IMPLEMENTATION OF INDEPENDENT MODULE FOR VBA PROGRAM

Seung Ju Jang
College of ICT Engineering, Dong-Eui University
Korea
sjjang@deu.ac.kr

ABSTRACT

In this paper, we design a module to prevent cracks in the password set in Excel VBA. After the Excel VBA program, set the password to block access to the source code for critical code. Design the module so that access to the source code can be blocked by changing the password for the password set in Excel VBA. Design the Excel VBA program file separation structure to block access to the password set in VBA. We implemented the proposed design in this paper. Experiments were performed on actual implementation of the Excel file. As a result of experiments, it was confirmed that Excel VBA password access can be blocked.

Keywords: Excel VBA, Implementation VBA Program File Separation, Excel, Experiment.

1. INTRODUCTION

This paper proposes a structure to protect the Excel VBA program source code that we frequently use in everyday life. Excel VBA (Visual Basic for Applications) is used to more easily and conveniently handle repetitive work on data in complex Excel sheets. VBA program functionality is included in Excel basically.

VBA programs are used in many areas. VBA programs can be used to automatically process large amounts of data in Excel. Excel VBA is available in many areas. Excel VBA is used in various fields such as customer management, student performance management, budget measurement and forecast management, data processing and analysis related to science, and chart management using data.

The advantage of the Excel VBA program is that it can be used if you have only the Excel program installed, without having to install the compiler like a normal program language. Because of this convenience, many people are using Excel VBA programs. That is, the VBA code is run as a host application program rather than as a stand-alone program.
The Excel VBA program structure supports the OLE (Object Linking and Embedding) structure of existing MS Office programs. The inside of the OLE file format has the same file system structure as one small hard disk. The concept of folders and storage and streams is inside the OLE file.
This paper implements a separation module for safe use of VBA programs in Excel. This allows you to safeguard and manage VBA programs in Excel.

In this paper, we describe related works on Chapter 2, VBA program separation structure implementation on Chapter 3, experiment on Chapter 4, and conclusion on Chapter 5.

2. RELATED RESESRCHES

Spreadsheets are often used as user-friendly tasks. We use VBA programs to perform data processing in Excel more smoothly and quickly. Spreadsheet macro programming has changed and developed tremendously over the last 12 years. Spreadsheets have evolved into the current Excel format through various developments and changes [1, 3, 5].

When Excel 5 came out in 1994, it was very popular. Like previous versions, spreadsheets have been evaluated and used as having various features. Excel 5 is the first version of VBA to be introduced. Since then, the version of Excel 95 has been released, and improvements in functionality and performance have been made. The Excel 95 version also has some improvements in VBA functionality.

In recent years, there has been an increase in cases of Excel file security violation using VBA macros. The VBA macro facility is convenient to use. On the other hand, it is easy for an attacker to attack [6, 7].

Easy access for general users could be disadvantages to spreadsheet applications. It is convenient for anyone to easily access the application in the spreadsheet, but it can cause problems in the management of the application.

In many cases, a programming language is used to process data in a spreadsheet. The ability to effectively protect source code for these programming languages is important. And it is also important to guarantee data integrity within the spreadsheet. [1, 2, 8, 9].

3. MODULE WITHOUT VBA PROGRAM

The VBA program provides a variety of programs that can conveniently handle Excel data. Unlike existing programs, VBA programs do not require a compiler, and can be used by anyone who has Excel installed.

A VBA program does not generate object code in an interpreter manner. Therefore, it runs in the form of executing the program using the source code. Running in this way, it is not easy to keep the VBA source code from being exposed.

Program source code is an important intellectual property. Therefore, it is important to manage and keep them from exposure. The VBA program has an OLE (Object Linking and Embedding) structure. OLE files (or COM Structured Storage) formats have been used as document formats for Word, Excel, and PowerPoint in the Microsoft Office. The inside of the OLE file format has the same file system structure as one small hard disk. Therefore, the inside of an OLE file has a concept of a folder and a file. An OLE file is largely divided into two blocks. They are a header block and a data block. The header block has a size of 512 bytes, and the data block has 512 bytes or more. The header block contains the main information of the whole OLE file, and the data block has the information below.

In particular, the Excel file has a compressed file structure. The internal structure of the compressed file is shown in Figure 1.

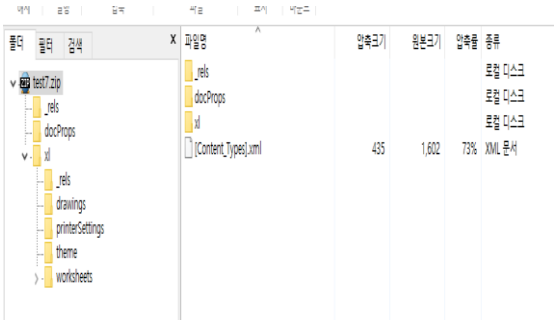


Figure 1 Internal Structure after Decompressing Excel File

The internal structure of the file after decompression is shown in the above Figure 1. The configuration in the file consists of folders _rels, docPros, and xl.

•_rels (Folder): "/_rels/.rels" is an XML file that stores the relationship at the start package level.

• docProps (Folder): This folder contains two files, app.xml and core.xml. The files in this folder have some metadata. The core.xml file contains information about the author, the modifier, the data created, and the date stored. On the other hand, app.xml has information about the contents of the file.

•xl (Folder): This folder contains most of the information. Basically, there are subfolders of "_rels", "theme", and "worksheets" and two additional XML files, styles.xml and workbook.xml. If the VBA program is included, the vbaProject.bin file is included.

• [Content_Types] .xml: The XML file [Content_Types] is the only file at this level. It contains reference information for the XML files contained in the above folders.

Excel provides VBA program functionality for convenient data processing. It is stored and managed as data in an Excel file. As mentioned earlier, when you unzip the file, most of the information is in the xl folder. Such a structure is very vulnerable to security.
In this paper, we propose a security enhancement structure in VBA program operation structure. For VBA program source code security, we propose the following structure that is different from the existing one.

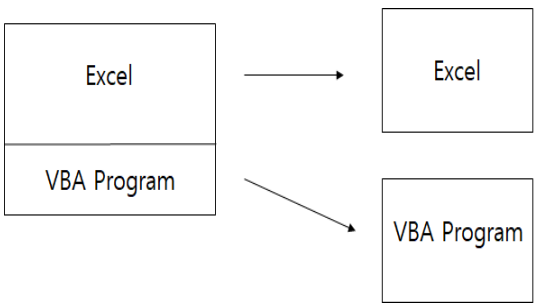


Figure 2 Structure of VBA Independent Modules

The structure for separating VBA program into independent modules from excel files is shown in Figure 2. Existing Excel file contains VBA program. In this paper, we want to separate VBA program into independent modules. VBA program modules separated from Excel will be separated from Excel into independent modules.

To separate the VBA program module from the Excel file, you need to convert the Excel file to a compressed file and decompress it. When it is unzipped, there is a vbaproject.bin file that stores the VBA program source code in the xl folder. The vbaproject.bin file will be detached from the archive. If this file is separated from the Excel file, the structure of Excel will be the same as without VBA program.

By separating vbaproject.bin VBA program source code module from Excel file, it is impossible to use VBA programs.

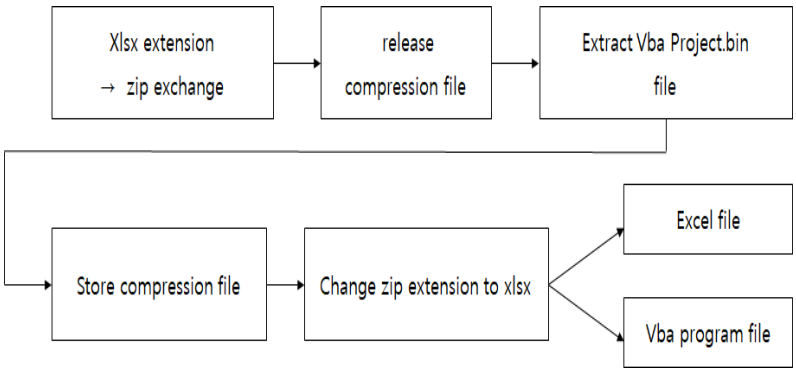


Figure 3 VBA Separation as Independent Modules.

To separate the VBA program from the Excel file, it converts the extension xlsx of the Excel file to zip. It releases the compressed file for an Excel file whose extension has been converted to zip. When unzipped the file, only the vbaproject.bin file will be extracted. It can avoid security risks by isolating the vbaproject.bin file from the Excel file.
After extracting the vbaproject.bin file, it saves the compressed file newly. It converts the extension of the newly saved zip file to xlsx. Then the Excel file will be saved as without the VBA program module.

It inserts a guiding module for the use of vbaproject.bin instead of the vbaproject.bin file, indicating that the VBA program cannot be run if there is no permission.

If the user can not use vbaproject.bin module, he / she will be informed that it can not be used. On the other hand, if the user can use the vbaproject.bin module, he / she can use the module connection guide.
For users who can use the vbaproject.bin module, the vbaproject.bin file can be added to the Excel file.

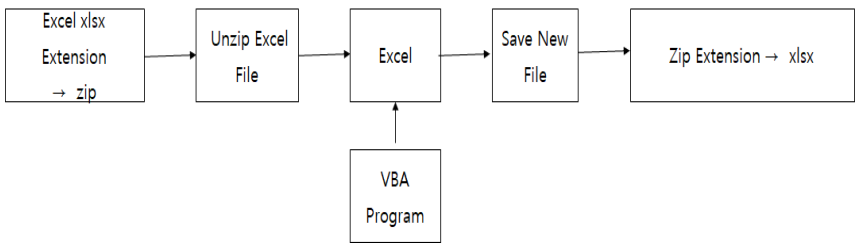


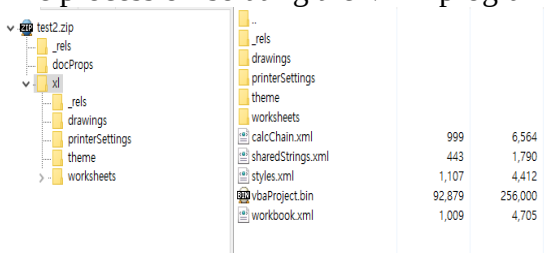
Figure 4 Add VBA Independent Module into Excel

The process of adding a VBA program module to an Excel file is shown in Figure 4. To add a VBA program module, it changes the extension of the Excel file to zip. Then it decompresses the zip file. After decompressing, it inserts the vbaproject.bin file into the uncompressed Excel file. After adding vbaproject.bin to the excel file, it saves the excel file. It changes the

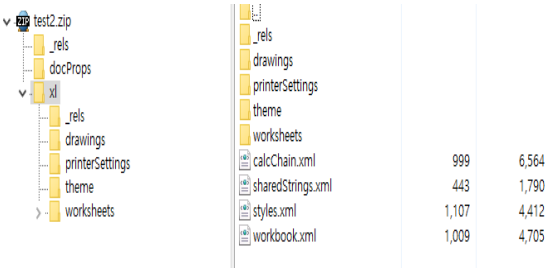
extension of the saved Excel file from zip to xlsx.

4. EXPERIMENT

In this paper, we implemented and experimented VBA program independent module system. The environment for implementation and experiment use Excel 2013 version. We designed and implemented VBA program independent module using Excel 2013 version.
We performed experiments on the implementation of the VBA program independent module system proposed in this paper. In the experiment, we first isolate the VBA program module. The process of isolating the VBA program module is shown Figure 5.



(a) Unzip the file

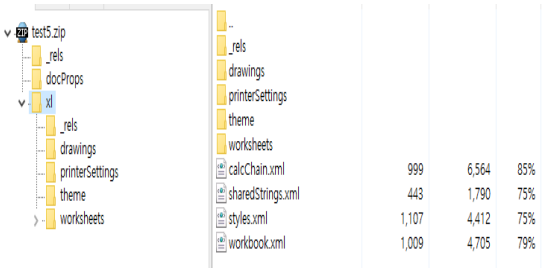


(b) Contents of the file after extracting vbaproject.bin

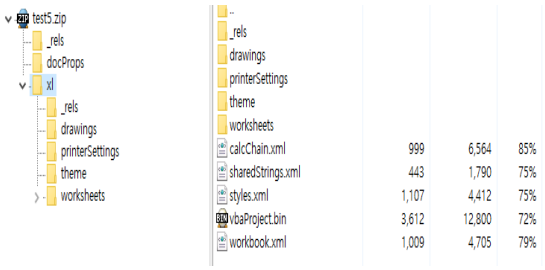
Figure 5 Separating into a VBA Independent Module

Figure 5 shows the process of isolating a VBA program module from an Excel file. To separate the VBA program module from the excel file, we change it to zip extension and decompress it. After decompressing, we extract vbaproject.bin file and then save the zip file. we convert the extension of the compressed archive to xlsx.

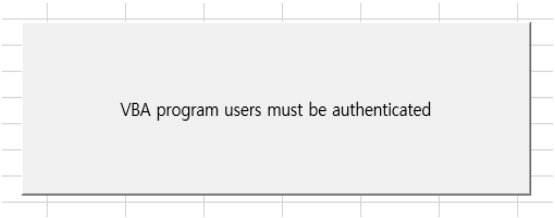
Once you have removed the VBA program module, there are no more VBA program modules in the Excel file. If you want to use the VBA program, you will install the user guide module instead. Figure 6 shows the installation process of VBA program module.



(a) Case without Guide Module



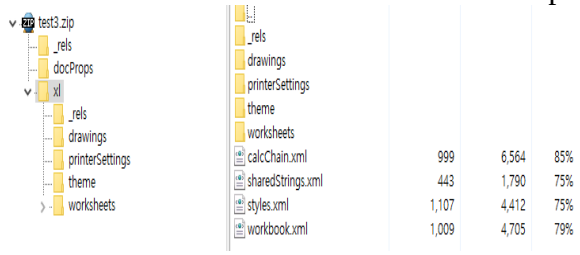
(b) Case with Guide Module



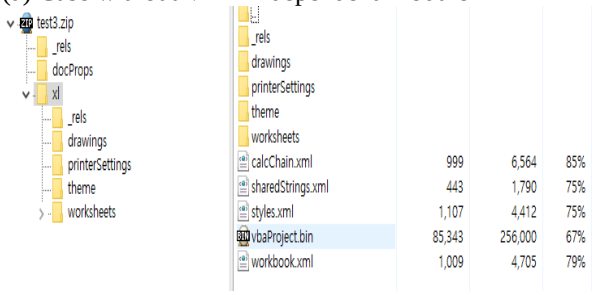
(c) Guide Module Screen

Figure 6 Equipped with Guide Module Function

Figure 6 shows the installation process of the guide module for using the VBA program. With this module, users can get guidance on using the VBA program. It is a function that can be informed about how to use VBA program. If he/she is an authorized user who can use the VBA program module, he/she needs to install the VBA program module in Excel. This allows the user to install and use the VBA program.



(a) Case without VBA Independent Module



(b) Case with VBA Independent Module

Figure 7 VBA Independent Module Mounting

Figure 7 shows the process of loading a VBA program module. If the user is an authorized user, he/she can use the VBA program in Excel. To install the VBA program module, it converts the extension of the Excel file to zip. It extracts the compressed file. Then, the guide module program is removed. Then, it inserts the VBA program module. After inserting, it saves the file. It converts the extension of the compressed archive to xlsx.

5. CONCLUSION

This paper proposes a system structure for secure protection of VBA program source code

used for data processing in Excel. For the safe use of VBA programs, we propose an independent structure from Excel. We separate VBA program module from Excel file and add VBA program module to excel file only for authorized user. This makes VBA program modules more secure.

In order to separate VBA program modules from Excel, it follows the fixed procedure. It changes the extension of the Excel file to zip archive. Then, the compressed file is released. It extracts the VBA program module from the compressed file. It then saves the zip file. It converts the extension of the compressed file to xlsx. After this process, the VBA program will be saved as Excel file.

After saving it as an Excel file without a VBA program, it adds a module to guide users. Users can get instructions on using the VBA program. If the user is an authorized user, he/she inserts the VBA program module so that it can be used.

We performed experiments on the VBA independent module structure proposed in this paper. As a result of the experiment, it was confirmed that the extraction and insertion of VBA program is normally performed well.

REFERENCES

[1] Jang, Seung Ju, "DESIGN OF THE EXCEL VBA PASSWORD CHANGING MODULE", European Journal of Engineering and Technology, Vol. 6 No. 2, 2018.
[2] Hong, Sug Myung, "Detection of Obfuscated VBA Macro in Microsoft Office Documents", Department of Computer and Radio Communications, Engineering, Korea University, Thesis for the Degree of Master, May 2017.
[3] Suryam Tiwari, Vijay Shrivastav, "Development of Design Spreadsheet Tool for R.C.C. Beam Design using V.B.A., IJIRST –International Journal for Innovative Research in Science & Technology| Volume 4, Issue 11, April 2018
[4] Blessing O. Abisoye, "Simulation of Electric Power Plant Performance Using Excel®-VBA", *Information Engineering and Electronic Business*, 8-14, 2018, 3.
[5] Michael R. Bartolacci, Larry J. LeBlanc, Yasanur Kayikci, Thomas A. Grossman, Optimization Modeling for Logistics: Options and Implementations, *Journal of Business Logistics*, Volume 33, Issue 2, June, 2012 pp.118–127, 13 June, 2012.
[6] J Pichitlamken Email author, S Kajkamhaeng, P Uthayopas, R Kaewpuang, "High performance spreadsheet simulation on a desktop grid", *Journal of Simulation*, Volume 5, Issue 4, pp266–278, November, 2011.
[7] Jeong-Keun Lee, Jin-Shik Yoon, Hee-Jun Chung, "A development of an automatic design module of motor analysis using VBA and OPERA-3d", Proceedings of The Korean Institute of Electrical Engineers, pp. 692-693. 2014.7
[8] Wook Hee Koh, "Study of Dynamical Characteristics of Nonlinear Pendulum using Excel VBA", Journal of The Korean Society for School Science, pp. 60-71, 2015.2,
[9] Excel_VBA_Macro_Programming-Richard_Shepherd-Mc_Graw_Hill._MS_Excel_2016.