       The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types

Abstract

   Several MIME type/subtype combinations exist that can contain
   different media formats. A receiving agent thus needs to examine the
   details of such media content to determine if the specific elements
   can be rendered given an available set of codecs. Especially when
   the end system has limited resources, or the connection to the end
   system has limited bandwidth, it is helpful to know from the Content-
   Type alone if the content can be rendered.

   This document specifies two parameters, 'codecs' and 'profiles', that
   are used with various MIME types or type/subtype combinations to
   allow for unambiguous specification of the codecs employed by the
   media formats contained within, or the profile(s) of the overall
   container format. This document obsoletes RFC 4281; RFC 4281 defines
   the 'codecs' parameter, which this document retains in a backwards
   compatible manner with minor clarifications; the 'profiles' parameter
   is added by this document.

   By labeling content with the specific codecs indicated to render the
   contained media, receiving systems can determine if the codecs are
   supported by the end system, and if not, can take appropriate action
   (such as rejecting the content, sending notification of the
   situation, transcoding the content to a supported type, fetching and
   installing the required codecs, further inspection to determine if it
   will be sufficient to support a subset of the indicated codecs,
   etc.).

   Similarly, the profiles can provide an overall indication, to the
   receiver, of the specifications with which the content complies.
   This is an indication of the compatibility of the container format
   and its contents to some specification. The receiver may be able to
   work out the extent to which it can handle and render the content by
   examining to see which of the declared profiles it supports, and what
   they mean.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6381.

Copyright Notice

Table of Contents

1. Introduction

   One of the original motivations for MIME is the ability to identify
   the specific media type of a message part. However, due to various
   factors, it is not always possible from looking at the MIME type and
   subtype to know which specific media formats are contained in the
   body part or which codecs are indicated in order to render the
   content.

   There are several media type/subtypes (either currently registered or
   deployed with registration pending) that contain codecs chosen from a
   set. In the absence of the parameters described here, it is
   necessary to examine each media element in order to determine the
   codecs or other features required to render the content. For
   example, video/3gpp may contain any of the video formats H.263
   Profile 0, H.263 Profile 3, H.264, MPEG-4 Simple Profile, and/or any
   of the audio formats Adaptive Multi Rate (AMR), Adaptive Multi Rate -
   WideBand (AMR-WB), Extended AMR-WB, Advanced Audio Coding (AAC), or
   Enhanced aacPlus, as specified in [3GPP-Formats].

In some cases, the specific codecs can be determined by examining the
header information of the media content. While this isn't as bad as
examining the entire content, it still requires specialized knowledge
of each format and is resource consumptive.

This ambiguity can be a problem for various clients and servers. For
example, it presents a significant burden to Multimedia Messaging
(MMS) servers, which must examine the media sent in each message in
order to determine which codecs are required to render the content.
Only then can such a server determine if the content requires
transcoding or specialized handling prior to being transmitted to the
handset.

Additionally, it presents a challenge to smart clients on devices
with constrained memory, processing power, or transmission bandwidth
(such as cellular telephones and PDAs). Such clients often need to
determine in advance if they are currently capable of rendering the
content contained in an MMS or email message.

Ambiguity:

o audio/3gpp can contain AMR, AAC, AMR-WB, Extended AMR-WB, or
   Enhanced aacPlus contents as specified in [3GPP-Formats].

o audio/3gpp2 can contain AMR, AAC, 13K (as per [RFC3625]), Enhanced
   Variable Rate Codec (EVRC), Selectable Mode Vocoder (SMV), or
   Variable Multi Rate WideBand (VMR-WB), as specified in
   [3GPP2-Formats].

o video/3gpp can contain H.263 Profile 0, H.263 Profile 3, H.264,
   MPEG-4 Simple Profile, and/or AMR, AMR-WB, Extended AMR-WB, AAC,
   or Enhanced aacPlus, as specified in [3GPP-Formats].

o video/3gpp2 can contain H.263 Profile 0, H.263 Profile 3, H.264,
   MPEG-4 Simple Profile, and/or AMR, AAC, 13K (as per [RFC3625]),
   EVRC, SMV, or VMR-WB, as specified in [3GPP2-Formats].

o audio/mp4 can include any codec defined in MPEG-1, MPEG-2, MPEG-4
   or registered at the MP4 registration authority [MP4RA].

o video/mp4 has the same issues as audio/mp4, and in addition many
   video codecs, and especially the MPEG codecs, have a variety of
   profiles and levels, not all of which are supported by every
   implementation.

Note that there are additional media types that are ambiguous, but
are outside the scope of this document, including:

o video/mpeg4-generic, which can contain anything allowed by the
   MPEG-4 specification, or any codec registered with the MP4
   registration authority [MP4RA];

With each "bucket" type, a receiving agent only knows that it has a
container format. It doesn't even know whether content that is
labeled video/3gpp or video/3gpp2 contains video; it might be audio
only, audio and video, or video only.

A solution that permits a receiving agent to determine the specific
codecs or profiles required to render media content would help
provide efficient and scalable servers, especially for Multimedia
Messaging (MMS), and aid the growth of multimedia services in
wireless networks.

2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in "Key words for use in
RFCs to Indicate Requirement Levels" [RFC2119] .

The syntax in this document uses the BNF rules specified in [RFC2045]
and [RFC2231].

3. The 'Codecs' Parameter

3.1. Introduction

This section adds a parameter to allow unambiguous specification of
all codecs indicated to render the content in the MIME part. This
parameter is optional in all current types to which it is added.
Future types that contain ambiguity are strongly encouraged to
include this parameter.

This parameter applies to:

1. Files in the family based on the ISO Base Media File Format
    [ISO14496-12] called "ISO family files" in this specification.

2. The QuickTime file format, owned by Apple, Inc.

This includes the media types:

1. audio/3gpp, video/3gpp [RFC3839]

2.   audio/3gpp2, video/3gpp2 [RFC4393]

3.   audio/mp4, video/mp4, application/mp4 [RFC4337]

4.   video/quicktime

5.   application/mp21 (see note below)

Note that MPEG-21 files under the type application/mp21 may, but are
not required to, contain a top-level 'moov' atom providing a timed,
coded, resource. The 'codecs' parameter SHOULD only be used for
MPEG-21 files when this timed material is also present in the file.

Parameter name: codecs

Parameter value: A single value, or a comma-separated list of values
identifying the codec(s) indicated to render the content in the body
part.

Each value consists of one or more dot-separated elements. The name
space for the first element is determined by the MIME type. The name
space for each subsequent element is determined by the preceding
element. The precise syntax is given below in the Generic Syntax
(Section 3.2).

Note that, per [RFC2045], some characters (including the comma used
to separate multiple values) require that the entire parameter value
be enclosed in quotes.

An element MAY include an octet that [RFC2045] requires encoding. In
this case, [RFC2231] is used: an asterisk ("*") is placed at the end
of the parameter name (becoming 'codecs*' instead of 'codecs'), the
parameter value usually starts with two single quote ("'") characters
(indicating that neither character set nor language is specified),
and each octet that requires encoding is represented as a percent
sign ("%") followed by two hexadecimal digits. Note that, when the
[RFC2231] form is used, the percent sign, asterisk, and single quote
characters have special meaning and so MUST themselves be percent
encoded.

        Examples of Generic Syntax:
            codecs=a.bb.ccc.d
            codecs="a.bb.ccc.d, e.fff"
            codecs*=''fo%2e
            codecs*="''%25%20xz, gork"

When the 'codecs' parameter is used, it MUST contain all codecs indicated by the content present in the body part. The 'codecs' parameter MUST NOT include any codecs that are not indicated by any media elements in the body part.

In some cases, not all indicated codecs are absolutely required in order to render the content. Therefore, when a receiver does not support all listed codecs, special handling might be required. For example, the media element(s) could be examined in order to determine if an unsupported codec is actually required (e.g., there may be alternative tracks (such as English and Spanish audio), there may be timed text that can be dropped, etc.).

Although the encoder MUST create parameter values that are complete and accurate in 'breadth' (that is, the encoder MUST report all four-character codes used in all tracks for ISO family files, for example) receivers MUST NOT rely on the parameter values being complete in 'depth'. (If the hierarchical rules for a given code (e.g., 'qvxy') were written after a server was implemented, for example, that server would not know what elements to place after 'qvxy').

Although a mismatch is not permitted by this specification, the body part is definitive of the actual codecs needed; the parameter supplied here is informative. If a receiver encounters a body part whose 'codecs' parameter contains codecs that are not indicated by any media elements, then the receiver SHOULD process the body part by discarding the information in the 'codecs' parameter.

If a receiver encounters a body part whose 'codecs' parameter does not contain all codecs indicated by the media elements, then the receiver MAY process the body part by discarding the information in the 'codecs' parameter.

3.2. Generic Syntax

The 'codecs' parameter takes either of two forms. The first form is used when the value does not contain any octets that require encoding. The second form uses [RFC2231] to allow arbitrary octets to be encoded. With either form, quotes allow for commas and other characters in <tspecials> (quotes MAY be used even when not required).

This BNF uses the rules specified in [RFC2045] and [RFC2231].

While [RFC2231] allows specification of character set and language, this parameter does not contain items intended for human consumption, and hence makes no use of language. The language element SHOULD be

      omitted; the character set SHOULD also be omitted. A receiver MAY
      ignore language and MAY choose to support only US-ASCII [RFC1345] and
      UTF-8 [RFC3629].

      Implementations MUST NOT add comments and/or folding white space
      (CFWS) between the tokens except after ",". TOKEN is defined in
      [RFC2045], and <ext-octet> and <attribute-char> are defined in
      [RFC2231].

      The BNF syntax is as follows:

```
      codecs      := cod-simple / cod-fancy
      cod-simple  := "codecs" "=" unencodedv
      unencodedv  := id-simple / simp-list
      simp-list   := DQUOTE id-simple *( "," id-simple ) DQUOTE
      id-simple   := element
                  ; "." reserved as hierarchy delimiter
      element     := 1*octet-sim
      octet-sim   := <any TOKEN character>

                  ; Within a 'codecs' parameter value, "." is reserved
                  ; as a hierarchy delimiter
      cod-fancy   := "codecs*" "=" encodedv
      encodedv    := fancy-sing / fancy-list
      fancy-sing  := [charset] "'" [language] "'" id-encoded
                  ; Parsers MAY ignore <language>
                  ; Parsers MAY support only US-ASCII and UTF-8
      fancy-list  := DQUOTE [charset] "'" [language] "'" id-list DQUOTE
                  ; Parsers MAY ignore <language>
                  ; Parsers MAY support only US-ASCII and UTF-8
      id-list     := id-encoded *( "," id-encoded )
      id-encoded  := encoded-elm *( "." encoded-elm )
                  ; "." reserved as hierarchy delimiter
      encoded-elm := 1*octet-fancy
      octet-fancy := ext-octet / attribute-char

      DQUOTE      := %x22 ; " (double quote)
```

   Initial name space: This document only defines values for files in
   the ISO Base Media File Format and QuickTime families. Other file
   formats may also define codec naming.

3.3. ISO Base Media File Format Name Space

   For the ISO Base Media File Format, and the QuickTime movie file
   format, the first element of a 'codecs' parameter value is a sample
   description entry four-character code as registered by the MP4
   Registration Authority [MP4RA]. Values are case sensitive.

Note that there are potentially multiple tracks in a file, each
potentially carrying multiple sample entries (some but not all uses
of the ISO Base Media File Format restrict the number of sample
entries in a track to one).

When the first element of a value is 'mp4a' (indicating some kind of
MPEG-4 audio), or 'mp4v' (indicating some kind of MPEG-4 part-2
video), or 'mp4s' (indicating some kind of MPEG-4 Systems streams
such as MPEG-4 BInary Format for Scenes (BIFS)), the second element
is the hexadecimal representation of the MP4 Registration Authority
ObjectTypeIndication (OTI), as specified in [MP4RA] and [MP41]
(including amendments). Note that [MP4RA] uses a leading "0x" with
these values, which is omitted here and hence implied.

One of the OTI values for 'mp4a' is 40 (identifying MPEG-4 audio).
For this value, the third element identifies the audio
ObjectTypeIndication (OTI) as defined in [MP4A] (including
amendments), expressed as a decimal number.

For example, AAC low complexity (AAC-LC) has the value 2, so a
complete string for AAC-LC would be "mp4a.40.2".

One of the OTI values for 'mp4v' is 20 (identifying MPEG-4 part-2
video). For this value, the third element identifies the video
ProfileLevelIndication as defined in [MP4V] (including amendments),
expressed as a decimal number.

For example, MPEG-4 Visual Simple Profile Level 0 has the value 9, so
a complete string for MPEG-4 Visual Simple Profile Level 0 would be
"mp4v.20.9".

When the first element of a value is a code indicating a codec from
the Advanced Video Coding specification [AVC], specifically one of
the sample entries defined in [AVC-Formats] (such as 'avc1', 'avc2',
'svc1', 'mvc1', and 'mvc2') -- indicating AVC (H.264), Scalable Video
Coding (SVC), or Multiview Video Coding (MVC), the second element
(referred to as 'avcoti' in the formal syntax) is the hexadecimal
representation of the following three bytes in the (subset) sequence
parameter set Network Abstraction Layer (NAL) unit specified in
[AVC]:

(1) profile_idc,

(2) the byte containing the constraint_set flags (currently
     constraint_set0_flag through constraint_set5_flag, and the
     reserved_zero_2bits), and

(3) level_idc.

Note that the sample entries 'avc1' and 'avc2' do not necessarily indicate that the media only contains AVC NAL units. In fact, the media may be encoded as an SVC or MVC profile and thus contain SVC or MVC NAL units. In order to be able to determine which codec is used, further information is necessary (profile_idc). Note also that reserved_zero_2bits is required to be equal to 0 in [AVC], but other values for it may be specified in the future by ITU-T or ISO/IEC.

This is as previously defined in the 3GPP File Format specification 3GPP TS 26.244 [3GPP-Formats], Section A.2.2.

When SVC or MVC content is coded in an AVC-compatible fashion, the sample description may include both an AVC configuration record and an SVC or MVC configuration record. Under those circumstances, it is recommended that the two configuration records both be reported as they may contain different AVC profile, level, and compatibility indicator values. Thus, the codecs reported would include the sample description code (e.g., 'avc1') twice, with the values from one of the configuration records forming the 'avcoti' information in each.

3.4. ISO-Family Syntax

```
   id-simple   :=/ id-iso
   id-encoded  :=/ id-iso
   id-iso      := iso-gen / iso-mpega / iso-mpegv / iso-avc
   iso-gen     := cpid *( element / encoded-elm )
               ; <element> used with <codecs-simple>
               ; <encoded-elm> used with <codecs-fancy>
               ;
               ; Note that the BNF permits "." within <element>
               ; and <encoded-elm> but "." is reserved as the
               ; hierarchy delimiter
   iso-mpega   := mp4a "." oti [ "." aud-oti ]
   iso-mpegv   := mp4v "." oti [ "." vid-pli ]
   iso-avc     := avc1  / avc2 / svc1 / mvc1 / mvc2 [ "." avcoti  ]
   cpid        := 4(octet-simple / octet-fancy)
               ; <octet-simple> used with <codecs-simple>
               ; <octet-fancy> used with <codecs-fancy>
   mp4a        := %x6d.70.34.61 ; 'mp4a'
   oti         := 2(DIGIT / "A" / "B" / "C" / "D" / "E" / "F")
               ; leading "0x" omitted
   avc1        := %x61.76.63.31 ; 'avc1'
   avc2        := %x61.76.63.32 ; 'avc2'
   svc1        := %x73.76.63.31 ; 'svc1'
   mvc1        := %x6d.76.63.31 ; 'mvc1'
   mvc2        := %x6d.76.63.32 ; 'mvc2'
   avcoti      := 6(DIGIT / "A" / "B" / "C" / "D" / "E" / "F")
               ; leading "0x" omitted
   aud-oti     := 1*DIGIT
   mp4v        := %x6d.70.34.76 ; 'mp4v'
   vid-pli     := 1*DIGIT
```

3.5. Use in Additional Media Types

   This parameter MAY be specified for use with additional MIME media
   types.

   For ISO family file formats where the name space as defined here is
   sufficient, all that needs to be done is to update the media type
   registration to specify the 'codecs' parameter with a reference to
   this document. For existing media types, it is generally advisable
   for the parameter to be optional; for new media types, the parameter
   MAY be optional or required, as appropriate.

   For ISO family file formats where the name space as defined here
   needs to be expanded, a new document MAY update this one by
   specifying the additional detail.

   For non-ISO family file formats, a new document MAY update this one
   by specifying the name space for the media type(s).

3.6. Examples

   Content-Type: video/3gpp2; codecs="sevc, s263"
       (EVRC audio plus H.263 video)
   Content-Type: audio/3gpp; codecs=samr
       (AMR audio)
   Content-Type: video/3gpp; codecs="s263, samr"
       (H.263 video plus AMR audio)
   Content-Type: audio/3gpp2; codecs=mp4a.E1
       (13K audio)
   Content-Type: video/3gpp2; codecs="mp4v.20.9, mp4a.E1"
       (MPEG-4 Visual Simple Profile Level 0 plus 13K voice)
   Content-Type: video/mp4; codecs="avc1.640028"
        (H.264/AVC video, High Profile, Level 40,
         e.g., DVB 720p 50Hz HDTV)
   Content-Type: video/mp4; codecs="svc1.56401E, avc1.4D401E"
        (SVC video, Scalable High Profile, Level 30,
         with a Main Profile AVC base layer, e.g., DVB 25 Hz SDTV)
    Content-Type: video/mp4; codecs="mvc1.800030, avc1.640030"
        (MVC video, Stereo High Profile, Level 42,
         with a High Profile base layer, e.g., as adopted in Blu-ray)

   Note: OTI value 20 ("0x20" in [MP4RA]) says "Includes associated
   Amendment(s) and Corrigendum(a). The actual object types are defined
   in [MP4V] and are conveyed in the DecoderSpecificInfo as specified in
   [MP4V], Annex K." (references adjusted).

3.7. Additional Media Feature Details

   It is sometimes helpful to provide additional details for a media
   element (e.g., the number of X and Y pixels, the color depth, etc.).
   These details are sometimes called "media features" or "media
   characteristics".

   When such additional features are included, the content-features
   [RFC2912] header provides a handy way to do so.

4. The 'Profiles' Parameter

4.1. Introduction

   Just as some codecs have a variety of profiles (subsets of their
   functionality within which a bitstream can be coded), some media
   files can also be profiled and be associated with one or more profile
   identifiers of the profiles to which they conform. These profiles

can indicate features of the file format itself, which codecs may be
present, the profiles of those codecs, and so on. It can be
advantageous to a receiving system to know the overall file
profile(s) of a file; indeed, under these circumstances it may not be
necessary to know the codecs themselves if they are implied by the
profile.

The 'profiles' parameter reports on the profile(s) of the overall
container format. A profile of the container format may have
restrictions on not only the features of the container format itself
but also on what codecs may be included, and it may indeed have
restrictions on the profiles of those codecs. The 'profiles'
parameter does not, however, report directly any profiles of the
contained media: when such codec-specific profiles are reported, this
report is part of the 'codecs' parameter. The 'profiles' parameter
reports only the profile(s) applying to the complete container.

When the use of the 'profiles' parameter is defined for a given
format, that definition SHOULD indicate that it directly reflects
information in the body part, i.e., that it does not convey
information beyond, or different from, what can be learnt by
inspecting the body part. Although a mismatch is not permitted by
this specification, the body part is definitive of the actual
profiles; the parameter supplied here is informative.

4.2. Formal Declaration

This section adds a parameter to allow unambiguous specification of
the profiles to which a file claims conformance. This parameter is
OPTIONAL in all current types to which it is added.

This parameter applies to Box-structured (also known as atom-
structured) files that have an initial box containing compatibility
brands, as registered at the MP4 Registration Authority [MP4RA], such
as a filetype or segment-type box. Principally, this includes files
in the family based on the ISO Base Media File Format [ISO14496-12]
and the QuickTime file format, owned by Apple, Inc. (A brand can
indicate conformance with restrictions regarding which codecs and
file format features are used, adherence to quantitative limits such
as the length/size of the file, and so on.)

This includes the media types:

1. audio/3gpp, video/3gpp [RFC3839]

2. audio/3gpp2, video/3gpp2 [RFC4393]

3. audio/mp4, video/mp4 [RFC4337]

   4. video/quicktime

   5. application/mp21

   Parameter name: profiles

   Parameter value: A single value, or a comma-separated list of values
   identifying the profiles(s) to which the file claims conformance.

   The name space is determined by the MIME type.

   Note that, per [RFC2045], some characters (including the comma used
   to separate multiple values) require that the entire parameter value
   be enclosed in quotes.

   An element MAY include an octet that [RFC2045] requires encoding. In
   this case, [RFC2231] is used: an asterisk ("*") is placed at the end
   of the parameter name (becoming 'profiles*' instead of 'profiles'),
   the parameter value usually starts with two single quote ("'")
   characters(indicating that neither character set nor language is
   specified), and each octet that requires encoding is represented as a
   percent sign ("%") followed by two hexadecimal digits. Note that,
   when the [RFC2231] form is used, the percent sign, asterisk, and
   single quote characters have special meaning and so MUST themselves
   be percent encoded.

           Examples of Generic Syntax:
               profiles="isom,mp41,qvXt"
               profiles*="''%25%20xz, gork"

4.3. 'Profiles' Parameter Definition

   The 'profiles' parameter is an OPTIONAL parameter that indicates one
   or more profiles to which the file claims conformance. Like the
   'codecs' parameter described above, it may occur as either 'profiles'
   or 'profiles*', with the same encoding rules. The value is, as for
   the 'codecs' parameter, a comma-separated list of profile
   identifiers.

4.4. Profiles for Files Carrying MP4RA-Registered Brands

   For any file format carrying a brand registered at the MP4
   Registration Authority [MP4RA], notably files based on the ISO Base
   Media File Format ISO/IEC 14496-12 [ISO14496-12] and QuickTime movie
   files, the 'profiles' parameter MUST list exactly the major-brand,
   followed by the compatible-brands, as listed in the filetype box
   ('ftyp') or segment-type box ('styp'). The major-brand MUST be
   first, and MAY be removed from the compatible-brands list. (The file

   format requires that it be repeated in the compatible-brands, but
   this requirement is relaxed here for compactness.)

   An example might be profiles="mp41,isom,qvXt", indicating that MPEG-4
   version 1 is the major-brand and preferred use, that the file is
   compatible with the version of the base file format identified by
   'isom', and that it is also compatible with the specification/profile
   'qvXt' (whatever that may be).

4.5. 'Profiles' Parameter BNF Definition

   profiles   := pro-simple / pro-fancy
   pro-simple := "profiles" "=" unencodedv
   pro-fancy   := "profiles*" "=" encodedv

5. IANA Considerations

   IANA has replaced references to [RFC4281] with references to this
   document in the "MIME Media Types" registry, thereby indicating that
   the 'codecs' and/or 'profiles' parameters are optional for the
   following media types (as listed in Sections 3 and 4):

   1. audio/3gpp, video/3gpp [RFC3839]

   2. audio/3gpp2, video/3gpp2 [RFC4393]

   3. audio/mp4, video/mp4, application/mp4 [RFC4337]

   4. video/quicktime

   5. application/mp21

6. Registration

   The MPEG4 Registration Authority can be consulted for the most up-to-
   date registration of sub-parameters for the codecs type, for specific
   codecs.

7. Security Considerations

    The 'codecs' parameter itself does not alter the security
    considerations of any of the media types with which it is used. Each
    audio and video media type has its own set of security considerations
    that continue to apply, regardless of the use of the 'codecs'
    parameter.

    An incorrect 'codecs' parameter might cause media content to be
    received by a device that is not capable of rendering it or might
    cause media content not to be sent to a device that is capable of
    receiving it. An incorrect 'codecs' parameter is therefore capable
    of some types of denial-of-service attacks. However, this is most
    likely to arise by accident, as an attacker capable of altering media
    data in transit could cause more harm by altering the media format
    itself, or even the content type header, rather than just the
    'codecs' parameter of the content type header.

    To the extent that a receiver reacts to a 'codecs' parameter that
    indicates an unsupported codec, by fetching and installing the
    required codecs, such reaction needs to be performed carefully and in
    accord with the system's normal validity and security checks and
    procedures.

8. Differences from RFC 4281

    1. Improved the introduction and other supporting and explanatory
        text;

    2. improved the references;

    3. clarified the MIME types to which the parameters apply, and
        clarified the consequent IANA actions;

    4. added the 'profiles' parameter;

    5. fixed an error in the BNF, where it did not correspond to either
        the examples or common usage;

    6. added the definition of the sub-parameters for the AVC family of
        codecs;

    7. added a security consideration for possible triggering of
        downloads;

    8. updated acknowledgments.

9. Acknowledgements

   Harinath Garudadri provided a great deal of help, which is very much
   appreciated. Mary Barnes and Bruce Lilly provided detailed and
   helpful comments. Reviews and comments by Sam Hartman, Russ Housley,
   and Bert Wijnen were much appreciated. Chris Newman carefully
   reviewed and improved the BNF.

   Christian Timmerer helped with the MPEG-21 material, and Thomas
   Schierl and Yago Sanchez helped with SVC and MVC.

10. References

10.1. Normative References

   [3GPP-Formats]   3rd Generation Partnership Project, "Technical
                     Specification Group Services and System Aspects;
                     Transparent end-to-end packet switched streaming
                     service (PSS); 3GPP file format (3GP)", 3GPP
                     TS 26.244.

   [AVC]   "Advanced video coding for generic audiovisual
                     services", ITU-T Recommendation H.264, ISO/
                     IEC 14496-10:2009.

   [AVC-Formats]   "Information technology -- Coding of audio-visual
                     objects -- Part 15: Advanced Video Coding (AVC) file
                     format", ISO/IEC 14496-15:2010.

   [ISO14496-12]   "Information technology -- Coding of audio-visual
                     objects -- Part 12: ISO base media file format",
                     ISO/IEC 14496-12:2008.

   [MP4RA]   "MP4REG, The MPEG-4 Registration Authority",
                     <http://www.mp4ra.org>.

   [RFC2045]   Freed, N. and N. Borenstein, "Multipurpose Internet
                     Mail Extensions (MIME) Part One: Format of Internet
                     Message Bodies", RFC 2045, November 1996.

   [RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
                     Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC2231]   Freed, N. and K. Moore, "MIME Parameter Value and
                     Encoded Word Extensions:
                     Character Sets, Languages, and Continuations",
                     RFC 2231, November 1997.

   [RFC2912]   Klyne, G., "Indicating Media Features for MIME
                      Content", RFC 2912, September 2000.

   [RFC3629]   Yergeau, F., "UTF-8, a transformation format of ISO
                      10646", STD 63, RFC 3629, November 2003.

   [RFC3839]   Castagno, R. and D. Singer, "MIME Type Registrations
                      for 3rd Generation Partnership Project (3GPP)
                      Multimedia files", RFC 3839, July 2004.

   [RFC4281]   Gellens, R., Singer, D., and P. Frojdh, "The Codecs
                      Parameter for "Bucket" Media Types", RFC 4281,
                      November 2005.

   [RFC4337]   Y Lim and D. Singer, "MIME Type Registration for
                      MPEG-4", RFC 4337, March 2006.

   [RFC4393]   Garudadri, H., "MIME Type Registrations for 3GPP2
                      Multimedia Files", RFC 4393, March 2006.

10.2. Informative References

   [3GPP2-Formats] Third Generation Partnership Project 2, "3GPP2 File
                      Formats for Multimedia Service", <http://
                      www.3gpp2.org/Public_html/specs/
                      C.S0050-0_v1.0_121503.pdf>.

   [MP41]   "Information technology--Coding of audio-visual
                      objects -- Part 1: Systems", ISO/IEC 14496-1:2010.

   [MP4A]   "Information technology--Coding of audio-visual
                      objects -- 3: Audio", ISO/IEC 14496-3:2009.

   [MP4V]   "Information technology--Coding of audio-visual
                      objects -- Part 2: Visual", ISO/IEC 14496-2:2004.

   [RFC1345]   Simonsen, K., "Character Mnemonics and Character
                      Sets", RFC 1345, June 1992.

   [RFC3625]   Gellens, R. and H. Garudadri, "The QCP File Format
                      and Media Types for Speech Data", RFC 3625,
                      September 2003.

Authors' Addresses

   Randall Gellens
   QUALCOMM Incorporated
   5775 Morehouse Drive
   San Diego, CA 92121
   US

   EMail: rg+ietf@qualcomm.com

   David Singer
   Apple, Inc.
   1 Infinite Loop
   Cupertino, CA 95014
   US

   Phone: +1 408 996 1010
   EMail: singer@apple.com

   Per Frojdh
   Ericsson AB
   Ericsson Research
   Stockholm SE-164 80
   Sweden

   Phone: +46 10 7190000
   EMail: Per.Frojdh@ericsson.com