A review of cryptographic algorithm recognition technology for binary code

Chenxia Zhao¹,²,a, Fei Kang²b, Ju Yang²c and Hui Shu²,d*

¹Zhengzhou University, Zhengzhou, Henan 450001, China
²State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, Henan 450001, China

*aemail: zhaoctx96@126.com, bemail: 39323690@qq.com, cemail: 18511286202@163.com,

*Corresponding author’s e-mail: 202022332015356@gs.zzu.edu.cn, shuhui123@126.com

Abstract. Aiming at the cryptographic algorithm that may be contained in the binary program, combined with existing research results, several cryptographic algorithm identification techniques are analyzed, including control flow graphs, data flow graphs, feature matching, selection of command statistical attributes, and semantic-based behavior analysis. The recognition effects of related technologies are discussed respectively.

1. Introduction
The cryptographic algorithm is used for encryption and decryption to ensure the security of the transmitted information. In the era of rapid information development, network and information security cannot be guaranteed without a cryptographic algorithm. In terms of computer security protection, the security of data transmission and software systems often relies on a certain cryptographic algorithm, and computer viruses often use the cryptographic algorithm protection mechanisms to hide their characteristics. Identifying the cryptographic algorithm in the malicious code can better protect the computer system. The application of cryptographic algorithm recognition is of great value in both commercial and military fields. Finding a more efficient and accurate cryptographic algorithm recognition technology has strong theoretical and practical significance.

Cryptographic algorithm recognition is a part of algorithm recognition, which belongs to the category of program comprehension. Binary-based cryptographic algorithm identification refers to the use of various tools and techniques to identify algorithms with encryption and decryption functions used in the program. At present, there are many methods to identify the binary cipher algorithm, and different methods have their advantages and disadvantages. We will review the technical research on binary cryptographic algorithm identification.

2. Review
The existing cryptographic algorithm recognition technologies about binary files mainly include control flow graph and data flow graph, feature matching and instruction statistical attribute screening, and semantic-based behavior analysis. These technologies are discussed in the following sections.
2.1. Control flow graph and data flow graph

Control flow graphs and data flow graphs often contain a lot of information. In many fields, control flow graphs and data flow graphs are used to judge the similarity of code fragments. Of course, in the field of cryptographic algorithm recognition, some people use control flow graphs and data flow graphs to detect or classify cryptographic algorithms.

2.1.1 Control flow graph

Through the control flow diagram of the code, we can know the calling relationship of the function in the execution of the program. In 2011, Ruoxu Zhao et al. [1] adopted the open-source Bochs full-system simulator to verify cryptographic functions. The Bochs full-system simulator tracks and records the CPU running status and memory read and write operation status. According to the control flow executed by the program, Ruoxu Zhao divided the recorded traces into control flow diagrams, code blocks, dynamic basic blocks, and instruction data units, and then performed the cryptographic function verification in combination with the association between the input and output data of the cipher function (assuming that input and output parameters are read from memory). The system is difficult to recognize custom symmetric cryptographic algorithms and has many limitations.

To solve the problem of low recognition efficiency of traditional control flow graph loop detection methods, Li Jizhong [2] designed a disassembly result control structure loop recognition algorithm based on back-edge traversal and DFS path discovery to identify cryptographic functions in 2014. By introducing the loop body identification set, he symbolically defined the nested loop structure and designed an instruction sequence loop recognition algorithm of loop body reduction from inside to outside. Although this method can identify some cryptographic functions, the recognition rate is not high.

In 2017, Léonard Benedetti et al. [3] used grap to detect cryptographic algorithms. The disassembly code of an executable program can represent a graph (control flow graph) of possible instruction sequences. Grap can be used to detect cryptographic algorithms which have available patterns. Although the grap technology does not affect the detection result when the code has a small change, it is necessary to create a design pattern that accurately detects the cryptographic algorithm, which is generally unpredictable in reality.

2.1.2 Data flow graphs

The data flow graph contains a large amount of data information, and the data flow graph is used to check the algorithm of the code in many fields. In 2012, Li Yang et al. [4] identified the cryptographic algorithm with the support of the binary analysis platform DynamoRIO. DynamoRIO dynamically records data information during program execution. Then he comprehensively used techniques such as filtering and classification based on statistical characteristics, matching based on the constant characteristics of cryptographic algorithms, and function parameter identification based on data stream analysis to identify cryptographic algorithms. Compared with the traditional cryptographic algorithm identification method based on static constant feature codes, this method basically eliminates the interference caused by software packing and code confusion, and can accurately identify and locate the vast majority of cryptographic algorithms which have dynamically generated constant features or no obvious constant features. But the intermediate files generated by dynamic data stream records are too large, the identification of specific cryptographic algorithms may increase the rate of false positive. In 2014, Li Jizhong [5] defined the encryption and decryption process dependency graph in the form of nine-tuples, and proposed a functional dependency graph construction framework based on dynamic data flow analysis as shown in the figure below, and designed the construction algorithm. He simplified the dependency graph using the method of garbage call removal and loop reduction. He also designed a similarity discriminant algorithm to distinguish the dependency graph set, based on the standard cryptographic algorithm dependency graph knowledge base. The dependency analysis of the encryption and decryption process is an important step in the reconstruction of cryptography, which plays an important role in the security analysis of the target code. As the scale of application software
continues to increase, the constructed encryption and decryption process dependency graph continues to expand, which makes the workload continue to increase. In addition, the paper does not give the recognition accuracy of the cryptographic algorithm.

Figure 1. Dependency graph construction framework for encryption and decryption process.

In 2015, Lestringant et al. [6] proposed the Data Flow Graph (DFG) isomorphism to automatically identify the symmetric cipher algorithm and its parameters in binary files. But their DFG isomorphism method is limited to known primitives and relies on heuristics to select code fragments for analysis. This limitation was subsequently overcome. In 2020, Carlo Meijer [7] proposed combining DFG with symbolic execution to automate the identification and classification of (proprietary) cryptographic primitives in binary codes. Because DFG cannot represent code flow information, it is lacking in loop detection. Later work can be carried out for the information merging of the code flow.

The existing cryptographic algorithm recognition based on control flow graph and data flow graph has the problem of false positives or excessive workload. Looking for other methods to cooperate with the use of control flow and data flow can better make up for these deficiencies.

2.2. Feature word matching and command statistical attributes filtering

Cryptographic algorithms often include characteristic constants and statistical attributes of instructions. These characteristic constants and statistical attributes of instructions can generally identify a part of the cryptographic algorithms that contain these constants. Next, we discuss the specific methods of using these characteristic constants and statistical attributes in the process of cryptographic algorithm identification, as well as the existing problems.

Ian Harvey[8] first put forward the problem of identifying cryptographic algorithms in binary codes in a research report in 2001. The report summarized the overall and partial characteristics of cryptographic algorithms in binary files. Ian Harvey classified instruction statistical characteristics according to instruction function. These categories include function call classes, data movement
classes, flow control classes, arithmetic and logic operations, system calls, multiplication and division operations, and other classes. According to the relevance of various instructions and cryptographic algorithms, different weights are assigned to them. A simple linear weighting method is used to calculate the similarity of the instruction fragments. The area with dense similarity values is the location of the cryptographic algorithm. The report summarized the overall and local characteristics of cryptographic algorithms in binary code, but the proposed simple linear weighted cryptographic algorithm identification method can only locate the approximate location of the cryptographic algorithm, the accuracy of discrimination is not high, due to the lack of code structured call control information and the subjectivity of statistical attribute weighting. N.Lutz[9] pointed out in the 2008 master's paper that arithmetic and bit operation instructions in the dynamic execution of cryptographic functions account for a large proportion, and proposed the view that the important components of the cryptographic function are loops, the cryptographic function uses a lot of integer operations and the process of encryption and decryption will change the entropy of information that pollutes the memory. He used the Valgrind dynamic instrumentation tool under the Linux platform to record the first read memory data set and the last write memory data set during the dynamic execution of the function and combined with the cyclic characteristics in the implementation of cryptographic algorithms, selected cryptographic functions according to the high entropy characteristics of cryptographic operation data. Although the high entropy feature of cipher function operation data is proposed, the mechanism of generating the high entropy feature and the data type of cipher function operation are not analyzed in detail. Besides, no effective solutions have been proposed for the recognition of the cycle characteristics of dynamic instruction sequences. In the same year, Zhi Wang [10] adopted a similar method and used data life cycle analysis techniques (including taint analysis, data flow analysis, etc.) to locate the turning point of cryptographic functions for processing plaintext messages and ciphertext messages based on dynamic statistical characteristics. Based on this, the approximate location of the cryptographic function is determined, and the decrypted plaintext information of the encrypted protocol is searched in the context memory unit of the cryptographic function. This method is only effective for the first cryptographic function in the dynamic instruction execution sequence, and cannot handle the combined use of multiple cryptographic functions. In addition, the process of obtaining decrypted encryption protocol plaintext information from the cryptographic function context memory unit is not highly automated.

In 2009, for the mathematical-statistical characteristics of cryptographic functions, Jason extracted the logic instruction frequency (xor, ror, rol, shr, shl) in the disassembly result. According to the OpenBSD-C function library and OpenSSL function library for statistical sample extraction and data training under different optimization settings, he used an artificial neural network model [11] (Neural Net for Locating Cryptography, NNLC) to determine whether the objective function is a cryptographic function, and analyzed the false positives presented by the discrimination result. In the process of feature attribute selection, this method only selects several representative logic operation instructions, does not consider other arithmetic operation instructions and the structural characteristics of cryptographic functions. In terms of public-key cipher and stream cipher identification, the reliability of this method needs to be further verified. Wang[12] introduced a heuristic method based on code structure changes when executing cryptographic algorithms. In 2009, Caballero[13] improved this heuristic. He pointed out that encryption routines use a high percentage of bitwise arithmetic instructions. Although these methods are very useful for detecting cryptographic algorithms, it has been found that they sometimes miss code examples and cause false positives.

Commonly used identification tools for cryptographic algorithms based on static feature codes have some inherent shortcomings. Felix Gröbert[14] built a dynamic binary prototype system based on PinTools in 2010, which can record the dynamic execution instruction information of the application. On this basis, a cryptographic algorithm feature instruction identification method combining instruction opcode and feature code is proposed, and it is verified against cryptographic function libraries such as OpenSSL, CryptoPP, and BeeCrypt. According to the dynamic analysis results of PinTools on the application, Felix analyzed the source of cryptography, such as dynamic characteristic
code, execution loop, and memory information presented during the implementation of cryptography algorithm, and verified the typical cryptography library.

Aiming at the cryptographic algorithm in the obfuscated binary code, Joan Calvé[15] identified the loop body in the instruction trajectory in 2012 and carried out I/O verification based on the input and output data of the loop body. Due to the instruction as the basic granularity of loop recognition, the efficiency of loop recognition is not high. And in the verification process of the cryptographic algorithm, the loop body I/O is traversed and verified without preprocessing, which makes the verification efficiency not high.

In 2014, Li Jizhong[16] studied the encryption and decryption mechanisms of various cryptographic algorithms and used binary code reverse analysis technology to extract various dynamic and static features of cryptographic algorithms. On this basis, cryptographic algorithms were identified and analyzed. The overall framework is shown in the figure. 2.

Kok, S et al.[17] proposed a pre-encrypted detection algorithm (PEDA) consisting of two stages in 2019. In PEDA-Phase-I, a learning algorithm (LA) is used to capture and analyze the Windows application programming interface (API) generated by the suspicious program. Through API pattern recognition, LA can determine whether the suspicious program is cryptographical transmission software. This method is used to ensure the most comprehensive detection of cryptographical transmission software, but it may have a high false positive rate (FPR). If the prediction is a cryptographic transmission software, PEDA will generate the signature of the suspicious program and store it in the signature repository, which is in the second stage. In PEDA-Phase-II, the signature repository allows the detection of encrypted transmission software at an earlier stage, which is the pre-execution stage through the signature matching method. This method can only detect known cryptographical transmission software.
The recognition effect of the cryptographic algorithm based on feature matching and instruction statistical attribute screening is not high. There are several cases where this method cannot be used: First, for algorithms that are implemented by calling multiple custom cipher algorithms to each other. Second, feature words are dynamically generated or do not exist. Third, some cryptographic algorithms have no obvious directive statistical properties.

2.3. Semantic
To identify cryptographic algorithms based on semantics, only Jingwei Zhang et al.[18] proposed the cryptographic behavior analysis method of the public-key cryptographic algorithm in 2011 to identify the public key cryptographic algorithm that may be included in the embedded system. Based on the research on the encryption principle of the public key cryptography algorithm and the analysis of the characteristics of the assembly level algorithm, Zhang Jingwei described the encryption behavior and identified the public key cryptography algorithm with the model detection technology. The abstract model of public-key cryptographic algorithm identification uses finite states automata (FSA) to describe the encryption behavior of public-key cryptographic algorithms; pushdown automata (PDA) describes the global state space of the target file, that is the execution path space of the target file; then the model detection method is used to check whether the encryption behavior of the public key cryptographic algorithm exists in the target file. This semantic-based public key cryptographic algorithm recognition abstract model is shown in Figure 3.

![Figure 3. Semantic-based public key cryptographic algorithm recognition abstract model.](image)

Although there have been some research results based on semantic analysis of binary, there is still a large gap in the use of semantics to identify binary file cryptographic algorithms.

2.4. Other methods and existing tools
A. Soni[19] proposed to use the Adaboost algorithm to classify cryptographic functions. By constructing a weak classifier that is slightly better than random guessing, each weak classifier is trained, and a strong classifier is obtained to identify the cryptographic functions.

In addition, the following tools are currently available for detecting encryption algorithms in binary programs. YARA describes rules in binary or text mode for the classification and detection of malware. Its’ Rules can also be written to detect specific constants or instructions related to encryption operations. FindCrypt2 and Signsrch use constants to detect cryptographic algorithms. IDAScope uses the same technique as searching constants and searches for binary areas that represent a large number of arithmetic and logic instructions. Aligot is a dynamic tool for detecting encryption algorithms, even if the binary files are obfuscated. Its mode of operation includes dynamic detection loops, and then sends some input values as parameters of these loops to test whether the output value is the same as the input value of a known algorithm. These tools rely on some static constants or loops, which belong to the category of several cryptographic algorithm identification technologies discussed above.
3. Discussion
In the cryptographic algorithm identification methods based on control flow graphs and data flow graphs, there are some problems with unsatisfactory identification results. Ruoxu Zhao's research can detect the symmetric ciphers and hash functions in most programs. Their approach can successfully identify the same algorithm with different implementations. One defect of this method is that the AES key expansion process can be determined, but the AES encryption cannot be determined. The average analysis speed is about 15k instructions per second, and the analysis time is usually within or around 10 minutes.

Several methods of data flow graphs and control flow graphs used by Li Jizhong and Li Yang in cryptographic algorithm identification are simply compared in Table 1. It can be seen from the table that the recognition speed is gradually improving. Besides, the recognition time of each method increases with the size of the identified software.

Table 1. Li Jizhong and Li Yang's cyclic characteristics, dynamic data flow and other methods to achieve a simple comparison of cryptographic recognition effect.

| Cryptographic algorithm recognition based on dynamic binary analysis | Binary code level cyclic feature recognition of cryptographic algorithm | Analysis of encryption and decryption process of cryptographic function based on dynamic data stream |
|---|---|---|
| software | Software size /B | recognition time/s | software | Software size /B | recognition time/s | software | Software size /B | recognition time/s |
| Feiq/2.4 | 3228 | 63 | Feiq/2.4 | 3228 | 17.6980 | Feiq/2012 | 15367168 | 33 |
| RAR/3.70 | 4713472 | 890 | RAR/3.91 | 1014 | 3.0270 | RAR/3.70 | 4713472 | 559 |

Carlo Meijer's method can successfully identify most of the cryptographic primitives in the various binaries, but the analysis time is positively correlated with the depth of inline.

Among the cryptographic algorithm recognition methods based on feature matching and instruction statistical attribute filtering, Felix Grubbert's method can indeed automatically recognize the cryptographic primitives of a given binary sample. However, there are also obvious defects. Firstly, the code must be executed; secondly, the custom encryption algorithm may not be able to detect successfully; finally, the cryptographic code generated by the compiler with different settings and optimization methods cannot be captured by the heuristic method.

For dynamically-generated features or cryptographic algorithms that do not contain obvious features in code fragments, methods based on statically identifying feature words, constants, and command statistical attributes in the code fragments may not be able to correctly identify cryptographic algorithms.

In the aspect of semantic analysis of binary file cryptosystem, there are only some research results on public-key cryptosystem. Zhang Jingwei's method is accurate and effective for the behavior characteristic model of the public key cryptography algorithm, but the recognition time of the public key cryptography algorithm increases linearly with the increase of the target application program, and the automation is insufficient. For other types of the cryptosystem, further research is needed.

4. Conclusion
In the identification of cryptographic algorithms for binary files, the recognition effect of a single method is not ideal. If several methods are combined to make up for each other's shortcomings, the recognition accuracy will be improved. If the control flow graphs, characteristic words, signature, and semantics are combined to assist each other in the recognition of the cryptographic algorithm, the recognition accuracy can be greatly improved. Besides, if these works are realized with machine
learning, their working efficiency can be greatly improved. In the later work, how to combine various methods needs further experimental research.

Acknowledgments
We thank the reviewers for taking the time to review this article. Thanks to the teachers and classmates in the laboratory for their many suggestions for this article. They helped me a lot in the process of writing this paper.

References
[1] Zhao, R., Gu, D., Li, J. and Yu, R. (2011) Detection and Analysis of Cryptographic Data Inside Software. The 14th Information Security Conference (ISC): 182-196.
[2] Li, J., Jiang, L., Shu, H. (2014) Binary code level cyclic feature recognition of cryptographic algorithm. Computer engineering and design, 35(08): 2628-2632.
[3] Benedetti, L., T. Aurélien and J.F.A. (2017) CyberSecurity, Detection of cryptographic algorithms with grap. 2017.
[4] Li, Y., Kang, F. and Shu, H. (2012) Cryptographic algorithm recognition based on dynamic binary analysis. Computer Engineering, 38(17): 106-109+115.
[5] Li, J., Jiang, L., Shu, H. and Dai, L. (2014) Analysis of encryption and decryption process of cryptographic function based on dynamic data stream. Computer application research, 31(04): 1185-1188.
[6] Lestringant, P., Guihéry, F. and Fouque, P. Automated identification of cryptographic primitives in binary code with data flow graph isomorphism. (2015) In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, pages 203–214.
[7] Meijer, C., Mooensamy, V. and Wetzels, J. (2020) Where's Crypto?: Automated Identification and Classification of Proprietary Cryptographic Primitives in Binary Code.
[8] Ian, H. (2001) Cipher Hunting: How to Find Cryptographic Algorithms in Large Binaries. nCipher Corporation Ltd.
[9] Lutz, N. (2008) Towards Revealing Attackers' Intent by Automatically Decrypting Network Traffic. Master's thesis, ETH Zurich and Google.
[10] Wang, Z., Jiang, X., Cui, W. and Wang, X. (2008) ReFormat: Automatic Reverse Engineering of Encrypted Messages. NC State University.
[11] Jason, L., Wright and Manie, M. (2009) Neural Network Approach to Locating Cryptography in Object Code, Emerging Technologies & Factory Automation (ETFA): 1-4.
[12] Wang, Z., Jiang, X., Cui, W., Wang, X. and Grace, M. (2009) ReFormat: Automatic Reverse Engineering of Encrypted Messages. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 200–215. Springer, Heidelberg.
[13] Caballero, J., Poosankam, P., Kreibich, C. and Song, D. (2009) Dispatcher: Enabling Active Botnet Infiltration using Automatic Protocol Reverse-Engineering. In: ACM Conference on Computer and Communications Security.
[14] Gröbert, F. (2010) Automatic Identification of Cryptographic Primitives in Software. Ruhr University Bochum.
[15] Calvet, J., José, M., Fernandez and Aliot, J. (2012) Cryptographic Function Identification in Obfuscated Binary Programs, Proceeding of the 2012 ACM Conference on Computer and Communications Security: 169-182.
[16] Li, J. (2014) Research on Key Technologies of cryptographic algorithm identification and analysis, PLA University of Information Engineering.
[17] Kok, S. (2019) Prevention of Crypto-Ransomware Using a Pre-Encryption Detection Algorithm, Computers, 8(4): p. 79.
[18] Zhang, J., Shu, H., Jiang, L. and Li, J. (2011) Research on identification technology of public key cryptography algorithm. Computer engineering and design, 32(10): 3243–3246+3273.
[19] Soni, A. Learning Encryption Algorithms from Ciphertext. BTP report, Department of Computer Science and Engineering, Indian Institute of Technology.