An Effective Security Model for Removing Distrustful Macros from Office Documents

Somchai Chatvichienchai
Department of Information and Media Studies
University of Nagasaki, Faculty of Global Communication
Nagasaki, Japan

***Abstract*—Macros of Microsoft Office documents are used in many organizations to improve efficiency of operations on the office documents. However, when created with malicious intentions, macros contain viruses that steal sensitive information of users or cause damage to files and systems of users. Although antivirus programs can remove the office documents containing macro viruses from users' systems, it can't detect new viruses that are not registered in its virus definition databases. Although users pay attention not to activate macros of the office documents come from unknown senders, macro virus creators have successfully used social engineering techniques that lead users of Microsoft office to run the infected macros. Therefore, viruses in macros of the office documents are still dangerous threats for the organizations. The objective of this paper is to propose an effective security model that solves this problem. This proposed model employs digital signature technology for examining trusted macro creators and detecting uncertified modification of macros of Microsoft Office documents.**

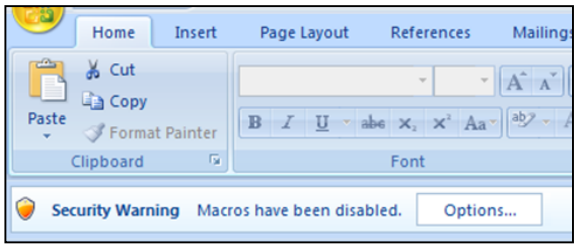
Keywords*— *Documents; macro; virus; XML; digital signature; certificates

I. INTRODUCTION

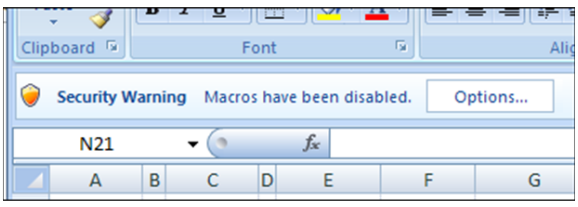
Microsoft Office Suite [11] is a collection of applications which are popularly used in many organizations. An outstanding feature of Microsoft Office is macro [3] feature. A macro is a set of computer instructions for Microsoft Office programs to do repetitive document production tasks, streamline cumbersome tasks, or the creation of documents that users use regularly. Therefore macros are useful in improving efficiency of office work of organizations. Users can easily create a macro by using the Visual Basic Editor of Microsoft Visual Basic for Applications (VBA) [10] to write their own macro, or to copy all or part of a macro to a new macro.

However, when created with malicious intentions, a macro can contain a virus that is a dangerous program, which steals sensitive user information and/or causes harm to files and systems of users. Macro viruses [16] are classified to be application-specific since they infect macro utilities that accompany Microsoft Word, Excel and PowerPoint. Therefore, macros viruses can be detected in files with extensions common to macro capable applications such as file with file extensions: *.doc*, *.xls*, and *.ppt* of Microsoft Office 97-2003, and *.docm*, *.xlsm* and *.pptm* of Microsoft Office 2007-2013. Macro viruses can be spread through e-mail attachments, USB flash drives, networks, and the Internet and is notoriously difficult to detect. A well-known example of macro virus in March 1999 was the Melissa virus [21]. In the beginning of 2013, a resurgence of malicious VBA macros has been observed [19]. Some of them are reported by the National Cyber Security Center (NCSC) of the Dutch CERT. The NCSC reported that many organizations have macros enabled - for instance to support corporate house styles [14] to ensure a consistent and professional look in documents and publications.

In Microsoft Office Suite starting from Office 2007, the ability of executing macros of is disabled by default. When users open a Microsoft Word or Excel File embedded with a macro, they are warned on the Word or Excel menu bar about the situation that macros have been disabled, as shown in Fig. 1. However, macro virus creators have already overcome this obstacle by using simple social engineering techniques to lead users to allow the macros to run. For instance, malicious persons send unsolicited emails with attached invoice of Microsoft Word documents. The rather vague messages of emails are obviously designed to trick recipients into opening the attached files in the hope of getting more information. However, when users attempt to open the attached documents, they will receive a message that asks whether they wish to enable macros to see the content. If they enable macros as requested, a blank Word document will be opened and displayed. Other versions may display some content that



(a)



(b)

Fig. 1. Examples of security warning screenshots of (a) Microsoft Word and (b) Microsoft Excel.

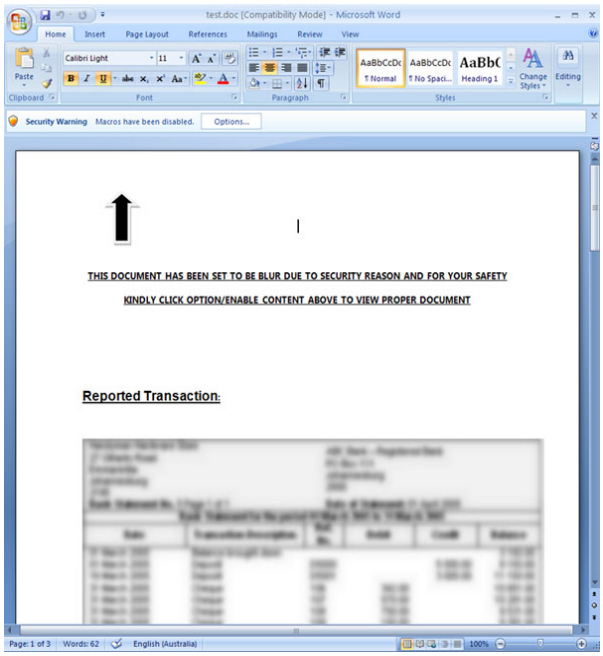


Fig. 2. An example of a screenshot of Microsoft Word document requesting users to enable macro execution.

requests that users should enable macros to gain access to the full document and to unblur document content (See Fig. 2), or get a password to unlock the remainder of the document. After enabling macros, malicious macros inside the attached document will download other viruses to the users' computers. This mechanism is being used by some advanced persistent threats (APT) which is a set of continuous computer hacking processes, often controlled by humans targeting a specific entity. Many APT attacks start with someone at the targeted organization receiving an email with an infected office document attached to it.

Based on the above observation, it can be concluded that damage of macro virus cannot be completely solved by antivirus programs and user caution. The objective of this paper is to propose an effective security model that removes distrustful macros from Microsoft office documents. In the proposed model, a trusted macro is a macro sent with its digital signature [17] and macro creator's certificate which has been already registered in certificate database of the document recipient. A macro without correct digital signature and macro creator's certificate is justified to be a distrustful macro. The goal of employing digital signature technology is to certify macro creators and detecting modification of the macro after macro digital signing. The proposed security model is applied for macros of Microsoft Office 2007-2013.

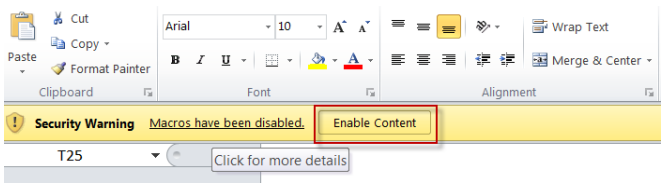


Fig.3. An example of a screenshot of security warning when a user opened an Excel file with a macro whose security level is set to the default setting.

The rest of the paper is organized as follows. In Section 2, basic concepts of hash function, digital signature and digital certificates are given. Section 3 discusses the drawbacks of current solutions of macro viruses and some research of malware detection. In Section 4, the proposed security model is presented. Section 5 explains in depth about the technique of handling distrustful macro from office documents without executing Microsoft Office Suite. Finally, the last section concludes this paper.

II. BASIC CONCEPTS

A. *Hash Functions*

A hash function usually means a function used to map digital data of arbitrary size to digital data of fixed size. The input data of hash function is often called the message, and the output value of the hash function is often called the message digest or the digest. The ideal hash function has four following properties:

- It is easy to compute the hash value for any given message.
- It is infeasible to generate a message from its message digest.
- It is infeasible to modify a message without changing the message digest.
- It is infeasible to find two different messages with the same message digest.

Hash functions have many information security applications, notably in digital signatures and message authentication. There are a number of different hash functions in use including Rivest's MD5 [20], which reduces a file to a 128-bit message digest, and NIST's Secure Hash Algorithm (SHA3) [4], which creates a 224-bit, 256-bit, 384-bit and 512-bit message digests.

B. *Digital Signatures*

A digital signature is a mathematical scheme that presents the authenticity of a digital message. A valid digital signature allows a message recipient to confirm three following items.

- Sender Authentication meaning that the message was created by a known sender,
- Sender Non-Repudiation meaning that the sender cannot deny having sent the message, and
- Message Integrity meaning that the message was not altered in transit.

Digital signatures are commonly used for software distribution, financial information interchange, and in other cases where it is important to detect forgery or tampering. Digital signatures are based on public key cryptography. Using a public key algorithm such as RSA [6], one can generate two keys that are mathematically linked: one private key and one public key. To create a digital signature, signing program (such as an email program) creates a message digest of the digital data to be signed. The private key is then used to encrypt the message digest. Digital signature of a message consists of the encrypted message digest and other information, such as the hash function, etc. The reason for encrypting the message digest instead of the entire message or document is that a hash function can convert an arbitrary input data into a fixed length value, which is usually much

shorter. This saves time since time spent on hashing is much shorter than time spent on signing.

C. Digital Certificates

A digital certificate is a seal of approval that enables an entity (such as a person, a computer or an organization, etc.) to exchange information securely over the Internet using the public key infrastructure (PKI) [5]. A digital certificate may also be referred as a public key certificate. The main purpose of the digital certificate is to ensure that the public key contained in the certificate belongs to the entity to which the certificate was issued. Encryption techniques using public and private keys require a PKI to support the distribution and identification of public keys. A digital certificate contains a public key, used hash functions, owner or subject data, the digital signature of a Certificate Authority who has verified the subject data, and a date range during which the certificate can be considered valid. Without certificates, anyone can create a new key pair and distribute the public key, and claim that it is the public key of other person. One could send data encrypted with the private key and the public key would be used to decrypt the data, but there would be no guarantee that the data was originated by anyone in particular. All the message recipient would know is just a fact that a valid key pair was used.

III. RELATED WORK OF MALWARE SOLUTIONS

Many solutions have been proposed to prevent computers from different kinds of viruses, spywares, etc. The following three solutions are widely used to handle macro viruses.

A. Signature-Based Malware Detection Method

The simplest and most widely used detection method is signature-based method which requires forensic experts to study each malware's behavior and to update virus signatures in the database [1]. Typical malware detection methods based on signatures therefore it has difficulty in detecting polymorphic viruses [9] when viruses first appear because their signatures are not yet analyzed. However, the drawback of this solution is that it cannot protect zero day viruses [18]. Zero-day viruses (also known as next-generation viruses) are previously unknown computer viruses or other viruses for which specific antivirus signatures are not yet available.

B. Macro Usage Restriction

This solution is generally recommended by security experts. The main point of the solution is to leave macros disabled and not to believe any message claiming that users must enable macros to view or interact with Microsoft Office documents. However, the drawback of this solution is that it obstructs users from using helpful macros developed by trusted macro creators.

C. Digital Signature of Microsoft Office

This solution provided by Microsoft Office Suite allows users to digitally sign macros. The digital signature allows a user to know that a macro comes from a trusted source and that it hasn't been modified since it was originally saved by that trusted source. In order to digitally sign a macro, users need to first obtain a digital certificate. A certificate is a seal of approval from a trusted third party that proves the identity of a user. A user can get digital certificates from a variety of commercial certificate authorities, which have different requirements of how the user goes on certifying her identity. However, the drawback of this solution is that Microsoft Office Suite allows unsigned macro to be easily enabled thru warning dialog box (see Fig. 3) when macro security setting is set to the default setting (Disable all macros with notification). Furthermore, macro virus creators can prevent users from checking instructions of the macro by locking VBA project. Busy users tend to enable macro without enough effort to confirm macro creator thoroughly. This makes computers of these users are easily infected by macro viruses. A method that solves this drawback is set macro security setting to "*Disable all macros except digitally signed macros*" and to register digital signatures of all trusted macro creators in computers of all users. However, this method has big overhead in registering and maintaining digital signatures of all computers of organizations. As far as users can change macro security setting, there still is potential that users carelessly allow executing infected macros.

D. Research Work of Malware Detection Methods

Kim and Moon [7] proposed a method that uses dependency graph analysis for detecting script malwares. A script malware is represented by a dependency graph and then the detection is transformed to the problem which finds maximum sub-graph isomorphism in that polymorphism still maintains the core of logical structures of malwares. Ko [8] proposed a flow analysis of macro operations to determine whether the investigated macro is a malware. Based on associated values on variables, the system extracts the control and data flow from the macro, compares the flow with that of the known suspect, and measures similarity. Otsubo et.al [15] analyzed the file structure difference between normal Microsoft document (Rich Text or Compound File Binary) files with malicious ones. They proposed methods that detect malicious Microsoft document by inspecting specific characteristics of file structures of the malicious documents. However, these are some malwares that can't be detected by these methods.

IV. THE PROPOSED SECURITY MODEL

The author claims that antivirus program is still essential to prevent users from malwares. The author proposes a new security model that solves the drawback of the solutions described in the previous section. This security model is based on the prerequisite condition that macros developed by trusted macro creators contain no malware. This security model has two procedures (see Fig. 4). The first procedure performs macro digital signing after including the macro in an office document. The second procedure performs macro's digital signature check before document recipient opens the office document.

A. Macro Digital Signing Procedure

Here, let *X* be a macro creator, and let *Y* be a document recipient. Note that *X* and *Y* can be a person or an organization. Before generating a digital signature of a macro, *X* has to register its private key and public key pair to a reputable certificate authority (such as, GlobalSign, Inc., Thawte, Inc., etc.). The certificate authority will issue a

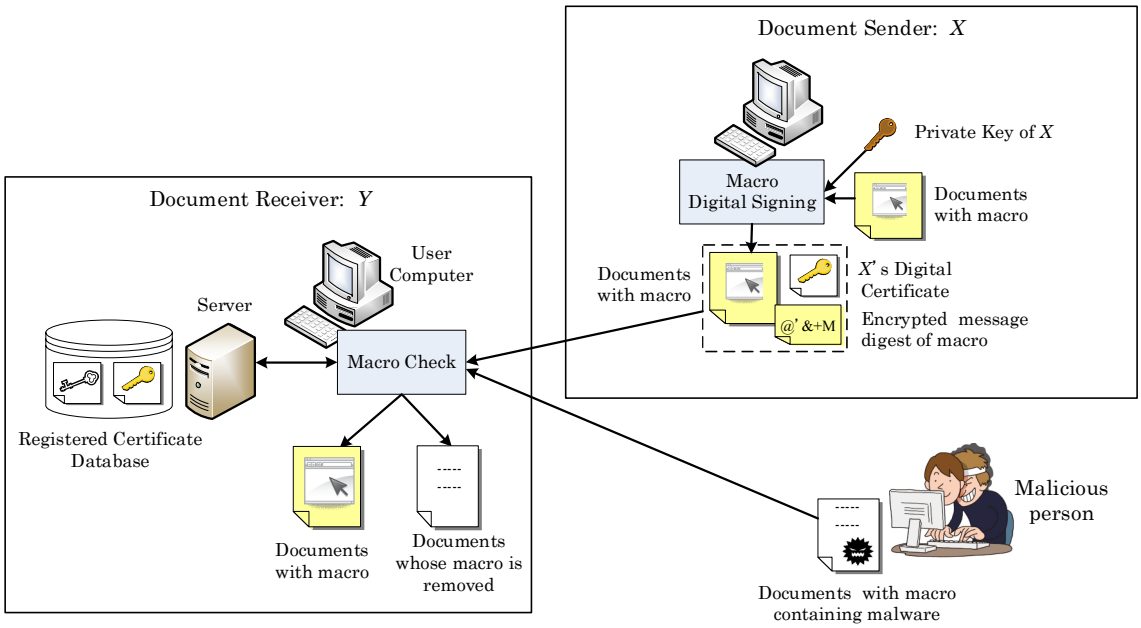


Fig. 4. Overview of the proposed security model.

digital certificate that certifies the ownership of the public key by the named sender *X* appearing as the subject of the certificate. As shown in Fig. 5, after creating a macro, *X* uses hash function described in *X*'s digital certificate to generate a digest from the macro. The technique that accesses macros of Microsoft office documents will be explained in the next section. *X* uses registered private key to encrypt the digest. Note that digital signature of a macro consists of encrypted digest of the macro and *X*'s certificate. Finally *X* sends the document and digital signature of the macro to *Y*.

B. Macro's Digital Signature Check Procedure

In this security model, a macro is trusted if all following conditions are satisfied.

(1) Digital certificate of the macro creator is registered in Certificate Database stored in a file server of the document recipient (see Fig. 4).

(2) Macro digest generated by the hash function of the macro creator is the same as the macro digest generated by decrypting the encrypted digest by the public key of the macro author (see the right hand side of Fig.5).

Note that the first condition is necessary because the document recipient needs to investigate reliability of macro creators. The reliability investigation is based on profile of the macro creator and comments of macro users. If macro creators pass reliability investigation, their digital certificates will be registered in certification database of the document recipient. Therefore, document recipients of the same

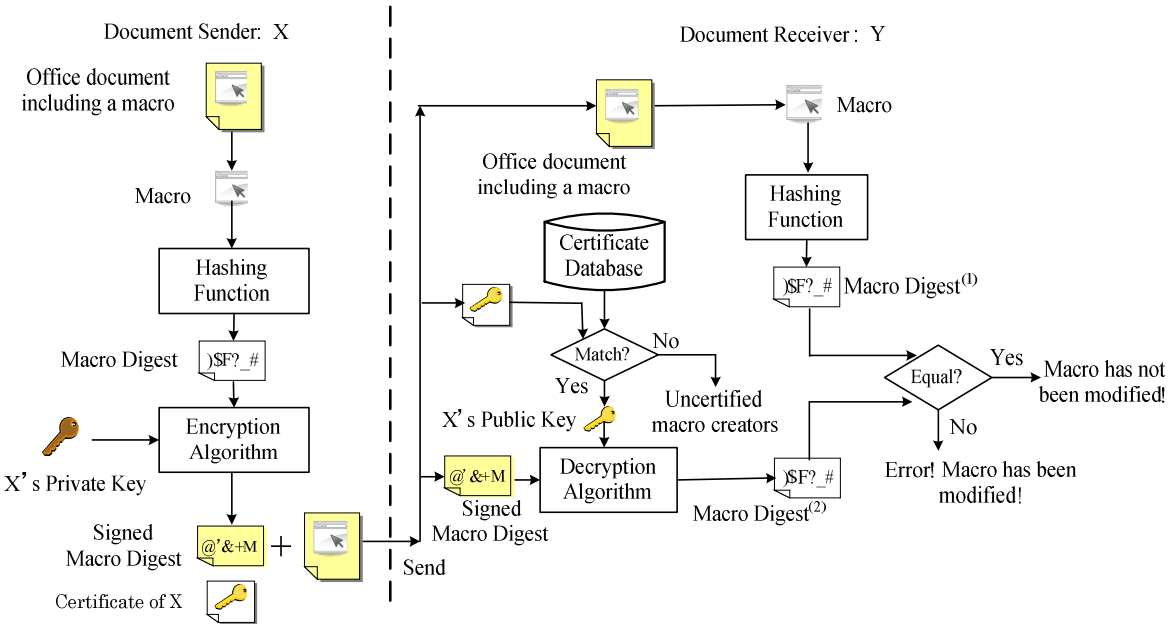


Fig. 5. A Flowchart showing Macro Digital Signature Generation and Checking.

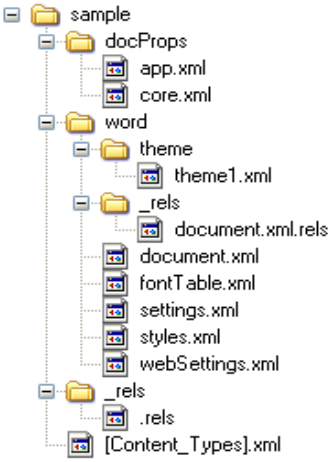


Fig. 6. Hierarchical file structure of a typical Word 2007 document.

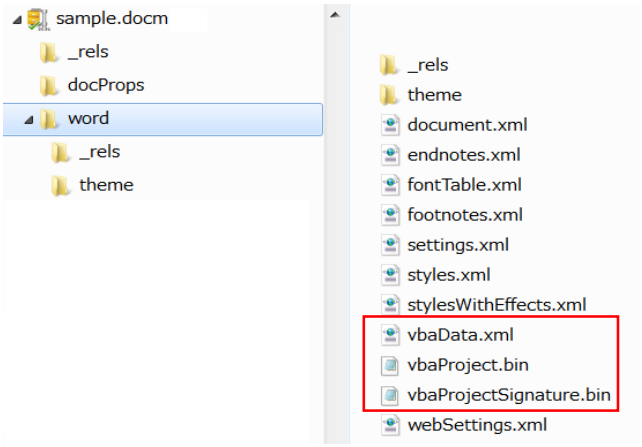


Fig. 7. Hierarchical file structure of a sample macro-enabled document which is created by adding a macro to the document of Fig. 5.

organization can share these digital certificates stored in the database. The certificate database should be maintained daily so that it contains up-to-date information.

The process of checking the second condition is explained as follow. As shown in Fig. 5, the document recipient *Y* produces macro digest(1) by the same hash function used by the document sender *X*. *Y* also obtains macro digest(2) by decrypting the signed macro digest with the public key of *X* which is described in the digital certificate of *X*. If both macro digest(1) and macro digest(2) are the same, the document recipient can conclude that the macro of the received document is the same as the macro signed by the document sender. This means that there is no modification of the macro by the third person. In this case, the document recipient can enable the macro. Otherwise, the document recipient can conclude that the macro may be modified by someone after *X* signed it. There is risk that someone may add virus into the macro. Therefore, the macro is justified to be a distrustful macro. The macro should be removed from the document to prevent users from the damage caused by the macro.

V. TECHNIQUE OF HANDLING A MACRO OF AN OFFICE DOCUMENT WITHOUT EXECUTING MICROSOFT OFFICE SUITE

Microsoft Office 2007 introduced a new file format, which is called Office Open XML [2], as the default file format. Office Open XML (also informally known as OpenXML or OOXML) is a zipped, XML-based file format that represents spreadsheets, word processing documents, charts and presentations. Such files are saved using an extra *x* letter in their extension (*.docx*/*.xlsx*/*.pptx*, etc.). Files containing macros are saved with an extra *m* letter in their extension instead (*.docm*/*.xlsm*/*.pptm*, etc.). Microsoft Office 2010 and Office 2013 also employ OpenXML as default.

Consider the new file format of Word 2007. This format makes a Word file to become a ZIP archive file containing XML and binaries. Therefore developers can easily create, update, or delete data in a Word file programmatically or manually without the need of editing by Microsoft Word. Figure 6 shows the file structure of Contact.docx which is a sample Word 2007 document. Note that the folders and files in a ZIP package combine to create a single document. The main folders and files of the ZIP package of a Word document are described [12] as follow.

1) The *docProps* folder contains file properties of the document.
2) The main document folder, such as the word folder showed here, stores the main document content, any media (such as pictures) in the document, and various document elements such as settings, headers, and themes. Note that the main document folder also contains its own *_rels* folder, where relationships between elements in the main document folder are defined.
3) The *_rels* folder contains a file named *.rels* that stores information about the relationships between items in the ZIP package. The *.rels* file is how the Office system programs know where to find document components when opening a document. This is an important file to keep in mind when developer adds or removes content in an Office Open XML ZIP package.
4) The [Content_Types].xml file contains definitions of the types of content included in the ZIP package, such as the main document, the file properties, and the document theme. This file also stores definitions of the file extensions used in the ZIP package, such as the file formats (such as *jpeg* or *png*) of pictures included in a document. The XML files that make up an Office Open XML ZIP package, such as document.xml, are often referred to as XML parts or document parts.

Figure 7 shows the hierarchical file structure of a sample macro-enabled document which is created by adding a macro to the document of Fig. 6. Macro definition is stored in *vbaData.xml* and *vbaProject.bin*. If developers digitally signed the macro at VBA editor, the digital signature of the macro is stored in *vbaProjectSignature.bin*. However, the signed macro digest of the proposed security model will not be saved in the Zip package. In case the validity check result of macro denotes that the macro of the document is untrusted, *vbaData.xml*, *vbaProject.bin* and *vbaProjectSignature.bin* will be removed from the zip package by using the Open

XML Application Programming Interface [13]. This API allows developers to create packages and manipulate the files that comprise the packages.

CONCLUSION AND FUTURE WORK

In this paper, a new security model that removes distrustful macros from Microsoft Office documents has been proposed. The prerequisite condition of this security model is that macros contain no virus at the time when the macros are digitally signed. This security model is based on digital signatures of the macros of the office documents. A macro is trustful if all following conditions are satisfied. The first condition is that digital certificate of creator of the macro is registered in Certificate Database stored in a file server of document recipient. The last condition is that digital signature of the macro pass sender authentication/non-repudiation investigation and macro integrity investigation. If both conditions are not satisfied, the macro is justified as a distrustful macro. Therefore, the distrustful macro will be removed from the office document.

As the future work, the author is going to develop a prototype that is based on the proposed security model. Furthermore, experiment of prototype will be conducted to evaluate the completeness and correctness of the proposed security model.

REFERENCES

[1] J. Aycock, Computer Viruses and Malware. Springer, 2006.
[2] S. Böttcher, R. Hartel, C. Messinger, Searchable compression of office documents by XML schema subtraction, Proceedings of the 7th international XML database conference on Database and XML technologies, pp. 103-112, 2010.
[3] CareerTrack, Unlocking the Secrets of Microsoft® Office® Macros 2010, CareerTrack, 2013.
[4] F. M. Fernández, P. Caballero-Gil, Analysis of the New Standard Hash Function, Proceeding Revised Selected Papers of the 14th International Conference on Computer Aided Systems Theory - EUROCAST 2013 - Volume 8111, pp. 142-149, 2013.
[5] R. Hunt, PKI and digital certification infrastructure, 9th IEEE International Conference on Networks, pp.234-239, Oct 2001.
[6] J. Jonsson, B. Kaliski, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, RSA Laboratories, 2003.
[7] K. Kim and B. Moon, Malware detection based on dependency graph using hybrid genetic algorithm, Proceedings of the 12th annual conference on Genetic and evolutionary computation (GECCO '10), July 2010.
[8] C. W. Ko. Method and apparatus for detecting a macro computer virus using static analysis, February 2004. United States Patent #6,697,950 B1.
[9] X. Li, P. K. K. Loh, F. Tan, Mechanisms of Polymorphic and Metamorphic Viruses, Proceedings of the 2011 European Intelligence and Security Informatics Conference, pp. 149-154, 2011.
[10] R. Mansfield, Mastering VBA for Microsoft Office 2013, Sybex, 1 edition, 2013.
[11] Microsoft, Microsoft Office Suite, https://products.office.com/en-us/home, accessed on 2015/9/12.
[12] Microsoft, Building Word 2007 Documents Using Office Open XML Formats, http://msdn.microsoft.com/en-us/library/bb264572%28v=office.12%29.aspx, accessed on 2014/10/12.
[13] Microsoft, Open XML API Roadmap, http://msdn.microsoft.com/en-us/library/office/cc471945%28v=office.12%29, accessed on 2014/10/12.
[14] The National Cyber Security Center, the Dutch CERT, https://www.ncsc.nl/dienstverlening/expertise-advies/factsheets/factsheet-office-macros.html, accessed on 2014/9/30.
[15] Y. Otsubo, M. Mimura, H. Tanaka, Methods to Detect Malicious MS Document File Using File Structure Inspection, Journal of Information Processing Society of Japa, Vol. 55, No.5, pp.1530-1540, May 2014.
[16] P. Szor, The Art of Computer Virus Research and Defense, Addison-Wesley Professional, 2005.
[17] S.R. Subramanya and B.K. YI, Digital Signatures, IEEE Potential, pp.5-8, Mar 2006.
[18] S. M. Tabish, M. Z. Shafiq, M. Farooq, Malware detection using statistical analysis of byte-level file content, Proceedings of the ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics, pp. 23-31, 2009.
[19] Virus Bulletin, https://www.virusbtn.com/virusbulletin/archive/2014/07/vb201407-VBA, accessed on 2014/9/30.
[20] Y. Wang, Q. Zhao, L. Jiang, Y. Shao, Ultra high throughput implementations for MD5 hash algorithm on FPGA, Proceedings of the Second international conference on High Performance Computing and Applications, pp.433-441, 2009.
[21] N. Wearver, V. Paxson, S. Staniford, R. Cunningham, A taxonomy of computer worms, Proceedings of the 2003 ACM workshop on Rapid malcode, pp.11-18, 2003.