CA 616 - Forensic Computing, Assignment #2, Hidden data in popular office file formats

Stefan Seufert & Sebastian Wolfgarten

April 20, 2006

Abstract

Proprietary file formats such as Adobe's Portable Document Format (PDF) or Microsoft's Word .doc are the de-facto standards for storing, publishing or exchanging information in home or office environments. This paper will take a closer look at these formats and will discuss some of the issues involved with using them.

Contents

1 Introduction

Adobe's Portable Document Format (PDF) and Microsoft's Word .doc are the de-facto standards for storing, publishing or exchanging information in home or business environments. Although both file formats are proprietary, they are reasonably well documented[1] and are available for more than ten years now. Additionally they are also supported by various alternative word-processing applications such as OpenOffice, AbiWord or KOffice enabling even users of alternative operating systems such as Linux to use these formats. However it is not widely known that especially Microsoft Word[2] tends to attach an enormous amount of so-called metadata to every document file. Metadata[3] is literally "data about data" and refers to information describing the characteristics of a certain piece of data rather than its content. The meta information Microsoft Word tends to add to every document are[4](digest):

- Name of the author of a document
- The author's initials
- Name of the computer used to create/edit a document
- Name of the network server or hard disk a document was saved to
- The names of previous document authors
- Hidden text
- Comments
- Document revisions
- Document versions
- Other file properties and summary information

Although Microsoft freely provides tools and documentation to remove or even minimize[5] the amount of descriptive metadata a document contains, only a few users are aware of this issue. Therefore even today many documents still contain these information as they are being published on the Internet or send from one business partner to another. In 2003 the issue of Word metadata hit the press[6] big time after a document published by the British government (the so-called "Iraq dossier") and cited by Colin Powell in an address to the United Nations,

[1] Wotsit's Format, "Microsoft Word 8/Word 97 Format - more complete version (HTML FILE)", URL: http://www.wotsit.org/download.asp?f=wword8 and Wotsit's Format, "Portable Document Format Reference Manual Version 1.3 - March 11, 1999 (Acrobat) [Adobe]", URL: http://www.wotsit.org/download.asp?f=pdfspec

[2] Wikipedia, "Microsoft Word", URL: http://en.wikipedia.org/wiki/Microsoft_Word

[3] Wikipedia, "Metadata", URL: http://en.wikipedia.org/wiki/Metadata

[4] Microsoft, "WD97: How to Minimize Metadata in Microsoft Word Documents", URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;223790&sd=tech

[5] Microsoft,"How to minimize metadata in Word 2003", URL: http://support.microsoft.com/kb/825576/EN-US/

[6] New York Times, "THREATS AND RESPONSES: INTELLIGENCE ASSESSMENT; Britain Admits That Much of Its Report on Iraq Came From Magazines", URL: http://query.nytimes.com/gst/abstract.html?res=F20B1EFD395F0C7B8CDDAB0894DB404482

was found to be partially copied from various magazines and academic journals. The plagiarism was identified in February 2003 by Dr. Glen Rangwala, a lecturer of the Cambridge University who published an analysis of the document in a posting to a mailing list[7] shortly after it was released. Dr. Glen Rangwala found out that the document mostly is based on content which was directly copied (without acknowledgement) from an article written by Ibrahim al-Marashi, a postgraduate student at the Monterey Institute of International Studies in Monterey, California. In June 2003 Richard M. Smith published on article on his website[8] in which he examined the metadata included in the document published by the British government. Therewith Smith was able to obtain a revision log of the file which revealed file names, file locations and the usernames of several individuals (including Paul Hamill, the Foreign Office official and Murtaza Khan, the junior press officer for the Prime Minister) that had access to the file and had edited it. In July/August 2004 Dr. Maximillian Dornseif, a German computer security researcher, gave a very enjoyable presentation[9] at the DefCon security conference in Las Vegas/USA in which he described the problems involving metadata and presented a large-scale study on hidden information in documents published on the Internet. Based on his presentation and the Iraq dossier, it is obvious that metadata in Office documents (especially Word files) is definitely a very valuable resource in any forensic investigation. For instance a typical fraud case might involve individuals creating fake invoices in order to make a financial gain. Hence the metadata could be used to identify the persons that opened, edited and saved a document as well as the computers involved. Furthermore metadata will reveal the locations a file was saved to as well as the exact version of Microsoft Word used. Therefore in the next section we would like to present some tools that can be used to extract and examine metadata from Word documents or Microsoft Office files in general.

2 Tools and analysis

In order to extract metadata from a Word document (or in fact any Office file), one can use a variety of both commercial and open source tools. The easiest analysis program to start with is obviously the "strings" command which will display strings of printable characters contained in a file.

2.1 Strings

For the aforementioned "Iraq dossier"[10] (blair.doc) the following strings can be recovered using the "strings" command on a Linux system (actual text was mostly removed):

[7] Dr. Glen Rangwala, "[casi] Intelligence? the British dossier on Iraq's security infrastructure", URL: http://www.casi.org.uk/discuss/2003/msg00457.html

[8] Richard M. Smith, "Microsoft Word bytes Tony Blair in the butt", URL: http://www.computerbytesman.com/privacy/blair.htm

[9] Dr. Maximillian Dornseif, "Far More Than You Ever Wanted To Tell - Hidden Data In Document Formats", URL: http://www.defcon.org/images/defcon-12/dc-12-presentations/Dornseif/dc-12-dornseif.pdf

[10] British Government, "IRAQ - ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION", backup copy available at http://www.computerbytesman.com/privacy/blair.doc

```
$ strings blair.doc
bjbjt+t+
IRAQ
ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
This report draws upon a number of sources, including
intelligence material, and shows how the Iraqi regime is
constructed to have, and to keep, WMD, and is now engaged in
a campaign of obstruction of the United Nations Weapons Inspectors.

[ ... ]

 PAGE
The Special
Republican Guard
Saddam
s Martyrs
Project 858
The Tribal
Chief
s Bureau
The Directorate
General Intelligence
The Directorate of
General Security

[ ... ]

Iraq- ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
default
Normal.dot
MKhan
Microsoft Word 8.0
default
Iraq- ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
Title
_PID_GUID
Microsoft Word Document
MSWordDoc
Word.Document.8
```

Based on the output above (and some information about the people working for the British government), one can identify the username "mkhan" which (as mentioned before) refers to Murtaza Khan, the junior press officer for the Prime Minister. Furthermore one can identify Microsoft Word 8.0 (also known as Word 97) as the version of Word used to create, edit or save the document. Unfortunately the "strings" command on Linux will by default only return a subset of the actual strings embedded in the file as it is not capable of processing the Unicode characters in the Word document. Therefore one has to use "tr" to translate and delete the null characters from the document prior to extracting its strings. Alternatively one can use the Windows variant of "strings" provided

by Mark Russinovich of sysinternals.com[11] which is capable of handling Unicode characters. However on Linux, one must use the following command to retrieve all strings embedded in the document:

```
$ tr -d \\0 < blair.doc | strings
```

Therewith one can recover even more metadata from the file:

```
$ tr -d \\0 < blair.doc | strings
bjbjt+t+
-$t/
.o#x
o#o#o#
o#{'
-$IRAQ
ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
This report draws upon a number of sources, including intelligence
material, and shows how the Iraqi regime is constructed to have, and
to keep, WMD, and is now engaged in a campaign of obstruction of the
United Nations Weapons Inspectors.
Part One focusses on how Iraq

[ ... ]

Normal
8 Heading 1
<Heading 2
<Heading 3
<Heading 4
<Heading 5
@ Heading 6
@ Heading 7
CJ OJ
@ Heading 8
CJ$OJ
@ Heading 9
Default Paragraph Font@C
Body Text Indent
2 Body Text
Header
Footer
Body Text 2
( Hyperlink
Blockquote
Body Text 3

[ ... ]

cic22JC:\DOCUME~1\phamill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd
```

[11] Marc Russinovich, "Strings", URL: http://www.sysinternals.com/utilities/strings.html

```
cic22JC:\DOCUME~1\phamill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd
cic22JC:\DOCUME~1\phamill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd
JPratt
C:\TEMP\Iraq - security.doc
JPratt
A:\Iraq - security.doc
ablackshaw!C:\ABlackshaw\Iraq - security.doc
ablackshaw#C:\ABlackshaw\A;Iraq - security.doc
ablackshaw
A:\Iraq - security.doc
MKhan
C:\TEMP\Iraq - security.doc
MKhan(C:\WINNT\Profiles\mkhan\Desktop\Iraq.doc

[ ... ]

Times New Roman5
SymbolG&
ArialHelveticaA&
Arial Narrow?&
Arial Black"
CIraq- ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
default
MKhan
DIraq- ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
default
Normal.dotN
MKhan.d
Microsoft Word 8.0C@
default
DIraq- ITS INFRASTRUCTURE OF CONCEALMENT, DECEPTION AND INTIMIDATION
Title
_PID_GUID
AN{5E2C2E6C-8A16-46F3-8843-7F739FA12901}

[ ... ]

O02WordDocument
SummaryInformation(
DocumentSummaryInformation8
CompObj
jObjectPool
Microsoft Word Document
MSWordDoc
Word.Document.8
```

With this command, one can obtain extremely interesting metadata including more usernames (e.g. ablackshaw, jpratt) of individuals involved in the creation of the document as well as locations the file was saved to (e.g. `C:\WINNT\Profiles\mkhan\Desktop\Iraq.doc`). As mentioned before, the ver-

sion of Microsoft Word used to produce the document is leaked (e.g. Word.Document.8 refers to Word 97[12]). More importantly this command reveals the so-called "Globally Unique Identifier" (GUID)[13] of the computer that the document was created on. The GUID[14] is a 128-bit pseudo-random number (in hexadecimal, e.g. 5E2C2E6C-8A16-46F3-8843-7F739FA12901) which is used in Microsoft's Component Object Model (COM) to uniquely identify various objects (e.g. files, components etc.) and which gets automatically added to Microsoft Office documents. It used to be based on a computer's network card MAC address and can in a forensic analysis ideally be used to track back the origin of a document. In 1999 this "feature" helped identifying the author of the famous Melissa macro virus/worm[15]. For the document of the British government this means an investigator would be able to exactly determine the computer that was used to initially create the document within the organisation. In Microsoft Excel files it is even common to find the name and the (network) address of the printer used to print the document.

2.2 Hex editors

The information one can reveal from an Office document using the "strings" command is pretty comprehensive, especially when handling Unicode encoding correctly (see above). Therefore a hex editor such as "Hex Workshop[16]" (shareware) or "XVI32[17]" (freeware) will reveal about the same or very similar information than the "strings" command. However in some situations, a hex editor could be used to bypass Word's form protection[18]. For entirely encrypted documents (Word's encryption is relatively weak), one still has to use special "recovery" software[19] to gain access to such a document.

2.3 Antiword

Antiword[20] is an open-source and free reader for Microsoft Word files. It is available for a variety of platforms including (but not limited to) Linux, RISC OS, FreeBSD, Mac OS X, Amiga, NetWare as well as Windows/DOS and is able to convert files from Word 2, 6, 7, 97, 2000, 2002 and 2003 to plain text (or PostScript/PDF).

2.3.1 Installing Antiword

Compiling and installing Antiword on a Linux system is relatively easy and firstly involves downloading the source code of the application from Antiword's

[12] Wikipedia, "Microsoft Word", URL: http://en.wikipedia.org/wiki/Microsoft_Word

[13] junkbusters.com, "Privacy Advisory on Microsoft Hardware IDs", URL: http://www.junkbusters.com/microsoft.html

[14] Wikipedia, "Globally Unique Identifier", URL: http://en.wikipedia.org/wiki/Globally_Unique_Identifier

[15] Wikipedia, "Melissa (computer worm)", URL: http://en.wikipedia.org/wiki/Melissa_worm

[16] BreakPoint Software, "Hex Workshop", URL: http://www.hexworkshop.com

[17] Christian Maas, "XVI32", URL: http://www.chmaas.handshake.de/delphi/freeware/xvi32/xvi32.htm

[18] Securityfocus.com, "Microsoft Word Form Protection Password Removal Weakness", URL: http://www.securityfocus.com/bid/9342/info

[19] Elcomsoft, "Advanced Office Password Recovery", URL: http://www.elcomsoft.com/aopr.html

[20] Adri van Os, "Antiword - a free MS Word document reader", URL: http://www.winfield.demon.nl/

website:

```
$ wget http://www.winfield.demon.nl/linux/antiword-0.37.tar.gz
```

Please note that if a proxy server is required to access the Internet, one can specify the proxy server by typing `export http_proxy="http://host:port"` (e.g. export http_proxy="http://proxy3.dcu.ie:3128") prior to executing wget. Furthermore some distributors (e.g. SuSE, Debian, Ubuntu or Gentoo) provide pre-compiled and/or pre-packed versions of Antiword one can install using the software management application of the given distribution (e.g. YaST for SuSE, dpkg for Debian or emerge for Gentoo). However if building the software from scratch, one needs to extract the archive of the software first:

```
$ tar xvzf antiword-0.37.tar.gz
```

After changing to the newly created directory "antiword-0.37", one must compile the software by issuing the "make" command:

```
$ make
gcc -Wall -pedantic -O2 -DNDEBUG -c main_u.c
gcc -Wall -pedantic -O2 -DNDEBUG -c asc85enc.c
gcc -Wall -pedantic -O2 -DNDEBUG -c blocklist.c
gcc -Wall -pedantic -O2 -DNDEBUG -c chartrans.c
gcc -Wall -pedantic -O2 -DNDEBUG -c datalist.c
gcc -Wall -pedantic -O2 -DNDEBUG -c depot.c
gcc -Wall -pedantic -O2 -DNDEBUG -c dib2eps.c
gcc -Wall -pedantic -O2 -DNDEBUG -c doclist.c

[ ... ]

gcc  main_u.o asc85enc.o blocklist.o chartrans.o
datalist.o depot.o dib2eps.o doclist.o fail.o
finddata.o findtext.o fmt_text.o fontlist.o fonts.o
fonts_u.o hdrftrlist.o imgexam.o imgtrans.o jpeg2eps.o
listlist.o misc.o notes.o options.o out2window.o
output.o pdf.o pictlist.o png2eps.o postscript.o
prop0.o prop2.o prop6.o prop8.o properties.o propmod.o
rowlist.o sectlist.o stylelist.o stylesheet.o summary.o
tabstop.o text.o unix.o utf8.o word2text.o worddos.o
wordlib.o wordmac.o wordole.o wordwin.o xmalloc.o xml.o
-o antiword
```

Please note that in order to be able to compile the source code, one must have appropriate tools (i.e. gcc, make etc.) installed. The last step is the actual installation of the software which can be done by running "make install" (as root):

```
# make install
```

The software is then ready to be used.

2.3.2 Using Antiword

Antiword just consists of a single executeable called "antiword" which when executed without any parameters, simply provides a list of all its parameters[21]:

```
$ antiword
        Name: antiword
        Purpose: Display MS-Word files
        Author: (C) 1998-2005 Adri van Os
        Version: 0.37  (21 Oct 2005)
        Status: GNU General Public License
        Usage: antiword [switches] wordfile1 [wordfile2 ...]
        Switches: [-f|-t|-a papersize|-p papersize|-x dtd][-m mapping][-w #][-i #][-Ls]
                -f formatted text output
                -t text output (default)
                -a <paper size name> Adobe PDF output
                -p <paper size name> PostScript output
                   paper size like: a4, letter or legal
                -x <dtd> XML output
                   like: db (DocBook)
                -m <mapping> character mapping file
                -w <width> in characters of text output
                -i <level> image level (PostScript only)
                -L use landscape mode (PostScript only)
                -r Show removed text
                -s Show hidden (by Word) text
```

The software is not only capable of converting Word documents into plain text or PostScript/PDF format, it also claims to be able to show hidden or even removed text. Our testing with documents created in Word 2002 and 2003 revealed it is only able to show hidden text (i.e. somebody marking a text section in Word and formatting it as Format->Font->Hidden) but at least when using newer versions of Word it is *not* capable of recovering removed text. However according to various sources[22] earlier versions of Microsoft Word (Word 95/97?) tended to include even deleted text within a document. Finally it should be noted that hidden text can also be recovered using the "strings" command.

2.4 Other products

Beside the almighty "strings" command, various hex editors and Antiword, there are only three more programs dedicated to extracting information from Word documents (on non-Windows platforms): "catdoc", "wvWare" and "Laola". Catdoc[23] is very similar to the well-known "cat" command except that it operates on binary Word documents rather than on plain text files. Therefore it concatenates a Word document and prints its content to the standard output (e.g. the display). Here's an example session of "catdoc":

[21] For further documentation, see "man antiword".

[22] Rich Mulligan, "Re: Risks of using Microsoft Word", URL: http://catless.ncl.ac.uk/Risks/17.80.html#subj15

[23] Vitus Wagner, "catdoc & xls2csv", URL: http://www.45.free.net/~vitus/software/catdoc/

```
$ catdoc test_document_word2002.doc
[This was fast-saved 15 times. Some information is lost]
This is a test!

This is a test with tracked changes!

This is a test with tracked changes!
```

Its output is similar to the output of Antiword or "strings". Interestingly it reveals that the given document was fast-saved 15 times while being edited, a piece of information which was picked up by no other program so far. Although this information probably is not of any immediate use, it might possibly be valuable in some cases. However it should be noted that in comparison with "strings" the program partially fails to recover text from the test document as some hidden data is not displayed by "catdoc". Additionally the program does not extract any metadata from the document (beside the fast-save information). The next program is "wvWare"[24] which essentially is a library as well as a set of programs to parse Microsoft Word documents and to convert them into various other formats (e.g. PDF, PS, DVI, HTML etc.). WvWare is available for several platforms including Linux, *BSD, Solaris, AIX and Windows and is also internally used by other applications such as AbiWord and KWord. Although the homepage of wvWare states that all its utilities should be considered deprecated in favor of AbiWord, the software includes some programs (e.g. wvText or wvSummary) that are still very useful from a forensic point of view. WvText is able to extract the contents from a document and therewith is very similar to Antiword. However the program is also capable of combining textual content with the metadata of a document resulting in a brief revision log. Here's an example of wvText extracting both the textual and metadata information from a document and "cat" displaying the result:

```
$ wvText test_document_word2002_tracking_changes.doc test.txt
$ cat test.txt

   This is the 1st test: We just added some text.[1][Author ID1: at Fri
   Apr 14 22:22:00 2006 ]

   This is the 2nd test: We just added some more text.[2][Author ID1: at
   Fri Apr 14 22:24:00 2006 ]

   This is the 3rd test: We just added some more and more text[3][Author
   ID1: at Fri Apr 14 22:24:00 2006 ] (which is hidden in the
   document!)[4][Author ID1: at Fri Apr 14 22:25:00 2006 ].[5][Author
   ID1: at Fri Apr 14 22:24:00 2006 ]

References

   1. file://localhost/tmp/wv22733.html#author1
   2. file://localhost/tmp/wv22733.html#author1
```

[24] Dom Lachowicz, "wvWare, library for converting Word documents", URL: http://wvware.sourceforge.net

```
3. file://localhost/tmp/wv22733.html#author1
4. file://localhost/tmp/wv22733.html#author1
5. file://localhost/tmp/wv22733.html#author1
```

The output of the wvText command is extremely confusing and hinders a solid forensic recontruction of the revision history of a document. Next the recovered timestamps are inaccurate and do not reflect the times a document section was edited. In the example above the software correctly detects the first two additions (marked as [1] and [2]) to the document. For the next three modifications we have added some text to our document ([3]), saved it and then closed the file. Additionally we have then reopened the file, marked the previous modification as invisible ([4]) and then added as well as deleted some more text ([5]). Although this revision log and its reconstruction might sound trivial as we have made it up, especially when dealing with longer or more complex documents, it may be impossible to reconstruct the exact history of a document given the limited and confusing output of wvText. Additionally wvText is only able to create these information if version tracking is enabled in the original document (i.e. Tools->Track Changes). Another mentionable program from the wvWare suite is "wvSummary" which will display a brief summary of the metadata included in a document:

```
$ wvSummary test_document_word2002.doc

The title is This is a test
The subject is
The author is Sebastian Wolfgarten
The keywords are
The comments are
The template was Normal.dot
The last author was Sebastian Wolfgarten
The rev # was 6
The app name was Microsoft Word 10.0
PageCount is 1
WordCount is 14
CharCount is 92
Security is 0
Codepage is 0x4e4 (1252)
```

For some reason "wvSummary" tends to get invalid information for the pagecount and does not report any comments although they are included in the original document. Therefore it should not be fully trusted as the tool tends to report some information wrongly. In case it reports a difference between the author of a document and the last author, this could indicate that one person initially created the document and then passed it on to another person. WvSummary can also be used on other Office files:

```
$ wvSummary test.xls

no title found
no subject found
The author is Andrea Childress
```

```
no keywords found
no comments found
no template found
The last author was Cathy Tourney
no rev no found
The app name was Microsoft Excel
no pagecount
no wordcount
no charcount
Security is 0
Codepage is 0x4e4 (1252)
```

As the output above indicates the amount of metadata it is able to obtain from other Office documents is very limited. In summary the wvWare suite does a reasonable job in recovering metadata from Word documents although it tends to misinterpret and overlook information contained in some documents. Therefore it should be considered deprecated in favor of AbiWord or "strings"). An alternative might be WordLeaker[25]. The last program to mention is "Laola"[26] which "is a collection of documentations and perl programs dealing with binary file formats of Windows program documents". Although it looks quite promising, its development has in fact been ceased for the last six years and is therefore not further discussed in this document.

3 Finding sample data

Finding sample data (i.e. Word documents) is relatively trivial and can be done by using any major search engine (e.g. Google or AlltheWeb). Google offers a variety of advanced search terms[27] that can be used to find information of a specific type (e.g. files of a certain type). The keyword "filetype" for instance enables the user to search for files of a certain type (e.g. .doc, .xls or .pdf) and has be combined with a search term. Therefore in order to find Word documents containing the term "Dublin City University" one would use:

```
filetype:doc Dublin City University
```

Subversive individuals could also search for Word documents containing the word "confidential" stored on servers of those domains ending in .mil or .gov (NOT encouraged!):

```
inurl:mil filetype:doc confidential
inurl:gov filetype:doc confidential
```

Obviously this could also be done for various other file types (e.g. PDF) or search terms. In fact there is even a website[28] dedicated to find "interesting" or sensitive information with Google (try "filetype:doc confidential", "filetype:doc

[25]Madelman, "WordLeaker, extracting info from Word files", URL: http://www.elligre.tk/madelman/madelman/index.php/archivos/2005/02/23/wordleaker-extracting-info-from-word-files/

[26]Martin Schwartz, "LAOLA", URL: http://user.cs.tu-berlin.de/ schwartz/pmh/

[27]Google, "Advanced Google Search Operators", URL: http://www.google.com/help/operators.html

[28]Johnny Long, "Google dorks", URL: http://johnny.ihackstuff.com

secret"). Automated programs (e.g. crawl[29]) or scripts (e.g. in Python[30]) could be used to automate this process.

4 PDF files

4.1 Introduction

The Portable Document Format[31] (PDF) is a file format developed by Adobe in the 1990's which supports storing information in plain text, hyperlinks, images and graphics. It has become an open and well-accepted standard for exchanging information in home and business environments because due to its internal structure, it ensures a PDF file always looks exactly the same way on any given platform. Therefore it is device as well as platform independent with various viewer and creator applications[32] available for all sorts of platforms (e.g. Windows, Unix, MacOSX). Another reason for the huge success of the PDF format is that PDF files usually do not contain as much metadata as Word documents do. Generally speaking, the only metadata available in a PDF file are the title, author, subject, keywords, creator, producer application and the creation date. However unlike Microsoft Word, these information will not be automatically appended to a PDF file and can easily be modified with applications such as Adobe Acrobat (full version, not the reader) or various other tools[33]. Therefore in order to minimize the amount of information leaked from a Word document, one should convert it into PDF format which will almost always[34] remove the metadata included in the original Word document and afterwards verify the metadata with the aforementioned programs.

4.2 Hidden data

Due to the lack of metadata and its platform independence, PDF files are very popular with both corporate and governmental organisations. Especially governmental organisations tend to use PDF files to publish reports or evidence from court cases, meetings and sometimes even military events. Prior to publishing these files, critical pieces of information will usually be blackened to prevent sensitive information from being leaked to a general public. A classic example of such a document is an official report[35] published by the US military in March 2005 in which (among other things) the incidents involving the shooting of the Italian secret agent Nicola Calipari and the wounded Italian journalist Giuliana Sgrena are described. Figure 1 (see last page of this paper) provides a screenshot of the blackened document.

[29]Niels Provos, "crawl - a small and efficient HTTP crawler", URL: http://monkey.org/ provos/crawl/

[30]Philipp Lenssen, "Google Web API with Python", URL: http://blog.outercourt.com/archive/2004_01_09_index.html

[31]Wikipedia, "Portable Document Format", URL: http://en.wikipedia.org/wiki/Pdf

[32]Wikipedia, "List of PDF software", URL: http://en.wikipedia.org/wiki/List_of_PDF_software

[33]Brian High, "Free Graphical PDF Metadata Editors", URL: http://www.accesspdf.com/article.php/20050529160835361/print

[34]Ernest Svenson, "Overstating the threat of metadata in PDF documents", URL: http://www.planetpdf.com/enterprise/article.asp?ContentID=6877&fa

[35]US military, title unknown, URL: http://www.corriere.it/Media/Documenti/Classified.pdf

The blackened text sections are hiding the names and the grades of the US soldiers involved in the incident. Additionally the document contains information on the general situation in Iraq (partially blackened) and mentions some problems with the Voice over IP-technology used. Unfortunately (or luckily?) even the almighty PDF format has its problem when dealing with hidden data enabling an individual to reveal blackened information. In the aforementioned presentation at the DefCon security conference, Dr. Maximillian Dornseif illustrated several ways of copying blackened information from PDF files into new documents and provided some interesting videos demonstrating his findings. His techniques include (please refer to his videos for a proper demonstration):

- Select text, copy & paste
- Select text, copy/export as image
- Removing blackened areas

The first (and by far easiest) method is to simply select a text section containing a blackened passage in any PDF viewer (e.g. Adobe Acrobat or KPDF) and copying & pasting it into a new text file. Depending on the technique used to blacken the text, one can therewith fully recover the hidden text[36]. The second technique is very similar and also involves selecting a text section containing a blackened passage in any PDF viewer. However rather than simply copying & pasting the text into a new file, one must copy/export the selected text as an image in order to be able to fully recover the hidden data[37]. Lastly (depending on the obfuscation technique used) one can simply remove text boxes that are being used to hide information in a PDF file by using appropriate software (e.g. Adobe Illustrator on MacOSX)[38]. Consequently the only way to reliably hide information from a target audience is to simply remove the piece of information from the PDF document in the first place.

5 Summary

Although Microsoft has introduced various tools and documents to remove or (at least) minimize the amount of metadata contained in Word documents, it continues to be a current threat to the privacy of both corporate and home users which is widely underestimated. Beside its content, simple programs such as "strings" or Antiword can extract various information from a document including name(s) of the author(s) of a given document, file name(s) used to save a document, any comments or hidden text contained in a document as well as different revisions. Furthermore by using programming languages such as Perl etc. one can implement a program to automatically extract metadata from Word documents[39]. Therefore in order to minimize the amount of information leaked from a Word document, one should consider following Microsoft's tips on

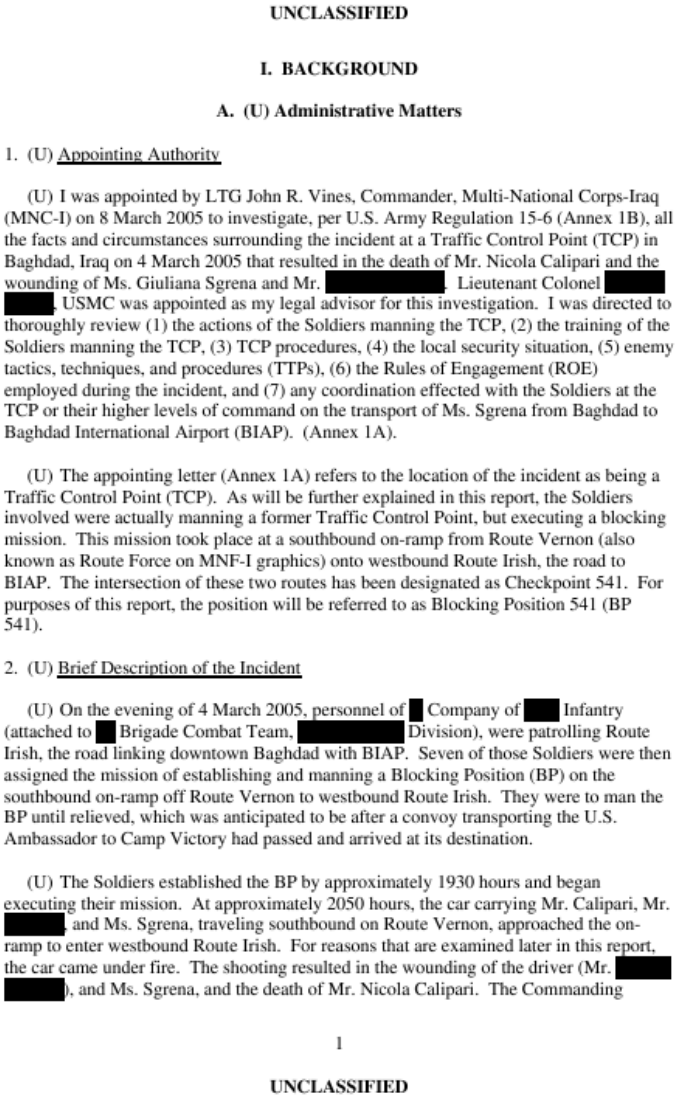
[36] Dr. Maximillian Dornseif, untitled, URL: http://md.hudora.de/presentations/2004-BlackHat/diversity.mov

[37] Dr. Maximillian Dornseif, untitled, URL: http://md.hudora.de/presentations/2004-BlackHat/sniper.mov

[38] Dr. Maximillian Dornseif, untitled, URL: http://md.hudora.de/presentations/2004-BlackHat/Illustrator-Iran.mov

[39] Harlan Carvey, "File::MSWord", URL: http://www.cpan.org/modules/by-authors/id/H/HC/HCARVEY/File-MSWord-0.1.readme

how to remove these metadata or alternatively convert the document into PDF format. Programs for converting a Word document into PDF (on Windows) are for instance Adobe Acrobat (commercial) or FreePDF (freeware). Newer versions of Microsoft Word (e.g. 2003) also provide the user with the ability to automatically remove the metadata from a document (see Tools->Options->Security). Additionally (as mentioned) tools for managing metadata in PDF files are also widely available. When trying to hide information contained in PDF documents, one should consider completely removing these information rather than simply blacken them in order to minimize the risk of accidental information leakage.



UNCLASSIFIED

I. BACKGROUND

A. (U) Administrative Matters

1. (U) Appointing Authority

(U) I was appointed by LTG John R. Vines, Commander, Multi-National Corps-Iraq (MNC-I) on 8 March 2005 to investigate, per U.S. Army Regulation 15-6 (Annex 1B), all the facts and circumstances surrounding the incident at a Traffic Control Point (TCP) in Baghdad, Iraq on 4 March 2005 that resulted in the death of Mr. Nicola Calipari and the wounding of Ms. Giuliana Sgrena and Mr. [redacted]. Lieutenant Colonel [redacted], USMC was appointed as my legal advisor for this investigation. I was directed to thoroughly review (1) the actions of the Soldiers manning the TCP, (2) the training of the Soldiers manning the TCP, (3) TCP procedures, (4) the local security situation, (5) enemy tactics, techniques, and procedures (TTPs), (6) the Rules of Engagement (ROE) employed during the incident, and (7) any coordination effected with the Soldiers at the TCP or their higher levels of command on the transport of Ms. Sgrena from Baghdad to Baghdad International Airport (BIAP). (Annex 1A).

(U) The appointing letter (Annex 1A) refers to the location of the incident as being a Traffic Control Point (TCP). As will be further explained in this report, the Soldiers involved were actually manning a former Traffic Control Point, but executing a blocking mission. This mission took place at a southbound on-ramp from Route Vernon (also known as Route Force on MNF-I graphics) onto westbound Route Irish, the road to BIAP. The intersection of these two routes has been designated as Checkpoint 541. For purposes of this report, the position will be referred to as Blocking Position 541 (BP 541).

2. (U) Brief Description of the Incident

(U) On the evening of 4 March 2005, personnel of [redacted] Company of [redacted] Infantry (attached to [redacted] Brigade Combat Team, [redacted] Division), were patrolling Route Irish, the road linking downtown Baghdad with BIAP. Seven of those Soldiers were then assigned the mission of establishing and manning a Blocking Position (BP) on the southbound on-ramp off Route Vernon to westbound Route Irish. They were to man the BP until relieved, which was anticipated to be after a convoy transporting the U.S. Ambassador to Camp Victory had passed and arrived at its destination.

(U) The Soldiers established the BP by approximately 1930 hours and began executing their mission. At approximately 2050 hours, the car carrying Mr. Calipari, Mr. [redacted], and Ms. Sgrena, traveling southbound on Route Vernon, approached the on-ramp to enter westbound Route Irish. For reasons that are examined later in this report, the car came under fire. The shooting resulted in the wounding of the driver (Mr. [redacted]), and Ms. Sgrena, and the death of Mr. Nicola Calipari. The Commanding

1

UNCLASSIFIED

Figure 1: American report on the shooting of an Italian secret agent in Iraq