WAVE File ISG Testing

### WAVE 4.1: WAVE File Header

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Check that ChunkID is RIFF in bytes 1-4. | 411.wav: changed first 4 bytes to ‘Test’ instead of ‘RIFF’ | Blocked; file type read as ‘DIY Thermocam raw data’ – file type unexpected and blocked |
| Check that Size is equal to 8 less than the byte-length of the file in bytes 5-8 (little-endian). Replace with correct size if size is wrong. | 412.wav: changed bytes 5-8 to reflect a size of 500000 bytes, when the value should be 234300 | Passed with change; size corrected |
| Check that FormType is WAVE in bytes 9-12. | 413.wav: changed bytes 9-12 to ‘Test’ instead of ‘WAVE’ | Blocked; file type unexpected and blocked |

### WAVE 4.2: Format Chunk Compression Code

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Validate the two-byte CompressionCode against the list of allowable entries | 421.wav: changed bytes 21-22 to 0xFFFF (not a valid compression code) instead of 0x0100 (PCM) | Blocked; filter\_gstaudio\_decode error code -115 |
| If the compression format is not PCM, verify that a Fact chunk exists | 422.wav: changed bytes 21-22 to 0x0300 (a valid, non-PCM code). File does not contain a fact chunk, so it should be rejected | Blocked; filter\_gstaudio\_decode error code -115 |

### WAVE 4.3: Format Chunk Number of Channels

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the value of nChannels is between 1-65536 | 431.wav: changed bytes 23-24 to 0x0000 instead of 0x0100 | Blocked; filter\_gstaudio\_decode error code -115 |
| Verify that the value of nChannels is less than some predefined threshold (More than 2 is not usually needed, and extra channels could be used to hide data) and remove all channels above the threshold | 432.wav: contains an extra two channels (for 3 total). Extra channel should be removed | Blocked; filter\_gstaudio\_decode error code -104 |

### WAVE 4.4: Format Chunk Parameter Consistency

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Validate that the format chunk parameters hold true for the above equations | 441.wav: changed bytes 25-28 (nSamplesPerSec) to 0xFFFFFFFF, invalidating the first equation. Changed bytes 33-34 to 0xFFFF | Blocked; filter\_gstaudio\_decode error code -115 |

### WAVE 4.5: Format Chunk Quantization

BitsPerSample values are rounded up to the nearest multiple of 8. For example, 9 bits per sample gets rounded up to 16 bits, leaving 7 bits that could be used to hide data

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| If BitsPerSample is not a multiple of 8, extra bits should be overwritten to zero | 451.wav: 9 bits of audio rounded up to 16 bits |  |
| If BitsPerSample exceeds a predefined threshold, reject the file | 452.wav: 64 bit audio |  |

### WAVE 4.6: Fact Chunk Parameters

When the Fact chunk is present (required for non-PCM formats), bytes 9-12 specify the length of the audio data in bytes (dwSampleLength)

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that dwSampleLength is correct. If not, replace with the correct value | 461.wav: dwSampleLength changed to 1 |  |

### WAVE 4.7: Data Chunk Parameter Consistency

Information is stored in the Data chunk in instances of time starting with the oldest time and progressing towards the newest. An instance consists of N samples, where N is the number of channels, and a sample consists of P bits, where P is the bits per sample

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the ChunkID is data, reject if not | 471.wav: Change bytes 37-40 to say ‘Test’ instead of ‘data’ |  |
| Verify that the data chunk is preceded by the Format chunk | 472.wav: Changed bytes 13-36 to zero, overwriting format chunk |  |

### WAVE 4.8: Data Chunk Audio Samples

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Check for steganographic material by comparing bit statistics with known bit statistics | 481.wav: Speech from <https://freesound.org/people/malak1236/sounds/253740/>  Includes a steganographic python script |  |
| Zero the least significant bit, or apply a bitmask to the least significant bit in a way that can’t be easily reversed | 482.wav: White noise  Includes a steganographic python script |  |

### WAVE 4.9: WAVE List Chunk

List Chunks are optional and rarely used; their purpose is to reduce file size for audio that contains lots of silence.

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the ChunkID is LIST | 491.wav: Changed ChunkID to ‘Test’ |  |
| Verify that the FormType is wavl | 492.wav: Changed FormType to ‘Test’ |  |
| If a primary data chunk is not present, verify that at least one data chunk is present in the List chunk. Reject if not | 493.wav: Contains a wavl chunk with only one slnt chunk and no data |  |
| If at least one silent chunk is present, verify that the data and silent chunks alternate | 494.wav: wavl chunk has two concurrent silent chunks |  |

### WAVE 4.10: Silent Chunk

If no fact chunk is present, where size is the size of the data chunk in bytes, nSamplersPerSec is the sampling rate in the fmt\_ chunk, and wBitsperSample is the sample size from the fmt\_ chunk

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the Chunk ID is slnt | 4101.wav: changed chunk id to ‘test’ |  |
| If a fact chunk is present, the silence length (dwSamples) should be less than the fact chunk’s dwSampleLength. If not, replace with a value that is | 4102.wav: slnt chunk dwSamples changed to 2000000, longer than fact chunk dwSampleLength |  |
| If no fact chunk is present, verify dwSamples satisfies the equation above. | 4103.wav: slnt chunk dwSamples changed to 2000000, no longer satisfies above equation |  |

### WAVE 4.11: Cue Chunk

Cue chunks are optional and are used to specify ‘points of interest’ in audio track. No more than one cue chunk can be present.

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the Chunk ID is ‘cue ‘ (including the space) |  |  |
| Verify that there are dwCuePoints items in the cue point list |  |  |
| For each cue point, verify that dwName is unique |  |  |
| For each item in the cue points list, verify that dwPosition points to a valid segment number in the Playlist chunk |  |  |
| For each item in the cue points list, verify that fccChunk is either data or slnt |  |  |
| For each item in the cue points list, verify that the cue point refers to a valid section within the data or slnt chunk specified in the fccChunk field |  |  |
| Check that dwChunkStart is the beginning of the data or slnt chunk that the cue point references. If there is only one data chunk, then ensure this value is zero |  |  |
| If a WAVE-List chunk is present that contains multiple data or slnt chunks, then check that the dwChunkStart value is the byte offset to the beginning of the data or slnt chunk relative to the Data section of the wave-LIST chunk |  |  |
| If compressed, check that dwBlockStart points to the beginning of the block of compressed data within a data or slnt chunk that the cue point references |  |  |
| Check that dwSampleOffset is pointing to the correct location |  |  |
| Remove the cue points chunk and associated playlist chunk to avoid skipping around the audio data during playback |  |  |

### WAVE 4.12: Playlist Chunk

Playlist chunks specify the play order for the cue points in a cue chunk

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the Chunk ID is plst |  |  |
| Check the cue length and loop time values to ensure that they are reasonable values (i.e., cue length should be no greater than the length of the data or slnt chunk data portion; the loop time should be no greater than a predetermined time value) |  |  |
| Ensure that every cue point name in the playlist chunk is contained in the cue chunk |  |  |
| Remove the playlist chunk and associated cue points chunk to avoid skipping around the audio data during playback |  |  |

### WAVE 4.13: Associated Data List Chunk

Contains labels, titles, and comments associated with cue points in the cue chunk. Could be used to hide data, especially if no cue chunk is present

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the Chunk ID is adtl |  |  |
| Verify that the file contains a cue chunk |  |  |
| Verify that dwCountry is a valid entry |  |  |
| Verify that dwLanguage is a valid entry |  |  |
| Verify that dwDialect is a valid entry |  |  |
| Remove the associated data list chunk and associated label, note, and labeled text subchunks to avoid skipping around the audio data during playback |  |  |

### WAVE 4.14: Label Subchunk

Associates a text label with a cue point. Must be a subchunk of the associated-data-list chunk

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the Chunk ID is labl |  |  |
| Verify that the cue point ID name dwName corresponds to a valid cue point |  |  |
| Remove the associated data list chunk and associated label, note, and labeled text subchunks to avoid skipping around the audio data during playback |  |  |
| Replace the Text field with a standard template or null characters |  |  |
| Pass the contents of the Text fields to an external filter |  |  |

### WAVE 4.15: Note Subchunk

Associates text with a cue point. Must be a subchunk of the associated-data-list chunk.

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the chunk identifier value is note |  |  |
| Verify that the cue point ID name dwName corresponds to a valid cue point |  |  |
| Remove the associated data list chunk and associated label, note, and labeled text subchunks to avoid skipping around the audio data during playback |  |  |
| Replace the Text field with a standard template or null characters |  |  |
| Pass the Text field to an external filter |  |  |

### WAVE 4.16: Labeled Text Subchunk

Similar to Note subchunk, with more information

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that the ChunkID is ltxt |  |  |
| Verify that the cue point ID name dwName corresponds to a valid cue point |  |  |
| Verify that dwCountry is a valid entry |  |  |
| : Verify that dwLanguage is a valid entry |  |  |
| Verify that dwDialect is a valid entry |  |  |
| Remove the associated data list chunk and associated label, note, and labeled text subchunks to avoid skipping around the audio data during playback |  |  |
| Replace the Comment field with a standard template or null characters |  |  |
| Pass the Comment field to an external filter |  |  |

### WAVE 4.17: Junk Chunk

Contains filler bytes to align RIFF chunks to certain boundaries. Information could be hidden in the filler chunks and ignored by most WAVE applications.

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that that the ChunkID is JUNK | 4171.wav: Changed Junk Chunk ID from ‘JUNK’ to ‘TEST’ |  |
| Verify that the byte length of Filler is Size – 8 | 4172.wav: Changed size to 1000 |  |
| Overwrite Filler data with zeros, keeping boundaries intact | 4173.wav: filler data contains phrase ‘THIS IS A TEST’ |  |

### WAVE 4.18: Pad Chunk

Essentially the same as the junk chunk

|  |  |  |
| --- | --- | --- |
| Recommendation | Test | Result |
| Verify that that the ChunkID is PAD\_ (underscore is a space) | 4181.wav: changed Pad Chunk ID from ‘PAD ‘ to ‘test’ |  |
| Verify that the byte length of Filler is Size – 8 | 4182.wav: Changed size to 1000 |  |
| Overwrite Filler data with zeros, keeping boundaries intact | 4183.wav: filler data contains phrase ‘THIS IS A TEST’ |  |

### 