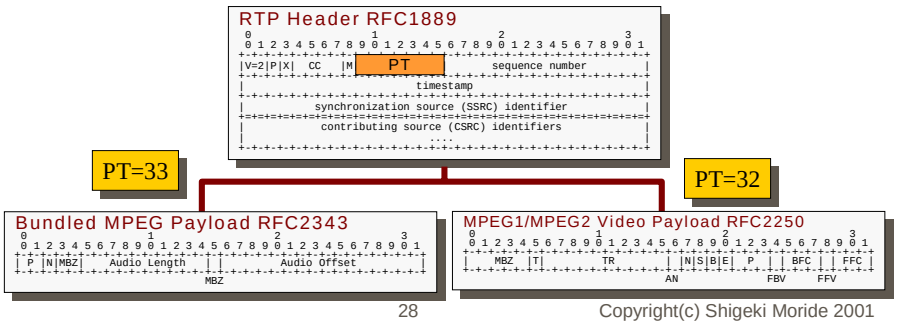
RTP関連のRFC

RTP基本規格	
RFC1889	RTP: A Transport Protocol for Real-Time Applications.
RFC1890	RTP Profile for Audio and Video Conferences with Minimal Control.
その他のペイロードに関する規格	
RFC2198	RTP Payload for Redundant Audio Data.
RFC2793	RTP Payload for Text Conversation.
RFC2833	RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals.
RTP全般に関する規格	
RFC2508	Compressing IP/UDP/RTP Headers for Low-Speed Serial Links.

ペイロードフォーマットの規格	
RFC2029	Sun's CellB Video Encoding.
RFC2032	H.261 VideOStreams.
RFC2035	JPEG-compressed Video.
RFC2038	MPEG1/MPEG2 Video.
RFC2190	H.263 VideOStreams.
RFC2250	MPEG1/MPEG2 Video.
RFC2343	Bundled MPEG EXPERIMENTAL
RFC2429	the 1998 Version of ITU-T Rec. H.263 Video (H.263+).
RFC2431	BT.656 Video Encoding.
RFC2435	JPEG-compressed Video.
RFC2658	PureVoice(tm) Audio.
RFC2862	Real-Time Pointers.
RFC3016	MPEG-4 Audio/Visual Streams.
RFC3047	ITU-T Recommendation G.722.1.

RTPパケット

- ヘッダーの後に各種ペイロードが連結
  - PTフィールド7ビットでペイロードを識別
  - 標準の型が PT=0～34 で定義済み
  - 標準以外のPayloadType は SDP rtpmap でダイナミックに規定



RTP Payload Type

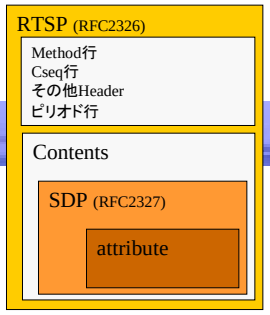
- 標準のPayloadType
- これ以外はダイナミックに
  - MPEG4なども標準外
  - IANAが採番
    rtp-parameters

PT	encoding name	audio/video (A/V)	clock rate (Hz)	channels (audio)	
0	PCMU	A	8000	1	[RFC1890]
1	1016	A	8000	1	[RFC1890]
2	G726-32	A	8000	1	[RFC1890]
3	GSM	A	8000	1	[RFC1890]
4	G723	A	8000	1	[Kumar]
5	DVI4	A	8000	1	[RFC1890]
6	DVI4	A	16000	1	[RFC1890]
7	LPC	A	8000	1	[RFC1890]
8	PCMA	A	8000	1	[RFC1890]
9	G722	A	8000	1	[RFC1890]
10	L16	A	44100	2	[RFC1890]
11	L16	A	44100	1	[RFC1890]
12	QCELP	A	8000	1	
13	Reserved	A			
14	MPA	A	90000		[RFC1890,2250]
15	G728	A	8000	1	[RFC1890]
16	DVI4	A	11025	1	[DiPol]
17	DVI4	A	22050	1	[DiPol]
18	G729	A	8000	1	
19	reserved	A			
20	unassigned	A			
21	unassigned	A			
22	unassigned	A			
23	unassigned	A			
24	unassigned	V			
25	CelB	V	90000		[RFC2029]
26	JPEG	V	90000		[RFC2435]
27	unassigned	V			
28	nv	V	90000		[RFC1890]
29	unassigned	V			
30	unassigned	V			
31	H261	V	90000		[RFC2032]
32	MPV	V	90000		[RFC2250]
33	MP2T	AV	90000		[RFC2250]
34	H263	V	90000		[Zhu]
35--71	unassigned	?			
72--76	reserved for RTCP conflict avoidance				[RFC1889]
77--95	unassigned	?			
96--127	dynamic	?			[RFC1890]

RTSP Real Time Streaming Protocol RFC2326

- 1998年ストリーミングを制御する方式として策定
  - RTPと協調して動作することを前提
  - ストリーミングサーバに対して再生・停止・早送り・巻き戻しなどを行う
  - ビデオの操作をするリモコン
  - ライブ放送の受信の制御も可能
- 基本的にはRTSP自身ではデータは配信せず、RTPが配信
  - 例外的にRTSPセッションの中にデータを埋め込む方法も用意
- RTSPはHTTPの拡張
  - HTTPではクライアントからのリクエストしか出来ない
  - RTSPではサーバ側からクライアントに情報を通知可
  - 同じコンテンツに対する操作を続けるためのセッションの概念
- ストリーミングソフトによって独自拡張あり

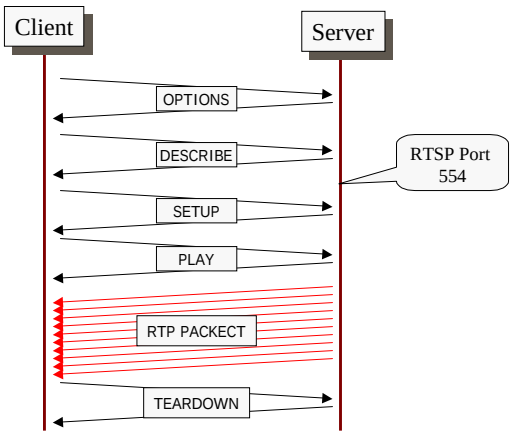
RTSPの特徴



- RTSPにはセッションの概念がある
  - セッション番号はサーバがランダムに生成
- RTSPには状態がある
  - SETUP, PLAY, RECORD, PAUSE, and TEARDOWN
- RTSP Methodはシーケンスで管理
  - クライアントはMethodにシーケンス番号をつけて投げる
  - サーバはどのMethodへの応答かをシーケンス番号をつけて応答
  - 応答を待たずに次のシーケンス番号で問い合わせるのもOK
- RTSPの文法には階層性がある
  - 階層ごとに独自の文法

RTSP/RTPのシーケンス概略

- OPTIONS
  - 使用可能な機能の交換
- DESCRIBE
  - コンテンツの情報取得
- SETUP
  - 伝送方法の決定
- PLAY
  - 再生
- TEARDOWN
  - 停止



OPTIONS RTSP Method

- 受け付けられるメソッドの交換
- C→Sは必須。S→Cはオプショナル

```
C->S:  OPTIONS * RTSP/1.0
       CSeq: 1
       Require: implicit-play
       Proxy-Require: gzipped-messages

S->C:  RTSP/1.0 200 OK
       CSeq: 1
       Public: DESCRIBE, SETUP, TEARDOWN, PLAY, PAUSE
```


DESCRIBE RTSP Method

- コンテンツの情報要求
- サーバーはSDPで応答

```
C->S: DESCRIBE rtsp://server.example.com/fizzle/foo RTSP/1.0
      CSeq: 312
      Accept: application/sdp, application/rtsl, application/mheg

S->C: RTSP/1.0 200 OK
      CSeq: 312
      Date: 23 Jan 1997 15:35:06 GMT
      Content-Type: application/sdp
      Content-Length: 376

      v=0
      o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
      s=SDP Seminar
      i=A Seminar on the session description protocol
      u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
      e=mjh@isi.edu (Mark Handley)
      c=IN IP4 224.2.17.12/127
      t=2873397496 2873404696
      a=recvonly
      m=audio 3456 RTP/AVP 0
      m=video 2232 RTP/AVP 31
      m=whiteboard 32416 UDP WB
      a=orient:portrait
```

SETUP RTSP Method

- 伝送方法とポートのネゴシエーション
- クライアントは使用可能なTransportを列挙
- サーバは選択して応答
- セッション番号の付与

```
C->S: SETUP rtsp://example.com/foo/bar/baz.rm RTSP/1.0
      CSeq: 302
      Transport: RTP/AVP;unicast;client_port=4588-4589

S->C: RTSP/1.0 200 OK
      CSeq: 302
      Date: 23 Jan 1997 15:35:06 GMT
      Session: 47112344
      Transport: RTP/AVP;unicast; client_port=4588-4589;server_port=6256-6257
```


PLAY RTSP Method

- サーバーにSETUPで示した方法での再生開始を要求
- 再生場所を時間で指定可能
- 例では3箇所を連続して再生指定
- Ranege 無しでもOK
  - デフォルトは最初から最後まで

```
C->S: PLAY rtsp://audio.example.com/audio RTSP/1.0
      CSeq: 835
      Session: 12345678
      Range: npt=10-15

C->S: PLAY rtsp://audio.example.com/audio RTSP/1.0
      CSeq: 836
      Session: 12345678
      Range: npt=20-25

C->S: PLAY rtsp://audio.example.com/audio RTSP/1.0
      CSeq: 837
      Session: 12345678
      Range: npt=30-
```

TEARDOWN RTSP Method

- 再生の停止

```
C->S: TEARDOWN rtsp://example.com/fizzle/foo RTSP/1.0
      CSeq: 892
      Session: 12345678
S->C: RTSP/1.0 200 OK
      CSeq: 892
```


SDP Session Description Protocol RFC2327

```
Optional items are marked with a `*'.
Session description
        v=  (protocol version)
        o=  (owner/creator and session identifier).
        s=  (session name)
        i=* (session information)
        u=* (URI of description)
        e=* (email address)
        p=* (phone number)
        c=* (connection information –
                not required if included in all media)
        b=* (bandwidth information)
        z=* (time zone adjustments)
        k=* (encryption key)
        a=* (zero or more session attribute lines)
Time description
        t=  (time the session is active)
        r=* (zero or more repeat times)
Media description
        m=  (media name and transport address)
        i=* (media title)
        c=* (connection information - optional if included at session-level)
        b=* (bandwidth information)
        k=* (encryption key)
        a=* (zero or more media attribute lines)
```

```
a=rtpmap:<payload type> <encoding name>/<clock rate>[/<encoding parameters>]
a=cat:<category>
a=keywds:<keywords>
a=tool:<name and version of tool>
a=ptime:<packet time>
a=recvonly
a=sendrecv
a=sendonly
a=orient:<whiteboard orientation>
a=type:<conference type>
a=charset:<character set>
a=sdplang:<language tag>
a=lang:<language tag>
a=framerate:<frame rate>
a=quality:<quality>
a=fmtp:<format> <format specific parameters>
```

実際のRTSPフロー(Real)

- サンプルムービー再生時のプロトコルをダンプ・解析
- RTSPパケットの採取・解析方法

```
# tcpdump -s 1518 –w rtsp.log port rtsp
# tcpshow < rtsp.log
```

または

```
# tcpdump -s 1518 -lenx port rtsp | tcpshow –cooked
```

- RTPパケットの採取・再生ツール rtptools も

  rtpdump, rtpplay, rtpsend, rtptrans

実際のRTSPフロー(1) OPTIONS

```
IP:     RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=421 id=A4CA
DATA:   OPTIONS rtsp://real.example.com:554 RTSP/1.0.
        CSeq: 1.
        User-Agent: RealMedia RealPlayer Version 6.0.7.1503 (win32).
        ClientChallenge: a3e77a6aab4fbcccb004a5bdbb3d1a3e.
        PlayerStarttime: [27/10/2001:20:04:21 09:00].
        CompanyID: L6oRDJT2B7kChymyPJHFRA==.
        GUID: 00000000-0000-0000-0000-000000000000.
        RegionData: .
        ClientID: WinNT_5.0_6.0.9.450_play32_SF8J_ja_686.
        Pragma: initiate-session.
        .
-------------------------------------------------------------------------
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=333 id=BE05
DATA:   RTSP/1.0 200 OK.
        CSeq: 1.
        Date: Sat, 27 Oct 2001 11:05:12 GMT.
        Session: 1481386453-1.
        Server: RealServer Version 8.0.1.367 (freebsd-3.0-i386).
        Public: OPTIONS, DESCRIBE, ANNOUNCE, SETUP, GET_PARAMETER, SET_PARAMETER, TEARDOWN.
        RealChallenge1: 3e57ec4a52851a2b4f6ad885a85c9579.
        StatsMask: 3.
        .
```

実際のRTSPフロー(2) DESCRIBE

IP: RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=465 id=A4CB
DATA:
DESCRIBE rtsp://real.example.com:554/real8video.rm RTSP/1.0.
CSeq: 2.
Accept: application/sdp.
Session: 1481386453-1.
Cookie: cbid=ffjjihjidgdkjidmeoprourtfrjrktlufkfgkidldjgkelpllsmrmpqtrrlnrqcuikdghhdi.
Bandwidth: 115200.
GUID: 00000000-0000-0000-0000-000000000000.
RegionData: .
ClientID: WinNT_5.0_6.0.9.450_play32_SF8J_ja_686.
SupportsMaximumASMBandwidth: 1.
Language: ja, *.
Require: com.real.retain-entity-for-setup.
.

実際のRTSPフロー(3) DESCRIBE

IP: RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=1500 id=BE07
DATA:
RTSP/1.0 200 OK.
CSeq: 2.
Date: Sat, 27 Oct 2001 11:05:12 GMT.
vsrc: http://real.example.com:8080/viewsource/template.html?nuyhtggaysz63Evlrfnb53me1bcfngA1v1yeC3d4ngEt5o5gwuw4t6x05jbhcv66ngE8xg8f.
Last-Modified: Sat, 27 Oct 2001 10:31:13 GMT.
Content-base: rtsp://real.example.com:554/real8video.rm/.
ETag: 1481386453-1.
Session: 1481386453-1.
Content-type: application/sdp.
Content-length: 5262.
.
v=0
o=- 1004178673 1004178673 IN IP4 real.example.com
s=White Rain
i=<No author> .2000
t=0 0
a=SdpplinVersion:1610643188
a=Flags:integer;11
a=IsRealDataType:integer;1
a=StreamCount:integer;2
a=Title:buffer;"V2hpdGUgUmFpbgA="
a=Copyright:buffer;"qITwMDAA"
a=Keywords:string;""
a=ASMRuleBook:string;"#($Bandwidth < 15000),Stream0Bandwidth = 8000, Stream1Bandwidth = 4000;#($Bandwidth >= 15000) && ($Bandwidth < 20000),Stream0Bandwidth = 8000, Stream1Bandwidth = 7000;#($Bandwidth >= 20000) && ($Bandwidth < 23025),Stream0Bandwidth = 8000, Stream1Bandwidth = 12000;#($Bandwidth >= 23025) && ($Bandwidth < 33999),Stream0Bandwidth = 11025, Stream1Bandwidth = 12000;#($Bandwidth >= 33999) && ($Bandwidth < 59999),Stream0Bandwidth = 11025, Stream1Bandwidth = 22974;#($Bandwidth >= 59999) && ($Bandwidth < 79999),Stream0Bandwidth = 20672, Stream1Bandwidth = 39327;#($Bandwidth >= 79999),Stream0Bandwidth = 20672, Stream1Bandwidth = 59327;"
a=Abstract:string;""
a=range:npt=0-0
m=audio 0 RTP/AVP 101
b=AS:21
a=control:streamid=0
a=range:npt=0-52.990000
a=length:npt=52.9900

v= (protocol version)
o= (owner/creator and session identifier).
s= (session name)
i= (session information)
t= (time the session is active)
m= (media name and transport address)
b= (bandwidth information)

実際のRTSPフロー(4) DESCRIBE

IP: RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=1500 id=BE08
DATA: 00
a=rtpmap:101 x-pn-realaudio
a=mimetype:string;"audio/x-pn-realaudio"
a=MinimumSwitchOverlap:integer;200
a=StartTime:integer;0
a=AvgBitRate:integer;20672
a=EndOneRuleEndAll:integer;1
a=AvgPacketSize:integer;600
a=EndTime:integer;51092
a=SeekGreaterOnSwitch:integer;0
a=Preroll:integer;4642
a=MaxPacketSize:integer;600
a=MaxBitRate:integer;20672
a=RMFF 1.0 Flags:buffer;"AAgAAgAAAAIAAAACAAAAAgAA"
a=OpaqueData:buffer;"TUxUSQAIAAMAAwAAAAAAAQABAAIAAgAEAAAAVi5yYf0ABQAALnJhNWYFUucABQAAAEYAAAAASAAAM8AAADqYG9vYzsACAEgACAAAAAAH0AAB9AAAAAEAABZ2VucmNvb2sBBwAAAAACAEAAAEBAAAMAAAAVi5yYf0ABQAALnJhNWYFUucABQAAAEYAAQAAAWAAARMAAAFC/wAAAAAACAFgACAAAAAAKxEAACsRAAAAEAABZ2VucmNvb2sBBwAAAAACAEAAAEBAAAMAAAAXi5yYf0ABQAALnJhNWYFUucABQAAAE4AEwAAIgAAgOgAAJdnwAAAAACgJYADwAAAAAViIAAFYiAAAAEAACZ2VucmNvb2sBBwAAAAAAEAEAAAMEAAAXAAAAAAABAAMAAABWLnJh/QAFAAAucmE1ZgVS5wAFAAAARgAAAABIAAAzwAAAOpgb29jOwAIASAAIAAAAAAfQAAAH0AAAAAQAAFnZW5yY29vawEHAAAAAAAIAQAAAQEAAAw="
a=StreamName:string;"audio/x-pn-multirate-realaudio logical stream"
a=ASMRuleBook:string;"#($OldPNMPlayer),AverageBandwidth=8000,priority=5,PNMKeyframeRule=T;#($OldPNMPlayer),AverageBandwidth=0,priority=5,PNMNonKeyframeRule=T;#($Bandwidth < 11025),AverageBandwidth=8000,Priority=5;#($Bandwidth < 11025),AverageBandwidth=0,Priority=5,OnDepend=¥"2¥", OffDepend=¥"2¥";#($Bandwidth >= 11025) && ($Bandwidth < 20672),AverageBandwidth=11025,Priority=5;#($Bandwidth >= 11025) && ($Bandwidth < 20672),AverageBandwidth=0,Pri

a=rtpmap:<payload type> <encoding name>/<clock rate>[/<encoding parameters>]

実際のRTSPフロー(5) DESCRIBE

IP: RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=1500 id=BE09
DATA: ority=5,OnDepend=¥"4¥", OffDepend=¥"4¥";#($Bandwidth >= 20672),AverageBandwidth=20672,Priority=5;#($Bandwidth >= 20672),AverageBandwidth=0,Priority=5,OnDepend=¥"6¥", OffDepend=¥"6¥";"
m=video 0 RTP/AVP 101
b=AS:59
a=control:streamid=1
a=range:npt=0-50.750000
a=length:npt=50.750000
a=rtpmap:101 x-pn-realvideo
a=mimetype:string;"video/x-pn-realvideo"
a=MinimumSwitchOverlap:integer;0
a=StartTime:integer;0
a=AvgBitRate:integer;59327
a=EndOneRuleEndAll:integer;1
a=AvgPacketSize:integer;509
a=EndTime:integer;50750
a=SeekGreaterOnSwitch:integer;1
a=Preroll:integer;20353
a=MaxPacketSize:integer;607
a=MaxBitRate:integer;59327
a=RMFF 1.0 Flags:buffer;"ABAAAgAAAAAAAgACAAAAAgAAAAIAAAACAAAAAgAAAAIAAA=="
a=OpaqueData:buffer;"TUxUSQAQAAYABgAHAAgAAgACAAEAAQAAAAAAAwADAAUABQAEAAQACQAAACgAAAAoVklET1JWMzAA8AC0AAwAAAAAAA8AAAGrkDAwICACLCBQPDwtAAAAKAAAAChWSURPUlYzMADwALQADAAAAAAADwAAauQMDAgIAIsIFA8PC0AAAAoAAAAKFZJRE9SVjMwAPAAtAAMAAAAAAAPAAABq5AwMCAgAiwgUDw8LQAAACgAAAAoVklET1JWMzAA8AC0AAwAAAAAA8AAGrkDAwICACLCBQPDwtAAAAKAAAAChWSURPUlYzMADwALQADAAAAAAADwAAauQMDAgIAIsIFA8PC0AAAAoAAAAKFZJRE9SVjMwAPAAtAAMAAAAAAAPAAABq5AwMCAgAiwgUDw8LQAAACgAAAAoVklET1JWMzAA8AC0AAwAAAAAA8AAGrkDAwICACLCBQPDwtAAAAKAAAAChWSURPUlYzMADwALQADAAAAAAADwAAauQMDAgIAIsIFA8PC0AAAAoAAAAKFZJRE9SVjMwAPAAtAAMAAAAAAAPAAABq5AwMCAgAiwgUDw8LQ==="
a=StreamName:string;"video/x-pn-multirate-realvideo logical stream"
a=ASMRuleBook:string;"#(($Bandwidth >= 12000) && ($OldPNMPlayer)),AverageBandwidth=12

m= (media name and transport address)
b= (bandwidth information)
a=rtpmap:<payload type> <encoding name>/<clock rate>[/<encoding parameters>]

実際のRTSPフロー(6) DESCRIBE

```
IP:   RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=1319 id=BE0C
DATA: 000,priority=9,PNMKeyframeRule=T;#(($Bandwidth >= 12000) &&
      ($OldPNMPlayer)),AverageBandwidth=0,priority=5,PNMNonKeyframeRule=T;#(($Bandwidth < 12000) &&
      ($OldPNMPlayer)),TimestampDelivery=T,DropByN=T,priority=9,PNMThinningRule=T;#($Bandwidth <
      4000),TimestampDelivery=T,DropByN=T,priority=9;#($Bandwidth >= 4000) && ($Bandwidth <
      7000),AverageBandwidth=4000,Priority=9;#($Bandwidth >= 4000) && ($Bandwidth <
      7000),AverageBandwidth=0,Priority=5,OnDepend=¥"4¥";#($Bandwidth >= 7000) && ($Bandwidth <
      12000),AverageBandwidth=7000,Priority=9;#($Bandwidth >= 7000) && ($Bandwidth <
      12000),AverageBandwidth=0,Priority=5,OnDepend=¥"6¥";#($Bandwidth >= 12000) && ($Bandwidth <
      22974),AverageBandwidth=12000,Priority=9;#($Bandwidth >= 12000) && ($Bandwidth <
      22974),AverageBandwidth=0,Priority=5,OnDepend=¥"8¥";#($Bandwidth >= 22974) && ($Bandwidth <
      39327),AverageBandwidth=22974,Priority=9;#($Bandwidth >= 22974) && ($Bandwidth <
      39327),AverageBandwidth=0,Priority=5,OnDepend=¥"10¥";#($Bandwidth >= 39327) && ($Bandwidth <
      59327),AverageBandwidth=39327,Priority=9;#($Bandwidth >= 39327) && ($Bandwidth <
      59327),AverageBandwidth=0,Priority=5,OnDepend=¥"12¥";#($Bandwidth >=
      59327),AverageBandwidth=59327,Priority=9;#($Bandwidth >= 59327),AverageBandwidth=0,Priority=5,OnDepend=¥"14¥";"
```


実際のRTSPフロー(7) SETUP

```
IP:   RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=420 id=A4CE
DATA: SETUP rtsp://real.example.com:554/real8video.rm/streamid=0 RTSP/1.0.
      CSeq: 3.
      RealChallenge2: 5ef5555fcad4124c05ab233757b7dca801d0a8e3, sd=55c1025d.
      RDTFeatureLevel: 2.
      Transport: x-real-rdt/mcast;client_port=7070;mode=play,x-real-rdt/udp;client_port=7070;mode=play,x-pn-
      tng/udp;client_port=7070;mode=play,rtp/avp;unicast;client_port=7070-7071;mode=play.
      If-Match: 1481386453-1.
      .
------------------------------------------ - ----------------------------
IP:   RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=281 id=BE0E
DATA: RTSP/1.0 200 OK.
      CSeq: 3.
      Date: Sat, 27 Oct 2001 11:05:12 GMT.
      Session: 1481386453-1.
      RealChallenge3: f8081bb888b5bd97507bcd09dcac0d944f213d09,sdr=f18b5cd0.
      RDTFeatureLevel: 2.
      Transport: x-real-rdt/udp;client_port=7070;server_port=23116.
      .
```

実際のRTSPフロー(8) SETUP

```
IP:    RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=216 id=A4CF
DATA:  SETUP rtsp://real.example.com:554/real8video.rm/streamid=1 RTSP/1.0.
       CSeq: 4.
       RDTFeatureLevel: 2.
       Transport: x-real-rdt/udp;client_port=7070;mode=play.
       Session: 1481386453-1.
       .
-------------------------------------------------------------------------
IP:    RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=210 id=BE0F
DATA:  RTSP/1.0 200 OK.
       CSeq: 4.
       Date: Sat, 27 Oct 2001 11:05:12 GMT.
       Session: 1481386453-1.
       RDTFeatureLevel: 2.
       Transport: x-real-rdt/udp;client_port=7070;server_port=23116.
       .
```


実際のRTSPフロー(9) SET_PARAMETER

```
IP:    RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=380 id=A4D1
DATA:  SET_PARAMETER rtsp://real.example.com:554/real8video.rm RTSP/1.0.
       CSeq: 5.
       Subscribe: stream=0;rule=6,stream=0;rule=7,stream=1;rule=14,stream=1;rule=15.
       Session: 1481386453-1.
       .
       PLAY rtsp://real.example.com:554/real8video.rm RTSP/1.0.
       CSeq: 6.
       Session: 1481386453-1.
       Range: npt=0-51.092000.
       .
       SET_PARAMETER * RTSP/1.0.
       CSeq: 7.
       Ping: Pong.
       .
-------------------------------------------------------------------------
IP:    RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=128 id=BE14
DATA:  RTSP/1.0 200 OK.
       CSeq: 5.
       Date: Sat, 27 Oct 2001 11:05:12 GMT.
       Session: 1481386453-1.
       .
```

実際のRTSPフロー(10) SET_PARAMETER

```
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=348 id=BE17
DATA:   RTSP/1.0 200 OK.
        CSeq: 6.
        Date: Sat, 27 Oct 2001 11:05:12 GMT.
        RTP-Info: url=rtsp://real.example.com:554/real8video.rm/streamid=0;seq=0;rtptime=0,
        url=rtsp://real.example.com:554/real8video.rm/streamid=1;seq=0;rtptime=0.
        .
        RTSP/1.0 451 Parameter Not Understood.
        CSeq: 7.
        Date: Sat, 27 Oct 2001 11:05:12 GMT.
        .
-------------------------------------------------------------------------
IP:     RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=188 id=A4D4
DATA:   SET_PARAMETER rtsp://real.example.com:554/real8video.rm RTSP/1.0.
        CSeq: 8.
        SetDeliveryBandwidth: Bandwidth=96000;BackOff=0.
        Session: 1481386453-1.
        .
-------------------------------------------------------------------------
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=128 id=BE26
DATA:   RTSP/1.0 200 OK.
        CSeq: 8.
        Date: Sat, 27 Oct 2001 11:05:13 GMT.
        Session: 1481386453-1.
        .
```


実際のRTSPフロー(11) SET_PARAMETER

```
IP:     RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=189 id=A4D9
DATA:   SET_PARAMETER rtsp://real.example.com:554/real8video.rm RTSP/1.0.
        CSeq: 9.
        SetDeliveryBandwidth: Bandwidth=104000;BackOff=0.
        Session: 1481386453-1.
        .
-------------------------------------------------------------------------
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=128 id=BE71
DATA:   RTSP/1.0 200 OK.
        CSeq: 9.
        Date: Sat, 27 Oct 2001 11:05:16 GMT.
        Session: 1481386453-1.
        .
```

実際のRTSPフロー(12) SET_PARAMETER

```
IP:     RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=190 id=A507
DATA:   SET_PARAMETER rtsp://real.example.com:554/real8video.rm RTSP/1.0.
        CSeq: 10.
        SetDeliveryBandwidth: Bandwidth=105040;BackOff=0.
        Session: 1481386453-1.
        .
-------------------------------------------------------------------------
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=129 id=C308
DATA:   RTSP/1.0 200 OK.
        CSeq: 10.
        Date: Sat, 27 Oct 2001 11:05:49 GMT.
        Session: 1481386453-1.
        .
```


実際のRTSPフロー(13) SET_PARAMETER

```
IP:     RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=408 id=A50B
DATA:   SET_PARAMETER rtsp://real.example.com:554/real8video.rm RTSP/1.0.
        CSeq: 11.
        Session: 1481386453-1.
        PlayerStats: Stat1:     220     0       0       0       0 20_Kbps_Stereo_Music_High_-_RA8][Stat2:     20672     0
        0       0       0       0       0       0       0       0       40 20_Kbps_Stereo_Music_High_-_RA8].
        .
-------------------------------------------------------------------------
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=129 id=C436
DATA:   RTSP/1.0 200 OK.
        CSeq: 11.
        Date: Sat, 27 Oct 2001 11:06:21 GMT.
        Session: 1481386453-1.
        .
```

実際のRTSPフロー(14) TEARDOWN

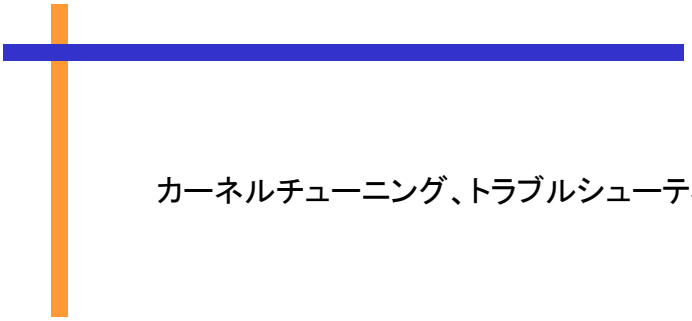
```
IP:     RealPlayer -> RealServer hlen=20 TOS=00 dgramlen=135 id=A50C
DATA:   TEARDOWN rtsp://real.example.com:554/real8video.rm RTSP/1.0.
        CSeq: 12.
        Session: 1481386453-1.
        .
-------------------------------------------------------------------
IP:     RealServer -> RealPlayer hlen=20 TOS=00 dgramlen=106 id=C437
DATA:   RTSP/1.0 200 OK.
        CSeq: 12.
        Date: Sat, 27 Oct 2001 11:06:21 GMT.
        .
```


FEC Forward Error Correction RFC2733

- 「ハードディスクRAID5のパケット版」というのが一番単純
  - 冗長なデータをあらかじめ送っておく
  - パケットが紛失してもクライアントで残りのパケットから計算で再生
  - RealでもRealServer8 から実装されている
  - その他のベンダーのサーバでも実装予定がアナウンス
- 一番簡単な例
  - パケット数個ごとにパケットの論理和を計算、冗長パケットを生成
  - 失われた時は残りのパケットの論理和を計算、結果を反転することにより再生
- 余分なパケットを常に送る
  - 帯域幅が余分に必要になるという副作用
  - 必要な冗長パケットの頻度は、パケットの損失率に依存
- マルチキャストと併用することにより大きな効果を発揮

FECに関する規格	
RFC2354	Options for Repair of Streaming Media.
RFC2733	An RTP Payload Format for Generic Forward Error Correction.

サーバ構築実践



カーネルチューニング、トラブルシューティング

FreeBSDへのインストール(1)

- FreeBSSD版「RealProducer」は無い、「RealServer」のみ
- インストールとサーバの実行にはroot権限が必要
  - インストールするディレクトリを作成
    ```
    #mkdir /usr/local/realserver
    ```
  - ダウンロードしたファイルとライセンスをコピー
    ```
    #cp rs-8-01-freebsd-3.bin /usr/local/realserver
    #cp xxxx-xx-xx-xxxxxxxxxxxxxxx.lic /usr/local/realserver
    ```
  - インストールするディレクトリに移動
    ```
    #cd /usr/local/realserver
    ```
  - ダウンロードしたファイルを実行
    ```
    #chmod +x rs-8-01-freebsd-3.bin
    #./rs-8-01-freebsd-3.bin
    ```

FreeBSDへのインストール(2)

- 管理用ポート番号はインストールする毎にランダムに変わる
- インストール終了時にそのまま起動しない事
  - デフォルトはYES
- インストーラ終了後ライセンスファイルを移動

```
#mv xxxx-xx-xx-xxxxxxxxxxxxxxx.lic License/
```

- 起動

FreeBSDインストールのノーハウ

- RealServerはネットワークI/Fが複数あるマシンでは手動設定が必要
- インストール状態では127.0.0.1のループバックアドレスで起動
- WEBブラウザーを使ってAdmin画面で設定
- または、コンフィグレーションファイルを書き直しておく
- インストールしたディレクトリにあるrmserver.cfg
- アドレスはIPアドレスでも、ドメイン表記でもOK

```
<List Name="IPBindings">
    <Var Address_1="AAA.AAA.AAA.AAA"/>
    <Var Address_1="BBB.BBB.BBB.BBB"/>
</List>
```

起動スクリプト

- OS起動時の自動起動スクリプトは付属せず
- 起動スクリプト例 **/usr/local/etc/rc.d/rmserver.sh**
- ファイルには実行権限を
  #chmod +x rmserver.sh

```
#!/bin/sh
rmdir=ここにインストールしたディレクトリを記述
rmserver=$rmdir/Bin/rmserver
rmconf=$rmdir/rmserver.cfg
rmpid=$rmdir/Logs/rmserver.pid

case "$1" in
start)
        if [ -f $rmconf -a -x $rmserver ]; then
                $rmserver $rmconf &
        fi
        ;;
stop)
        if [ -f $rmpid ]; then
                kill `cat $rmpid`
        fi
        ;;
*)
        echo "Usage: `basename $0` {start|stop}" >&2
        exit 64
        ;;
esac
exit 0
```


カーネルチューニング

- ちゃんと動かすにはカーネルのチューニングが必要
- WEBサーバなどと兼用になっている場合はカーネル資源も多く必要
- maxfiles
  - fstat , lsof などで調査
  - RealServerは起動しただけで1600程度をオープン
  - デフォルトカーネルでは最大値が2000
    /kernel: file: table is full
- mbuf
  - 現在の状況は netstat –m
  - 不足しそうな場合はカーネルの再構築
    - もしくは kern.ipc.nmbclusters で設定（最新カーネルのみ）
  - カーネルコンフィグレーションファイルにオプション追加
    options NMBCLUSTERS=8000
  - カーネル再構築の仕方は、
    http://www.jp.FreeBSD.org/www.FreeBSD.org/ja/FAQ/kernelconfig.html

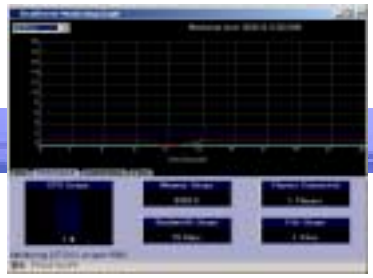
起動時に自動で設定
/etc/sysctl.conf

```
kern.maxfiles=4000
kern.maxfilesperproc=4000
```

設定コマンド

```
#sysctl -w kern.maxfiles=4000
#sysctl -w kern.maxfilesperproc=4000
```

サーバの設定



- サーバの設定はWEB画面で可
- 管理ツールは「RealAdministrator」
  http://real.example.com:NNN/admin/index.html
- 「Monitor」を選択するとサーバの状態を表示
  リアルタイムのサーバ監視Javaアプレットが起動
  「NEW window」を選択すると独立したウィンドウで監視
- ライブ中継などを行う場合はこの機能を使うと監視が容易

サーバ動作の確認

- サンプルコンテンツ
  /usr/local/realserver/Content/
- RealPlayer を起動して「場所」に
  rtsp://host.exmaple.com/real8video.rm
- 50秒ほどのWhite Rainという名前のビデオクリップ
  ビデオが60kbpsオーディオが20kbpsのステレオ
- この状況を「Real Administrator」の「Monitor」を見る
  クライアント数やバンド幅などの現在の状況グラフで表示

自宅でサーバ

- Free のサーバもいっぱい
  - Real, WMT, QuikTime, Shoutcast, Icecast ...
- バンド幅的には家庭でもStreaming可能に
  - ADSL だと上りバンド幅が数百kbps
  - FTTH だと 100Mbps?
- でも
  - IPが可変、DNSに登録できない
  - DynamicDNSでも自分のドメインが使えない
- こんなのもありますよ
  - DHIS（Dynamic Host Information System)
    http://www.portside.net/dhis/
- ペットの様子を中継？それとも子供？

トラブルシュート

ファイアーウォール・パケットロス
バッファリングと遅延時間
パケットサイズ・WEBサイト

ファイアーウォール

- イントラネットでユーザが最初に遭遇するトラブル
- ストリーミングプロトコルが通過出来ない
  - 通常HTTP・FTPのみ、PROXYサーバ経由に限定
  - RTSPやRTPを普通のPROXYサーバは扱えない
- 結果として「コンテンツが見えない」ということが起きる
- 別の手段でアクセスする必要がある

ユーザレベルの対策

- HTTPストリーミング
  - パケットをHTTPプロトコルでカプセル化
  - クライアントは一定時間ごとにコンテンツを細切れにHTTPでリクエスト
  - 擬似ストリーミングとの違いはサーバ・クライアントで連携動作する所
  - ストリーミングシステムごとに異り相互の互換性は無い
  - プロトコルは非公開
  - 自動設定ではうまく動かない場合あり
- クライアントソフトを個々に設定する必要が発生
  - 一般ユーザには敷居が高い
  - イントラネットの管理者にとっては頭の痛い問題
- この他にSOCKS
  - Apple QuickTimeはこのSOCKSを使える
  - 最新のSOCKS5にはUDPBIND

ネットワーク管理者による対策

- PROXYやSOCKSなど
  - ストリーミングプロダクトに依存しない機構
- 各ストリーミングプロダクト特有のPROXY機構
  - RealSystem Proxy 8（PROXY兼キャッシュサーバ）
- セキュリティに関するポリシーに依存
  - どちらがいいかは単純には言えない

パケットロス

- ストリーミングを受信していて
  - 絵が乱れる
  - 動きがギクシャク
  - 「ネットワークが混雑しています」
  - 再生が停止
- ご経験は？
- 何が起きているのでしょう

パケットロス – なぜ？

- いろいろな原因
- 多いのはネットワークやストリーミングサーバの負荷
  - 途中のルータやサーバの過負荷による通信バッファあふれ
  - 受信の際のパケットとりこぼし
  - WindowsCEとか、、、
- 伝送途中の回線そのものへのノイズによるビット誤り
  - 現状の日本ではノイズによる影響が問題になる環境は少ない
  - 最近のモバイル環境で電波や赤外線伝送を使った場合は問題
  - 無線LANとBluetoothとの干渉などモバイル機器同士の相互影響
- エンコーダ・サーバ間のパケットロスに注意
  - ユーザにはサーバ不調に見える

パケットロス – 影響(1)

- 結果としてクライアントに届くべきパケットが紛失
  - TCPではOSのIPスタックでエラー再送処理
    - クライアントソフトにはエラーの無いストリームが届く
    - パケットの到着遅延という形であらわれる
  - UDPパケットではOSでエラー再送が行われない
- パケットが紛失したままデコードしてしまった場合
  - 音声の場合音の途切れ
  - 画像の場合は画像の乱れ
  - 音声の途切れのほうが人間の感覚的には影響が大
- クライアントでは音声が途切れないように
  - 画像より高い優先順位
  - 再送頻度の調整

パケットロス – 影響(2)

- 高圧縮率の画像コーデックは基準フレームとの差分を使って情報圧縮
  - 一回乱れた画像は、次の基準フレームまで完全な回復は出来ない
  - あるフレームの欠損はそれ以後のフレームの欠損として後遺症を残す
  - この現象を起こさないような工夫がコーデック上の実装のノーハウ
- MPEG4では規格として後遺症を残さない工夫（エラーコンシールメント）
  - ビットエラーを想定してその影響を波及させない工夫
  - IPパケットの場合は1ビットのエラーでもIPスタックでパケット廃棄
  - エラーコンシールメントをするためにはIPスタック修正が必要
  - エラーを起こしたパケットをそのままアプリケーションプログラムに渡す
- このような工夫を施された汎用OSのIPスタックはまだない
- PDAや携帯電話などの組み込み用途の場合には検討してみる価値

パケットロス時の動作

- TCP
  - OSが再送要求を繰り返す
  - 通信そのものがそこで止まる、ストリーミングでは致命的な欠点
- UDP
  - クライアントソフトがサーバに再送要求
  - すべての処理はアプリケーションによって行われる
  - TCPとの違いは「努力してもパケットがこない時は、素直にあきらめる」
  - OS側ではエラーや紛失に対してパケットを捨てる以外は何もしない
  - パケットが届かない場合、そのままデコード処理を始ねる
- 再送を要求方法には標準がない
  - 各アプリケーションによる独自の実装
  - RTCPパケットAPPメッセージを使って実装されることが多い

パケットロス その対策は

- ユーザで対策できることはほとんどない
  - 無線アクセスの場合に電波状況のいいところに移動
  - 周囲の干渉原因やノイズ源を取り除くくらい
- もっぱらサーバもしくはネットワーク側で対策
  - パケットロスの原因がそこにあることの裏返し
- エンコーダ・オリジンサーバ間は特に重要
  - 専用のダイアルアップが安心
  - 中継中は常に監視を
  - トラフィックの中身と量に注意

バッファリングと遅延時間

- エラーパケット再送には再送要求をしたパケットの到着を待つ必要
  - 新しいデータが到着してもそれをすぐに音や画像には出来ない
  - 一定時間はデータを保存、すべてのデータがそろってから先の処理をする
  - それを実現するのがデータバッファ、その影響が再生遅延
- バッファタイムどのくらいにするかはアプリケーションの設計ポリシー
  - エラーレートが高くて遅い回線をフォローするには長くする必要
  - 経験から言えばReal Playerの遅延時間が目立って大きい
  - 多段構成の中継スプリッターを使った場合などは10分以上もざら
  - 逆にいうとRealPlayerが低速伝送にめっぽう強い事の裏返し
  - サーバ、スプリッターのデータバッファがそれぞれ分単位のバッファを持つ
  - クライアント自身とのバッファ時間の合計が遅延時間として体感
- 単純な一方向ストリームの場合は問題にならない
  - 遠隔授業や、フィードバックを要求する視聴者参加番組などでは問題
  - 運用でのがれる方法がない。チューニング手段の提供が望まれる

パケットサイズ(1)

- 混雑したネットワークでストリーミングを行う時に要注意
- UDPパケットサイズはアプリケーションごとに異なる
- 低速回線上を通過する時には大きなパケットは途中のルータで分割
  - LANなどでは問題にならない
  - WANで問題が起きる場合あり
  - 家庭でISDNなどを使っている時も同様
  - たいていのWANでは1500バイト前後
  - 最近はやりのADSLなどでも1500バイト前後
- Realの場合はこのパケットサイズは500～600バイト程度
- Windows Media Technologyでは数キロバイト
  - クライアントには二分割、三分割されてバラバラに届く

パケットサイズ(2)

- 分割だけでは何も問題ない
- 分割されたパケットはOSのIPスタックによって再結合される
- 途中の経路でパケットロスが発生した場合問題
  - パケットロスは分割されたパケットすべてに等しい確立で起きる
  - 二分割の場合には2倍、三分割の場合には3倍の確率でロスが起きる
  - 一つでもなくなってしまうとOSは全ての受信済み分割パケットを廃棄
- 最大パケットサイズはMTU呼ぶ
  - MTUを必要以上に小さくしてしまうとパフォーマンスが下がる
  - 小さい方が必ず有利ということではない
- クライアントへの伝送環境に依存
  - 「混雑してパケットロスするネットワークが悪い」と一刀両断したい
  - 現実には対策する必要がある
- MTU問題も頭の片隅に置いておくと何かの時に助けになる

WEBサーバの重要性(1)

- ライブ放送には要注意
- 視聴者数が設計値より下で飽和してしまうことが何回も
  - ライセンスや回線のバンド幅がいっぱいになるはるか手前
- あとから考えてみるとあたりまえ
  - 視聴者は「WEBサーバの上のコンテンツへのリンクをクリック」
  - これが無視できない負荷
  - ライブ放送というのは開始時間があらかじめ決まっている
  - ライブを見ようとする視聴者はその開始時間前後に集中してアクセス
- ライブ放送用のサーバがWEBサーバ兼用だと負荷が高くなる
  - ストリーミングのUDPパケットがロス
  - ストリーミングプレーヤには「接続できませんでした」というメッセージ
  - 視聴者は何度もクリック、悪循環の繰り返し
  - 一度起きるとあとは雪だるま

WEBサーバの重要性(2)

- サーバとネットワークの負荷はどんどん上がる
- ライブにはぜんぜんつながらないという悪夢が出現
- イントラの経営トップの挨拶ライブなどでもおきる
  - ライブ放送を担当者には悪夢が出現
  - 「社長の顔をつぶした」といわれるかも、
- 以下のようなことに注意
  - ストリーミングサーバとWEBサーバは兼用しない
  - できればそれぞれ別のネットワークに置く

WEBページの作り方

- ライブコンテンツへのリンクページ
  - 見栄えのする凝った物にしていませんか？
  - CGIなどにしていませんか？
- ライブリンクページは繰り返しクリックされます
  - なるべく画像を少なく
  - ページサイズも小さくする
- ページの中身にテーブルを使うことは避ける
  - MSエクスプローラではあまり関係ない
  - ネスケではテーブルが最後まで読みこまれないと表示されない
  - 途中までしか転送されない時でも視聴者にクリックできる可能性を与える
  - テーブルコンテンツを使わないことがお勧め

エンコーディングのノーハウ

一番重要なのはPCに入ってくる信号

撮影・録音のコツ

- ストリーミングを行うには映像や音を取り込む必要
- この部分にはいろいろノーハウ
- 簡単にいくと思っているとはまってしまう

ストリーミング向けカメラワーク（1）

- 普通に撮った映像だとコマ落ちしたり、画像の解像度がる場合がある
- コーデックで高い圧縮をかける仕組みに原因
  - 基本的には伝送する画像を基準フレームからの差分として扱う
  - 差分を利用するために速い動きの時には情報量が多くなる
  - スムーズさや解像度が追いつかなくなるという副作用
- ストリーミングが苦手なカメラワーク
  1. ズーム
  2. パン
  3. シーンチェンジ
- 「なんだ、カメラ動かしちゃダメって事？」⇒程度の問題
- 出来ることならエンコード後の映像をカメラマンが見ながら操作する
  - 通常ビデオ撮影時にはカメラのモニターを覗きながら操作
  - カメラワークの加減が良くわかる

ストリーミング向けカメラワーク（2）

- 本番時には撮影場所とエンコード場所が違うのはよくある
  - あらかじめカメラマンに感覚をつかんでもらう
  - 本番時と同条件でエンコード前後の絵を同時に見ながらリハーサル
- プロの手馴れたカメラマンほど自分のカメラワーク
  - この辺の加減はかえって難しいかも
- このような工夫も低ビットレートでの問題
  - 数百kbps 以上の帯域では通常のカメラワークでもOK

音量レベル

1万円ちょっとの低価格なミキサー
BEHRINGER MX802A

- オーディオ系はアナログ回路だという感覚が必要
  - 会場のマイクをパソコンに繋げて終わりというのは失敗の元
- 一番多いのは、いわゆるボリュームの問題
  - 会議では発言者によって声の大きさが何倍も違う
  - 会場には複数のマイクがあって司会者や、質問者は別のマイクを使う
  - この複数のマイクのボリュームを適切なレベルに調整する必要がある
  - PCへの過大入力はひずみ、過小入力はノイズに
- お薦めはミキサーを外付けしての手動調整
  - PA屋さんの音響マネージメントと同じ事が必要
  - 人手で音量調整をするのはある程度経験と技術が必要
  - AGCアンプとかALCアンプはマイクレベル自動調整してくれる
- 音響関係のデバイスは楽器屋にたくさんある
  - 楽器屋をウロウロのぞいて歩くのはストリーマーには必須？

ハムノイズ（1）

- 次はもっとややこしいノイズの話
  - ノイズにはそれこそ千差万別の原因と対策
  - 対策はノーハウのかたまり、でも理論なしでは泥沼
- ノイズの中で頻繁にお目（耳？）にかかるのはハムノイズ
  - ブーンという音、たいていの人は聞き覚えがある
  - 50Hzや60HzのAC100V電源の信号が何らかの経路でマイクに入る
  - ノイズが空間を飛んできてケーブルに飛び込むイメージは間違い
  - そういうモードでハムノイズが入ることは少ない
  - 大部分はアース経由でAC100V電源の一部が回り込んでくる

ハムノイズ（2）

- 中継時音響系を構成する機器は種類も数も多くなる
  - マイク、ミキサー、アンプ、録音機器、エンコードPCなど
- それぞれがAC100Vのコンセントにつながっている
  - 時にはAC200Vのアンプなども
- 日本の100Vコンセントでは普通アース端子は配線されていない
  - 各機器のグラウンドレベルは一致していない
  - テスターで機器の金属同士を計ると電圧の大きさに驚く
- プロ用の音響機材はこのアース電位の影響を受けない
  - アースと信号線を分離したバランス型で信号をやり取り
  - PCのマイク端子は分離されていないミニジャックがほとんど
  - PCとミキサーやマイクアンプ間にアース電位差があるとノイズの原因に

ハムノイズ ー 対策

- 一番簡単には
  - 簡単にはコンセントの指す方向を逆にする
  - 日本の100Vコンセントのはどちらか一方がアース
  - プラグを指す向きを変えるとアース電位差が少なくる場合がある
- 機材のケースにあるアース端子を直接太い線で結ぶ
  - 冷蔵庫や洗濯機を買うとついてくる緑色の太い銅線
  - モデムやTAなどにも付属している物が具合よく使える
- 自分でバランス変換器を作る方法
  - トランスを使ってバランス型とアンバランス型の変換が可能。
  - 楽器屋さんに行くとこの辺の変換箱とかも売っている

Streamingをささえる世界

ペタペタでジャブジャブ

WDMと海底ケーブル

- 多くの人がストリーミングするようになったら何が起きるか
  - ラスト1マイルがFTTH、ADSLやCATVなどで高速化
  - バックボーンネットワークの太さが問題に
- 光ファイバーの中にたくさんのビームを通せるWDM技術で解決
  - 東京・大阪間のファイバー網は現在では余っている
- 海底ケーブルにはWDM技術が簡単には使えない
  - 海底ケーブルには一定間隔で中継器が必要
  - 中継器がWDMに対応しきれない
    - 今のところ40km程度の間隔で入れる必要
    - 小型化も難しく、現状では海底に設置するのが難しい
- 陸上では何十倍何百倍もの高速化が可能
- 日本以外との通信がまたネックになる可能性が高い

沿岸海底ファイバー

- 半分冗談、半分真剣
- 日米間海底ケーブルはグアムやハワイなどの島を経由
  - 海底ケーブルは中継器を入れる都合上ファイバーの数が限られる
- 沿岸の40kmおきに陸上中継局を設け、ケーブル自体は海底に敷設
  - 地上にで問題になる土地問題と、WDMの中継問題を同時解決
  - 陸上と同じように数百芯のケーブルが使える。
  - 海底中継器が無いのでケーブルのコストがぐっと安く
- 陸地間の距離が40km以上の海峡があると成り立たない
  - ルートは北海道⇔ロシア⇔アラスカ⇔カナダ⇔アメリカ
  - 意外と海峡が少ない
    - 北海道と樺太間の宗谷海峡 43km
    - 樺太・ロシア間の間宮海峡 7km
    - ロシア・アラスカ間のベーリング海峡 100km
- ベーリング海峡がちょっと長い、、、でも

エンジェルリング 環北極圏ファイバー網

- ベーリング海峡のちょうど真中に島
  - ロシア領大ダイオミード島
  - アメリカ領小ダイオミード島
- まるでWDMファイバーを敷設するために作ったような島
  - 島の間の距離は3km
  - 島の大きさもあるので、結局40km+αの距離が最長
- さらにファイバーをシベリア経由でヨーロッパまで引く
  - 北極の周りを一周するファイバーネット⇒エンジェルリング
  - オーストラリアと南極大陸以外はすべてつながる
- 今世紀のキーワードは、「ペタペタでジャブジャブ」

Appendix

参考資料

国内のストリーミング関連ML

- **Streams-JP**
  ストリーミングが趣味・仕事な人の集まる場所
  各ベンダーの人や有名どころの人などが多数いる国内の総本山
  国内での大きなストリーミングイベントはだいたいこのML参加者が運用
  http://www.iij-mc.co.jp/bunji/Streams-JP/
- **SMIL-ML**
  SMILを始めとするコンテンツに関するML.
  http://www.takesato.com/smil/
- IPmulticast
  マルチキャストに関するML。技術的な話題が中心。
  http://www.iijnet.or.jp/IPmulticast/mailinglist.html

参考ホームページ

- Real Networks（日本）
  http://www.jp.realnetworks.com/
- Microsoft Media（日本）
  http://www.Microsoft.com/japan/windows/windowsmedia/
- Apple QuickTime（日本）
  http://www.apple.co.jp/quicktime/
- JPNIC RFC-JP プロジェクト
  http://rfc-jp.nic.ad.jp/
- IP Multicast Initiative(日本)
  http://www.iijnet.or.jp/IPmulticast/
- IETF(Internet Engineering Task Force)
  http://www.ietf.org/

参考文献

- わかる！ストリーミング技術（仮称）
  神田泰典・森出茂樹　共著
  ISBN4-274-07937
  オーム社 2001年12月発売予定
- インターネットストリーミング
  大澤 光 編著
  共立出版（2000）
- インターネット ストリーミング ブック
  Morley Robertson, 林 岳里, 原水真一, 姉歯康, 猪蔵, 佐藤めぐみ
  翔泳社
- はじめてのストリーミングWebで動画を見せよう！
  エーアイ出版

ご清聴ありがとうございました

ご質問をどうぞ
Any question?