IJDC | *Research Paper*

# The Generation of Revision Identifier (Rsid) Numbers in MS Word: Implications for Document Analysis

Dirk H.R. Spennemann
School of Agricultural, Environmental and Veterinary Sciences; Charles Sturt University, Australia

Clare L. Singh
School of Dentistry and Medical Sciences, Charles Sturt University, Australia.

**Abstract**

The 2007 implementation of the Office Open XML standard for Microsoft Word introduced the assignation of individual revision save identifiers (Rsid) to document editing sessions that end in a save action. The relevant standards ECMA (2016) and ISO/ IEC 29500-1:2016 (2016) stipulate that these Rsid should be allocated randomised but with increasing numerical value, thereby documenting the progress of the editing. As MS Word is the most ubiquitous word processing software, Rsid appear to be a useful tool to examine and provide evidence for a wide range of common document generation editing and modification processes and file management operations, with implications for document analysis including, but not limited to academic integrity issues in student assignment submissions (e.g. contract cheating).

This paper presents the results of a series of experiments conducted to assess whether and how well MS Word implements the ECMA and ISO/ IEC standards. The results show that the number of allocated Rsid indeed increases with each edit and save action, with the previous Rsids carried over and retained. The newly allocated Rsid, however, do not conform to the standard as the numerical value of a Rsid associated with a save action may be larger or smaller than any or all of those allocated during that of the previous save actions. The allocation of a new Rsid is not necessarily caused by an *edit* event but that a new Rsid can also be generated if a file is saved as rtf or if it is sent as an e-mail from *within* MS Word, although the file was not edited in any way. Rsid numbers are not generated if a person opens a MS Word document, reads it and closes the file without saving, making this action impossible to detect.

MS Word template files on a given machine contain document (root) Rsid numbers that are generated when a newly installed application is launched for the first time. As these will be embedded as legacy Rsid into every new file generated from that template file, they act as signatures for all MS Word documents that are created.

The experiments have shown that user behaviour has a direct influence on the number of Rsid represented in a given file. Although the implementation of Office Open XML chosen by Microsoft is not compliant with the relevant standards, and thus Rsid cannot be used determine the exact chronological order of all editing sequences within a given document, the Rsid retain their value for document forensics as they are associated with specific edit events, and illuminate the document writing and editing process.

*Submitted* 20 February 2023 ~ *Revision received* 27 September 2023 ~ *Accepted* 29 September 2023

Correspondence should be addressed to Dirk H.R. Spennemann, School of Agricultural, Environmental and Veterinary Sciences; Charles Sturt University; PO Box 789; Albury NSW 2640, Australia. Email: dspennemann@csu.edu.au

The *International Journal of Digital Curation* is an international journal committed to scholarly excellence and dedicated to the advancement of digital curation across a wide range of sectors. The IJDC is published by the University of Edinburgh on behalf of the Digital Curation Centre. ISSN: 1746-8256. URL: http://www.ijdc.net/





http://dx.doi.org/10.2218/ijdc.v18i1.870
DOI: 10.2218/ijdc.v18i1.870

# Introduction

The analysis of the inner workings of digital manuscript files created by word processing programs is of interest from a forensic science perspective for two reasons. One is the intentional manipulation of content for the purposes of hiding information (steganography). The other is to identify content generation and editing processes that may shed light on the origin and formation of the information contained in the files, which for example has a bearing on the investigation of integrity issues in student assignment submission. MS Word, supplied as part of the Microsoft Office suite, has become one of the most ubiquitously used word processing programs and is available for the main operating systems used in personal computing, Microsoft Windows and Apple Mac OS.

While MS Word files have long been used to hide data , the introduction of the Office Open XML standard in 2007 ensured that tighter control over editing sequences can be achieved. In response, numerous novel methods have been proposed adapting to the new standard with steganographic embedding of text in other elements of the XML file structure , file dead spaces ; the use of invisible or white spaces in the document text splitting or the adaptive use of track changes . Such steganographic techniques alter the integrity of the document and thus may be trackable if earlier versions of the text can be found. This paper will focus on the investigation of the sequencing of content generation and editing processes in MS Word.

While some work has been carried out on the relationship between temporary (.TMP) and final files (.DOCX) , as well as RAM dumps , these approaches require physical access to the computer or storage device in question. Such access is, under normal circumstances, only possible in law-enforcement situations. As the authors are specifically interested in the ability to examine stand-alone documents (for example as submitted to academic misconduct investigations), this paper will focus on the generation and deployment of revision save identifiers that are embedded in each MS Word document.

## Version tracking in in MS Word using Rsid numbers

Since 2007 Microsoft Word has been based on the Office Open XML standard . As part of this, the application assigns a revision save identifier (Rsid) number to every document editing session that ends with a save action (or an e-mail send). These Rsid numbers precede the respective text inserted during that session and are stored collectively in a settings file as a four-digit hexadecimal number. In the document XML file, each Rsid is then associated (in the form of tags) with specific actions such as text insertions, deletions (if "track changes" has been activated) and specific formatting, such as the insertion of page and section breaks as well as subsections of tables.

The manipulation of Rsid numbers has been investigated as means of steganographic embedding of text and watermarking which relies on substituting the automatically allocated Rsid numbers enclosing the encoded text with Rsid numbers that contain steganographic text or watermarks . The manipulation of Rsid numbers has also been raised as a means to "maliciously implicate an innocent computer user or create the appearance of a false correlation" .

The interpretation of Rsid information for digital document forensics has been discussed on a theoretical basis by a number of authors , with one paper presenting a case study of an actual implementation . In that paper, Langweg examined a copy a terrorist manual associated with a mass shooting event in Norway in 2011 with the aim of ascertaining whether the generation and editing of the text was consistent with the claims made by the apprehended suspect . This technique will gain in importance as the need for the conservation, curation and interpretation of 'born digital' content increases , in particular potential claims of authorship fraud.

Johnson and Davies have noted the potential usefulness of Rsid numbers for the detection of academic plagiarism, in particular contract cheating . The authors commented that Rsid numbers might be a tool to determine the chronological sequence of editing a given text, but

could not demonstrate the validity of the assumption . Theirs was a plausible assumption, given that the ECMA specifications for the Office Open XML standard stipulate that "every editing session shall be assigned a revision save ID that is larger than all earlier ones in the same file" . If the specifications are followed, then this feature would indeed allow to reconstruct the editing sequence (in chronological order) of any given file written in MS Word. This in turn would have unique benefits for document analysis and genealogical approaches to aid detailed understanding of content generation and editing processes.

While numerous authors have discussed Rsid numbers with varying objectives (see above), common to most papers is the assumption that Rsid implementation in MS Word indeed follows the ECMA and ISO/ IEC 29500-1:2016 standards . While the literature addresses the use of Rsid numbers, in reality there appears to be some ambiguity among authors in understanding how Rsid numbers are generated by MS Word. There is also no discussion on what the implications of this might be, in terms of document analysis.

The publicly accessible documentation by Microsoft is not helpful in that regard, as it is either silent or contradictory on this matter. In one set of documentation, Microsoft asserts that an "editing session shall be assigned a revision save ID that is larger than all earlier ones in the same file" thereby following the ECMA specification, while in another set of documentation Microsoft states that "Word generates a random number for every Rsid that is not necessarily larger than all earlier ones in the same file," because if "both are observed, the application would quickly run out of allowable Rsids" .

Clearly, a full understanding of the nature, the initial and subsequent allocation, as well as the persistence of Rsid numbers in MS Word documents has a bearing on their suitability for digital forensics. Consequently, this paper will examine through experimentation the allocation and modification of Rsid numbers by MS Word. The experimentation will focus on common document editing and file management operations. Findings will then be used to discuss the implications for document analysis.

# Methodology

A number of experiments were designed to examine the effects of a range of file management and document editing actions on the allocation of Rsid numbers in MS Word files. These included, for example, direct file copy, opening and reading files but not saving them, and editing files with single or multiple in-session saves. The experiments both simulate real world examples of workflows or provide baseline data to understand the functioning of Rsid allocation in MS Word.

## Accessing Rsid numbers

All Rsid numbers contained in a given document are stored in a summary table ('Rsidtbl') which lists them in increasing numerical order. In addition, the tag 'Rsidroot' shows the Rsid allocated to the document itself. There are two fundamental ways of accessing and extracting the Rsid information contained in MS Word files: by examining the XML files or by examining the data as exported in rich text format (rtf).

The former method, which relies on the fact that the Word docx format is in fact a zip archive, requires to 'crack open' the Word docx file (by replacing the .docx extension with .zip), extracting the file from the archive and accessing the Rsid information in the file *settings.xml*. In the settings.xml file all Rsid numbers are rendered as hexadecimal characters (e.g. Rsid004C4A41), which correspond with the text edits in the *document.xml* file.

The second method requires the file to be saved in rtf format and to open that file in a text editor (or as in the case of this paper) via MS Excel, where Rsid numbers are rendered as standard numerical strings (e.g. Rsid4999745).

Using either option, the list of Rsid numbers (decimal or hexadecimal) can be quickly located by searching for the summary table header 'Rsidtbl.'

## Experiment 1

Experiment 1 was designed to test whether it would be detectable if a person opens a MS Word document and reads it but closes the file without saving because MS Word might perform a hidden (background) save action in the process. It was posited that Rsid numbers would not be affected when a file was opened and then closed, but not saved. Experiment 1 started with a new, blank MS Word file created on a MacBook, using MS Word 2011 (version14.7.7). The file was saved as a *docx* file. From this, ten duplicates were generated. Two experiments were conducted:

1. Opening a set of ten files in MacBook MS Word 16.38 and closing without saving
2. Opening a set of ten files in MS Word 2013 (build 15.0.5233.1000) on a Windows machine and again closing without saving.

The relevant Rsid data were extracted from the XML Archive.

## Experiment 2

Experiment 2 was designed to illuminate the initial allocation of Rsid numbers. A copy of the base file generated for experiment 1 served as sample file (generated as file copy at the operating system level). The file was saved as a *docx* file. From this, 50 duplicates were generated, each of which was opened and saved in rtf format (experiment 2A). To assess whether MS Word for mobile devices would be different, the same was replicated on an Android phone running MS Word for mobile 1.0.1 (16.0.15629.20122). In this case a copy of the base file was emailed to and opened on the Android phone. From this, 50 duplicates were generated, each of which was opened in MS Word mobile and saved as *docx*. The files were then transferred to the MacBook (via file copy) and saved to rtf (experiment 2B). All subsequent processing of the rtf files followed the steps outlined in the section on accessing Rsid numbers.

## Experiment 3

Experiment 3 was designed to explore the nature of unique Rsid that were observed in the rtf files of experiment 2. As base file for this experiment served a file copy of the same base file that had been used for experiment 1. That file was opened and saved to rtf without any changes A further, renamed copy of that base file was created. The rtf file was opened and saved in docx format (but not closed), overwriting the initial base file copy. A renamed copy of that file was created. The open docx file was saved to rtf without any changes (under a new name). The last two steps were repeated twice over (Figure 1).

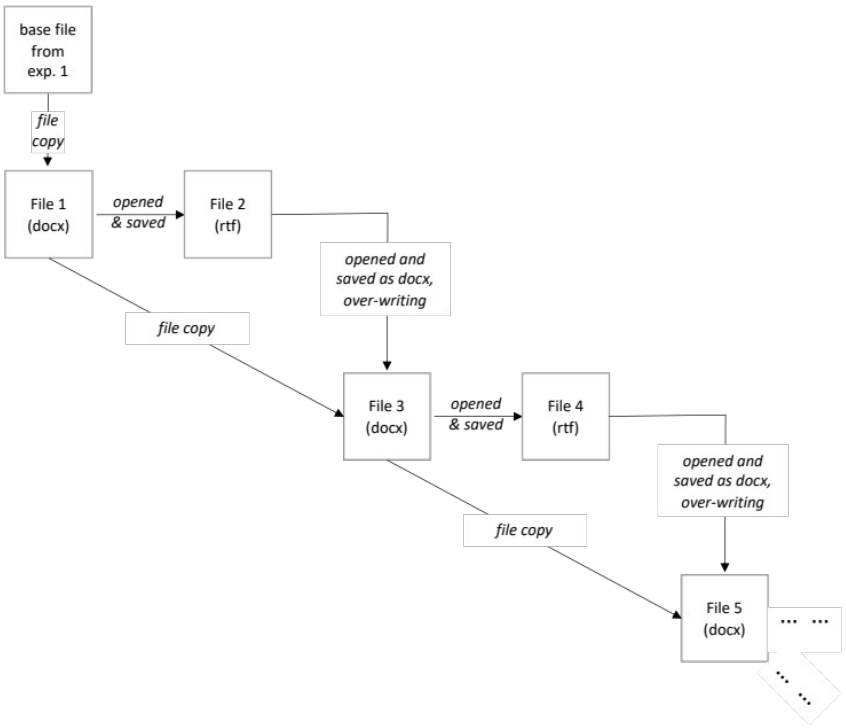


**Figure 1.** Flowchart for experiment 3.

## Experiment 4

Although computers tend to have fixed replacement schedules at least in the business and commercial world, and therefore a limited operational life span, the latency of legacy machines in the private sphere is much greater. Experiment 4 was designed to understand if different versions of MS Word, run on different operating systems, would assign Rsid in the same way as observed in experiment 2. Experiment 4 started with a copy of the base file generated for experiment 1. The file was opened, text pasted in and saved under a new file name. This file was reopened, further text pasted in, and again saved under a new file name. This continued iteratively until the ninth iteration, resulting in 11 files (experiment 4A) (Figure 2). The experiment was repeated by creating a new, blank MS Word file on a different MacBook using MS Word 2011 (version 14.3.9) (as experiment 4B) and creating a new, blank MS Word file on a Windows PC using MS Word 2013 (version 15.0) (as experiment 4C). Both were subjected to the same open-paste-save sequences as experiment 4A.

A variation copied the base file generated for experiment 1, but instead of adding text at each iteration, performed a single editing action before being saved as a new docx file. The editing actions were pasting text (with embedded Endnote entries); running Endnote; changing the Endnote output style; updating Endnote reference list (no new text pasted); making a serial change of words; colouring or bolding text; and changing paragraph styles. This was carried out using a MacBook with MS Word 16.38 (build 20061401) (experiment 4D) and on a Windows PC using MS Word 2008 (build 13127.21624) (Experiment 4E). All files were subsequently reopened and exported as rtf. All subsequent processing of the rtf files followed the steps outlined in the section on accessing Rsid numbers.

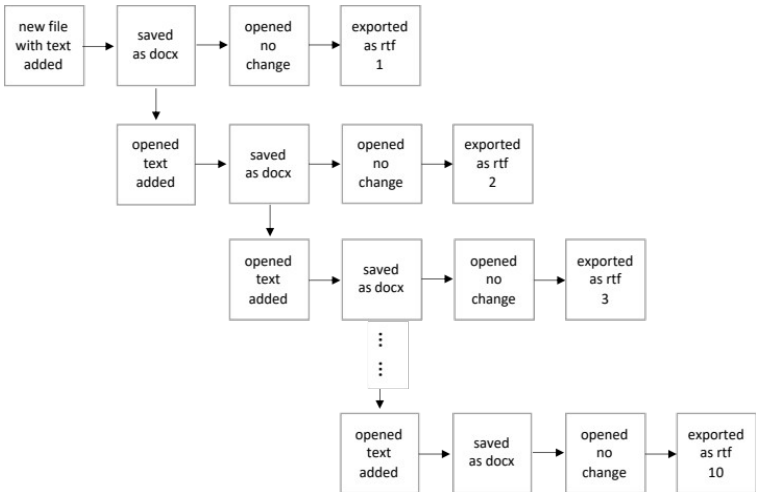


**Figure 2.** Workflow for the creation of files used in experiment 4a–c.

## Experiment 5

Experiment 5 was set up along similar lines to experiment 4, but the file was saved straight to rtf, omitting the interim step of saving to docx. The rtf file was then reopened, text added and again saved to rtf. That step was repeated until the ninth iteration, resulting in ten files. Five replicate sets were generated for this process (experiment 5A). The same process was repeated but only until the fourth iteration resulting in five files. Five replicate sets were generated for this series (experiment 5B) (Figure 3). All subsequent processing of the rtf files followed the steps outlined in the section on accessing Rsid numbers.

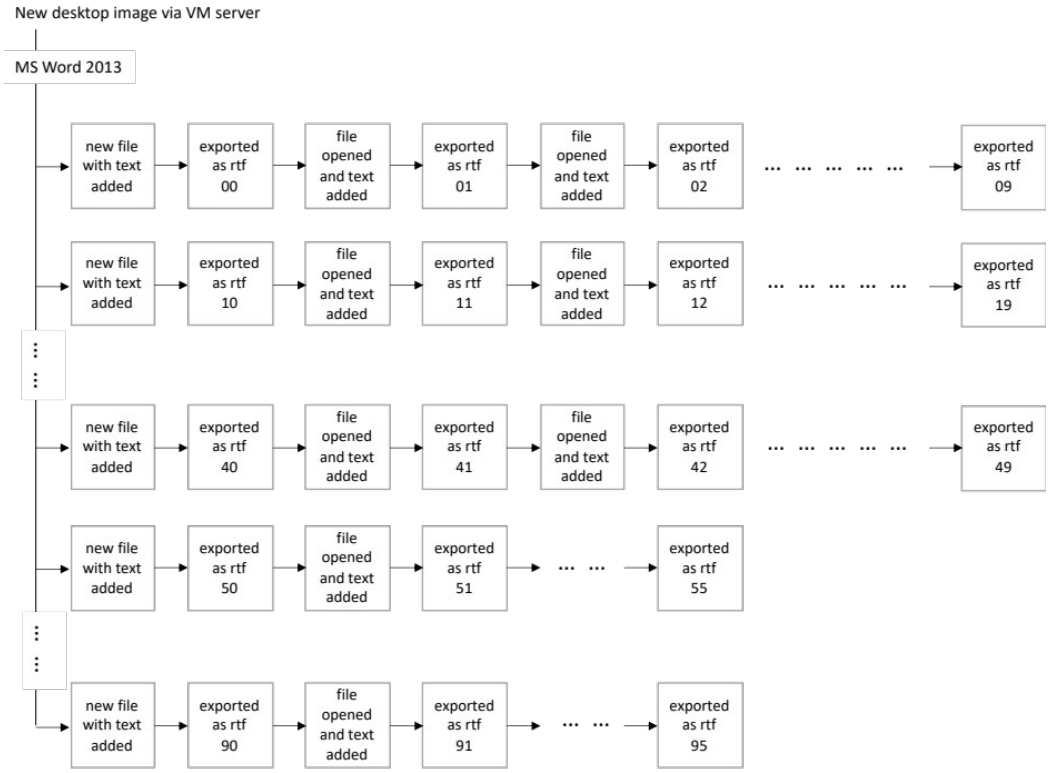


**Figure 3.** Workflow for the creation of files used in experiment 5

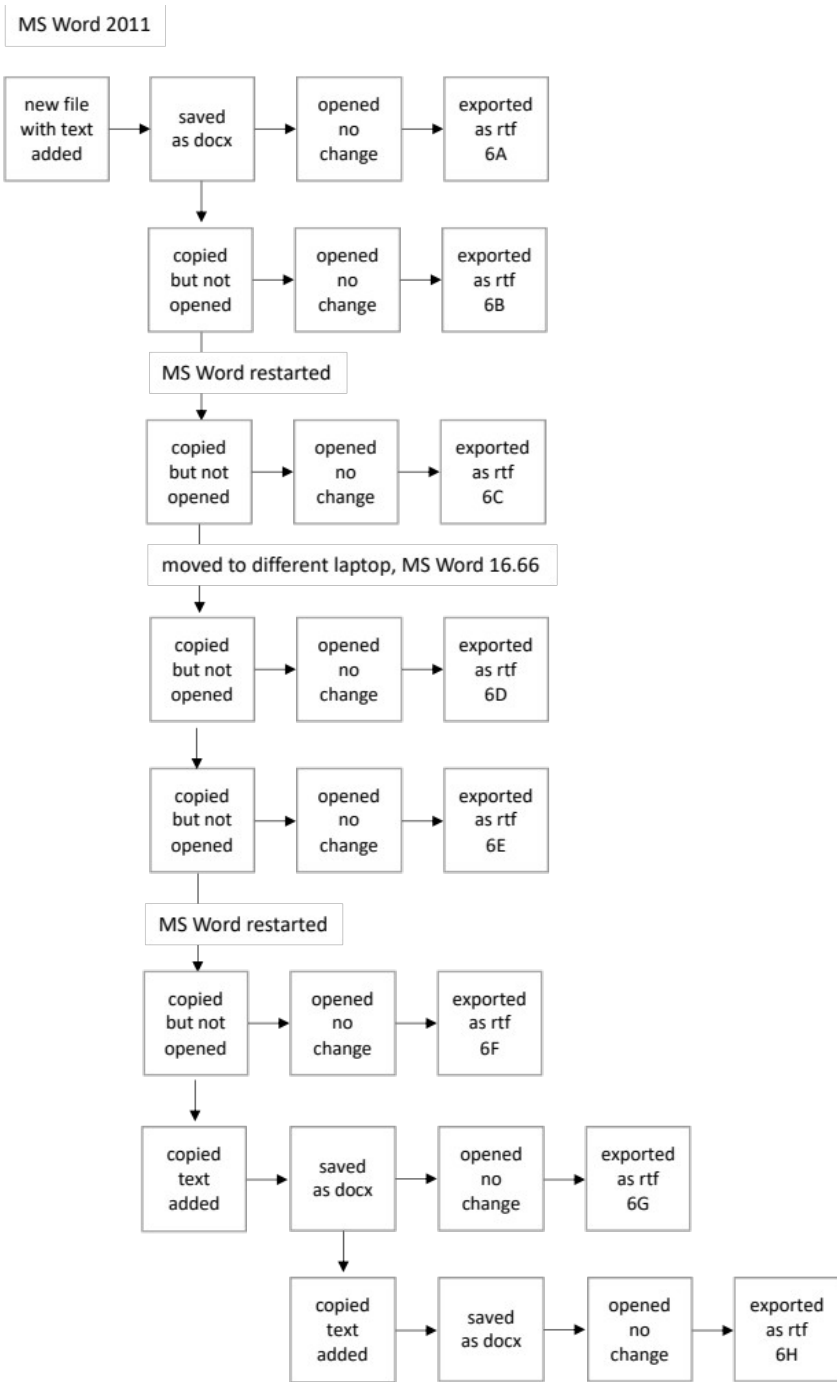


**Figure 4.** Workflow for the creation of files used in experiment 6.

The system used to generate the files was Windows MS Word 2013 (build 15.0.5233.1000), hosted on the institution's server and accessed via VM Horizon Client software. At the start of the session a new desktop was created, implying that no legacy files existed with the exception of the MS Word settings file that that comes preloaded with MS Office 2013.

## Experiment 6

Document development often occurs in a collaborative fashion, with files being shuttled between different authors and their computers (and possibly versions of MS Word). Experiment 6 was designed to examine the effects of opening and closing of the MS Word software instance after

the save action (and not merely closing the file), as well as movement of files between computers. A set of ten files was created on a MacBook, using MS Word 2011 (version 14.3.9). Copied sets of these files formed the basis of all variations (refer to Figure 4 for workflow). The variations were opening and saving to rtf (experiments 6A, 6B); restarting MS Word, opening and saving to rtf (experiment 6C); copying files to different a MacBook, opening and saving to rtf with MS Word (version 16.38) (experiments 6D, 6E); restarting MS Word on the second MacBook, opening and saving to rtf (6F); files opened, text added and saved to docx, reopened and saved to rtf (experiment 6G); files of experiment 6G copied, opened, text added and saved to docx, reopened and saved to rtf (experiment 6H). All subsequent processing of the rtf files followed the steps outlined in the section on accessing Rsid numbers.

## Experiment 7

The development of larger documents tends to an iterative process of text creation and editing that may span several days or weeks. Experiment 7 was designed to examine the effects of frequent saving *during* an editing process. As base file for this experiment served a copy of the same base that had been used for experiment 1. The file was opened, a single set of text pasted in and the file saved under a new name, but not closed. The same text was pasted twice and the file was again saved under a new name, but not closed. The same text was pasted three times and the was file again saved under a new name, but again not closed. This was continued for total of 12 iterations. The experiment was carried out on a MacBook with MS Word 16.38 (experiment 7A) and on a Windows PC MS Word 2008 (build 13127.21624) (experiment 7B) (Figure 5).

A variation was a similar approach, but in this case, a save action was performed after each paste event (experiments 7C, 7D) (Figure 6). The remainder of the process was the same as that for experiment 7A and 7B. All subsequent processing of the rtf files followed the steps outlined in the section on accessing Rsid numbers.

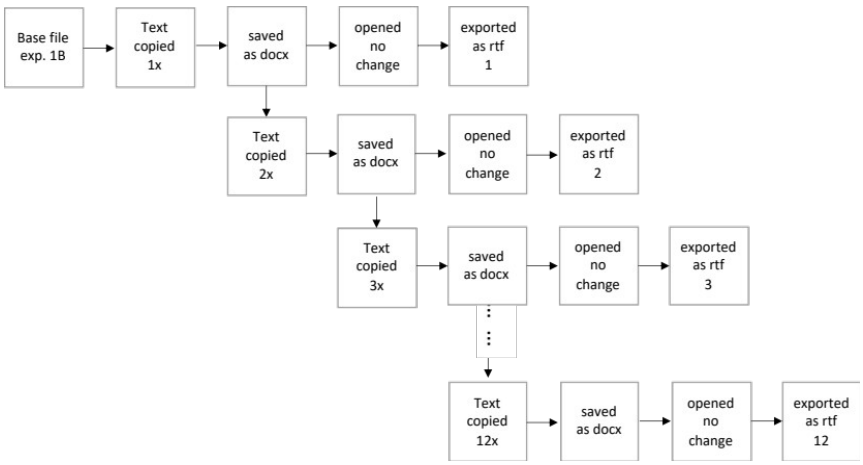


**Figure 5.** Workflow for the creation of files used in experiment 7A and 7B.

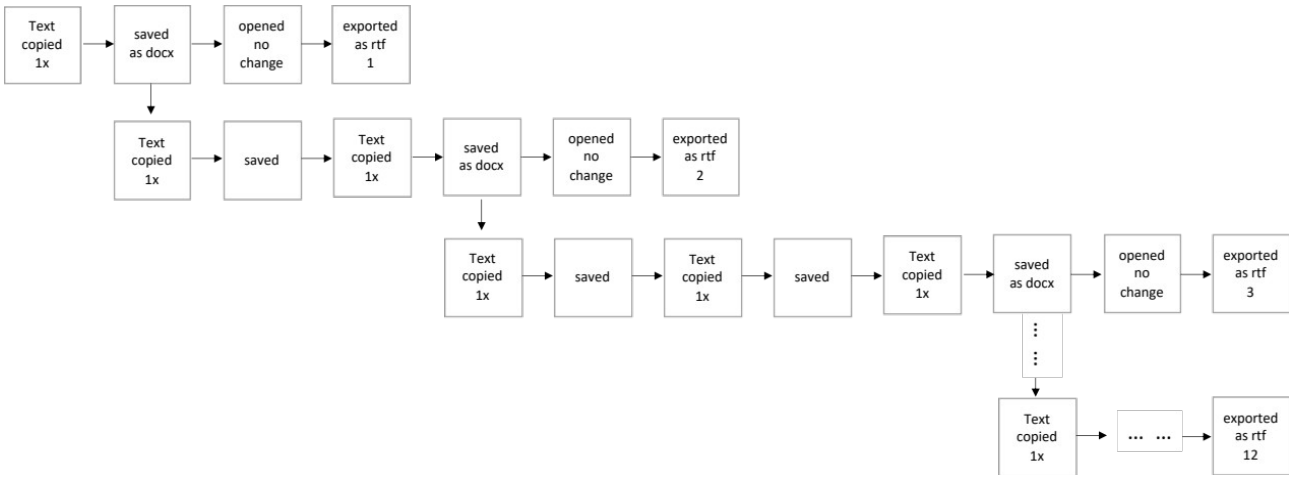


**Figure 6.** Workflow for the creation of files used in experiment 7C and 7D.

## Experiment 8

Experiment 8 was designed to ascertain whether the use of the AutoSave function in MS Word, a function with a user-modifiable default setting of time intervals, could cause an open unmodified file to be saved without user intervention and thereby generate a new Rsid. Two tests were carried out: the first test was on the MacBook, using MS Word 14.7.7, leaving file with a single unsaved edit open for three times the autosave period. The second test was on a Windows machine with MS Word 2009 build 13127.21624 and autosave to MS OneDrive activated.

## Experiment 9

Experiment 9 was designed to assess whether crashing MS Word with auto recovery would result in additional Rsid. MS Word on two MacBooks (MS Word versions (14.7.7 and 16.73 [23051401]) was artificially crashed (through 'force quit').

## Experiment 10

Experiment 10 was designed to assess whether an e-mail action would generate a new Rsid as prescribed in the ECMA specification. As base file for this experiment served a copy of the same base file that had been used for experiment 1. The file was opened and, without any edit action e-mailed to one of the authors, using the File > Share > Send Document function of MS Word. The experiment was carried out on a MacBook with MS Word 16.73 (23051401) (experiment 8A), a MacBook with MS Word 14.3.10 (131030) (experiment 10B) and a Windows laptop with MS Word 2208 (15601.20680) (experiment 10C). As a control, the file was added as an attachment to a mail message, without being opened (experiments 10D–10F).

## Supplementary data

The data files showing the alphanumeric equivalents of allocated Rsid numbers, as well as examples of raw data tables, can be found in the supplementary file. These tables are prefixed in the text with an 'S', e.g. Table S4.

## Visualisation

For visualisation of the incremental changes, the Rsid data tables were extracted from the word files (e.g. Tables S1–S3) and then arranged in columns, with the persistent and temporary Rsid

aligned. To better illustrate the pattern of allocating Rsid numbers, the Rsid numbers were replaced by alphanumeric equivalents (e.g. Table 1, Tables S4–S8), which were then recoded based on the sequence of their appearance in each step. The temporary Rsid which were allocated as part of the rtf save and which were not carried to the next step were replaced by an asterisk (Tables S4–S8). Using Experiment 4A as an example, Rsid data extracted from the Rsidtbl looked like this for the base file (0) and the first two iterations (1, 2):

| | |
|---|---|
| 0 | Rsid3034366\Rsid3236923\Rsid5520986\Rsid6498759\Rsid10830063 |
| 1 | Rsid3034366\Rsid5520986\Rsid6498759\Rsid6776360\Rsid10830063\Rsid13371717 |
| 2 | Rsid811884\Rsid3034366\Rsid5520986\Rsid5772199\Rsid6498759\Rsid6776360\Rsid10830063 |

For ease of use, they were transcribed thus:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | | A | B | C | | D | | E | |
| 1 | | A | | C | | D | F | E | G |
| 2 | H | A | | C | I | D | F | E | |

and visualised as follows with the temporary Rsids for the rtf save substituted by asterisks:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | | A | * | C | | D | | E | |
| 1 | | A | | C | | D | F | E | * |
| 2 | H | A | | C | * | D | F | E | |

# Results

The experiments showed that there are two kinds of Rsid numbers, the document Rsid (as stored in both 'Rsidroot and 'Rsidtbl') and a persistent Rsid that are carried from document version to document version. Both root and persistent Rsids are stored in 'Rsidtbl'. All Rsid tables extracted from the word files are consistent in that they list all Rsid numbers in sequence with increasing numerical value (e.g. Tables S1–S3).

## Experiment 1

The mere opening and closing of a file, without a save action did not alter the Rsid (on either Mac or Windows machines), conforming to ECMA standards.

The opening and reading of a file without saving can be detected, however, when perusing the file's meta data (by using 'Get Info' on Macintosh computers, for example, which do not require a file to be opened). That can be side-stepped, however, by creating a file copy at the operating system level and perusing that file.

## Experiment 2

Experiment 2 assessed the allocation of new Rsid numbers in relation to the document Rsid and persistent Rsid, using 50 replicates of the same file, which was opened, saved as rtf and closed. The experiment confirmed that both the initial persistent Rsid and the document Rsid were retained throughout, while a new persistent Rsid allocated during the save action.

The position of the new Rsid allocated was not consistent. In experiment 2A, the numeric Rsid pattern of the base file was *persistent Rsid* | *new Rsid* | *document Rsid*. The numeric value of that Rsid relative to the two pre-existing Rsid was not constant. In 52% of the cases the new Rsid was larger than the other Rsid, and in 18% it was smaller. In 30% of the cases the new Rsid was replaced the previous temporary Rsid (i.e. between persistent and document Rsid).

In experiment 2B (MS Word for mobile), the numeric Rsid pattern of the base file generated by MS Word for mobile devices running Android, was *persistent Rsid* | *document Rsid* | *new Rsid*. Again, the numeric value of that Rsid relative to the two pre-existing Rsid was not constant. In 50% of the cases the new Rsid was larger than the other Rsid, and in 36% it was smaller. In only 14% of the cases the new Rsid was inserted in between.

## Experiment 3

Experiment 3 explored the nature of Rsid allocation when a file was iteratively saved to rtf, but no text was added. The base file contained two Rsid numbers (the root Rsid and one persistent Rsid). Even though no edit action occurred, saving a file from docx format to rtf format generated an additional Rsid. Likewise, opening an rtf format file and saving it to docx format generated an additional Rsid even though no edit action occurred.

## Experiment 4

The observed pattern of Rsid number generation was that a document Rsid ('Rsidroot') was allocated at the time the document was created and then to add one persistent Rsid at each edit and save step. That is, a file when first saved, will have a minimum of two Rsid while the following edit&save actions resulted in one additional Rsid each. MS Word versions for the Mac will start with at three persistent Rsid at the time a new file is generated. At the time the file is next saved, and a further persistent Rsid added (Table 1; Tables S1–S2, S4–S8). While the numbers of Rsid in the listing table increase with each step, the sequential positions of the temporary Rsid are not fixed in the Rsid table, nor is there a pattern of where the persistent Rsid are located in the numerical sequence. This readily illustrated when considering the distribution of Rsid at the fourth iteration (Table 2). All five experiments show different sequences of Rsid provisioning, both in terms of the initial persistent Rsid in relation to the document Rsid and in relation to the persistent Rsids that were added at subsequent steps. This becomes more complex at the final iteration (Table 3)

**Table 1.** Allocation of Rsid numbers with subsequent opening and editing of files. Experiment 4B MacBook MS Word 2011 version 14.3.9. Note that C,E,F, for example, are temporary Rsid (allocated for the rtf save) that are not included in the next iteration.

| Iteration | Rsid (alphanumeric equivalent) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | | | | A | | | B | | C | | | | | | | |
| 1 | | | | A | | D | B | | E | | | | | | | |
| 2 | | | | A | | D | B | | F | | G | | | | | |
| 3 | | | | A | | D | B | | | | G | H | | | | I |
| 4 | | | J | A | | D | B | | | K | G | | | | | I |
| 5 | | L | M | A | | D | B | | | K | G | | | | | I |
| 6 | | L | | A | N | D | B | | | K | G | | O | | | I |
| 7 | P | L | | A | | D | B | | | K | G | Q | O | | | I |
| 8 | P | L | R | A | | D | B | | | K | G | | O | S | T | I |
| 9 | P | L | | A | | D | B | U | V | K | G | | O | S | | I |
| 10 | P | L | | A | W | D | B | U | | K | G | X | O | S | | I |

**Table 2.** Experiment 4. Locations of persistent and new Rsid in the numerical sequence. Fourth iteration. R indicates the document Rsid, the asterisk (*) indicates the allocation of the temporary Rsid allocated for the rtf save.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 4A | Mac 14.7.7 | 3 | 1 | 1 | R | 2 | 1 | * | 4 |
| 4B | Mac 14.3.9 | 1 | 2 | R | 3 | * | 4 | | |
| 4C | Windows 15.0 | R | 1 | 2 | 4 | * | 3 | | |
| 4D | Mac 16.38 | 1 | 4 | 2 | R | * | 1 | 1 | 3 |
| 4E | Windows 16.0 13127 | 4 | 2 | * | 1 | 3 | R | | |

**Table 3.** Allocation of Rsid numbers with subsequent opening and editing of files. Experiment 4. Locations of persistent Rsid allocated for the rtf save (*) in the numerical sequence. Final iteration. R indicates the document Rsid.

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4A | Mac 14.7.7 | 6 | 3 | 10 | 1 | * | 1 | * | R | 2 | 7 | | 8 | 1 | | 9 | 5 | 4 | |
| 4B | Mac 14.3.9 | 8 | 6 | | 1 | * | 2 | R | 10 | | 5 | 3 | * | 7 | 9 | | 4 | | |
| 4C | Windows 15.0 | * | 5 | R | | 10 | 6 | | 9 | 8 | | 1 | 2 | 4 | | 7 | * | 3 | |
| 4D | Mac 16.38 | | 1 | 6 | * | 10 | 5 | 4 | 2 | R | | 1 | 7 | | 8 | 9 | 1 | * | 3 |
| 4E | Windows 16.0 13127 | * | 10 | * | 6 | 4 | 8 | 2 | | 1 | 3 | R | 5 | 7 | | 9 | | | |

## Experiment 5

Experiment 5 was set up in a similar fashion to experiment 4, but the file was saved straight to rtf, omitting the interim step to docx. While numbers in the listing table increase with each step, a new Rsid for the rtf save step was not generated. As with experiment 5, the sequential positions of the new Rsid are not fixed in the Rsid table, nor is there a pattern of where the persistent Rsid are placed in the numerical sequence (Table 4, Table 5).

**Table 4.** Allocation of Rsid numbers with subsequent opening and editing of files. Experiment 5A. Locations of persistent and temporary Rsid in the numerical sequence. Fifth iteration. R indicates the document Rsid.

| Replicate | Rsid sequence | | | | | |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | R | 5 | 4 | 1 |
| 2 | 3 | 4 | 5 | 2 | R | 1 |
| 3 | 5 | 2 | 3 | 4 | 1 | R |
| 4 | 3 | R | 2 | 4 | 5 | 1 |
| 5 | 4 | 5 | 2 | 3 | R | 1 |
| 6 | 3 | 2 | 5 | R | 4 | 1 |
| 7 | 2 | 4 | 5 | R | 1 | 3 |
| 8 | 5 | 2 | 3 | 4 | R | 1 |
| 9 | 2 | 1 | | 4 | 3 | 1 |
| 10 | 4 | 2 | R | 4 | 1 | 3 |

**Table 5.** Experiment 5B. Locations of persistent and temporary Rsid in the numerical sequence. Final iteration. R indicates the document Rsid.

| Replicate | Rsid sequence | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 6 | 8 | 3 | 7 | 9 | R | 5 | 4 | 1 |
| 2 | 3 | 9 | 7 | 4 | 5 | 2 | 6 | 8 | R | 1 |
| 3 | 5 | 6 | 7 | 2 | 3 | 4 | 8 | 1 | 9 | R |
| 4 | 3 | R | 8 | 2 | 6 | 4 | 5 | 1 | 9 | 7 |
| 5 | 4 | 5 | 2 | 6 | 8 | 7 | 9 | 3 | R | 1 |

## Experiment 6

Experiment 6 was designed to examine the effects of opening and saving of MS Word (to rtf), as well as movement of files between computers. Given that the same source document was used for all variations, the document Rsid is the same for all (coded 'A' in Table 6 and Table 7, raw data Table S12–S13). Opening and closing of two copies of the file in the same session of Word Mac 14.3.9 created two sets of files, where the temporary Rsid differed in numerical value, but remained in the same position (Table 6, 6A, B). When Word was restarted, the temporary Rsid shifted in six of the ten replicates (Table 6, 6C). A set of files copied to a different machine with Word Mac 16.38, resulted in extensive changes to the positions of the Rsid. When opening and closing of two copies of the file in the same session, the positions changed in seven of the ten replicates, while a restart of MS Word resulted in changes to positions in six of the ten replicates.

Of interest, however, is replicate 3, where a persistent Rsid drops out after restart in Word Mac 14.3.9 (Table 6, 6C), as well as in two occurrences when opened in Word Mac 16.38 (Table 6, 6D, F).

**Table 6.** Allocation of Rsid numbers with files moved between computers. Experiment 6A to 6F. Files reopened, saved as RTF. The Rsid codes have been replaced by letters. The Rsid for the rtf save has been replaced by a '*'.

| | Computer A | | | | | | | | | Computer B | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | same session | | | | | | restarted | | | same session | | | | | | restarted | | |
| Replicate | 6A | | | 6B | | | 6C | | | 6D | | | 6E | | | 6F | | |
| 1 | A | * | B | A | * | B | A | * | B | A | * | B | A | * | B | * | A | B |
| 2 | A | * | C | A | * | C | A | * | C | A | * | C | A | C | * | A | * | C |
| 3 | A | * | D | A | * | D | A | * | * | A | * | * | A | D | * | * | A | * |
| 4 | A | E | * | A | E | * | A | E | * | A | E | * | A | E | * | A | E | * |
| 5 | * | F | A | * | F | A | F | A | * | * | F | A | F | A | * | F | A | * |
| 6 | A | * | G | A | * | G | * | A | G | A | G | * | * | A | G | A | * | G |
| 7 | A | H | * | A | H | * | A | H | * | A | H | * | * | A | H | A | * | H |
| 8 | A | * | I | A | * | I | A | I | * | A | I | * | A | * | I | A | * | I |
| 9 | * | A | J | * | A | J | * | A | J | A | * | J | A | * | J | A | * | J |
| 10 | A | * | K | A | * | K | A | K | * | A | * | K | A | K | * | A | * | K |

**Table 7.** Allocation of Rsid numbers with files moved between computers. Experiment 6F to 56H Files reopened, text added, saved as docx, reopened and resaved as rtf. The Rsid codes have been replaced by letters. The Rsid for the rtf save has been replaced by a *. The new persistent Rsid in replicate 3 is indicated by 'X'.

| Replicate | 6F | | | 6G | | | | 6H | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | * | A | B | A | * | * | B | * | A | * | * | B |
| 2 | A | * | C | * | A | C | * | A | * | C | * | * |
| 3 | * | A | X | A | X | * | * | A | * | X | * | * |
| 4 | A | E | * | A | E | * | * | A | E | * | * | * |
| 5 | F | A | * | F | A | * | * | * | * | F | A | * |
| 6 | A | * | G | * | A | * | G | * | * | A | * | G |
| 7 | A | * | H | A | * | * | H | A | * | * | * | * |
| 8 | A | * | I | A | I | * | * | A | I | * | * | * |
| 9 | A | * | J | A | J | * | * | A | * | * | J | * |
| 10 | A | * | K | A | * | K | * | A | * | * | K | * |

A variation used copies of the rtf files generated as experiment 6F, added text, saved them to docx and from there to rtf (experiment 6G) with that file being opened and treated the same fashion. The development of Rsid as tracked in Table 7 shows that while the number of Rsid increases with each saving step, the relative position of the added temporary Rsid in relation to the document Rsid and the persistent Rsid is not fixed.

## Experiment 7

Experiment 7 had been designed to examine the effects of frequent saving during an editing process but without closing the file. The data show that the number of Rsid increases by one every time the file is saved and closed at the same time (Table 8, 7A–7B), irrespective of the number of paste actions performed in each session. Where a paste action was followed by a save action, but without closing the file, the number of Rsid increases by one for each save action (Table 8, 7C–7D).

**Table 8.** Effects of the frequency of saving on the development of Rsid.

| | Experiment 7A Mac 16.38 | | | | Experiment7 Windows 16.0 13127 | | | | Experiment 7C Mac 16.38 | | | | Experiment 7D Windows 16.0 13127 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| File | Actions | Saves/Session | N° of Rsid | increment | Actions | Saves/Session | N° of Rsid | increment | Actions | Saves/Session | N° of Rsid | increment | Actions | Saves/Session | N° of Rsid | increment |
| 0 | 0 | 0 | 5 | — | 0 | 0 | 3 | — | 0 | 1 | 5 | — | 0 | 1 | 3 | — |
| 1 | 1 | 1 | 6 | 1 | 1 | 1 | 3 | 1 | 1 | 1 | 7 | 2 | 1 | 1 | 5 | 2 |
| 2 | 2 | 1 | 7 | 1 | 2 | 1 | 4 | 1 | 2 | 2 | 10 | 3 | 2 | 2 | 8 | 3 |
| 3 | 3 | 1 | 8 | 1 | 3 | 1 | 5 | 1 | 3 | 3 | 14 | 4 | 3 | 3 | 12 | 4 |
| 4 | 4 | 1 | 9 | 1 | 4 | 1 | 6 | 1 | 4 | 4 | 19 | 5 | 4 | 4 | 17 | 5 |
| 5 | 5 | 1 | 10 | 1 | 5 | 1 | 7 | 1 | 5 | 5 | 25 | 6 | 5 | 5 | 22 | 5 |
| 6 | 6 | 1 | 11 | 1 | 6 | 1 | 8 | 1 | 6 | 6 | 32 | 7 | 6 | 6 | 29 | 7 |
| 7 | 7 | 1 | 12 | 1 | 7 | 1 | 9 | 1 | 7 | 7 | 40 | 8 | 7 | 7 | 37 | 8 |
| 8 | 8 | 1 | 13 | 1 | 8 | 1 | 10 | 1 | 8 | 8 | 49 | 9 | 8 | 8 | 46 | 9 |
| 9 | 9 | 1 | 14 | 1 | 9 | 1 | 11 | 1 | 9 | 9 | 59 | 10 | 9 | 9 | 57 | 11 |
| 10 | 10 | 1 | 15 | 1 | 10 | 1 | 12 | 1 | 10 | 10 | 70 | 11 | 10 | 10 | 68 | 11 |
| 11 | 11 | 1 | 16 | 1 | 11 | 1 | 13 | 1 | 11 | 11 | 82 | 12 | 11 | 11 | 80 | 12 |
| 12 | 12 | 1 | 17 | 1 | 12 | 1 | 14 | 1 | 12 | 12 | 95 | 13 | 12 | 12 | 93 | 13 |

## Experiment 8

The Auto-Save test on the MacBook (MS Word 14.7.7) leaving a file with a single unsaved edit open for three times the autosave period did not result in a write to disk at the file's location. The test on the Windows machine with autosave to MS OneDrive activated, however, resulted in the allocation of additional Rsid.

## Experiment 9

Artificially crashing MS Word (through force quit) resulted in recoverable files, but these did not contain additional Rsid numbers until they were saved again after recovery. This can be explained by the fact that no edit action had occurred.

### Experiment 10

Experiment 10 had been designed to assess whether an e-mail action would generate a new Rsid (as prescribed in the ECMA specification). In the case of experiments 10A and 10B, the mail action added a Rsid number to each of the files, even though no change occurred. In the case of experiment 10C two Rsid numbers were added to the file.

The latter appears to be a function as to how MS Outlook handled the file in that instance. The mail action for experiments 10A and 10B opened the Apple Mail application and added the file as an attachment. The mail action for experiment 10C, however, added the file first into the file share of the user's mail institutional system, and then provided the options of sharing via OneDrive or via mail copy. It would appear, that this represents two discrete save actions and thus results in two Rsid.

Sending a closed file as an e-mail attachment resulted in no changes to the Rsid (experiments 10D–10F).

### The role of MS Word template files

An examination of a *new* file generated on one of the authors' (DHRS) MacBooks (MS Word 16.54 build 21101001) found that the Rsid table contained an astonishing 4,732 entries. When looking through backup files of documents that had been generated over the use-span of that laptop, it was noted that the number Rsid gradually, but not incrementally increased. Rather, over time there were some major jumps in overall numbers. As the files used for the analysis were drawn from the first daily backup for each newly generated document, the Rsid numbers closely reflect the blank file status for that day. While at first sight this addition of numerous Rsid numbers in the blank file appears non-sensical, it makes sense when considering that the template files (normal.dotx & normal.dotm) had become corrupted at various time. Rather than automatically generating a clean-slate file, it was regenerated an existing file (with all text deleted) as this allowed to preserve all embedded stylesheets and customised document specifications. At this point the Rsid numbers of that document were saved into and preserved within the .dotx /.dotm file.

To test this proposition, MS Word was exited, the template files were deleted and allowed to auto-regenerate upon oping MS Word again. An inspection of the .dotm file revealed three Rsid, one of which was the root Rsid. The originally embedded stylesheets and customised document specifications were then merged from a manuscript file into the template via the Style Manager with the newly saved .dotm file now containing four Rsid numbers. This implies that each new file will from now on have six Rsid upon its first save&exit action, four of which will be the legacy Rsid inherited from the template.

If the embedded style sheet of a given document is amended, but the added style or the changes to an existing style only apply to the current document, then the template file is not affected. If, however, such an additional style or a style alteration are also added to the template file (so that they are available to all new documents), then this will be saved into the template upon exiting MS Word. This results in an additional Rsid, which henceforth will be carried forward as a legacy Rsid in each new document.

## Discussion

At the time of writing, the underlying logic for the allocated values of the Rsid numbers could not be gleaned from the experiments and has also not been publicly documented. According to ECMA and ISO/ IEC 29500-1:2016 specifications, the Rsid should be randomly generated based on the current time . Theoretically, the Rsid should be a) randomly chosen and b) each editing session should have been assigned an Rsid "that is larger than all earlier ones in the same file" . Numerous authors reiterate this and assume it to be a basis of fact . The experiments showed that irrespective of version of MS Word or operating system, each new edit and save action does indeed generate an additional Rsid, but the value of that Rsid can be larger or

smaller than any or all of the pre-existing Rsids (experiments 2, 4–7). While for the most part, persistent Rsid seem to be permanent, replicate 3 of experiment 3 shows that they can indeed drop out of the sequence (Table 6). This has serious implications for the establishment of document genealogies especially where saves between edits are rare.

The allocation of a new Rsid is not necessarily caused by an *edit* event. The experiments showed that a new Rsid can be generated if a file is saved as rtf, although the file was not edited in any way (experiment 3). Likewise, sending a file as an e-mail from *within* MS Word, even it was not edited in any way, generates an additional Rsid (experiment 8). Finally, it is not possible to detect if a person opens a MS Word document, reads it, and closes the file without saving (experiment 1).

The experiments have shown that user behaviour has a direct influence on the number of Rsid represented in a given file. The more frequently a user saves a file, the more persistent Rsid will be generated and stored in the Rsidtbl.

The MS Word template files (.dotx /.dotm) on a given machine contain document (root) Rsid numbers that are generated when a newly installed application is launched for the first time. These will be embedded as legacy Rsid into every new file generated from that template file. As these numbers are autogenerated upon for initialisation of MS Word, each copy of MS Word has, at least in theory, a set of three Rsid that are unique to that specific copy of the application and essentially act as signatures. Thus, it will be possible to trace back a word document to a specific copy of MS Word as long as that machine can be accessed. Template files can be deleted, however, either out of necessity as they have become corrupted or in order to delete trace evidence. When the template file regenerates, it will have a new set of Rsid numbers, thereby breaking the chain of evidence.

Finally, we would like to briefly return to the observation that Microsoft does not implement the ECMA specification that the Rsid for each save&exit event should be larger than that for the previous event. Both ECMA and the Microsoft documentation specify a Rsid to be a four-digit hexadecimal number . When examining the Rsid numbers in a large number of documents, the observable pattern was that all numbers were indeed encoded as four-digit hexadecimal numbers. The use of four-digit hexadecimal numbers, however, allows for $16^8$ (4,29 billion) numbers without duplication. As every MS Word file has a root Rsid and at least one other Rsid, the starting combinations are $16^8 \times 16^8 - 1$. Thus the chance that two MS word files have the same Rsid combination is $1 : 1.84467^{19}$. Every additional step increases this by a factor of $16^8$-*n*, where *n* is the number of edit&save events. The assertion by Microsoft that the "application would quickly run out of allowable Rsids" does not hold true if we ignore the initial assignation of the root Rsid, which is clearly identified as such in the rootRSID field. If for example, an Rsid for each additional save&exit event were to be allocated as a random number drawn from sequential block of 100,000 numbers, then there would be $4.29497^{14}$ possible combinations for the initial file and first save&exit event, with $4.29497^{19}$ possible combinations for the second save&exit event, whereby each subsequent save&exit event increases by exponent 5. Given that there are 42,949 save&exit events before the sequential block of 100,000 numbers are exhausted, Microsoft's assertion is disingenuous.

# Conclusions

The process of allocating Rsid numbers as implemented by MS Word does not fully follow the ECMA and ISO/ IEC standards. Rather than adding Rsid with continually increasing numerical value for each edit & save action, MS Word allocated Rsid seemingly randomly (at least without a detectable pattern).

The MS Word template files contain Rsid numbers, will be embedded as legacy Rsid into every new file generated from that template file. This can allow to trace back a given document to a specific copy of MS Word. While successive saved states of a document have an increasing quantity of Rsid numbers, the allocated numbers *per se* do not allow determination of the sequence of all edits. Consequently, while Rsid numbers carry vital information on the creation

of a MS Word document, they cannot be used to develop an absolute chronological sequence of events, unless several versions or states of a document are in hand.

Opportunities for future research into Rsid allocation rest in the investigation of 'born digital' content. This includes looking at similarities between different versions of content; investigating the detectability of document cloning with subsequent alteration of content; and examining the ability of post-hoc manipulation of Rsid numbers for the purposes of steganography. Another line of fruitful enquiry might look at the persistence of Rsid allocated to aspects of document edit (e.g. pasting, deletions etc) followed by additional editorial manipulations and subsequent edit & save actions. Some of these will be explored by the authors in future papers.

## Authorship contributions

Conceptualization: DHRS; Methodology: DHRS; Investigation: DHRS and CS; Formal analysis: DHRS and CS; Writing - Original Draft: DHRS; Writing - Review & Editing: DHRS and CS; Visualization: DHRS.

## Disclosure statement

The authors report there are no competing interests to declare.

## References

Al-Baghdady, A. Y. (2009). Steganographic Method for Data Hiding in Microsoft Word Documents Structure by a Change Tracking Technique. (Master of Science in Computer Science). University of Technology, Baghdad.

Al-Sharif, Z. A., Bagci, H., Zaitoun, T. A., & Asad, A. (2018). Towards the memory forensics of ms word documents. In Information Technology-New Generations (pp. 179-185): Springer.

Castiglione, A., D'Alessio, B., De Santis, A., & Palmieri, F. (2017). New steganographic techniques for the OOXML file format. Paper presented at the International Conference on Availability, Reliability, and Security.

Castiglione, A., De Santis, A., & Soriente, C. (2007). Taking advantages of a disadvantage: Digital forensics and steganography using document metadata. Journal of Systems and Software, 80(5), 750-764.

Crasson, A., Lebrave, J.-L., & Pedrazzi, J. (2019). Le «siliscrit» de Jacques Derrida. Exploration d'une archive nativement numérique. Genesis. Manuscrits–Recherche–Invention, 49, 7-12. doi: https://doi.org/10.4000/genesis.4316

Dasare, A., & Dhore, M. (2015). Secured Approach for Hiding Data in MS Word Document Using MCDFF. Paper presented at the 2015 International Conference on Computing Communication Control and Automation.

Didriksen, E. (2014). Forensic Analysis of OOXML Documents. (Master of Science in Information Security). Gjøvik University College, Gjøvik.

Durães, B. G. d. S. (2010). Data flows of classified documents. (Mestrado em Segurança Informática). UniveRsidade de Lisboa, Lisbon.

ECMA. (2016). Office Open XML File Formats — Fundamentals and Markup Language Reference [ECMA-376-1:2016]. 5th. Retrieved from https://ecma-international.org/publications-and-standards/standards/ecma-376/

Fu, Z., & Sun, X. (2015). 基于冗余属性的OOX 文本数字水印算法 [Digital watermarking algorithm based on redundant attributes for OOX documents] [in Chinese]. Nanjing Xinxi Gongcheng Daxue Xuebao, 7(1), 40-45.

Fu, Z., Sun, X., Liu, Y., & Li, B. (2011). Forensic investigation of OOXML format documents. Digital Investigation, 8(1), 48-55.

Fu, Z., Sun, X., Liu, Y., & Li, B. (2012). Text split-based steganography in OOXML format documents for covert communication. Security and Communication Networks, 5(9), 957-968.

Fu, Z., Sun, X., & Xi, J. (2015). Digital forensics of Microsoft Office 2007–2013 documents to prevent covert communication. Journal of Communications and Networks, 17(5), 525-533.

Fu, Z., Sun, X., Zhang, J., & Li, B. (2011). A novel watermark embedding and detection scheme based on zero-knowledge proof. International Journal of Digital Content Technology and Its Applications, 5(3), 273-286.

Garfinkel, S. L., & Migletz, J. J. (2009). New xml-based files implications for forensics. IEEE Security & Privacy, 7(2), 38-44.

Hong, K., Cho, J., Kim, S., & Kim, J. (2017). OOXML 문서에 대한 향상된 데이터 은닉 및 탐지 방법 [Improved Data Concealing and Detecting Methods for OOXML Document] [in Korean]. [Improved Data Concealing and Detecting Methods for OOXML Document]. Journal of the Korea Institute of Information Security & Cryptology, 27(3), 489-499. doi: https://doi.org/10.13089/JKIISC.2017.27.3.489

ISO/IEC. (2008). Information technology — Document description and processing languages — Office Open XML File Formats — Part 1: Fundamentals and Markup Language Reference. In (Vol. ISO/IEC 29500-1:2008(en), pp. 5560): International Organization for Standardization / International Electrotechnical Commission.

ISO/IEC. (2016). Information technology — Document description and processing languages — Office Open XML File Formats — Part 1: Fundamentals and Markup Language Reference. In (Vol. ISO/IEC 29500-1:2016(en), pp. 5024): International Organization for Standardization / International Electrotechnical Commission.

Jeong, D., & Lee, S. (2017). Study on the tracking revision history of MS Word files for forensic investigation. Digital Investigation, 23, 3-10.

Johnson, C. S., & Davies, R. (2020). Using digital forensic techniques to identify contract cheating: A case study. Journal of Academic Ethics, 18(2), 105-113.

Joun, J., Chung, H., Park, J., & Lee, S. (2021). Relevance analysis using revision identifier in MS word. Journal of Forensic Sciences, 66(1), 323-335. Retrieved from: https://onlinelibrary.wiley.com/doi/epdf/10.1111/1556-4029.14584

Joun, J., Han, J., Jung, D., & Lee, S. (2018). Study on History Tracking Technique of the Document File through RSID Analysis in MS Word [in Korean]. Journal of the Korea Institute of Information Security & Cryptology, 28(6), 1439-1448.

Kaczyński, K. (2012). Stego.docx : hidden communication system using docx files. Biuletyn Wojskowej Akademii Technicznej, 61(4), 325-342.

Langweg, H. (2012). OOXML file analysis of the July 22nd terrorist manual. Paper presented at the IFIP International Conference on Communications and Multimedia Security.

Lebrave, J.-L. (2011). Computer forensics: la critique génétique et l'écriture numérique. Genesis. Manuscrits–Recherche–Invention, 33, 137-147. doi: https://doi.org/10.4000/genesis.633

Lee, H., & Lee, H.-W. (2019). Forgery Detection Mechanism with Abnormal Structure Analysis on Office Open XML based MS-Word File. International journal of advanced smart convergence, 8(4), 47-57.

Lee, H., & Lee, H.-W. (2020). Hidden Message Detection in MS-Word File by Analyzing Abnormal File Structure. Paper presented at the 2020 International Conference on Green and Human Information Technology (ICGHIT).

Liang, O. W., & Iranmanesh, V. (2016). Information hiding using whitespace technique in Microsoft word. Paper presented at the 2016 22nd International Conference on Virtual System & Multimedia (VSMM).

Lin, I.-C., & Hsu, P.-K. (2010). A data hiding scheme on word documents using multiple-base notation system. Paper presented at the 2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing.

Liu, T.-Y., & Tsai, W.-H. (2007). A new steganographic method for data hiding in microsoft word documents by a change tracking technique. IEEE Transactions on Information Forensics and Security, 2(1), 24-30.

Mahato, S., Khan, D. A., & Yadav, D. K. (2020). A modified approach to data hiding in Microsoft Word documents by change-tracking technique. Journal of King Saud University-Computer and Information Sciences, 32(2), 216-224.

Mahato, S., Yadav, D. K., & Khan, D. A. (2014). A novel approach to text steganography using font size of invisible space characters in microsoft word document. In Intelligent Computing, Networking, and Informatics (pp. 1047-1054): Springer.

Microsoft. (n.d.-a). 2.1.415 Part 1 Section 17.15.1.70, Rsid (Single Session Revision Save ID). Retrieved from https://docs.microsoft.com/en-us/openspecs/office_standards/ms-oi29500/9e7cca9a-56a7-4c99-9e30-7aeedfa4c995

Microsoft. (n.d.-b). Rsid Class. Retrieved from https://learn.microsoft.com/en-us/dotnet/api/documentformat.openxml.wordprocessing.Rsid?view=openxml-2.8.1

Microsoft. (n.d.-c). Rsid Class. Retrieved from https://docs.microsoft.com/en-us/dotnet/api/documentformat.openxml.wordprocessing.Rsids?view=openxml-2.8.1

Park, B., Park, J., & Lee, S. (2009). Data concealment and detection in Microsoft Office 2007 files. Digital Investigation, 5(3-4), 104-114.

Raffay, M. A. (2011). Data hiding and detection in office open XML (OOXML) documents. (Master of Applied Science in Electrical and Computer Engineering). University of Ontario Institute of Technology, Oshawa, Ontario.

Ries, T. (2010). „die geräte klüger als ihre besitzer ": Philologische Durchblicke hinter die Schreibszene des Graphical User Interface. Editio, 24(2010), 149-199.

Ries, T. (2017). Filologia i proces pisania cyfrowego–metody i wyzwania. Wielogłos(31), 103-127.

Ries, T. (2018). The rationale of the born-digital dossier génétique: Digital forensics and the writing process: With examples from the Thomas Kling Archive. Digital Scholarship in the Humanities, 33(2), 391-424.

Sabir, M., Jones, J., & Liu, H. (2017). A Non-Algorithmic File-Type Independent Method for Hiding Persistent Data in Files. Las Vegas, Nevada, USA.

Stojanov, I., Mileva, A., & Stojanovic, I. (2014). A new property coding in text steganography of Microsoft Word documents. Paper presented at the Securware 2014: The Eighth International Conference on Emerging Security Information, Systems and Technologies, 16-20 Nov 2014, Lisbon, Portugal.

Xiang, L., Sun, C., Liao, N., & Wang, W. (2016). A characteristic-preserving steganographic method based on revision identifiers. International Journal of Multimedia and Ubiquitous Engineering, 11(9), 29-38.

Yang, B., Sun, X., Zhang, J., Xiang, L., Chen, X., & Li, X. (2012). A Novel Reversible Text Data Hiding Scheme. Information Technology Journal, 11, 1084-1090.