HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

COMPUTER SCIENCE

# TPM 2.0 Library

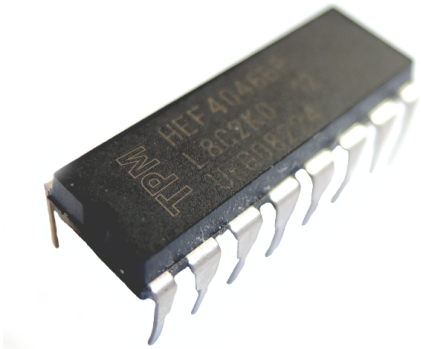
# Studienarbeit

Department of Computer Science
University of Applied Science Rapperswil

Autumn Term 2013


Author(s): Manuel Frei, Qi Zhang
Advisor: Prof. Dr. Andreas Steffen

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz



Abteilung Informatik

Student Research Project - Autumn Term 2013

# Trusted Platform Module 2.0 Library

*Authors:*
Manuel Frei

Qi Zhang

*Supervisor:*
Prof. Dr. Andreas Steffen

December 20, 2013

# Abstract

Platform security is very crucial for protecting data which is stored on our devices we use daily to work. Therefore it is very significant to construct a reliable security, based on The Trusted Platform Module (TPM), on kinds of systems, like Linux and Windows. TPM is a cryptographic microchip designed to enhance security for hardware devices and TPM 2.0 library is the successor of the TPM 1.2 specification released by the Trusted Computing Group. The TPM 2.0 specification is a "library specification", which means that it supports a wide variety of functions, algorithms and capabilities upon which future platform-specific specifications will be based. And TPM 2.0 library includes 4 parts. Part 2 described all the structures and type definitions used by the TPM. Part 3 described series of commands. Part 4 consists only of C code which represented algorithms and methods used by the commands from part 3.

Our project is based on the use of TPM 2.0 library. In this whole project we created a program, via Python, which takes the TPM 2.0 library document at first, and then translated the relevant parts to C code and writes it to the Header files and C files. And we need to use the extracted source files to build a TPM simulator and to compile the whole project. The most important part in our project is to extract the table, and C code in TPM 2.0 library into header and C files, and we have already finished with a very high quality.

It is found that by simulating TPM chip with our program could improved protection for people's private and sensitive information. In future, we need to compile our program in Linux or Windows systems, and to create a test program which can enter the interface of TPM simulator and execute TPM command.

# Definition of Task

### Einführung

Microsoft hat für die Trusted Computing Group die vierteilige Spezifikation der TPM 2.0 Library Funktionen verfasst:

- Part 1: Architecture
- Part 2: Structures
- Part 3: Commands
- Part 4: Supporting Routines

Das API der Trusted Platform Module (TPM) Library ist in Form von C Code definiert, der in die Word-Dokumente eingebettet ist und gleichzeitig auch eine lauffähige Referenz-Implementation der TPM Funktionalität darstellt.

Der Pflichtteil dieser Arbeit besteht darin, den C Code aus der XML Version der Word Dokumente zu extrahieren (Kapitel 4 "Automation" im Part 4 der Spezifikation gibt eine kurze Übersicht was getan werden.

Falls die Zeit dazu reicht, soll anschliessend der extrahierte C Code der TPM 2.0 Library mit dem gcc Compiler unter Linux kompilliert werden. Ein kleines Anwendungsbeispiel soll die Lauffähigkeit der übersetzten Referenzimplementation nachweisen.

### Aufgabenstellung

- Einarbeitung in die TPM 2.0 Library Spezifikation der Trusted Computing Group.
- Wahl einer geeigneten Skriptsprache für die Implementation des Word Document Parsers.

- Extraktion der Typendefinitionen und Datenstrukturen aus Part 2, sowie der Funktionsdefinitionen inklusive C Source Code aus Part 3 und 4.

- Automatisches Abfüllen der C Header und Code Dateien in eine Verzeichnisstruktur.

- Dokumentation des Parsers.

### Optionale Aufgaben

- Übersetzen des extrahierten TPM 2.0 Source Codes unter Linux mittels des gcc Compilers in ein lauffähiges Programm.

- Erstellen eines einfachen Anwendungsbeispiels, um die Lauffähigkeit des TPM 2.0 Emulators nachzuweisen. Dies könnte auch auf der Basis des Visual Studio Codes von Microsoft geschehen.

### Optionale Aufgaben

- Trusted Computing Group, TPM 2.0 Library Specification (Revision 0.99), `http://www.trustedcomputinggroup.org/resources/tpm_library_specification`

Rapperswil, 16 September 2013

Prof. Dr. Andreas Steffen

# Declaration of Authorship

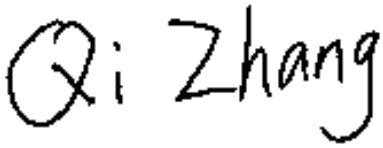
We declare,

- that this thesis and the work presented in it was done by our own and without any assistance, except what was agreed with the supervisor.
- that we mentioned all sources and cited them in the common academic way.
- that we did not use copyright protected materials (ex. images) unauthorizedly in this thesis.

Place, date: Rapperswil, December 18th 2013

Name, Sign:

Manuel Frei



Qi Zhang

# Contents

# 1 Technical Report of the Project

## 1.1 Introduction and Overview

This chapter gives the reader a general knowledge about the project. It clarifies important terms and points out the decisions we made.

### 1.1.1 Basics

To understand what it is all about it is required to explain some terms in this chapter.

#### 1.1.1.1 TPM

The Trusted Platform Module (TPM) is a cryptographic microchip designed to enhance security for hardware devices. TPM is integrated on most of the newer motherboards or can be attached so a system as a separate module. The TPM is an isolated system but it allows the host system to communicate over an interface which is described in the specification. The most common usage of the TPM is to generate and store encryption keys of hard disk drives. This function can be used for example by Microsoft's BitLocker[MsBl] instead to store the keys on insecure external flash drives.

Another purpose a solution called remote attestation to solve the bring-your-own-device (BYOD) problematic. A TPM is able to perform cryptographic measurements of BIOS and files while a system boots and runs. This measurements will be sent to a server in the network which compares the measurements and is able to detect changes on the system made by the user or malicious software so it can deny access to the network for an infected device.

#### TPM 2.0

In the new version of the TPM brings a lot of changes[TPM2ChL]. One of them is an important architecture change to make it easy to adapt the TPM 2.0 for future requirements. New and stronger algorithms can easy be added so it is a lot more flexible than the fixed algorithm set of TPM 1.2.

### 1.1.1.2 Specification

The specification is defined by the Trusted Computing Group (TCG) , a nonprofit organization created for developing industry standards[TCG]. At the time of writing revision 00.99 is open for public review and available for download on the TCG website[1] in form of separated PDF documents. Involved people are able to get the source of this documents in form of Microsoft Office Word documents. To implement a fully standard compliant software all four parts are required.[TLP1] The following paragraphs contain a brief description about the content of each part.

#### Part 1 - Architecture

Part 1 is just an informative document without code or fragments used for the implementation. It clarifies the terms used in the specification, shows the diagrams with a prose description. With flowcharts it is visualized how the TPM will react to different available actions. This part is irrelevant for extracting code but it helps to understand the content.

#### Part 2 - Structures

This part begins with an description how the content have to be translated to C code. All the structures and type definitions used by the TPM are described in this part. The information is stored in tables. With this tables as base the marshal and unmarshal code have to be generated which is used to serialize the TPM commands over the interface.

#### Part 3 - Commands

A working TPM is a series of commands which are described in Part 3. They resort to the structures provided by part 2 and supporting functions by part 4. Commands are used for internal tasks like the initialization and self-tests but also for the communication over the interface. Content to extract is stored in a mixed way. The main code of the command function is available as C code and the information to generate header files are stored in tables.

1 `http://www.trustedcomputinggroup.org/resources/tpm_library_specification`

**Part 4 - Supporting Routines**
The content of this part consists only of C code, there are no table with required information. This C code represents algorithms and methods which are used by the commands which can be extracted from the last part.

### 1.1.1.3 Implementation by Microsoft

Part 2 and 3 are created in a special form that makes it easier to create a parser for it.[TLP4] Microsoft which is part of the TCG created a set of parsers which were used to extract the code and create a project in Microsoft Visual Studio. The resulting project is Microsoft confidential and not open for public use. About the parser it is only known that it is written in Perl[2][TLP4], but never published. The working Visual Studio project is classified as Microsoft Confidential and is available to TGC members and invited experts under a non-disclosure agreement, which is seen by the comment in the heading of each file of the project.

**Listing 1.1:** Copyright note in Visual Studio files.

```
/*(Copyright)

        Microsoft Copyright 2009, 2010, 2011, 2012, 2013
        Confidential Information

*/
```

On a closer look at this files we can get more information about the TCG parsers.

**Listing 1.2:** Used parsers in the Visual Studio implementation.

```
$ grep -R "Automatically Generated" * | awk -F":" '{print $2}' | sort -u
Automatically Generated by DoImplemented.pl
Automatically Generated by ExtractPrototypes version 0.93, Oct 9, 2012
Automatically Generated by Part2AnnexParser
Automatically Generated by Part2Parser.pl
Automatically Generated by Part3Parser version 00.98.00 June 6, 2013
```

Although the name of the implementation indicates that it is revision 00.99, it is obvious that older and different revision were parsed. Towards this information there were five parsers used for two documents, part 2 and 3. About the last part there is no information that a parser was used. Not in a file header neither in the automation information of the documents.

## 1.1.2 Goal

1. Analyze the word documents and write a program that translates the content into code.

2 Perl is a high-level interpreted programming language.

2. Verify if the resulting code is the entire code used for the implementation.

3. Build the simulator out of the code.

4. Create a small program which access the simulator interface and test if it is working like expected.

### 1.1.3 Approach

As source we have the specification either as PDF documents or as Microsoft Word documents. Now we have to choose how we access them.

#### 1.1.3.1 Text File

There are tool to extract plain text from a Word document. One of this tools is managed by the Apache Software Foundation, it is called Apache POI[3]. This library is available for Java, a short example how to use it is available on its web site. The resulting file contains just the plain text like in the following snippet.

**Listing 1.3:** Snippet of extracted text file.

```
5.4 TPMB.h
This file contains extra TPM2B structures
1 #ifndef _TPMB_H
2 #define _TPMB_H
3 #include "TPM_Types.h"
This macro helps avoid having to type in the structure in order to create a new TPM2B type that is used in a function.
4 #define TPM2B_TYPE(name, bytes)             \
5     typedef union {                         \
6         struct  {                           \
7             UINT16  size;                   \
8             BYTE    buffer[(bytes)];         \
9         } t;                                \
10         TPM2B   b;                          \
11     } TPM2B_##name
Macro to instance and initialize a TPM2B value
12 #define TPM2B_INIT(TYPE, name)  \
13     TPM2B_##TYPE    name = {sizeof(name.t.buffer), {0}}
A 2B structure for a seed
14 TPM2B_TYPE(SEED, PRIMARY_SEED_SIZE);
```

It is hard to extract the code from a file like this.

- Line starts with an ordinary integer ⇒ Code
- Line starts with a floating-point number ⇒ Header (depending on document: file name)

3 http://poi.apache.org/text-extraction.html

- Line starts with alphabetic letters ⇒ comment or description

If the document get more complex, for example if it contains tables and some of the cells are just notes, it is not possible to extract the correct information. So this is not the way to go.

### 1.1.3.2 XML

If a Word document is opened by an appropriate program like LibreOffice[4] it is possible to export it to an XML file. The structure of this file is like this snipped:

**Listing 1.4:** Snippet of exported XML file.

```
<txt ptr="0x7f6d5f95ea58" id="339" symbol="8SwTxtFrm" next="347" upper="341" txtNodeIndex="1147">
    <infos>
        <bounds left="2056" top="372289" width="0" height="0" />
    </infos>
TPMB.h    <Finish /></txt>
<txt ptr="0x7f6d5f95eeb8" id="347" symbol="8SwTxtFrm" next="348" prev="339" upper="341" txtNodeIndex="1148">
    <infos>
        <bounds left="2056" top="372289" width="0" height="0" />
    </infos>
This file contains extra TPM2B structures    <Finish /></txt>
<txt ptr="0x7f6d5f95f000" id="348" symbol="8SwTxtFrm" next="349" prev="347" upper="341" txtNodeIndex="1149">
    <infos>
        <bounds left="2056" top="372289" width="0" height="0" />
    </infos>
#ifndef _TPMB_H    <Finish /></txt>
<txt ptr="0x7f6d5f95f118" id="349" symbol="8SwTxtFrm" next="350" prev="348" upper="341" txtNodeIndex="1150">
    <infos>
        <bounds left="2056" top="372289" width="0" height="0" />
    </infos>
#define _TPMB_H    <Finish /></txt>
<txt ptr="0x7f6d5f95f230" id="350" symbol="8SwTxtFrm" next="351" prev="349" upper="341" txtNodeIndex="1151">
    <infos>
        <bounds left="2056" top="372289" width="0" height="0" />
    </infos>
#include "TPM_Types.h"    <Finish /></txt>
```

A well-known advantage of XML is that there exist a lot of libraries for all languages to access an XML file in a comfortable way. One problem here is that there is no obvious indicator which shows the type of content. This solution is similar to the plain text file in the previous section but it is encapsulated in XML. If it is not possible to differentiate the content this is also not the way to go.

4 http://www.libreoffice.org/

#### 1.1.3.3 PDF

An interesting approach is to directly parse the PDF documents which are available on the TCG web site [5]. There are already PDF parsing tools available, for example PDFMiner[6]. Even if it sounds as a good idea because you do not have to expend the effort to get the Word documents it is bad practice. PDF is a presentation oriented format[Pdf] and it is not focused on the content. So at the latest we well stuck at the same problem like had on the previous sections.

#### 1.1.3.4 Office Open XML

Office Open XML also called OpenXML or OOXML is the new default format of Office documents since Microsoft Office 2007.[MsOXML07] It is a zipped archive which contains a bunch of XML files and the containing resources.

On a Linux/Unix system the unzip utility can be called with the document as argument.

**Listing 1.5:** Unzip a Word document.

```
1 $ unzip "TPM Rev 2.0 Part 4 - Supporting Routines 00.99-code.docx"
```

Now the content is extracted to the current directory.

**Listing 1.6:** Content of a word document.

```
1  $ find
2  .
3  ./word
4  ./word/header12.xml
5  ./word/endnotes.xml
6  ./word/document.xml
7  ./word/header8.xml
8  ./word/header10.xml
9  ./word/webSettings.xml
10 ./word/header4.xml
11 ./word/header5.xml
12 ./word/header6.xml
13 ./word/footer2.xml
14 ./word/embeddings
15 ./word/embeddings/oleObject1.bin
16 ./word/footer3.xml
17 ./word/media
18 ./word/media/image1.wmf
19 ./word/header1.xml
20 ./word/header9.xml
21 ./word/footer1.xml
22 ./word/header2.xml
23 ./word/footnotes.xml
24 ./word/footer5.xml
25 ./word/stylesWithEffects.xml
26 ./word/numbering.xml
27 ./word/theme
28 ./word/theme/theme1.xml
29 ./word/_rels
30 ./word/_rels/document.xml.rels
```

5 http://www.trustedcomputinggroup.org/resources/tpm_library_specification
6 http://www.unixuser.org/ euske/python/pdfminer/index.html

```
./word/header11.xml
./word/footer4.xml
./word/footer6.xml
./word/settings.xml
./word/styles.xml
./word/fontTable.xml
./word/header7.xml
./word/header3.xml
./docProps
./docProps/app.xml
./docProps/core.xml
./_rels
./_rels/.rels
./[Content_Types].xml
./customXml
./customXml/itemProps1.xml
./customXml/_rels
./customXml/_rels/item1.xml.rels
```

The most important file for this project is **./word/document.xml**. It contains the whole text of the document and the formatting encapsulated in XML.

For a better reading by humans a pretty print utility was used.

**Listing 1.7:** Pretty print XML file.

```
$ cat document.xml  | tidy -utf8 -xml -w 255 -i -c -q -asxml > pretty_document.xml
```

The relevant tags for this project are contained in the namespace `http://schemas.openxmlformats.org/wordprocessingml/2006/main`[7]. The **document.xml** of an empty document looks like this:

**Listing 1.8:** OpenXML structure of empty document (simplified).

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<w:document xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
    <w:body></w:body>
</w:document>
```

The content of a Word document will be pasted in the body element. For each line in the Word document a **p** element is created. The child called **pPr** contains formatting settings of the line. A variable number of **r** elements in the **p** element contain in their child element **t** the text parts of the line. This are the most important elements for this project, there are a lot of other irrelevant ones. To get an impression on what we are talking about in this paragraph, the following snipped shows how a couple of lines in the Word document looks like in **document.xml**.

**Listing 1.9:** Extract of exported XML file.

```
        <w:p w:rsidR="00530999" w:rsidRPr="00757E70" w:rsidRDefault="00530999" w:rsidP="00530999">
            <w:pPr>
                <w:pStyle w:val="Heading2" />
                <w:pageBreakBefore />
            </w:pPr>
            <w:r>
                <w:lastRenderedPageBreak />
                <w:t>TPMB</w:t>
            </w:r>
            <w:r w:rsidRPr="00757E70">
                <w:t>.h</w:t>
            </w:r>
```

7 This link does not point to a web site, it just describes the namespace

```
13          <w:bookmarkEnd w:id="86" />
14      </w:p>
15      <w:p w:rsidR="00FF05F8" w:rsidRDefault="00FF05F8" w:rsidP="00FF05F8">
16          <w:pPr>
17              <w:pStyle w:val="BodyText" />
18              <w:rPr>
19                  <w:rStyle w:val="Italic" />
20                  <w:i w:val="0" />
21              </w:rPr>
22          </w:pPr>
23          <w:bookmarkStart w:id="87" w:name="_Toc362535795" />
24          <w:r>
25              <w:rPr>
26                  <w:rStyle w:val="Italic" />
27                  <w:i w:val="0" />
28              </w:rPr>
29              <w:t>This file contains extra TPM2B structures</w:t>
30          </w:r>
31      </w:p>
32      <w:p w:rsidR="00FF05F8" w:rsidRPr="008C313E" w:rsidRDefault="00FF05F8" w:rsidP="00FF05F8">
33          <w:pPr>
34              <w:pStyle w:val="Code" />
35              <w:numPr>
36                  <w:ilvl w:val="0" />
37                  <w:numId w:val="23" />
38              </w:numPr>
39              <w:spacing w:before="240" w:after="240" />
40              <w:rPr>
41                  <w:rStyle w:val="CodeBlack" />
42                  <w:color w:val="auto" />
43              </w:rPr>
44          </w:pPr>
45          <w:r>
46              <w:rPr>
47                  <w:rStyle w:val="CodeMacro" />
48              </w:rPr>
49              <w:t>#ifndef</w:t>
50          </w:r>
51          <w:r>
52              <w:rPr>
53                  <w:rStyle w:val="CodeBlack" />
54              </w:rPr>
55              <w:t xml:space="preserve">_TPMB_H</w:t>
56          </w:r>
57      </w:p>
```

This approach allows us to extract the most possible information that is why we decided to go this way.

### 1.1.4 Decision of Programming Language

There are a lot of capable languages out in the world to fulfill this project.

Our modest requirements:

- Object oriented.
- Compilable on Linux/Unix systems, Windows.
- Reliable XML library available preferably already containing in the standard library.

Due this open requirements personal reasons had some influences to our choice. We decided to use Python[8].

Why Python is attractive:

- Runs on Windows, Linux/Unix, Mac OS X.[PyFe]
- Free to use.[PyFe]
- Clear syntax.[PyAb]
- Big standard library.
- Pre-installed on most Linux systems.
- Interactive shell available.

## 1.2 Software Architecture

### 1.2.1 General

All of the three specification parts contain different forms of content:

- 2: Tables only.
- 3: Tables and Code.
- 4: Code only.

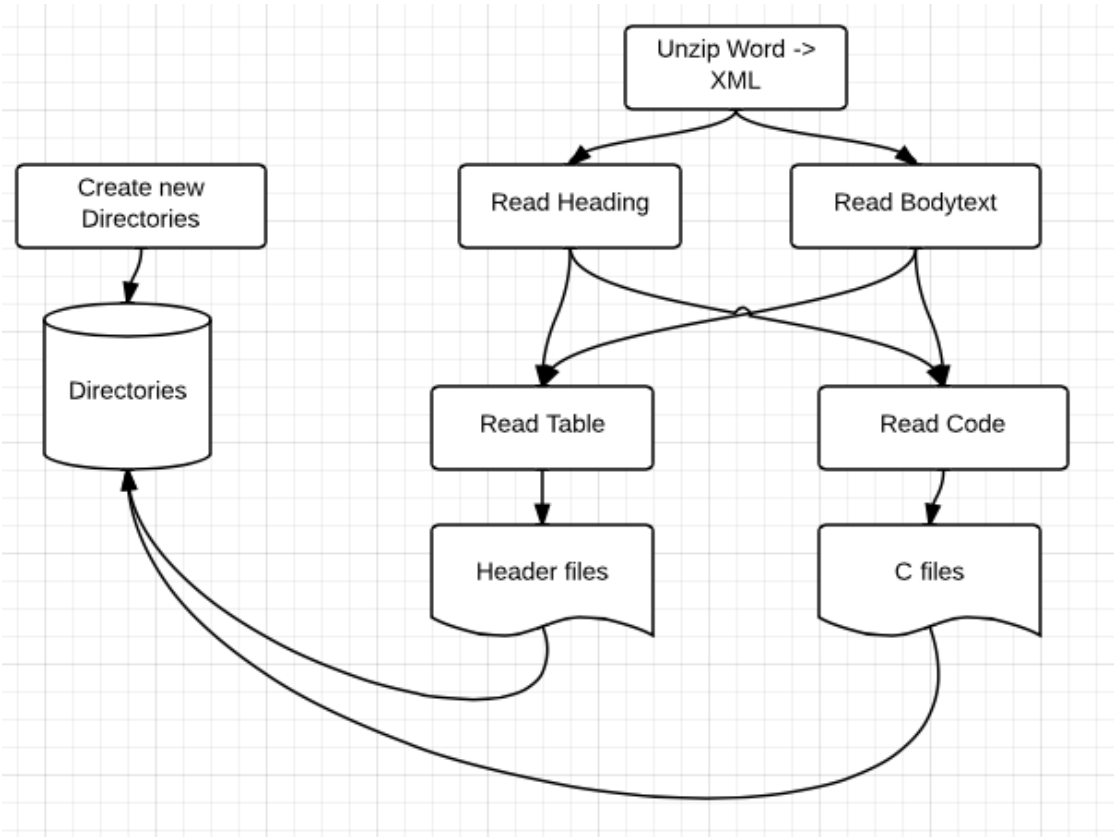
But the base part of accessing the specification file and the loop through the elements is similar on each of them. The architecture is visualized in figure 1.1.

To parse the XML file the Python module **xml.etree.ElementTree** is used. A change to this module has been performed to reduce the parsing time of 15 seconds with **xml.dom.minidom** to 5 seconds[9].

In one go the Word document is unzipped and parsed by **xml.etree.ElementTree**.

8 `http://www.python.org/`
9 Time data measured while parsing part 4 document.

**Figure 1.1:** Architecture of the parsers.




**Listing 1.10:** Read document to element tree.

```
1 with zipfile.ZipFile(DOC_TO_PARSE) as zfile:
2     doc = ET.parse(zfile.open("word/document.xml"))
```

All child elements of the body are stored in a variable.

**Listing 1.11:** Get child elements of body.

```
1 elements = doc.getroot().getchildren()[0].getchildren()
```

The following namespace is used in all tag, so it is once defined.

**Listing 1.12:** Namespace definition.

```
1 NAMESPACE = "{http://schemas.openxmlformats.org/wordprocessingml/2006/main}"
```

Now the loop starts. The two relevant elements have to be handled in it. One of them is the **p** element which contains one line of the Word document the other element is the **tbl** which is only relevant if in the previous **p** element a table description was found.

**Listing 1.13:** Main parser loop.

```
for element in elements:
    if element.tag == "{0}p".format(NAMESPACE):
        # handle text
    elif element.tag == "{0}tbl".format(NAMESPACE):
        # handle table
```

## 1.2.2 Part 2

### 1.2.2.1 Approach

In this part, all useful information is stored in tables. The first chapter describes how each type of table have to be translated.

### 1.2.2.2 Challenges

#### Table Parsing

This part of the specification contains a lot of tables with different numbers of rows and columns. This figure shows how a table look like in the document:

The title indicates the object the table contains but like the notes before the table it isn't required for parsing. The first important thing is the table description. All needed information how the table content have to be translated into code is indicated by the table description.

This little table from the last figure produces a lot of code in the XML file:

**Listing 1.14:** Table in OpenXML (shortened).

```
        <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="00975A44">
            <w:pPr>
                <w:pStyle w:val="Heading2" />
            </w:pPr>
            <w:r>
                <w:t>TPMI_DH_OBJECT</w:t>
            </w:r>
        </w:p>
        <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="00975A44">
            <w:pPr>
                <w:pStyle w:val="BodyText" />
            </w:pPr>
            <w:r>
                <w:t>The TPMI_DH_OBJECT interface type is a handle that references a loaded object. The handles in this set are
      used to refer to either transient or persistent object. The range of these values would change according to the TPM
        implementation.</w:t>
```

**Figure 1.2:** Table in part 4 of the specification.

### 9.3 TPMI_DH_OBJECT

The TPMI_DH_OBJECT interface type is a handle that references a loaded object. The handles in this set are used to refer to either transient or persistent object. The range of these values would change according to the TPM implementation.

NOTE These interface types should not be used by system software to qualify the keys produced by the TPM. The value returned by the TPM shall be used to reference the object.

**Table 38 — Definition of (TPM_HANDLE) TPMI_DH_OBJECT Type**

| Values | Comments |
|---|---|
| {TRANSIENT_FIRST:TRANSIENT_LAST} | allowed range for transient objects |
| {PERSISTENT_FIRST:PERSISTENT_LAST} | allowed range for persistent objects |
| +TPM_RH_NULL | the conditional value |
| #TPM_RC_VALUE | |

```
16          </w:r>
17        </w:p>
18        <w:p w:rsidR="00975A44" w:rsidRPr="007F4942" w:rsidRDefault="00975A44" w:rsidP="00975A44">
19            <w:pPr>
20                <w:pStyle w:val="NOTE" />
21                <w:keepNext />
22            </w:pPr>
23            <w:r>
24                <w:t>NOTE</w:t>
25            </w:r>
26            <w:r>
27                <w:tab />
28                <w:t>These interface types should not be used by system software to qualify the keys produced by the TPM. The
    value returned by the TPM shall be used to reference the object.</w:t>
29            </w:r>
30        </w:p>
31        <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="00975A44">
32            <w:pPr>
33                <w:pStyle w:val="TABLE-title" />
34            </w:pPr>
35            <w:bookmarkStart w:id="922" w:name="_Toc362536010" />
36            <w:r>
37                <w:t xml:space="preserve">Table</w:t>
38            </w:r>
39            <w:r>
40                <w:fldChar w:fldCharType="begin" />
41            </w:r>
42            <w:r>
43                <w:instrText xml:space="preserve">SEQ Table \* ARABIC</w:instrText>
44            </w:r>
45            <w:r>
46                <w:fldChar w:fldCharType="separate" />
47            </w:r>
48            <w:r w:rsidR="00E82E05">
49                <w:rPr>
50                    <w:noProof />
51                </w:rPr>
52                <w:t>38</w:t>
53            </w:r>
54            <w:r>
55                <w:rPr>
56                    <w:noProof />
57                </w:rPr>
58                <w:fldChar w:fldCharType="end" />
59            </w:r>
60            <w:r>
61                <w:t xml:space="preserve">- D</w:t>
62            </w:r>
63            <w:r w:rsidRPr="00AA62B1">
64                <w:t>efin</w:t>
65            </w:r>
```

```
            <w:r>
                <w:t>ition of (TPM_HANDLE) TPMI_DH_OBJECT Type</w:t>
            </w:r>
            <w:r>
                <w:t xml:space="preserve"></w:t>
            </w:r>
        </w:p>
        <w:tbl>
            <w:tblPr>
                <w:tblW w:w="9360" w:type="dxa" />
                <w:jc w:val="center" />
                <w:tblBorders>
                    <w:top w:val="single" w:sz="6" w:space="0" w:color="auto" />
                    <w:left w:val="single" w:sz="6" w:space="0" w:color="auto" />
                    <w:bottom w:val="single" w:sz="6" w:space="0" w:color="auto" />
                    <w:right w:val="single" w:sz="6" w:space="0" w:color="auto" />
                    <w:insideH w:val="single" w:sz="6" w:space="0" w:color="auto" />
                    <w:insideV w:val="single" w:sz="6" w:space="0" w:color="auto" />
                </w:tblBorders>
                <w:tblLayout w:type="fixed" />
                <w:tblCellMar>
                    <w:left w:w="58" w:type="dxa" />
                    <w:right w:w="58" w:type="dxa" />
                </w:tblCellMar>
                <w:tblLook w:val="0000" w:firstRow="0" w:lastRow="0" w:firstColumn="0" w:lastColumn="0" w:noHBand="0"
    w:noVBand="0" />
            </w:tblPr>
            <w:tblGrid>
                <w:gridCol w:w="4680" />
                <w:gridCol w:w="4680" />
            </w:tblGrid>
            <w:tr w:rsidR="00975A44" w:rsidRPr="001F1A1B" w:rsidTr="007874D7">
                <w:trPr>
                    <w:tblHeader />
                    <w:jc w:val="center" />
                </w:trPr>
                <w:tc>
                    <w:tcPr>
                        <w:tcW w:w="4680" w:type="dxa" />
                        <w:tcBorders>
                            <w:top w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:left w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:bottom w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:right w:val="single" w:sz="8" w:space="0" w:color="auto" />
                        </w:tcBorders>
                    </w:tcPr>
                    <w:p w:rsidR="00975A44" w:rsidRPr="001F1A1B" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                        <w:pPr>
                            <w:pStyle w:val="TABLE-col-heading" />
                        </w:pPr>
                        <w:r w:rsidRPr="001F1A1B">
                            <w:t>Value</w:t>
                        </w:r>
                        <w:r>
                            <w:t>s</</w:t>
                        </w:r>
                    </w:p>
                </w:tc>
                <w:tc>
                    <w:tcPr>
                        <w:tcW w:w="4680" w:type="dxa" />
                        <w:tcBorders>
                            <w:top w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:left w:val="single" w:sz="8" w:space="0" w:color="auto" />
                            <w:bottom w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:right w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        </w:tcBorders>
                    </w:tcPr>
                    <w:p w:rsidR="00975A44" w:rsidRPr="001F1A1B" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                        <w:pPr>
                            <w:pStyle w:val="TABLE-col-heading" />
                        </w:pPr>
                        <w:r w:rsidRPr="001F1A1B">
                            <w:t>Comments</</w:t>
                        </w:r>
                    </w:p>
                </w:tc>
            </w:tr>
            <w:tr w:rsidR="00975A44" w:rsidRPr="004A61C2" w:rsidTr="007874D7">
                <w:trPr>
                    <w:jc w:val="center" />
                </w:trPr>
                <w:tc>
                    <w:tcPr>
                        <w:tcW w:w="4680" w:type="dxa" />
                        <w:tcBorders>
                            <w:top w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:left w:val="single" w:sz="12" w:space="0" w:color="auto" />
                            <w:bottom w:val="single" w:sz="6" w:space="0" w:color="auto" />
```

```
                        <w:right w:val="single" w:sz="8" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                    </w:pPr>
                    <w:r>
                        <w:t>{TRANSIENT_FIRST:TRANSIENT_LAST}</w:t>
                    </w:r>
                </w:p>
            </w:tc>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:top w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        <w:left w:val="single" w:sz="8" w:space="0" w:color="auto" />
                        <w:bottom w:val="single" w:sz="6" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="12" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRPr="001F7ECD" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                    </w:pPr>
                    <w:r>
                        <w:t>allowed range for transient objects</w:t>
                    </w:r>
                </w:p>
            </w:tc>
        </w:tr>
        <w:tr w:rsidR="00975A44" w:rsidRPr="004A61C2" w:rsidTr="007874D7">
            <w:trPr>
                <w:trHeight w:val="285" />
                <w:jc w:val="center" />
            </w:trPr>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:top w:val="single" w:sz="6" w:space="0" w:color="auto" />
                        <w:left w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="8" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                    </w:pPr>
                    <w:r>
                        <w:t>{PERSISTENT_FIRST:PERSISTENT_LAST}</w:t>
                    </w:r>
                </w:p>
            </w:tc>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:top w:val="single" w:sz="6" w:space="0" w:color="auto" />
                        <w:left w:val="single" w:sz="8" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="12" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                    </w:pPr>
                    <w:r>
                        <w:t>allowed range for persistent objects</w:t>
                    </w:r>
                </w:p>
            </w:tc>
        </w:tr>
        <w:tr w:rsidR="00975A44" w:rsidRPr="004A61C2" w:rsidTr="007874D7">
            <w:trPr>
                <w:jc w:val="center" />
            </w:trPr>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:left w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="8" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                    </w:pPr>
```

```
                    </w:pPr>
                    <w:r>
                        <w:t>+TPM_RH_NULL</w:t>
                    </w:r>
                </w:p>
            </w:tc>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:left w:val="single" w:sz="8" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="12" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="000F5B40">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                    </w:pPr>
                    <w:r>
                        <w:t xml:space="preserve">the</w:t>
                    </w:r>
                    <w:r w:rsidR="000F5B40">
                        <w:t>conditional</w:t>
                    </w:r>
                    <w:r>
                        <w:t xml:space="preserve" />
                    </w:r>
                    <w:r w:rsidR="000F5B40">
                        <w:t>value</w:t>
                    </w:r>
                </w:p>
            </w:tc>
        </w:tr>
        <w:tr w:rsidR="00975A44" w:rsidRPr="004A61C2" w:rsidTr="007874D7">
            <w:trPr>
                <w:jc w:val="center" />
            </w:trPr>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:left w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        <w:bottom w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="8" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                        <w:keepNext w:val="0" />
                    </w:pPr>
                    <w:r>
                        <w:t>#TPM_RC_VALUE</w:t>
                    </w:r>
                </w:p>
            </w:tc>
            <w:tc>
                <w:tcPr>
                    <w:tcW w:w="4680" w:type="dxa" />
                    <w:tcBorders>
                        <w:left w:val="single" w:sz="8" w:space="0" w:color="auto" />
                        <w:bottom w:val="single" w:sz="12" w:space="0" w:color="auto" />
                        <w:right w:val="single" w:sz="12" w:space="0" w:color="auto" />
                    </w:tcBorders>
                </w:tcPr>
                <w:p w:rsidR="00975A44" w:rsidRDefault="00975A44" w:rsidP="006C6220">
                    <w:pPr>
                        <w:pStyle w:val="TABLE-cell" />
                        <w:keepNext w:val="0" />
                    </w:pPr>
                </w:p>
            </w:tc>
        </w:tr>
    </w:tbl>
```

At the beginning are some **p** elements. The first ones of them contains information which wont be used (heading, notes), this elements will be skipped. The last **p** before the table begins is required because it contains the description. To be sure the parser recognizes the correct element, **pStyle** is checked if the attribute **val** is equal to **TABLE-title**.

**Listing 1.15:** Indicator for table description.

```
<w:pStyle w:val="TABLE-title" />
```

To get the full description all values of existing **t** elements in **p** are concatenated.

The following loop element is a **tbl**. This one represents the total table. To access this table in a comfortable way later in the program it is stored in a class object. Internally lists are used to fill them with a dictionary (key, value). The reason to do it this way is that it is easy readable in the program and because we need to access specific single cells, they are directly addressed with the row number and the column name.

**Listing 1.16:** Class to store table data.

```
class Table(object):
    def __init__(self):
        self.head = []
        self.rows = []
        self.rows.append(dict())

    def add_head(self, name):
        self.head.append(name)

    def add_cell(self, value):
        if len(self.head) == 0:
            return
        if len(self.rows[-1]) >= len(self.head):
            self.rows.append(dict())
        self.rows[-1].update({self.head[len(self.rows[-1])]:value})

    def print_table(self):
        print("\t".join(self.head))
        print(self.head)
        for row in self.rows:
            print(row)

    def get_rows(self):
        return self.rows

    def get_keys(self):
        return self.rows[0].keys()
```

### Word Version Tracking

TCG use the Microsoft Word feature **Track Changes** for traceability. It is a good one to use it in collaborations, because everyone knows what changed since the last version, who applied the change and also the modification time.[MsWrd13] The protection in the XML file looks simple. All modification on the document will be encapsulated into an **ins** element. Depending of the modification type some parts in the **ins** element will be encapsulated in a **del** element to show the deleted content (strike through in the Change-View).

The following listing shows how an **ins** element looks like.

**Listing 1.17:** Element to identify changes.

```
<w:ins w:id="1107" w:author="Ken Goldman" w:date="2013-07-11T16:43:00Z" />
```

A **del** element contains the same attributes. This listing shows some deleted text:

**Listing 1.18:** Element to identify changes with a deleted line.

```
<w:ins w:id="1086" w:author="Ken Goldman" w:date="2013-07-11T16:43:00Z">
    <w:del w:id="1087" w:author="David Wooten" w:date="2013-07-25T17:11:00Z">
        <w:r w:rsidDel="00E82E05">
            <w:rPr>
                <w:noProof />
            </w:rPr>
            <w:delText>46</w:delText>
        </w:r>
    </w:del>
    <w:r>
        <w:rPr>
            <w:noProof />
        </w:rPr>
        <w:fldChar w:fldCharType="end" />
    </w:r>
    <w:r>
        <w:t xml:space="preserve">- D</w:t>
    </w:r>
    <w:r w:rsidRPr="00AA62B1">
        <w:t>efin</w:t>
    </w:r>
    <w:r>
        <w:t>ition of (TPM_HANDLE) TPMI_RH_</w:t>
    </w:r>
    </w:r>
</w:ins>
```

For parsing this mean we have check every element if it is an **ins** one. If it applies we have to encapsulate it and check again for a modification element. If a **del** is found, the content have to be dropped.

In consultation with the supervisor we decided to drop support for Word documents with unaccepted changes. To use the program it is required to open the specification document, accept all changes and save it.

### Table Formatting

Some cells are not formatted like they should be. Maybe it is because in some earlier versions of the document the cell was just a note and now they are filled with content used to create code. In this few cells the formating is **Table-cell-note** instead of **TABLE-cell**. To get around this problem an extra query was added:

**Listing 1.19:** Solution to formating problem.

```
elif formatting_value == "Table-cell-note" and not full_cell_value.startswith("NOTES"):
    table.add_cell(full_cell_value)
```

### 1.2.3 Testing

#### 1.2.3.1 Unit Tests

To check this part a unit test was created. Python offers micro testing ability with the module **unittest**. It is simple to use. To create a test, a new Python program have to be created. This file imports the module to test, which is **part4parser** in this part.

**Listing 1.20:** Unit test example.

```
#!/usr/bin/env python
import part4parser
class Testpart4parser(unittest.TestCase):

    def setUp(self):
        pass

    def tearDown(self):
        pass

    def test_example(self):
        self.assertEqual(1, 1)
```

The program have to be modified that it does not automatically execute if it is imported. To implement this, the variable __**name**__ is checked for the content **"__main__"**

**Listing 1.21:** Module modification for testing.

```
if __name__ == "__main__":
    # program code
```

#### 1.2.3.2 Internal Error Handling

For the program it is required that the names of the columns are like described in the first chapter of the part 4 document. If there are incorrect names the program will produce bad code. The program was modified to write information about bad tables to the file **tpm.log** in the output directory.

**Listing 1.22:** Error handling on column names.

```
    if table.has_col("Type") and table.has_col("Name"):
        # ...
    else:
        write_file("[ERROR] Incorrect cols at table " + table_description, FILE.LOG)
```

### 1.2.4 Part 3

Part 3 describe series of commands, and each TPM command has its specific Response and Command table to define the variables used in C code.

#### 1.2.4.1 Approach

1. Unzip Word document into XML files.
2. Read all elements, tables and codes in XML file.
3. Extract the contents in tables and C codes into Header files and C files.
4. Put those header files and C files into respectively directories.

#### 1.2.4.2 Challenges

**Table Parsing**
There are 222 tables in part 3 and every second table consist of a header file, which defines the attributes of variables.

**Solution**

- Write a table class, which contains functions that can store the properties of each column in the table.
- Create a table class.
- Put all we read from the table into this class.
- Take out the properties in the table class into header file.

**Formatting Problem**
The Code in Header file which contains the properties of table does not align at the end of each code.

**Figure 1.3:** Table at part 3.

#### 11.4.2 Command and Response

**Table 7 — TPM2_Shutdown Command**

| Type | Name | Description |
|---|---|---|
| TPMI_ST_COMMAND_TAG | tag | |
| UINT32 | commandSize | |
| TPM_CC | commandCode | TPM_CC_Shutdown {NV} |
| TPM_SU | shutdownType | TPM_SU_CLEAR or TPM_SU_STATE |

**Table 8 — TPM2_Shutdown Response**

| Type | Name | Description |
|---|---|---|
| TPM_ST | tag | see clause 8 |
| UINT32 | responseSize | |
| TPM_RC | responseCode | |

**Listing 1.23:** Unformatted code.

```
#ifndef CERTIFY_FP.H
#define CERTIFY_FP.H
 typedef struct {
TPMI_DH_OBJECT      objectHandle;
TPMI_DH_OBJECT      signHandle;
TPM2B_DATA      qualifyingData;
TPMT_SIG_SCHEME      inScheme;
}
Certify_In;
typedef struct {
TPM2B_ATTEST      certifyInfo;
TPMT_SIGNATURE      signature;
}
Certify_Out;
#define Certify_certifyInfo             (TPM_RC_P + TPM_RC_1)
#define Certify_signature            (TPM_RC_P + TPM_RC_2)

TPM_RC
TPM2_Certify_(Certify__In *in, Certify__Out *out);
#endif
```

## Solution

- Use the command "pretty print" in Python.

**Listing 1.24:** Formatted code.

```
#ifndef _CERTIFY_H
#define _CERTIFY_H

typedef struct {
    TPMI_DH_OBJECT                  objectHandle;
    TPMI_DH_OBJECT                  signHandle;
    TPM2B_DATA                      qualifyingData;
```

```
8       TPMT_SIG_SCHEME                     inScheme;
9   } Certify_In;
10
11  typedef struct {
12      TPM2B_ATTEST                        certifyInfo;
13      TPMT_SIGNATURE                      signature;
14  } Certify_Out;
15
16  #define Certify_certifyInfo           (TPM_RC_P + TPM_RC_1)
17  #define Certify_signature             (TPM_RC_P + TPM_RC_2)
18
19  TPM_RC
20  TPM2_Certify(Certify_In *in, Certify_Out *out);
21
22  #endif
```

### 1.2.5 Part 4

#### 1.2.5.1 Approach

The structure of this document is mostly like this:

- 5 Header Files
  - 5.2 bool.h
  - 5.3 Capabilities.h
  - 5.4 TPMB.h
  - ...
- 9 Support
  - 9.1 AlgorithmCap.c
  - 9.2 Bits.c
  - 9.3 CommandCodeAttributes_fp.h
  - ...

The level two headings represents the file name where the content goes into. On a closer look to one of this chapters you see a lot of sub chapters. Each of this contains some code snippets and a small description. To get the total code of the file which is described in a chapter all code snippets have to be joined in the existing order.

The program loops through all **p** elements. To recognize the content which is stored in this element, the format name from **pStyle** is read.

- Code ⇒ source code
- Heading ⇒ file name
- Italic ⇒ comment
- ANNEX-heading ⇒ file name
- ListNumber ⇒ comment
- NOTE ⇒ comment

In case it is supposed to be a file name the heading number is tracked. This is required to determine if it is a level two heading because this is the only relevant level. Other headings are ignored.

If content was defined as code, it will be written to the pre-defined file.

Because it is useful to have the description of a code snipped in a source file, content which was defined as comment will also be written to the file. This content have to be edited. For a better reading, the text will break after 80 chars and a comment mark (//) is added on the beginning of lines. For wrapping a Python module is used:

**Listing 1.25:** Comment Wrapping.

```
1 import textwrap
2 multiline_text = "\n// ".join(textwrap.wrap(complet_text, 80))
3 write_file("".join(["\n// ", multiline_text, "\n"]), file_name)
```

### 1.2.5.2 Challenges

#### Hidden File Names

There are a few chapters in this part which aren't composed of just file name. Some of them are hidden in brackets and they occur in the middle or at the end of some text.

- 7.2 Attestation Command Support (**Attest_spt.c**)
- 7.3 Context Management Command Support (**Context_spt.c**)

- 6.1 CommandDispatcher() In the reference implementation, a program that uses part 3 as input automatically generates the command dispatch code. The function prototype header file (**CommandDispatcher_fp.h**) is shown here.

- 6.2 ParseHandleBuffer() In the reference implementation, the routine for unmarshaling the command handles is automatically generated from part 3 command tables. The prototype header file (**HandleProcess_fp.h**) is shown here.

To get around this problem regular expressions are used to cut out the file names.

**Listing 1.26:** Definition of RegEx's.

```
file_pattern = re.compile(r'^\w+.[ch]$')
file_pattern_with_brackets = re.compile(r'\(\w+.[ch]\)')
```

Later in the program the expression is used to check if the file name is found (RegEx matched) and the name is set to the match.

**Listing 1.27:** RegEx to find file name.

```
match = file_pattern_with_brackets.search(complet_text)
if match:
    complet_text = match.group()[1:-1]
```

## 1.3 Results

This chapter describes the output which we are able to extract.

### 1.3.1 Part 2

This part describes just the content of four files:

- ./tpm/support/marshal.c
  - marshal and unmarshal functions

- ./tpm/include/prototypes/marshal_fp.h
  - marshal and unmarshal function prototypes

- ./tpm/include/TPM_Types.h
  - all type definitions except from annex types

- ./tpm/include/Implementation.h
  - types defined in the annex of the document

### 1.3.2 Part 3

Extract all the tables and C codes in part 3, so that the TPM commands in part 3 can be used by later works.

### 1.3.3 Part 4

This part describes a lot of header and source files. Which are required by the other parts of the specification. The content of the files was verified by a couple of random picks which were compared to the Visual Studio implementation.

**Listing 1.28:** Extracted files from part 4.

```
$ ls -1 /tmp/TPMCmd
AlgorithmCap.c
Attest_spt.c
Bits.c
bool.h
Cancel.c
Capabilities.h
Clock.c
CommandAudit.c
CommandCodeAttributes.c
CommandDispatcher_fp.h
Commands.c
Context_spt.c
CpriCryptPri.c
CpriData.c
CpriDataEcc.c
CpriDataEcc.h
CpriECC.c
CpriHash.c
CpriRNG.c
CpriRSA.c
CpriSym.c
CryptDataEcc.c
CryptoBaseTypes.h
CryptUtil.c
DA.c
DRTM.c
Entity.c
Entropy.c
ExecCommand.c
Global.c
Global.h
Handle.c
HandleProcess_fp.h
Hierarchy.c
InternalRoutines.h
Locality.c
LocalityPlat.c
Manufacture.c
Marshal.c
MathFunctions.c
MemoryLib.c
NV.c
NVMem.c
NV_spt.c
Object.c
Object_spt.c
PCR.c
PlatformData.c
PlatformData.h
```

```
Platform.h
Policy_spt.c
Power.c
PowerPlat.c
PP.c
PPPlat.c
PropertyCap.c
RSAData.c
RSAKeySieve.c
RSAKeySieve.h
Session.c
SessionProcess.c
swap.h
TcpServer.c
Ticket.c
Time.c
TPMB.h
TpmBuildSwitches.h
TPMCmdp.c
TPMCmds.c
TpmError.h
TpmFail.c
TpmTcpProtocol.h
VendorString.h
```

## 1.4 Conclusions

Because the project is very complex, it was not possible to finish it. We created parsers for each document and all of them will write source files. The open task are described here.

### General

There are some classes which are used in multiple parsers which can be extracted into a separate module. Following classes are affected: Table, Timer.

The file for the input and the directory for the output are at the moment hard coded in each parser. It would be more comfortable to execute the parser with this information as arguments like it is common for command line programs.

An interesting feature would be the handling of Word documents which contains unaccepted changes.

### Part 2

The only thing in this part which does not work correctly is the generation of marshal and unmarshal code. A detailed comparison between the marshal and unmarshal code of the Visual Studio implementation and the tables in the specification will be required to figure out if it is even possible to generate this code automatically or if some work by hand is required.

### Part 3

This part consist of two separate programs. One generates source code and the other generates the header files. It would be more comfortable to process the total part with one program.

### Part 4

All files are correctly extracted and are similar to the Visual Studio implementation. One thing which is not done yet is the splitting of the files to subdirectories.

# List of Abbreviations

| | |
|---|---|
| BIOS | Basic Input/Output System |
| BYOD | bring-your-own-device |
| OOXML | Office Open XML |
| PDF | Portable Document Format |
| TCG | Trusted Computing Group |
| TPM | Trusted Platform Module |

# Listings

# List of Figures

# Bibliography

[HpTpm] *HP TPM Information*, http://h18004.www1.hp.com/products/servers/proliantstorage/module.html

[MsSecB] *Microsoft Windows, Secure Boot.*, http://technet.microsoft.com/en-us/library/hh824987.aspx

[UbSecB] *Canonical Blog Post about Secure Boot implementation, 2012.*, http://blog.canonical.com/2012/06/22/an-update-on-ubuntu-and-secure-boot/

[TPM2ChL] *Changes of TPM version 2.0 since 1.2.*, https://www.trustedcomputinggroup.org/resources/tpm_20_library_specification_faq

[TCG] *Trusted Computing Group*, http://www.trustedcomputinggroup.org/about_tcg

[TLP1] *TPM 2.0 Library Specification Revision 00.99 Part 1*, Public Review_TPM Rev 2.0 Part 1 - Architecture 00.99.pdf

[TLP4] *TPM 2.0 Library Specification Revision 00.99 Part 4*, Public Review_TPM Rev 2.0 Part 4 - Supporting Routines 00.99-code.pdf

[Pdf] *About PDF* http://www.adobe.com/products/acrobat/adobepdf.html

[MsOXML07] *OpenXML White Paper* http://www.ecma-international.org/news/TC45_current_work/OpenXML%20White%20Paper.pdf

[PyFe] *Python.org* http://www.python.org/

[PyAb] *About Python* http://www.python.org/about/

[MsWrd13] *Office 2013 change tracking.* http://office.microsoft.com/en-001/word-help/track-changes-HA102840151.aspx

[MsBl] *Microsoft BitLocker* `http://windows.microsoft.com/en-us/windows-vista/bitlocker-drive-encryption-overview`