                            Ogg Media Types

Status of This Memo

Abstract

   This document describes the registration of media types for the Ogg
   container format and conformance requirements for implementations of
   these types. This document obsoletes RFC 3534.

Table of Contents

1. Introduction

   This document describes media types for Ogg, a data encapsulation
   format defined by the Xiph.Org Foundation for public use. Refer to
   "Introduction" in [RFC3533] and "Overview" in [Ogg] for background
   information on this container format.

   Binary data contained in Ogg, such as Vorbis and Theora, has
   historically been interchanged using the application/ogg media type
   as defined by [RFC3534]. This document obsoletes [RFC3534] and
   defines three media types for different types of content in Ogg to
   reflect this usage in the IANA media type registry, to foster
   interoperability by defining underspecified aspects, and to provide
   general security considerations.

   The Ogg container format is known to contain [Theora] or [Dirac]
   video, [Speex] (narrow-band and wide-band) speech, [Vorbis] or [FLAC]
   audio, and [CMML] timed text/metadata. As Ogg encapsulates binary
   data, it is possible to include any other type of video, audio,
   image, text, or, generally speaking, any time-continuously sampled
   data.

   While raw packets from these data sources may be used directly by
   transport mechanisms that provide their own framing and packet-
   separation mechanisms (such as UDP datagrams or RTP), Ogg is a
   solution for stream based storage (such as files) and transport (such
   as TCP streams or pipes). The media types defined in this document
   are needed to correctly identify such content when it is served over
   HTTP, included in multi-part documents, or used in other places where
   media types [RFC2045] are used.

2. Changes Since RFC 3534

   o The type "application/ogg" is redefined.

   o The types "video/ogg" and "audio/ogg" are defined.

   o New file extensions are defined.

   o New Macintosh file type codes are defined.

   o The codecs parameter is defined for optional use.

   o The Ogg Skeleton extension becomes a recommended addition for
      content served under the new types.

3. Conformance and Document Conventions

   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 BCP 14, [RFC2119] and
   indicate requirement levels for compliant implementations.
   Requirements apply to all implementations unless otherwise stated.

   An implementation is a software module that supports one of the media
   types defined in this document. Software modules may support
   multiple media types, but conformance is considered individually for
   each type.

   Implementations that fail to satisfy one or more "MUST" requirements
   are considered non-compliant. Implementations that satisfy all
   "MUST" requirements, but fail to satisfy one or more "SHOULD"
   requirements, are said to be "conditionally compliant". All other
   implementations are "unconditionally compliant".

4. Deployed Media Types and Compatibility

   The application/ogg media type has been used in an ad hoc fashion to
   label and exchange multimedia content in Ogg containers.

   Use of the "application" top-level type for this kind of content is
   known to be problematic, in particular since it obfuscates video and
   audio content. This document thus defines the media types,

   o video/ogg

   o audio/ogg

   which are intended for common use and SHOULD be used when dealing
   with video or audio content, respectively. This document also
   obsoletes the [RFC3534] definition of application/ogg and marks it
   for complex data (e.g., multitrack visual, audio, textual, and other
   time-continuously sampled data), which is not clearly video or audio
   data and thus not suited for either the video/ogg or audio/ogg types.
   Refer to the following section for more details.

   An Ogg bitstream generally consists of one or more logical bitstreams
   that each consist of a series of header and data pages packetising
   time-continuous binary data [RFC3533]. The content types of the
   logical bitstreams may be identified without decoding the header
   pages of the logical bitstreams through use of a [Skeleton]
   bitstream. Using Ogg Skeleton is REQUIRED for content served under

   the application/ogg type and RECOMMENDED for video/ogg and audio/ogg,
   as Skeleton contains identifiers to describe the different
   encapsulated data.

   Furthermore, it is RECOMMENDED that implementations that identify a
   logical bitstream that they cannot decode SHOULD ignore it, while
   continuing to decode the ones they can. Such precaution ensures
   backward and forward compatibility with existing and future data.

   These media types can optionally use the "codecs" parameter described
   in [RFC4281]. Codecs encapsulated in Ogg require a text identifier
   at the beginning of the first header page, hence a machine-readable
   method to identify the encapsulated codecs would be through this
   header. The following table illustrates how those header values map
   into strings that are used in the "codecs" parameter when dealing
   with Ogg media types.

Codec Identifier	Codecs Parameter
char[5]: 'BBCD\0'	dirac
char[5]: '\177FLAC'	flac
char[7]: '\x80theora'	theora
char[7]: '\x01vorbis'	vorbis
char[8]: 'CELT '	celt
char[8]: 'CMML\0\0\0\0'	cmml
char[8]: '\213JNG\r\n\032\n'	jng
char[8]: '\x80kate\0\0\0'	kate
char[8]: 'OggMIDI\0'	midi
char[8]: '\212MNG\r\n\032\n'	mng
char[8]: 'PCM '	pcm
char[8]: '\211PNG\r\n\032\n'	png
char[8]: 'Speex '	speex
char[8]: 'YUV4MPEG'	yuv4mpeg

   An up-to-date version of this table is kept at Xiph.org (see
   [Codecs]).

   Possible examples include:

   o application/ogg; codecs="theora, cmml, ecmascript"

   o video/ogg; codecs="theora, vorbis"

   o audio/ogg; codecs=speex

5. Relation between the Media Types

   As stated in the previous section, this document describes three
   media types that are targeted at different data encapsulated in Ogg.
   Since Ogg is capable of encapsulating any kind of data, the multiple
   usage scenarios have revealed interoperability issues between
   implementations when dealing with content served solely under the
   application/ogg type.

   While this document does redefine the earlier definition of
   application/ogg, this media type will continue to embrace the widest
   net possible of content with the video/ogg and audio/ogg types being
   smaller subsets of it. However, the video/ogg and audio/ogg types
   take precedence in a subset of the usages, specifically when serving
   multimedia content that is not complex enough to warrant the use of
   application/ogg. Following this line of thought, the audio/ogg type
   is an even smaller subset within video/ogg, as it is not intended to
   refer to visual content.

   As such, the application/ogg type is the recommended choice to serve
   content aimed at scientific and other applications that require
   various multiplexed signals or streams of continuous data, with or
   without scriptable control of content. For bitstreams containing
   visual, timed text, and any other type of material that requires a
   visual interface, but that is not complex enough to warrant serving
   under application/ogg, the video/ogg type is recommended. In
   situations where the Ogg bitstream predominantly contains audio data
   (lyrics, metadata, or cover art notwithstanding), it is recommended
   to use the audio/ogg type.

6. Encoding Considerations

   Binary: The content consists of an unrestricted sequence of octets.

   Note:

   o Ogg encapsulated content is binary data and should be transmitted
      in a suitable encoding without CR/LF conversion, 7-bit stripping,
      etc.; base64 [RFC4648] is generally preferred for binary-to-text
      encoding.

   o Media types described in this document are used for stream based
      storage (such as files) and transport (such as TCP streams or
      pipes); separate types are used to identify codecs such as in
      real-time applications for the RTP payload formats of Theora
      [ThRTP] video, Vorbis [RFC5215], or Speex [SpRTP] audio, as well
      as for identification of encapsulated data within Ogg through
      Skeleton.

7. Security Considerations

   Refer to [RFC3552] for a discussion of terminology used in this
   section.

   The Ogg encapsulation format is a container and only a carrier of
   content (such as audio, video, and displayable text data) with a very
   rigid definition. This format in itself is not more vulnerable than
   any other content framing mechanism.

   Ogg does not provide for any generic encryption or signing of itself
   or its contained bitstreams. However, it encapsulates any kind of
   binary content and is thus able to contain encrypted and signed
   content data. It is also possible to add an external security
   mechanism that encrypts or signs an Ogg bitstream and thus provides
   content confidentiality and authenticity.

   As Ogg encapsulates binary data, it is possible to include executable
   content in an Ogg bitstream. Implementations SHOULD NOT execute such
   content without prior validation of its origin by the end-user.

   Issues may arise on applications that use Ogg for streaming or file
   transfer in a networking scenario. In such cases, implementations
   decoding Ogg and its encapsulated bitstreams have to ensure correct
   handling of manipulated bitstreams, of buffer overflows, and similar
   issues.

   It is also possible to author malicious Ogg bitstreams, which attempt
   to call for an excessively large picture size, high sampling-rate
   audio, etc. Implementations SHOULD protect themselves against this
   kind of attack.

   Ogg has an extensible structure, so that it is theoretically possible
   that metadata fields or media formats might be defined in the future
   which might be used to induce particular actions on the part of the
   recipient, thus presenting additional security risks. However, this
   type of capability is currently not supported in the referenced
   specification.

   Implementations may fail to implement a specific security model or
   other means to prevent possibly dangerous operations. Such failure
   might possibly be exploited to gain unauthorized access to a system
   or sensitive information; such failure constitutes an unknown factor
   and is thus considered out of the scope of this document.

8. Interoperability Considerations

   The Ogg container format is device-, platform-, and vendor-neutral
   and has proved to be widely implementable across different computing
   platforms through a wide range of encoders and decoders. A broadly
   portable reference implementation [libogg] is available under the
   revised (3-clause) BSD license, which is a Free Software license.

   The Xiph.Org Foundation has defined the specification,
   interoperability, and conformance and conducts regular
   interoperability testing.

   The use of the Ogg Skeleton extension has been confirmed to not cause
   interoperability issues with existing implementations. Third parties
   are, however, welcome to conduct their own testing.

9. IANA Considerations

   In accordance with the procedures set forth in [RFC4288], this
   document registers two new media types and redefines the existing
   application/ogg as defined in the following section.

10. Ogg Media Types

10.1. application/ogg

   Type name: application

   Subtype name: ogg

   Required parameters: none

   Optional parameters: codecs, whose syntax is defined in RFC 4281.
   See section 4 of RFC 5334 for a list of allowed values.

   Encoding considerations: See section 6 of RFC 5334.

   Security considerations: See section 7 of RFC 5334.

   Interoperability considerations: None, as noted in section 8 of RFC
   5334.

   Published specification: RFC 3533

   Applications which use this media type: Scientific and otherwise that
   require various multiplexed signals or streams of data, with or
   without scriptable control of content.

Additional information:

Magic number(s): The first four bytes, 0x4f 0x67 0x67 0x53,
correspond to the string "OggS".

File extension(s): .ogx

   RFC 3534 defined the file extension .ogg for application/ogg,
   which RFC 5334 obsoletes in favor of .ogx due to concerns where,
   historically, some implementations expect .ogg files to be solely
   Vorbis-encoded audio.

Macintosh File Type Code(s): OggX

Person & Email address to contact for further information: See
"Authors' Addresses" section.

Intended usage: COMMON

Restrictions on usage: The type application/ogg SHOULD only be used
in situations where it is not appropriate to serve data under the
video/ogg or audio/ogg types. Data served under the application/ogg
type SHOULD use the .ogx file extension and MUST contain an Ogg
Skeleton logical bitstream to identify all other contained logical
bitstreams.

Author: See "Authors' Addresses" section.

Change controller: The Xiph.Org Foundation.

10.2. video/ogg

Type name: video

Subtype name: ogg

Required parameters: none

Optional parameters: codecs, whose syntax is defined in RFC 4281.
See section 4 of RFC 5334 for a list of allowed values.

Encoding considerations: See section 6 of RFC 5334.

Security considerations: See section 7 of RFC 5334.

Interoperability considerations: None, as noted in section 8 of RFC
5334.

Published specification: RFC 3533

Applications which use this media type: Multimedia applications,
including embedded, streaming, and conferencing tools.

Additional information:

Magic number(s): The first four bytes, 0x4f 0x67 0x67 0x53,
correspond to the string "OggS".

File extension(s): .ogv

Macintosh File Type Code(s): OggV

Person & Email address to contact for further information: See
"Authors' Addresses" section.

Intended usage: COMMON

Restrictions on usage: The type "video/ogg" SHOULD be used for Ogg
bitstreams containing visual, audio, timed text, or any other type of
material that requires a visual interface. It is intended for
content not complex enough to warrant serving under "application/
ogg"; for example, a combination of Theora video, Vorbis audio,
Skeleton metadata, and CMML captioning. Data served under the type
"video/ogg" SHOULD contain an Ogg Skeleton logical bitstream.
Implementations interacting with the type "video/ogg" SHOULD support
multiplexed bitstreams.

Author: See "Authors' Addresses" section.

Change controller: The Xiph.Org Foundation.

10.3. audio/ogg

Type name: audio

Subtype name: ogg

Required parameters: none

Optional parameters: codecs, whose syntax is defined in RFC 4281.
See section 4 of RFC 5334 for a list of allowed values.

Encoding considerations: See section 6 of RFC 5334.

Security considerations: See section 7 of RFC 5334.

Interoperability considerations: None, as noted in section 8 of RFC
5334.

Published specification: RFC 3533

Applications which use this media type: Multimedia applications,
including embedded, streaming, and conferencing tools.

Additional information:

Magic number(s): The first four bytes, 0x4f 0x67 0x67 0x53,
correspond to the string "OggS".

File extension(s): .oga, .ogg, .spx

Macintosh File Type Code(s): OggA

Person & Email address to contact for further information: See
"Authors' Addresses" section.

Intended usage: COMMON

Restrictions on usage: The type "audio/ogg" SHOULD be used when the
Ogg bitstream predominantly contains audio data. Content served
under the "audio/ogg" type SHOULD have an Ogg Skeleton logical
bitstream when using the default .oga file extension. The .ogg and
.spx file extensions indicate a specialization that requires no
Skeleton due to backward compatibility concerns with existing
implementations. In particular, .ogg is used for Ogg files that
contain only a Vorbis bitstream, while .spx is used for Ogg files
that contain only a Speex bitstream.

Author: See "Authors' Addresses" section.

Change controller: The Xiph.Org Foundation.

11. Acknowledgements

   The authors gratefully acknowledge the contributions of Magnus
   Westerlund, Alfred Hoenes, and Peter Saint-Andre.

12. Copying Conditions

   The authors agree to grant third parties the irrevocable right to
   copy, use and distribute the work, with or without modification, in
   any medium, without royalty, provided that, unless separate
   permission is granted, redistributed modified works do not contain
   misleading author, version, name of work, or endorsement information.

13. References

13.1. Normative References

   [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.

   [RFC3533]   Pfeiffer, S., "The Ogg Encapsulation Format Version 0",
                RFC 3533, May 2003.

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

   [RFC4288]   Freed, N. and J. Klensin, "Media Type Specifications and
                Registration Procedures", BCP 13, RFC 4288,
                December 2005.

13.2. Informative References

   [CMML]   Pfeiffer, S., Parker, C., and A. Pang, "The Continuous
                Media Markup Language (CMML)", Work in Progress,
                March 2006.

   [Codecs]   Pfeiffer, S. and I. Goncalves, "Specification of MIME
                types and respective codecs parameter", July 2008,
                <http://wiki.xiph.org/index.php/MIMETypesCodecs>.

   [Dirac]   Dirac Group, "Dirac Specification",
                <http://diracvideo.org/specifications/>.

   [FLAC]   Coalson, J., "The FLAC Format",
                <http://flac.sourceforge.net/format.html>.

   [libogg]   Xiph.Org Foundation, "The libogg API", June 2000,
                <http://xiph.org/ogg/doc/libogg>.

   [Ogg]   Xiph.Org Foundation, "Ogg bitstream documentation: Ogg
                logical and physical bitstream overview, Ogg logical
                bitstream framing, Ogg multi-stream multiplexing",
                <http://xiph.org/ogg/doc>.

   [RFC3534]   Walleij, L., "The application/ogg Media Type", RFC 3534,
                May 2003.

   [RFC3552]   Rescorla, E. and B. Korver, "Guidelines for Writing RFC
               Text on Security Considerations", BCP 72, RFC 3552,
               July 2003.

   [RFC4648]   Josefsson, S., "The Base16, Base32, and Base64 Data
               Encodings", RFC 4648, October 2006.

   [RFC5215]   Barbato, L., "RTP Payload Format for Vorbis Encoded
               Audio", RFC 5215, August 2008.

   [Skeleton] Pfeiffer, S. and C. Parker, "The Ogg Skeleton Metadata
               Bitstream", November 2007,
               <http://xiph.org/ogg/doc/skeleton.html>.

   [Speex]   Valin, J., "The Speex Codec Manual", February 2002,
               <http://speex.org/docs/manual/speex-manual>.

   [SpRTP]   Herlein, G., Valin, J., Heggestad, A., and A. Moizard,
               "RTP Payload Format for the Speex Codec", Work
               in Progress, February 2008.

   [Theora]   Xiph.Org Foundation, "Theora Specification",
               October 2007, <http://theora.org/doc/Theora.pdf>.

   [ThRTP]   Barbato, L., "RTP Payload Format for Theora Encoded
               Video", Work in Progress, June 2006.

   [Vorbis]   Xiph.Org Foundation, "Vorbis I Specification", July 2004,
               <http://xiph.org/vorbis/doc/Vorbis_I_spec.html>.

Authors' Addresses

   Ivo Emanuel Goncalves
   Xiph.Org Foundation
   21 College Hill Road
   Somerville, MA 02144
   US

   EMail: justivo@gmail.com
   URI: xmpp:justivo@gmail.com

   Silvia Pfeiffer
   Xiph.Org Foundation

   EMail: silvia@annodex.net
   URI: http://annodex.net/

   Christopher Montgomery
   Xiph.Org Foundation

   EMail: monty@xiph.org
   URI: http://xiph.org