浙江工业大学

本科毕业论文（设计）

外文翻译及原稿

译文题目：Microsoft Office 2007 文件中的数据隐藏和检测

原稿出处：Digital Investigation, 2009, Elsevier

学　　院: 计算机科学与技术学院

专　　业: 软件工程（中外合作办学）

班　　级: 2019 软件工程（中外合作办学）02

学　　号: 201906150218

学生姓名: 唐晨宇

指导老师: 李小薪

提交日期: 2023 年 03 月

Microsoft Office 2007 文件中的数据隐藏和检测

摘要 随着越来越多的罪犯试图隐藏有罪的数据或盗取的信息，法证检验师和计算机安全专业人员知道在如何寻找隐藏的信息是很重要的。本文演示了如何隐藏 Microsoft Office 2007 文件中的数据。Office Open XML(OOXML) 格式构成了 Microsoft Office 2007 的基础，个人可以使用 OOXML 在 Microsoft Office 2007 文件中定义自定义部件、关系或是以上这两者以存储和隐藏信息。幸运的是对于数字调查人员来说，这种隐藏的数据可以通过寻找未知部分或关系的存在而被检测到。

1、引言

对电子文件的调查是计算机取证的一个重要方面，因为这类文件往往包含可用作犯罪证据的重要信息。虽然法证检验师经常可以从电子文档中获取重要信息，但已经出现的数据隐藏工具和技术可以完全隐藏或删除这些文件中的一些数据，从而使检查真实的文件内容变得困难。恐怖分子可以在电子文档中隐藏通信、计划和其他数据，被盗数据可以在电子文档的隐藏部分存储和传输。即使是被设计用于执行法证检查的应用程序也不会显示隐藏的数据，因此，法证计算机调查人员需要使用特殊技术来检测电子文档文件中的隐藏数据。虽然这项工作类似于“隐写分析”，但从检测和恢复文档文件中的隐藏数据的角度来看，应当更确切的将它归类为“文档取证”。

Microsoft(MS) Office 文件 (例如 MS Word、PowerPoint 和 Excel) 是最流行的电子文档类型。大多数新的计算机系统包括 MS Office 产品，特别是 MS Word、PowerPoint 和 Excel。出于这个原因，法证检查员必须能够检测这些文件中的可疑数据。从数据隐藏的角度来看，重要的是不仅要完全隐藏数据，而且要隐藏数据已被隐藏的事实。MS Office 文件是隐藏信息的一个很好的选择，因为这种常见文件的存在不太可能引起调查人员的怀疑[1]。

几种在 MS Office 文件中隐藏数据的方法已经被提出了，但并不是专门针对 Office 2007 文件[2]。MS Office 2007 文件的格式不同于以前版本 (如 MS Office 1997-2003) 使用的格式。MS Word、PowerPoint 和 Excel 2007 文件使用称为 Office Open XML(OOXML) 的新文件格式。在对 MS Office 2007 文件格式进行检验和测试的基础上，本文论证了在 MS Office 2007 文件中隐藏数据实际上是可能的。在 MS Office 2007 文件中管理数据的关键策略是创建不易察觉的内容。这些不易察

觉的数据可以使用“内容关系”的概念来创建。如果某人创建的数据与 MS Office 文件的主要数据内容无关，但只要数据满足正常 MS Office 文件的最低结构要求，这些数据就不会显示在 MS Office 屏幕上。

本文详细介绍了一种在 MS Office 2007 文件中隐藏数据的方法，并提供了一种检测此类文件中隐藏数据的算法。一旦法证人员注意到 MS Office 中有无法立即看到的数据，他们就可以使用本文的方法来揭示这些数据。本文中的大多数数据隐藏和检测示例都集中在 MS Word 2007 文件上，但这些方法也适用于 MS PowerPoint 和 Excel 2007 文件。

论文的其余部分结构如下。第二节介绍了前人的研究成果，并说明了本研究的必要性。第 3 节解释了 OOXML 格式。第 4 节重点介绍了一种基于 MS Office 2007 文件格式 OOXML 的数据隐藏方法。第 5 节描述了我们所提供的“Detector”检测工具用来发现隐藏数据的机制。最后，第 6 节得出结论并描述了仍需完成的工作。

2、数据隐藏的相关研究

对利用每个电子文档本身的原生文件格式的特征在电子文档中隐藏数据的方法的研究包括 MS Office 文档。例如，Office 2007 之前版本 (即 1997-2003) 的 MS Office 文件 (例如 MS Word、PowerPoint 和 Excel) 使用 Microsoft 定义的复合文档文件格式[3]。Castiglione 等人描述了一种使用复合文档文件格式的数据隐藏方法[2]。复合文档文件有助于组织电子文档的内容。它可以将文件数据分成几个流，并将这些流分别存储在文件中。通过这种方式，复合文档文件支持文件内部的完整文件系统，其中流就像文件系统中的文件，而存储区域就像子目录[3]。

复合文档文件由扇区和短扇区组成。扇区的大小为 512 字节，而短扇区的大小为 64 字节[3]。由于文件被保存为固定扇区和短扇区单元，因此产生了大量的垃圾空间 (已分配扇区中的空闲空间) 和许多空扇区 (未分配扇区)[2]。这些垃圾空间和空扇区可用于隐藏 MS Office 文件中的数据。隐藏过程包括四个步骤：

1. 计算文档的垃圾空间的大小。
2. 隐秘消息通过其报头进行压缩和加密。
3. 如有必要，将空扇区添加到该过程中。
4. 隐秘信息隐藏在垃圾空间和空扇区中。

Castiglione 等人从计算机取证和隐写术的角度提出了关于复合文档文件的第一项工作。与 MS Office 文档文件类似，复合文档文件存储的数据比用户保存的数据多

得多。从这些数据获得的信息可能与法证调查有关，但需要适当的工具和知识来检索和解释这些信息。

所有包含隐藏信息的空间，以及 MS Office 结构化存储所浪费的空间，可能被利用于隐写的目的。Castiglione 介绍了一个名为“StegOle”的工具，它利用这个空间来隐藏 MS Office 文档文件中的消息。

涉及 XML 的数据隐藏技术也在过去的工作中发表过。Inoue 等人提出了在 XML 文件中隐藏数据的方法，并指出很少有研究人员研究在电子文档文件中隐藏数据的方法[4]。在涉及 XML 数据交换或 XML 网页的应用中，隐秘数据可以在不改变原始内容的情况下嵌入到 XML 模块中。这些方法可以很容易地转换为现有的 XML 文档文件。Inoue 等人提出了五种在 XML 文件中隐藏数据的方法：(1) 表示空元素；(2) 标签中的空白；(3) 使用元素的表面顺序；(4) 使用属性的表面顺序；(5) 元素包含其他元素。这些方法主要涉及更改 XML 属性的顺序或在 XML 元素名称和“/”之间创建空格作为一种隐藏数据的机制。这些数据隐藏算法可能有些简单，并且它们的数据隐藏容量非常小，因为该算法实际上是用于文本隐藏。但由于在大多数上下文中要隐藏的数据不限于文本数据，因此本工作使用 XML 架构来隐藏数据，而不是使用 Inoue 文章中[4] 中建议的隐藏方法。

XML 架构定义数据流中可能存在的信息、可能出现的位置以及应该如何使用。此外，XML 架构建立了一个词汇表，用户和应用程序 (或另一个用户) 可以通过该词汇表交换信息[5]。换句话说，XML 架构反映了应用程序的需求。由于 MS Office 2007 文件格式使用指定的 XML 格式，因此 MS Office 2007 文件遵循 XML 架构，并可以使用指定的 OOXML 架构在 MS Office 2007 文件中隐藏数据。为了在 MS Office 2007 文件中隐藏数据，本研究试图在维护基本元素的同时操作 OOXML 架构的某些方面，以便 MS Office 2007 文件仍能正常打开。

3、Microsoft Office 2007 文件格式

MS Office 2007 文件由一系列压缩的组件部分组成，这些部分存储在称为包的容器中，每个解压缩的包符合 OOXML 文件格式要求。包是一个普通的 Zip 存档，其中包含包的内容类型项、关系项和部件[6]。对于每个包，都有一个包关系 Zip 项，其中包含有关包及其部件之间的关系的信息。同样，还有包含有关文档各部分之间关系的信息的部分关系 Zip 项。简而言之，这些包定义了文档的整体结构。

3.1 OOXML 格式

由于开放打包约定 (OPCs)，除了文件的特定内容之外，OOXML 文件的格式是可以相互拆分的。OPC 是以各种形式 (XML、图像、元数据等) 存储打包内容的方法之一，并用于完整地表示文档文件。最推荐的 OPC 格式是 Zip 存档。

如图1所示，OOXML 文件基于以下内容：

- 包：Zip 文档。
- 部件：Zip 文档中的文件。
- 关系：部件与包之间或部件之间的关系。

3.1.1 包

包是一个 Zip 容器，其中包含组成文档的组件 (也称为部件)，如 OPC 规范所定义的那样。其中许多元素也存在于 MS Office 2007 文件中。其中一些在所有 MS Office 应用程序之间通用，例如，文档属性、图表、样式表、超链接、图表和绘图。某些其他元素特定于每个应用程序，例如 Excel 中的工作表、PowerPoint 中的幻灯片和 Word 中的页眉和页脚。当用户使用 MS Office 2003 应用程序或以前版本的 MS Office 应用程序存储文档时，会将单个文件写入磁盘，用户可以轻松打开该文件。这个比喻在理解文档在实践中是如何存储、管理和共享方面很重要。由于 MS Office 2007 系统文件的各个部分包装在 Zip 容器中，因此文档仍是单个文件实例。使用单个包文件来表示单个文档实体，使用户能够获得与与以前版本的 MS Office 应用程序存储和打开 Office 2007 文件时相同的体验。也就是说，它们可以继续使用单个文件工作[7]。

3.1.2 部件

MS Office 文档的组成部分对应于一个包中的一个文件。例如，如果用户在 Excel 2007 文件上单击鼠标右键并选择将其解压缩，则他或她会看到几个文件，如“workbook.xml”文件和几个“sheetn.xml”文件。这些文件中的每一个都是包的一部分[7]。

每个部件可以有不同的内容类型。用于描述由 MS Office 2007 应用程序创建的数据的部件存储为 XML 文件。这些部件定义了 MS Office 2007 文件功能或对象，并符合相关的 XML 架构。所有表示为默认的 MS Office 2007 文件部件的 XML 架构都有完整的文档记录，并作为 MS Office 的一部分提供。

3.1.3 关系

关系是一种指定特定部件集合如何组合在一起以形成文档的方法。此方法指定源部件和目标资源之间的连接。关系存储在文档包的 XML 部件中 (例如，_rels\rels)[7]。

例如，通过关系，用户可以识别幻灯片和该幻灯片上显示的图像之间的连接。关系存储在包中的 XML 部件或“关系传输部件”中。如果源部件具有多个关系，则所有后续关系都将在同一个 XML 关系部件中列出。

关系文件在 MS Office XML 格式中起着重要作用。每个文档部分至少有一个关系引用。使用关系可以发现一个部分如何与另一个部分相关，而不需要查看

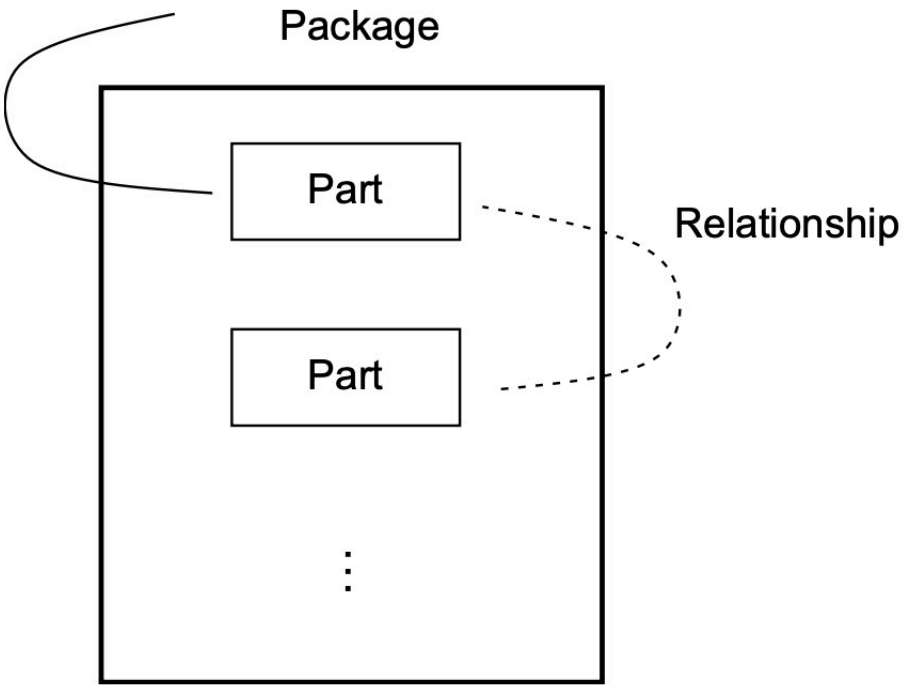

图 1: Office Open XML 格式

这些部分的内容。在部件中，对关系的所有引用都使用关系 ID 表示，这使得部件之间的所有连接都不受特定内容的架构的影响。关系文件包含基于文档开头部分的关系[7]。关系以以下格式定义：

```
<Relationship Id="ID" Type="relationshipType"
  Target="targetPart" (Targetmode="Internal/
  External")>
```

其中，ID 值可以是任何字符串，只要它在“.rels”文件中是唯一的。包需要有效的 XML 标识符。关系的 Type 将关系彼此区分开来，并指向定义 OOXML 格式类型的架构。此外，Target 指向包含关系目标的路径。Target 属性值取决于 TargetMode 的属性值。TargetMode 指示 Target 是描述包内还是包外的资源。有效值为“internal”和“external”。如果 TargetMode 为“internal”，则 TargetMode 将是相对于“父级”部件解释的相对引用。对于包关系，包实现者根据标识整个包资源的包统一资源标识符 (URI) 解析目标属性中的相对引用。如果 TargetMode 为“external”，则 TargetMode 可以是相对引用或 URI。如果目标属性是相对引用，则该引用相对于包的位置进行解析[6]。

OOXML 格式的关系可以是显式的，也可以是隐式的。在显式关系的情况下，使用关系标记的 ID 属性从源部件的 XML 引用资源。例如，只有当文档部件的

XML 显式引用了该超链接关系元素的 ID 属性值时，该文档部件才能与该超链接具有关系。由于该机制通常用于多种标记类型，因此可以从 OOXML 文档中提取显式关系，而无需预先了解标记语义。某些关系被定义为显性关系；所有其他关系都是隐性关系[6]。

为了说明这一点，这里提供了一个 OOXML 中的关系示例：

假设将特定的图像文件保存为 MS Word 2007 文件。如果随后重命名该图像并将其放置在 Media 文件夹中的“Document.xml”文件中，则会生成以下 XML 代码：

```
<w:pict>
    <v:image data r:Id="rId4"/>
</w:pict>
```

此 XML 代码表示该文档文件包含 ID 为“rId4”的图像。当应用程序遇到此代码时，它会尝试查找 ID 为“rId4”的目标。在关系文件中，列出了每个目标文件的 ID 信息。在这种情况下，关系文件中会出现以下 XML 代码：

```
<Relationships>
    <Relationship Id="rId4" Type = "/relationships/
        image" Target= "media/image1.jpeg"/>
</Relationships>
```

使用此代码，应用程序可以找到目标“rId4”并将图像插入到文档文件中。这是一个显式关系的简单例子。在隐式关系的情况下，会出现更复杂的联系。

4、数据隐藏

4.1 使用 OOXML 格式的数据隐藏

尽管 OPC 规范是为表示 OOXML 文档而设计的，但它也可以支持更多的应用程序。某些 OPC 功能的使用在 OOXML 文档中受到限制[6]。特别是存在的未知部分和未知关系，使隐藏数据成为可能。

4.1.1 未知部件

除了关系部分，OOXML 文档中不是有效关系目标的所有其他部分都被视为未知部分[6]。在读取文档时会忽略未知部分，并且在创建文档后可以（但没有必要）被丢弃的未知部分。换句话说，这些部件通常存在于任何给定的文件中，但无法被 MS Office 应用程序识别。

4.1.2 未知的关系

未在 ECMA376 标准中定义的任何关系都被视为未知关系。未知关系在 OOXML 文档中是有效的，只要它们符合 OPC 规范[6] 定义的关系标记指南。包含未知关系的文件正常打开，并且未知关系永远不会消失，即使用户以新名称保存这些文件。

4.1.3 使用未知部分和未知关系的数据隐藏

本节提供数据隐藏的具体示例。将数据插入载体存档文件是第一步，如图2所示。在解压缩 MS Word“.docx”文件后，将三个文件插入到解压缩的文件夹中：“mask.jpg”到主文件夹中，“BYE.mp3”到 word 文件夹中，以及“sysinders.zip”到媒体文件夹中。

图3显示了与提取的 MS Word 文档相关联的“[Content_Types].xml”文件，其中正常包中的所有文件扩展名都以粗体突出显示。由于包中每个部分的每个扩展名都必须在“[Content_Types].xml”中及其相关路径中声明，因此要隐藏其扩展名尚未出现在“[Content_Types].xml”中的数据，必须添加一些代码。在本例中，隐藏数据的扩展名为“jpg‘”、“mp3”和“zip”，它们不包括在“[Content_TYPE].xml”中。因此，有必要插入定义这些扩展及其关联路径的代码，如图4所示，添加的代码以粗体突出显示。

修改“[Content_Types].xml”后，“BYE.mp3”、“mask.jpg”和“sysinders.zip”文件将成为 MS Word 文档中的未知部分。包含这些隐藏文件的文档文件 (在本例中为 MS Word 2007 文件) 会正常打开。因此这三个文件是完全隐藏的。此时，如果定义了 MS Word 2007 文件和隐藏数据之间的关系，则发现隐藏数据变得更加困难。换句话说，即使当用户将数据存储在新文件中时，隐藏的数据仍然保留在文件中。消除隐藏数据的一种方法是从“[Content_Types].xml”中删除“jpg”、“mp3”和“zip”扩展名，并将文档另存为新文件。

图5显示了从 MS Office 文档中提取的具有“.rels”扩展名的正常关系文件。

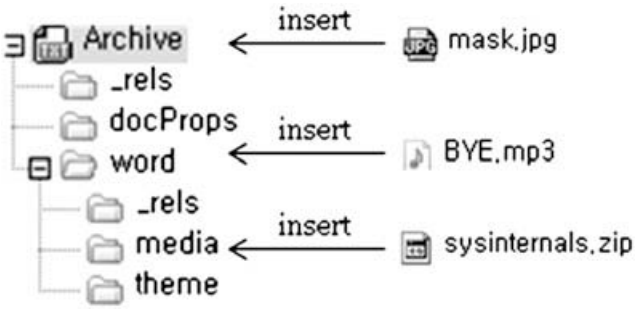

图 2: 数据隐藏示例

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Types xmlns="…">
<Override PartName="/word/footnotes.xml" ContentType="…"/>

<DefaultExtension="jpeg" ContentType="image/jpeg"/>
<DefaultExtension="rels" ContentType="application/vnd.openxmlformats-Package.relationship+xml"/>
<DefaultExtension="xml" ContentType="application/xml"/>

<Override PartName="/word/document.xml" ContentType="…"/>
                    ……
</Types>
```

图 3: [Content_Types].xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Types xmlns="…">
<Override PartName="/word/footnotes.xml" ContentType="…"/>

<DefaultExtension="jpeg" ContentType="image/jpeg"/>
<DefaultExtension="rels" ContentType="application/vnd.openxmlformats-Package.relationship+xml"/>
<DefaultExtension="xml" ContentType="application/xml"/>
<DefaultExtension="zip" ContentType="application/zip"/>
<DefaultExtension="mp3" ContentType="application/mp3"/>
<DefaultExtension="jpg" ContentType="application/jpg"/>

<Override PartName="/word/document.xml" ContentType="…"/>
                    ……
</Types>
```

图 4: 修改后 [Content_types].xml

在此关系部件中，ID、Type 和 Target ID 等属性必须是唯一的，并且 ID 连接文档和目标文件。文档部分的 XML 显式引用 ID 属性值[6]

图6示出了涉及隐藏数据的关系。例如，“BYE.mp3”的关系如下：“BYE.mp3”的 ID(在“Word”文件夹中) 是“rId102”，该 ID 的类型是“http://schemas.OpenXMLformats.org/officeDocument/2006/Relationship/c”。此时，如果用户将 Type 更改为 OOXML 规范中规定的类型，则在打开文档文件时会出现一条警告。因此，当设置类型时，用户需要使用 OOXML 规范中不存在的值 (例如，如图 6 所示的“a”，“b”，“c”)。

修改“.rels”文件和“[Content_Types].xml”文件后，用户可以正常打开文档，而不会出现任何警告。此外，如果用户使用 MS Word 2007 应用程序修改文档文件

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="…">
<Relationship Id="rId3" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/extended-properties"
Target="docProps/app.xml"/>
<Relationship Id="rId2" Type="http://schemas.openxmlformats.org/package/2006/relationships/metadata/core-properties"
Target="docProps/core.xml"/>
<Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument"
Target="word/document.xml"/>
</Relationships>
```

图 5: 关系文件

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="...">
<Relationship Id="rId3" Type="..." Target="docProps/app.xml"/>
<Relationship Id="rId2" Type="..." Target="docProps/core.xml"/>
<Relationship Id="rId1" Type="..." Target="word/document.xml"/>

<Relationship Id="rId100" Type="http://schemas.openxmlformats.org/officeDocument/2006/Relationships/a"
Target="word/media/sysinternals.zip"/>
<Relationship Id="rId101" Type="http://schemas.openxmlformats.org/officeDocument/2006/Relationships/b"
Target="mask.jpg"/>
<Relationship Id="rId102" Type="http://schemas.openxmlformats.org/officeDocument/2006/Relationships/c"
Target="word/BYE.mp3"/>

</Relationship>
```

图 6: 已修改关系文件

并再次保存，隐藏的数据将保留在文件中。由于 MS Office 2007 应用程序不检查未知部件和未知关系，因此可以使用它们来隐藏数据。

这种数据隐藏方法是 OOXML 中显式关系的自然结果。文档的主源部分依赖于每个组件的唯一 ID 值和关系部分中的关联信息来定位内容，如嵌入的文件。此隐藏过程的关键点是，为新目标分配新 ID 会导致该目标被 MS Office 应用程序忽略。因为新 ID 在关系部分中没有被引用，所以主源部分不知道新内容，并且隐藏的数据不会显示在屏幕上。然而，隐藏的数据不会被 MS Office 消除，因为这些隐藏的数据有 ID。MS Office 2007 应用程序确实有一个功能来检测“unnown.xml”，如图7所示。然而，如果有人使用本文描述的方法隐藏数据，该函数不会检测隐藏的数据。

4.1.4 使用注释隐藏数据

还可以使用 XML 注释在 MS Office 文档中隐藏数据。MS Office 2007 应用程序创建的普通文件中的 XML 部件没有 XML 注释。这种方法非常简单，但这种隐藏方法也适用于隐藏消息。

5、隐藏数据的检测

本节提供了使用本文中详细介绍的技术检测已隐藏数据的算法和伪代码。

如上所述，在 MS Office 2007 文件中使用未知部件和未知关系，可以隐藏文件中的数据。未知部件和未知关系的属性也可用于检测隐藏数据。图8示出了检测算法的概要。在下一段中，图 10-12 中的伪代码提供了对该算法的详细解释。

在加载了包含隐藏数据的文件之后，“Detector”检测程序 (使用 C# 开发) 收集了文件中的每个部件和每个关系的信息。有关部件的信息存储在“SetFileInfoList”结构中：

```
1 struct FileInfo {
```

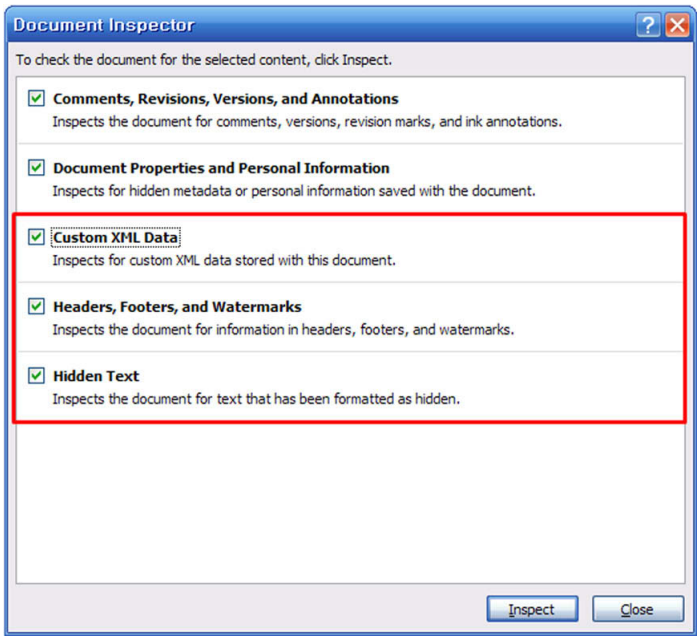

图 7: 文档检查器

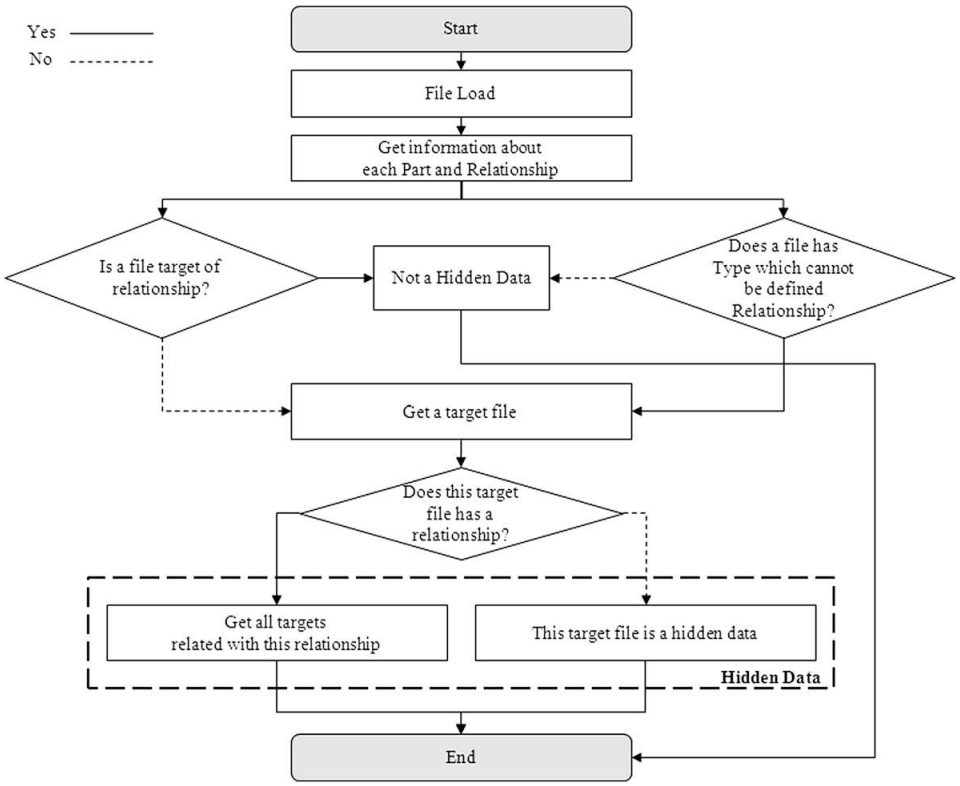

图 8: 隐藏数据检测算法

```
2       public bool IsUnknownPart;
3       public bool IsUnknownRel;
4       public bool HasComment;
5       public string DirPath;
6       public string FileName;
7       public string comment;
8   }
```

“IsUnnownPart”是一个变量，它指示某个特定部件是否未知。换句话说，这个变量表明每个部分是否被定义了[6]。不具有未知关系的部件可以简单地通过“IsUnnownPart”变量的值来检测。“IsUnnownRel”是一个变量，它指示某个特定部件是否为未知关系。如果输入部分具有未定义的类型，则此检测算法确定该部分具有未知关系。当输入部分有注释时，“HasComment”为“TRUE”，否则为“FALSE”。“DirPath”是输入部件的绝对路径，“filename”是部件的名称。如果检测到注释，则将其存储在“comment”变量中。关系信息被以同样的方式收集和处理。

在完成此数据收集工作后，将执行以下两个步骤：

1. 检查每个文件是否是关系文件中陈述的特定关系的目标。如果不是，则由隐藏数据组成。如果这些隐藏数据具有关系文件，则此关系文件中的所有目标都由隐藏数据组成。

2. 检查每个文件是否具有 OOXML 规范中未描述的异常内容类型。如果是，则该文件包含隐藏数据。如果此文件有关系文件，则此关系文件中的所有目标也包含隐藏数据。

现在将描述该算法的伪代码，并详细介绍检测算法。

图9示出了检测处理的顺序。在加载包文件 (例如，MS Word 2007 文件) 之后，检测程序提取包并收集关于每个部件和关系部件的信息。

图10、11、12示出了如何在文档文件中检测未知部分和未知关系。检测过程分三步执行：

首先，程序检查每个部件以确定它是否是关系部件的目标。如果不是，则该部分是未知部分，也就是说，它是隐藏数据。此外，如果部件在其子文件夹中有关系部件 (其中子文件夹的名称为“_rels”)，则关系部件和关系部件的所有目标都是隐藏数据。这个过程递归地持续进行。

其次，程序必须检查每个关系部分的类型，以确定它是否在 OOXML 规范中定义。如果 OOXML 规范中没有定义关系部分的类型，则该关系部分是未知关系，也就是说，它是隐藏数据。例如，在“.docx”文件 (MS Word 2007 文件) 的情况下，关系部分的类型包括：

```
Detect
{
    Detect.FileLoad
    Detect.ExtractZip

    Detect.SetFileInfoList
    Detect.SetRelationshipList

    Detect.FindUnknownParts
    Detect.FindUnknownRel_from_Root ("_rels\\.rels")
    Detect.FindUnknownRel_from_UnknownPart
    Detect.FindXMLComments
}
```

图 9: 伪代码 1

```
Detect.FindUnknownParts
{
    foreach (File in Package)
    {
        Get FileInfo of this file in FileInfoList
        Assume that this file is Unknown Part

        foreach (Relationships in RelationshipList)
        {
            foreach (Relationship in Relationships)
            {
                if (path of this file == path of Relationship.target )
                {
                    This file is not Unknown Part
                    break
                }
            }
        }

        if (This file is Unknown Part)
        {
            FileInfo.IsUnknowsPart = TRUE
        }
    }
}
```

图 10: 伪代码 2

```
Detect.FindUnknownRel_from_Root (rels_path)
{
    foreach (Relatioinship file in Package)
    {
        Get Relationships of this relationship file in RelationshipList

        if (Relationships.rels_path != rels_path)  break

        foreach (Relationship in Relationships)
        {
            if (Type and Target of Relationship are not in Known_Relationships_in_MS_Word_2007)
            {
                This relationship is Unknown Relationship

                Get FileInfo of Relationship.target in FileInfoList
                FileInfo.IsUnknowsRel = TRUE

                if (Relationship.target has sub relationship file)
                {
                    FindUnknownRel_from_Root (sub_rels_path)
                }
            }
            else
            {
                This relationship is not Unknown Relationship
            }
        }
    }
}
```

图 11: 伪代码 3

```
Detect.FindUnknownRel_from_UnknownPart
{
    foreach (FileInfo in FileInfoList)
    {
        if (FileInfo.IsUnknownPart == TRUE)
        {
            if (relationship file of this file is exist)
            {
                Relationships in this relationship file are Unknown Relationship
                FindUnknownRel_from_Root (path of this relationship file)
            }
        }
    }
}
```

图 12: 伪代码 4

- http://schemas.OpenXMLformats.org/officeDocument/2006/ relationships/office-Document
- http://schemas.OpenXMLformats.org/officeDocument/2006/elationships/extend-edproperties
- http://schemas.OpenXMLformats.org/package/2006/relationships/metadata/core-properties
- http://schemas.OpenXMLformats.org/package/2006/relationships/digitalsignature/origin
- http://schemas.OpenXMLformats.org/package/2006/relationships/metadata/thumb-nail

诸若此类。这些类型在关系部件的 OOXML 规范中。

图13、14、15展示了隐藏数据的检测结果。在前文使用的示例中，隐藏的数据是“mask.jpg”、“BYE.mp3”和“sysinders.zip”，这些都在图5中进行了表示。隐藏的数据由不同于正常数据的图标表示。该程序已验证“mask.jpg”隐藏在根文件夹中，“BYE.mp3”隐藏在“word”文件夹中，“sysinternals.zip”隐藏在“”word\media“”文件夹中。

最后，程序必须检查 XML 格式的每个部分，以确定它是否有 XML 注释。通常由 MS Office 2007 应用程序创建的所有 XML 格式的部件都没有 XML 注释。因此，如果在 MS Office 2007 文件中存在 XML 注释，则可以将其视为用户插入的隐藏数据。图16显示了该查询的结果，包括 XML 注释检测算法。

6、总结

除了在将 MS Office 2007 文档从 OpenXML 格式转换为复合格式 (由 MS Office 2003 版和更早版本使用) 时发生数据丢失之外，这里概述的数据隐藏方法

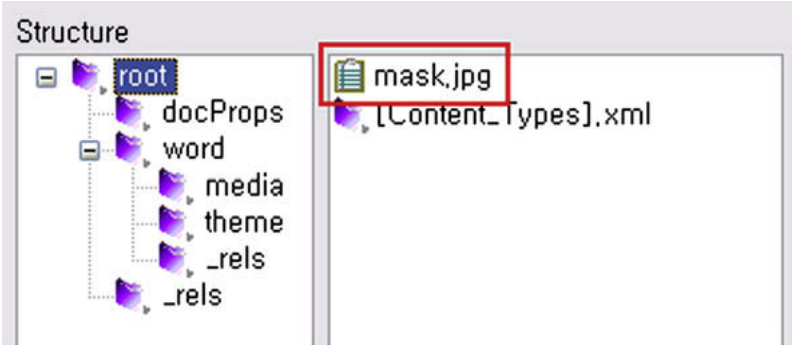

图 13: 检测结果 1

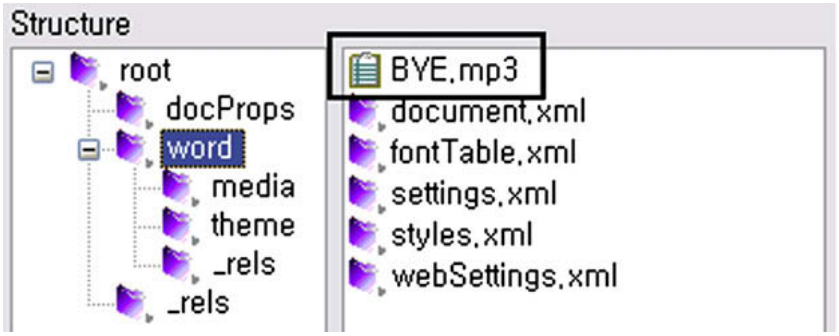

图 14: 检测结果 2

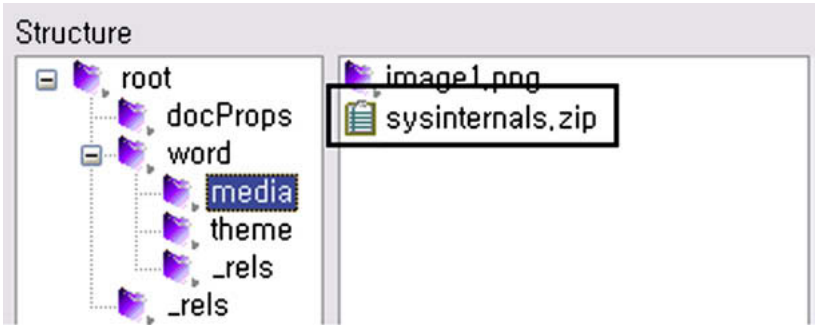

图 15: 检测结果 3

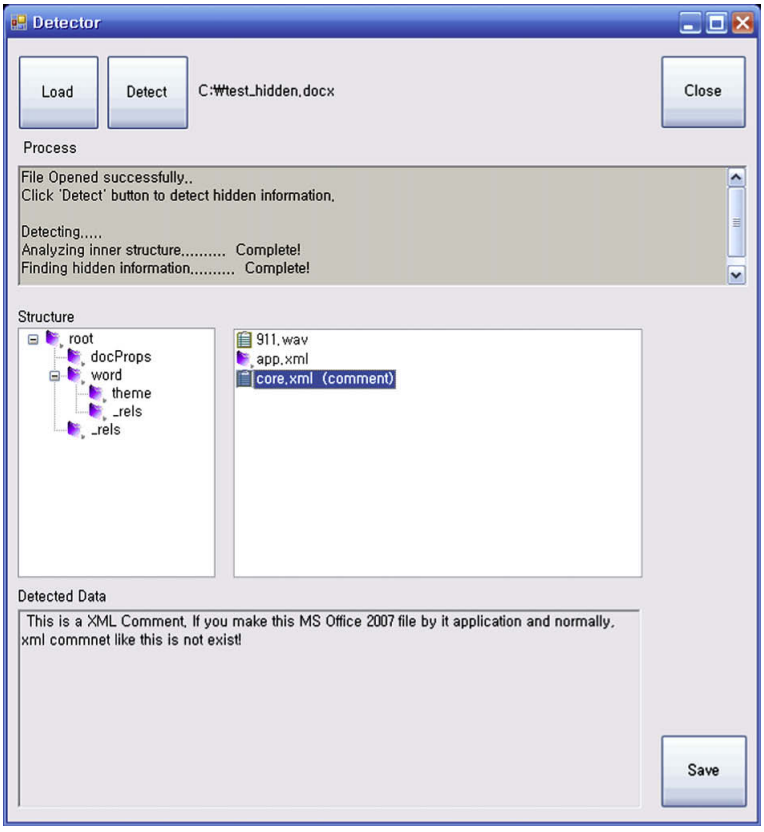

图 16: 使用“Detector”的最终检测结果

非常强大，特别是在 MS Office 2007 应用程序支持的任何功能都无法检测到隐藏数据的情况下。Microsoft Office Isolated Conversion Environment(MOICE)[8] 使用“Document.xml”以及与“Document.xml”相关的部件和关系从 2003 年或更早版本的 MS Office 应用程序中读取数据。因此，当在较旧的应用程序中使用 MOICE 打开 MS Office 2007 文件时，与“Document.xml”无关的隐藏数据，例如通过此处提供的方法 (不会修改“Document.xml”) 生成的数据，在复合重新格式化或在 2007 版本的应用程序中查看后消失。换句话说，MOICE 不能检查未使用的数据，如未知部件或未知关系，也不能进行任何验证测试。相反，它只能转换与“Document.xml”相关的数据。因此，可以得出结论，除了本文描述的检测算法之外，没有其他方法可以检测这里给出的数据隐藏。

考虑到隐写术将信息隐藏在封面数据中或某些隐藏数据正在被传输的事实，如果有可能将数据插入到文件中而不在文件打开时引起通知，则这种隐藏数据的方法对于隐写术至关重要。这项研究的结果得出的结论是，在 MS Office 2007 文件中隐藏数据是可能的，并且刚刚展示了如何检测此类隐藏数据。本文描述了如何创建工具来检测 OOXML 格式中可能隐藏的数据。

从计算机取证的角度来看，确认任何未被特定应用程序检查的数据的存在是非常重要的。在计算机取证调查中，调查人员可能会假设电子文档文件中的所有数据都可以通过他们的应用程序进行检查。然而，仅通过相关应用程序调查电子文档文件是不够的，因为大多数应用程序无法检查或检测到的数据仍可能存在于文件中。

在本文中，大部分描述都集中在 MS Word 2007 文件上。但是，如果 MS Office PowerPoint 和 Excel 2007 文件包含任何隐藏数据，也可以使用相同的算法检测出这些数据。

在未来，作者将继续寻找在文件中隐藏数据的其他方法，并创建工具来检测文件中的隐藏数据。

参考文献

[1] PROVOS N, HONEYMAN P. Hide and seek: An introduction to steganography[J]. IEEE security & privacy, 2003, 1(3) : 32 – 44.

[2] CASTIGLIONE A, DE SANTIS A, SORIENTE C. Taking advantages of a disadvantage: Digital forensics and steganography using document metadata[J]. Journal of Systems and Software, 2007, 80(5) : 750 – 764.

[3] RENTZ D. Microsoft compound document file format[J]. Internet]. Available: http://www. openoffice. org. zaxyproxy. com, 2007.

[4] INOUE S, MAKINO K, MURASE I, et al. A proposal on information hiding methods using XML[C] // The 1st Workshop on NLP and XML. 2001 : 707 – 710.

[5] GOLDFARB C F, PRESCOD P. XML handbook[M]. [S.l.] : Prentice Hall PTR, 2000.

[6] ECMA. ECMA-376, Office Open XML File Formats Part 1[J]. ECMA- 376. 1st ed, 2006.

[7] RICE F. Open XML file formats[J]. Microsoft Corporation, 2006.

[8] MICROSOFT. Description of the microsoft office isolated conversion environment update for the compatibility pack for word, excel, and powerpoint 2007 file formats. [J]. Microsoft Corporation, 2007.