Network Working Group
Internet-Draft
Intended status: Informational
Expires: 27 September 2021

Y. Shafranovich
Nightwatch Cybersecurity

26 March 2021

Common Format and MIME Type for Comma-Separated Values (CSV) Files
draft-shafranovich-rfc4180-bis-00

Abstract

This RFC documents the common format used for Comma-Separated Values (CSV) files and updates the associated MIME type "text/csv".

Status of This Memo

This Internet-Draft will expire on 27 September 2021.

Copyright Notice

Table of Contents

1. Introduction

The comma separated values format (CSV) has been used as a common way to exchange data between disparate systems and applications for many years. Surprisingly, while this format is very popular, it has never been formally documented and didn't have a media type registered. This was addressed in 2005 via publication of [RFC4180] and the concurrent registration of the "text/csv" media type.

Since the publication of [RFC4180], the CSV format has evolved and this specification seeks to reflect these changes as well as update the "text/csv" media type registration.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Motivation For and Status of This Document

The original motivation of [RFC4180] was to provide a reference in order to register the media type "text/csv". It tried to document existing practices at the time based on the approaches used by most implementations. This document continues to do the same, and updates the original document to reflect current practices for generating and consuming of CSV files.

Both [RFC4180] and this document are published as informational RFC for the benefit of the Internet community and and not intended to be used as formal standards. Implementers should consult [RFC1796] and [RFC2026] for crucial differences between IETF standards and informational RFCs.

2. Definition of the CSV Format

While there had been various specifications and implementations for the CSV format (for ex. [CREATIVYST], [EDOCEO], [CSVW] and [ART])), prior to publication of [RFC4180] there is no attempt to provide a common specification. This section documents the format that seems to be followed by most implementations (incorporating changes since the publication of [RFC4180]).

2.1. High level description

1. Each record is located on a separate line, ended by a line break (CR, LF or CRLF). For example:

   ```
   aaa,bbb,cccCRLF
   zzz,yyy,xxxCRLF
   ```

2. The last record in the file MUST have an ending line break. For example:

   ```
   aaa,bbb,cccCRLF
   zzz,yyy,xxxCRLF
   ```

3. The first record in the file MAY be an optional header with the same format as normal records. This header will contain names corresponding to the fields in the file and SHOULD contain the same number of fields as the records in the rest of the file. For example:

   ```
   field_name_1,field_name_2,field_name_3CRLF
   aaa,bbb,cccCRLF
   zzz,yyy,xxxCRLF
   ```

4. Within each record, there MAY be one or more fields, separated by commas. Each record SHOULD contain the same number of fields throughout the file. Spaces are considered part of a field and SHOULD NOT be ignored. The last field in the record MUST NOT be followed by a comma. For example:

   ```
   aaa,bbb,cccCRLF
   ```

5. Each field MAY be enclosed in double quotes (however some programs, do not use double quotes at all). If fields are not enclosed with double quotes, then double quotes MUST NOT appear

inside the fields. For example:

```
"aaa","bbb","ccc"CRLF
zzz,yyy,xxxCRLF
```

6. Fields containing line breaks (CR, LF or CRLF), double quotes, or commas MUST be enclosed in double-quotes. The same applies for the first field of a record that starts with a hash to avoid the field from being parsed as a comment. For example:

```
"aaa","b CRLF
bb","ccc"CRLF
zzz,yyy,xxxCRLF
"#aaa",#bbb,cccCRLF
```

7. A double-quote appearing inside a field MUST be escaped by preceding it with another double quote. For example:

```
"aaa","b""bb","ccc"CRLF
```

8. A hash sign MAY be used to mark lines that are meant to be commented lines. A commented line can contain any whitespace or visible character until it is terminated by a line break (CR, LF or CRLF). A comment line MAY appear in any line of the file (before or after an OPTIONAL header) but MUST NOT be mistaken with a subsequent line of a multi-line field. Subsequent lines of multi-line fields can start with a hash sign and MUST NOT interpreted as comments. For example:

```
#commentCRLF
aaa,bbb,cccCRLF
#comment 2CRLF
"aaa","this is CRLF
# not a comment","ccc"CRLF
```

2.2. Default charset and line break values

Since the initial publication of [RFC4180], the default charset for "text/*" media types has been changed to UTF-8 (as per [RFC6657]). This document reflects this change and the default charset for CSV files is now UTF-8.

Although section 4.1.1. of [RFC2046] defines CRLF to denote line breaks, implementers MAY recognize a single CR or LF as a line break (similar to section 3.1.1.3 of [RFC7231]). However, some implementations MAY use other values.

2.3. ABNF Grammar

The ABNF grammar (as per [RFC5234]) appears as follows:

```
file = *((comment / record) linebreak)
```

```
comment = hash *comment-data

record = first-field *(comma field)

linebreak = CR / LF / CRLF

first-field = (escaped / first-non-escaped)

field = (escaped / non-escaped)

escaped = DQUOTE *(data-with-hash / comma / CR / LF / 2DQUOTE) DQUOTE

first-non-escaped = [data *data-with-hash]

non-escaped = *data-with-hash

comma = %x2C

hash = %x23

comment-data = WSP / %x21-7E / UTF8-data
         ; characters without control characters

data = WSP / %x21 / %x24-2B / %x2D-7E / UTF8-data
         ; characters without control characters, comma, hash and DQUOTE

data-with-hash = data / hash

CR = %x0D ; as per section B.1 of [RFC5234]

DQUOTE = %x22 ; as per section B.1 of [RFC5234]

LF = %x0A ; as per section B.1 of [RFC5234]

CRLF = CR LF ; as per section B.1 of [RFC5234]

HTAB = %x09 ; as per section B.1 of [RFC5234]

SP = %x20 ; as per section B.1 of [RFC5234]

WSP = SP / HTAB ; as per section B.1 of [RFC5234]

UTF8-data = UTF8-2 / UTF8-3 / UTF8-4 ; as per section 4 of [RFC3629]
```

Note that the authoritative definition of UTF-8 is in [UNICODE].

3. Common implementation concerns

This section describes some common concerns that may arise when producing or parsing CSV files. All of these remain out of scope for this document and are included for awareness. Implementers may also use other means to handle these use cases such as [CSVW].

3.1. Null values

Some implementations (such as databases) treat empty fields and null values differently. For these implementations, there is a need to define a special value representing a null.

Example of a CSV file with nulls (if "NULL" is used to mark nulls):

```
field_name_1,field_name_2,field_name_3CRLF
aaa,bbb,cccCRLF
zzz,NULL,xxxCRLF
```

3.2. Empty files

Implementers should be aware that in accordance to this specification a file does not need to contain any comments or records (empty file with zero bytes).

3.3. Empty lines

This specification recommends but doesn't require having the same number of fields in every line. This allows CSV files to have empty lines without any fields at all. Some implementations can be configured to skip empty lines instead of parsing them.

Example of a CSV file with empty lines:

```
field_name_1,field_name_2,field_name_3CRLF
aaa,bbb,cccCRLF
CRLF
zzz,yyy,xxxCRLF
```

However, if the records are only made up of one field it is not possible to differentiate between an empty line, and an empty and unquoted field. This differentiation might play an important role in some implementations such as database exports/imports.

Example of a CSV file with empty lines and only one field per record:

```
aaa
CRLF
bbbCRLF
```

3.4. Fields spanning multiple lines

When quoted fields are used, it is possible for a field to span multiple lines, even when line breaks appear within such field.

3.5. Unique header names

Implementers should be aware that some applications may treat header values as unique (either case-sensitive or case-insensitive).

3.6. Whitespace outside of quoted fields

When quoted fields are used, this document does not allow whitespace between double quotes and commas. Implementers should be aware that some applications may be more lenient and allow whitespace outside the double quotes.

3.7. Other field separators

This document defines a comma as a field separator but implementers should be aware that some applications may use different values, especially with non-English languages. Those are outside the scope of this document and implementers should consult other efforts such as [CSVW].

3.8. Escaping double quotes

This document prescribes that a double-quote appearing inside a field must be escaped by preceding it with another double quote. Implementers should be aware that some applications may choose to use a different escaping mechanism.

3.9. BOM header

Applications that create text files with unicode character encoding might write a BOM (byte order mark) header in order to support multiple unicode encodings (like UTF-16 and UTF-32). Some applications might be able to read and properly interpret such a header, others could break. Implementors should review section 6 of [RFC3629] and section 23.8 of [UNICODE].

4. Update to MIME Type Registration of text/csv

The media type registration of "text/csv" should be updated as per specific fields below:

Encoding considerations:

CSV MIME entities can consist of binary data as per section 4.8 of [RFC6838]. Although section 4.1.1. of [RFC2046] defines CRLF to denote line breaks, implementers MAY recognize a single CR or LF as a line break (similar to section 3.1.1.3 of [RFC7231]). However, some implementations may use other values.

Published specification:

While numerous private specifications exist for various programs and systems, there is no single "master" specification for this format. An attempt at a common definition can be found in [RFC4180] and this document. Implementers should note that both documents are informational in nature and are not standards.

Optional parameters: charset

The "charset" parameter specifies the charset employed by the CSV content. In accordance with [RFC6657], the charset parameter SHOULD be used, and if it is not present, UTF-8 SHOULD be assumed as the default (this implies that US- ASCII CSV will work, even when not specifying the "charset" parameter). Any charset defined by IANA for the "text" tree may be used in conjunction with the "charset" parameter.

Interoperability considerations:

Due to lack of a single specification, there are considerable differences among implementations. Implementers should "be conservative in what you do, be liberal in what you accept from others" ([RFC0793]) when processing CSV files. An attempt at a common definition can be found in Section 2.

4.1. IANA Considerations

IANA is directed to update the MIME type registration for "text/csv" as per instructions provided in Section 4 of this document and include a reference to this document within the registration.

5. Security Considerations

All security considerations as discussed in [RFC4180] still apply.

6. Acknowledgments

In addition to everyone thanked previously in [RFC4180], the author would like to thank acknowledge the contributions of the following people to this document: Alperen Belgic, Abed BenBrahim, Benjamin Kaduk, Damon Koach, Barry Leiba, Oliver Siegmar, Marco Diniz Sousa and Greg Skinner.

A special thank you to L.T.S.

7. References

7.1. Normative References

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997,

<https://www.rfc-editor.org/info/rfc2119>.

[RFC4180] Shafranovich, Y., "Common Format and MIME Type for Comma-Separated Values (CSV) Files", RFC 4180, DOI 10.17487/RFC4180, October 2005, <https://www.rfc-editor.org/info/rfc4180>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC6657] Melnikov, A. and J. Reschke, "Update to MIME regarding "charset" Parameter Handling in Textual Media Types", RFC 6657, DOI 10.17487/RFC6657, July 2012, <https://www.rfc-editor.org/info/rfc6657>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <https://www.rfc-editor.org/info/rfc6838>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

7.2. Informative References

[ART] Raymond, E., "The Art of Unix Programming, Chapter 5", September 2003, <http://www.catb.org/~esr/writings/taoup/html/ch05s02.html>.

[CREATIVYST] Repici, J., "HOW-TO: The Comma Separated Value (CSV) File Format", 2010, <http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm>.

[CSVW] W3C, "CSV on the Web Working Group", 2016, <https://www.w3.org/2013/csvw/wiki/Main_Page>.

[EDOCEO] Edoceo, Inc., "Comma Separated Values (CSV) Standard File Format", 2020, <https://edoceo.com/dev/csv-file-format>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[RFC1796] Huitema, C., Postel, J., and S. Crocker, "Not All RFCs are Standards", RFC 1796, DOI 10.17487/RFC1796, April 1995, <https://www.rfc-editor.org/info/rfc1796>.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, DOI 10.17487/RFC2026, October 1996, <https://www.rfc-editor.org/info/rfc2026>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/info/rfc3629>.

[UNICODE] The Unicode Consortium, "The Unicode Standard, Version 13.0.0", March 2020, <https://www.unicode.org/versions/Unicode13.0.0/>.

Appendix A. Major changes since [RFC4180]

- Added a section clarifying motivation for this document and standards status
- Changing default encoding to UTF-8 and adding Unicode to the ABNF grammar
- Allowing CR, LF and CRLF for line breaks
- Allowing HTAB in text data
- Mandating a line break at the end of the last line in the file
- Making records and headers optional, thus allowing for an empty file
- Adding definition of commented lines
- Adding a section on common implementation concerns
- Removed "header" parameter for the MIME type

Appendix B. Note to Readers

Note to the RFC Editor: Please remove this section prior to publication.

Development of this draft takes place on Github at: https://github.com/nightwatchcyber/rfc4180-bis

Comments can also be sent to the ART mailing list at: https://www.ietf.org/mailman/listinfo/art

Full list of changes can be viewed via the IETF document tracker:

https://tools.ietf.org/html/draft-shafranovich-rfc4180-bis

Author's Address

Yakov Shafranovich
Nightwatch Cybersecurity

Email: yakov+ietf@nightwatchcybersecurity.com