# Processing of Cryptographic Function Identification Based on Multi-feature Progressive Model

Wei Lin, Yuefei Zhu, and Ruijie Cai

***Abstract*—Research on cryptographic function identification is of great significance in malicious code analysis, software vulnerability analysis and other fields. The current cryptographic function identification algorithm has the problem of low identification accuracy because of its single feature. In order to solve this problem, we proposed an improved method of cryptographic function identification based on multi-feature progressive approach to identify cryptographic functions by using data flow analysis in software. The experimental results show that, compared to current methods, it can identify the accurate cryptographic function accurately, and the accuracy is improved.**

***Index Terms*—Cryptographic function identification, multi-feature matching, progressive, decision tree, data flow analysis.**

## I. Introduction

At present, in order to achieve secure communication, the software mostly uses encryption to protect protocol data. Meanwhile, the malicious program also uses encryption against the reversing analysis. The cryptographic algorithm identification of the application can play a positive role in the analysis of malicious code, network data security transmission areas.

The existing cryptographic function identification can be divided into static and dynamic identification two. Principle of static identification is detecting whether exist static characteristic of cryptographic algorithm in binary program, may be a code fragment, special constant (magic number), and the special structure such as s-boxes or the digital codes of the encryption function at the entrance and so on. When the corresponding feature is found, these tools will output the name of the algorithm (such as MD5, RSA), or could achieve output implementation method of the algorithm (such as OpenSSL).

Caballero J. *et al*. [1]-[4] identify the cryptographic function by using the characteristic that bit operation instructions represent significant share of the total instruction. The Lutz N. [5] calculates entropy of the buffer to judge whether the buffer is decrypted, since the decryption process will reduce the information entropy of message. At the same time, the reference also uses another two characteristics: one is cryptographic algorithms generally have circular structure, the other is cryptographic algorithms generally use a large amount of arithmetic instructions. The reference mentioned above is limited to distinguish cipher function and non cryptographic function; security analysts need to know what kind of cipher algorithm in many situations. Calvet J. [6] present a cryptogram algorithm identification method based on cyclic I/O. the idea was to record program track information firstly, then identify the cycle structure in the track file, get the input and output parameters of the cycle structure, finally put the input and output memory data into the implementation of cryptographic algorithms to verify correctness. The limitation of this method is that, some cryptographic algorithm may take some optimization algorithm in the implementation process in order to improve efficiency, which will destroy some circulation structure of cryptographic algorithm in the original design, for instance, an optimization algorithm of AES will expand all the cycle structure , which lead to omissions.

In summary, the cryptographic function identification is the key of analysis of network software and malicious code, the existing static identification methods are generally feature matching methods with very limited applicability, and the dynamic identification are mostly classify the cryptographic function using instruction proportion which the normal function and the cryptographic function are different on. But because of cryptographic functions variety, it is very difficult to find a suitable classification model which makes the cryptographic function all only belong to one category by using instruction proportion, so the accuracy needs to be improved.

## II. Cryptogram Algorithm Identification

### A. Overall Idea

This paper proposes a method of the progressive cryptographic function what add the data flow analysis into the cryptographic function identification. Firstly, we use the cryptographic function constant and operation instruction ratio as a feature to eliminate the most of the normal functions interference, that identify the cryptographic function which include MD5, SHA1 who has the typical characteristics, and in the meantime we also filter the suspicious cryptographic function; then analyze the data dependency of the suspicious cryptographic function with data flow analysis to judge the types roughly. Finally, distinguish the cryptographic function specific type through instruction proportion characteristic of different instruction types. The whole idea of cryptographic function identification as shown in Fig. 1.

Manuscript received November 20, 2014; revised June 26, 2015. This work was supported in part by the National Key Technology Support Program of China, (2012BAH47B01); National Natural Science Foundation of China (61309007).

The authors are with the State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, P. R. China (e-mail: tiamo9880@gmail.com, zyf0136@sina.com, wsxcrj@163.com).

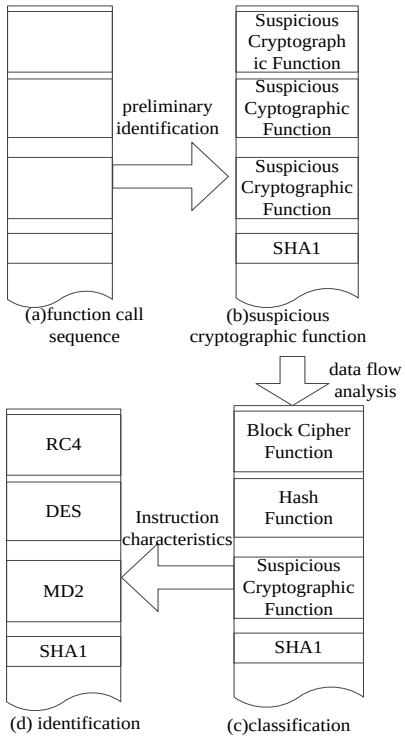


Fig. 1. system architecture diagram.

### B. Preliminary Identification of Cryptographic Functions

Therefore, First according to the characteristics of some simple cryptographic function of identifying the function selection, that the reason is the program execution path contains a function call frequently, then to eliminate the most function interference, last identify the suspicious cryptographic function depth. This paper chooses the following three characteristics:

1) The total number of instructions: The cryptographic function needs more instruction to realize the function of encryption or decryption what reason of the function is complex., This paper are as shown in Table I for the statistical results shown which through the analysis of an execution trace of Feiq software .we can see that the number of instructions is less than 10 of the number of function calls to account for all the function call number 42.8%, and the number of instructions is less than 200 of the function call is occupied 85.8%, interference therefore using this feature is the total number of instructions can eliminate most of the non cryptographic function. Therefore it using this feature which is the total number of instruction can eliminate most of the non cryptographic function.

TABLE I: THE PROPORTION OF FUNCTION WITH DIFFERENT INSTRUCTION COUNT

| Instruction Count | <10 | <50 | <100 | <200 |
|---|---|---|---|---|
| Proportion | 42.8% | 74.2% | 81.7% | 85.8% |

2) Arithmetic and logic operations instruction ratio: There is more cryptographic function arithmetic, logic commands, and less control transfer instructions, these types of instruction has a very distinct statistical properties in the realization of cryptographic algorithm in common, what is an important key for distinguish the cryptographic function and non code function.

3) Constant function used: implementation of cryptographic algorithms used in the constant what can be divided into two categories. The first is the immediate instructions, the second is constant which stored in memory, and that feature can be used to identify some special cryptographic function.

After the function information extraction, according to the characteristics of the function of cryptographic function was preliminarily identified filtering, in order to avoid duplication of identification of the same cryptographic function, this paper only to identify the cryptographic function innermost, if the sub function is identified as the cryptographic function, then identify right its parent function, if the child is not a function of the cryptographic function, then the information of the sub function is incorporated into the parent function identification. The concrete steps for each function call identification as follows:

Step 1: If the total instruction number execution is in a certain range of threshold values, go to step 2; otherwise, the function is thought not to be a cryptographic function, end all steps.

Step 2: Matches the immediate value and accessed memory which used by the function instruction with the characteristics library of cryptographic function, detect whether there contains a specific constant which only used by cryptographic algorithm, if contains ,that means the function is a realization of the cipher algorithm, end all steps.

Step 3: Count the proportion of arithmetic and logic operation instructions, if the proportion is greater than a certain threshold, then mark this function as a suspicious cryptographic function.

Preliminary identification and filtering of cryptographic function will filter out almost functions of call sequence, and identify some cryptographic function which contains the constant number, such as MD5, SHA1, and then filter the suspicious cryptographic function by the proportion of arithmetic and logic operation instructions for further analysis.

### C. Cryptographic Function Classification Based on Data Flow Analysis

In order to prevent cracking, Cipher algorithm is designed to try to meet the Avalanche Effect. Avalanche Effect means input little changes (such as change one bit)will result output large change (such as a half of output bit reversal). For example, the SHA-1 hash function shown in Fig. 2, the input only changed a bit, the output is entirely different. If the Avalanche Effect of a cipher or hash function is not good enough, so the cryptanalyst can speculate the input from the output, which will may cause the algorithm partly or wholly broken.

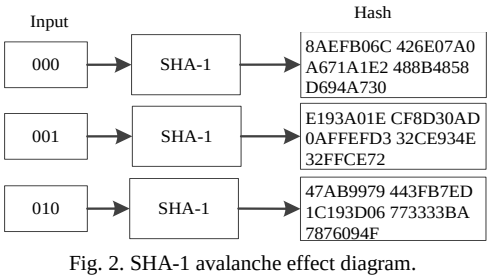


Fig. 2. SHA-1 avalanche effect diagram.

The Avalanche Effect of Cipher algorithm makes the dependent relationship complex between input and output of cryptographic functions. From the taint analysis perspective, each input byte will taint many output bytes, each output byte will rely on many input bytes, by use of this feature we can identify the cryptographic functions and non cryptographic functions on trace files, and also can be used for precisely positioning the cryptographic function input and output buffer.

At the same time since the input and output dependence of different cryptographic algorithm is different. As shown in Fig. 3, for block cipher algorithms, each input byte will affect every output byte, while for stream cipher each input byte only affect one output byte, It can identify the types of cryptographic function using this feature.

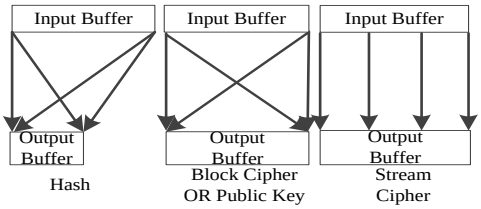


Fig. 3. the input and output dependence of different types of cryptographic function.

Positioning plaintext buffer is to find the input and output buffer which meet characteristics of cryptographic functions in all function input and output buffers. If it cannot be found, that means the function is not a cryptographic function. The common kinds of cryptographic function are Hash, block cipher function, public key cryptographic, and stream cipher. According to the characteristics of plaintext buffer, the four kinds of cryptographic functions can be divided into two types:

1) The first type is the former three kinds function, the function exist one input buffer and one output buffer, The each byte of input buffer will affect many output buffer bytes, while the each output buffer byte rely on many input buffer bytes.
2) The second is a stream cipher function, which contain one input buffer and one output buffer. The buffer size is equal, and each input byte only affects one output byte on corresponding position.

There are two interferential factors to locate the plaintext buffer. The one is the normal function may also exist the input and output buffer to meet above feature, the other is the buffer which meet the above feature may be not the plaintext or ciphertext, can be a encryption key, S-boxes or other temp Variable.

This paper adopts three methods to exclude the interference of the above two:

1) Filter out the majority of ordinary functions by the proportion of arithmetic and logic operation instructions, According to the practical experience, the rest of functions are cryptographic functions generally.
2) Record the value of the stack pointer ESP at the end of the function, since when the function go to the end, local variables in stack have been released, so the accessed address smaller than ESP value is seen as the local variables.
3) Set the encryption and decryption buffer is not less than 4 bytes, in case some temporary read-write memory or discrete access memory are mistaken for encryption and decryption buffer.

In order to identify the first kind of buffer, firstly the definition of memory dependency is given as follow:

**Memory dependence** refers to the number of taint source bytes which a single memory unit depends on in the taint analysis. For instance, when the instruction "mov byte ptr[edi],al" executed ,the taint status of one memory unit is {1,2,4},then the Memory dependence of this memory unit is 3.

For the first kind of buffer, in cryptographic functions each memory unit of output depends on all bytes of input due to Avalanche Effect, shown as high Memory dependence，so we Can use the following methods to locate the input and output buffer:

1) Mark all the address which size is not less than 4 in the all input memory unit of suspicious cryptographic functions as taint source.
2) Start the taint analysis for the execution track of suspicious cryptographic functions， then check each output unit of the taint source by bytes.
3) If find a input buffer and out buffer， which the each byte of input buffer will affect many output buffer bytes, and the each output buffer byte rely on many input buffer bytes， that means the successful positioning.

**Algorithm 1: The first kind of buffer positioning algorithm**

**Input**：Function execution trace(FuncTrace)，and All read and write of the memory unit (In and Out)
**Output**：if success，return InBuf and Outbuf，else return NULL

```
InitTaintSource(In) // Initialize the continuous memory unit
    which size is not less than 4 for initial taint
TaintAnalysis(FuncTrace)
for each byte i in taint source
  calculate T(i) // calculate the memory unit set propagated
    by "i" byte, denoted as T(i)
end for
InBuf = { the first byte of taint source }
OutBuf = T(0)
for (the rest byte of taint source)
  if (T(i) ⋂ T(i-1) ! =NULL)
      push i to InBuf;
      push T(i) to OutBuf;
    end if
    if sizeof(InBuf)<T then // T is the minimum threshold
    of input buffer size
      InBuf = { i byte of taint source }
      OutBuf = T(i);
    end if
  end for
  if sizeof(InBuf)<T then
    return NULL;
  end if
  return InBuf, OutBuf
```

For the second kind of buffer, since the ciphertext and plaintext has the bijection relationship, In the input and output buffer we can use sliding window method to find two memory blocks which meet the bijection relationship, that is the input and output of a stream cipher function. The algorithm is shown as follow:

**Algorithm 2: The second kind of buffer positioning algorithm**

**Input**：Function execution trace(FuncTrace)，and All read and write of the memory unit (In and Out)
**Output**：if success，return InBuf and Outbuf，else return NULL

1. InitTaintSource(In) // Initialize the continuous memory unit which size is not less than 4 for initial taint
2. TaintAnalysis(FuncTrace)
3. OutBuf = GetBytesWithContinuousLabel(Output) //get the memory unit of Output with continuous label
4. if sizeof(OutBuf)<T then // T is the minimum threshold of input buffer size
5. renturn NULL
6. end if
7. InBuf = GetDependencySource(OutBuf);// get the input buffer of OutBuf dependent
8. return InBuf, OutBuf

The input and output dependency relationship of different cryptographic function is different, which can be used to distinguish the cryptographic function. The input and output buffer size of stream cipher function is equal, and each input byte only affect one output byte; The output buffer size of HASH function is much smaller than output buffer size, and each input byte will affect every output byte; The input and output buffer size of block cipher function is equal, and each input byte will affect every output byte; The characteristic of public key cryptography is similar to block cipher function. Based on the foregoing analysis, the decision tree can be used as shown in Fig. 4 for a cryptographic function division.

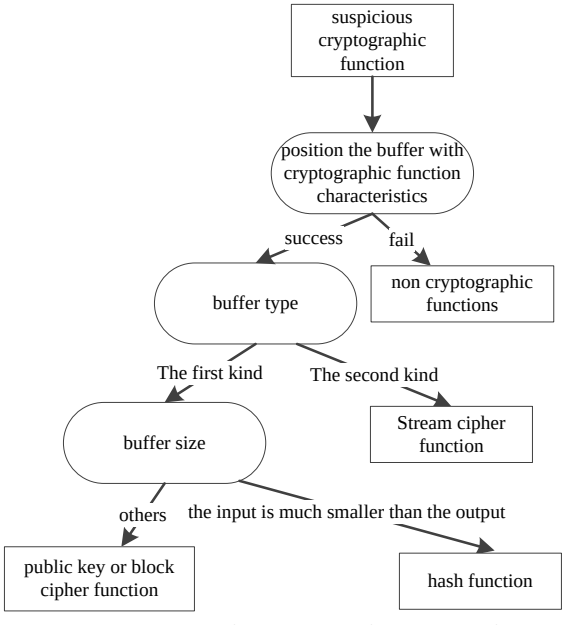


Fig. 4. The decision tree of cryptographic function classification.

Using data flow analysis will only divided suspicious cryptographic function into stream cipher function, Hash function and other large category, then according to the output buffer size, further distinguish can be made, for instance, we can get the decision tree as shown in Fig. 5. For one suspicious cryptographic function, the distinguish result is a set of all the possible types. For example, If the size of the output buffer is16 bytes, then the set is {MD2、MD4、MD5、RIPEMD...}.

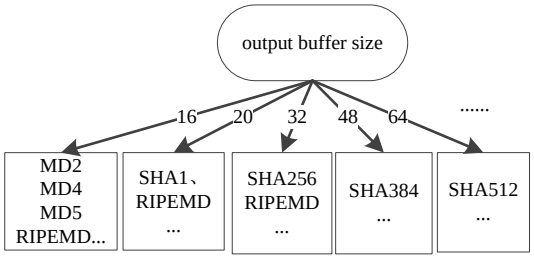


Fig. 5. The Hash function classification according to the buffer size.

### *D. Cryptographic Function Classification Based on Statistical Features*

Different cryptographic function has its own unique characteristics of the instructions statistics, which can be used to identify cryptographic functions. This paper defines a five dimensional feature vector to describe the cryptographic function. The meaning of each vector is shown as follow:

$N$: The total number of instructions.

$R_{arith}$: The proportion of arithmetic operation instructions. The arithmetic instruction reflects the function algorithm operation, Including ADC, ADD, DEC and other instructions.

$R_{logic}$: The proportion of logical operation instructions. Including AND、SHL、SHR、ROL、SAL and other instructions.

$R_{special}$: The proportion of special operation instructions. Public key cryptographic algorithms such as RSA contain many large number operations. Reflecting in the assembly instructions are MUL, IMUL, DIV, IDIV and other arithmetic instructions, which can be used to identify the public key cryptographic function.

$R_{conj}$: The proportion of conditional jump instructions. Through the analysis of cryptographic functions, the proportion of conditional jump instruction is much lower than the general function, Therefore we choose it as one of the cryptographic function characteristics.

TABLE II: PERFORMANCE COMPARISON OF THE CLASSIFICATION ALGORITHM

| Methods | Accuracy | Efficiency | Robustness | Scalability |
|---|---|---|---|---|
| Decision tree | high | fast | poor | poor |
| Artificial eural nets | high | slow | well | well |
| Bayesian | high | fast | moderately | well |
| SVM | high | moderately | moderately | well |

After the five dimensional feature vector of detection function is gained, a kind of data classification model is needed to distinguish the cryptographic function types. The common classification model is decision tree, artificial neural nets, Bayesian and support vector machine. Different classification methods have different feature to be suitable for

different types of data, Evaluation criteria include accuracy, execution efficiency, robustness, scalability and interpretability, Performance evaluation of the above four methods are shown in Table II, Considering the performance and implementation difficulty, this paper choose the Bayesian method to achieve this goal.

The basic process of Bias classification is as follows:

Step 1: Set $x = \{a_1, a_2, \ldots, a_m\}$, $a_i$ is a characteristic attribute of $x$;

Step 2: Get classes set $C = \{y_1, y_2, \ldots, y_n\}$;

Step 3: Calculate the conditional probabilities $P(y_1 \mid x)$, $P(y_2 \mid x)$ ,..., $P(y_n \mid x)$ ;

Step 4: If $P(y_k \mid x) = \max\{P(y_1 \mid x), P(y_2 \mid x), ..., P(y_n \mid x)\}$ , then $x \in y_k$ .

The problem is how to calculate the conditional probabilities in step 3; the following methods can be used:

Step 1: Find a pre classified training sample set;

Step 2: Calculate conditional probability distribution of each attribute: $P(a_1 \mid y_1), P(a_2 \mid y_1), ..., P(a_m \mid y_1), ..., P(a_m \mid y_n)$ ;

Step 3: According to the Bayesian theorem we has the following derivation: $P(y_i \mid x) = \frac{P(x \mid y_i)P(y_i)}{P(x)}$ .

Because the denominators are always the constant, the only need to do is maximizing the molecules, if each characteristic attribute is conditional independent, then

$$P(x \mid y_i)P(y_i) = P(a_1 \mid y_i)...P(a_m \mid y_i)P(y_i) = P(y_i)\prod_{j=1}^{m} P(a_j \mid y_i)$$

According to the above analysis, the steps of cryptographic function identification using the Bias classification as follows:

Step 1: Collect cryptographic algorithm samples through cryptographic algorithm library or Web forum etc, then get characteristics of the collected cryptographic algorithm.

Step 2: Classifier training, the main purpose of this step is generating classifier; calculate occurrence frequency of each class and the Conditional probability estimation of each characteristic attribute to each class.

Step 3: Classify the target function by using classifier. The input is the classifier and the target function, the output is the cryptographic algorithm.

## III. Implementation and Test

### A. Test Environment

In this paper, the test environment is shown in the Table III.

TABLE III: Test Environment

| Environment | Subject | Configuration Information |
|---|---|---|
| Hardware Environment | Processor | Intel(R) Core(TM) i5-2400 CPU@3.10GHz |
| | Memory | 4G |
| | Hard Disk | 500G |
| Software Environment | OS | Windows XP Professional SP3 |
| | Pin | Pin-2.13-61206 |

### B. Function Test — EncryptDemo

EncryptDemo is an own encryption communication program which to test system function. The software is divided into client and server, the client send a plaintext command while the server respond to an encrypted data, and then response to a confirmation packet. Method for encryption of data packet as shown in Fig. 6, Restore the packet encryption to plaintext and calculate after the client receives a data packet, and then make comparisons between the SHA-1 abstract and the data packet. Encryption and decryption function of the client and server based on open-encryption algorithm xyssl-0.8 is realized.

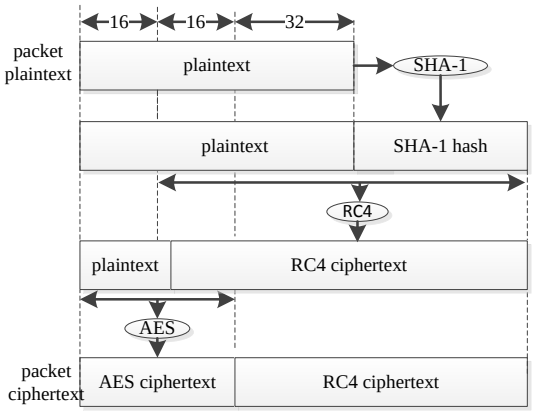


Fig. 6. EncryptDemo protocol format design.

First identify the cryptographic function on the basis of constant functions use and instruction ratio characteristics which the result is as shown in Fig. 7. Using this identification, we read off the SHA1-INIT, SHA1-update and AES-DE Function, in addition it found a cryptographic function seems to be SUB-44110d.

```
==========function==========
thread id:0
address:00441ef5
imm constants:0xc3d2e1f0, 0x10325476, 0x98badcf
description:SHA1_init
==========end function==========

==========function==========
thread id:0
address:00452980
imm constants:0x5a827999, 0x6ed9eba1, 0x8f1bbcd
description:SHA1_update
==========end function==========

==========function==========
thread id:0
address:004413ce
mem constants:(0xf4, 0xa7, 0x50, 0x51),(0x02, 0
description:AES_128_DE, AES_192_DE, AES_256_DE
==========end function==========

==========function==========
thread id:0
address:0044110d
inscount:5561
bitArithRate:0.23
description:suspected crpto function
==========end function==========
```

Fig. 7. EncryptDemo preliminary identification results.

Then use the Taint Analysis to position the encryption and decrtyption buffer of the suspicious cryptographic function, the sub-44110d function first calls the instance of the input and output results as shown in Fig. 8.

```
thread id:0
address:0044110d
inscount:5561
bitArithRate:0.23
memRead(0x43):
004df0a0(37) {96,}
004df0a1(fd) {97,}
004df0a2(e2) {98,}
004df0a3(d3) {99,}
004df0a4(13) {100,}
......
memWrite(0x43):
004df0a0(42) {96,}
004df0a1(42) {97,}
004df0a2(42) {98,}
004df0a3(42) {99,}
004df0a4(42) {100,}
......
```

Fig. 8. EncryptDemo further identification results.

The Fig. 8 omits some memory unit of the input and output, and each output byte depends on each input byte from input and output buffers which are all 0×43 bytes, therefore, we conclude that it is a stream cipher function. According to Bayes identification algorithm, which the probability of RC4 was 87% and the probability of SEAL was 13%, therefore the sub-44110d is RC4 function.

## IV. Conclusion

In the traditional cryptographic function identification algorithm, the static cryptographic function identification method are generally adopt feature matching method, and the cryptographic function can be classified according to his instruction proportion characteristics of cryptographic functions in the dynamic identification method. Meanwhile they both have low precision. Aiming at this problem, this paper presents a cryptographic function identification method which based on data flow analysis. It also introduced the data flow analysis into the cryptographic function identification, and the progressive multi-feature was used to identify the cryptographic function. The experiments results show this is effective for the cryptographic function and its location accuracy is better when compared with the existing method.

## References


[1] J. Caballero, H. Yin, Z. Liang *et al*., "Polyglot: Automatic extraction of protocol message format using dynamic binary analysis," in *Proc. the 14th ACM Conference on Computer and Communications Security*, ACM, 2007, pp. 317-329.

[2] Z. Lin, X. Jiang, D. Xu *et al*., "Automatic protocol format reverse engineering through context-aware monitored execution," *NDSS*, vol. 8, pp. 1-15, 2008.

[3] Z. Wang, X. Jiang, W. Cui *et al*., "ReFormat: Automatic reverse engineering of encrypted messages," in *Proc. European Symposium on Research in Computer Security*, 2009, pp. 200-215.

[4] G. Wondracek, P. M. Comparetti, C. Kruegel *et al*., "Automatic network protocol analysis," *NDSS*. vol. 8, pp. 1-14, 2008.

[5] N. Lutz, *Towards Revealing Attacker's Intent by Automatically Decrypting Network Traffic*, Mémoire de Maıtrise, ETH Zürich, Switzerland, 2008.

[6] F. Calvet, J. M. Fernandez, and J. Y. Marion, "Aligot: Cryptographic function identification in obfuscated binary programs," in *Proc. the 2012 ACM Conference on Computer and Communications Security*, ACM, 2012, pp. 169-182.


**Wei Lin** received the B.S. degree and M.S. degree from Zhengzhou Information Science and Technology Institute, Zhengzhou, China, in 2008 and 2011, respectively. Currently, he is Ph.D. candidate of Zhengzhou Information Science and Technology Institute. His research interest includes software security, protocol analysis and network security.

**Yuefei Zhu** is currently a professor of Zhengzhou Information Science and Technology Institute. His research interests include information security and network security. He is the author or co-author of more than 100 referred international journal and conference papers. He obtained the supports from the National Key Technology Support Program of China, and National Natural Science Foundation.

**Ruijie Cai** received the B.S. degree from Zhengzhou Information Science and Technology Institute, Zhengzhou, China, in 2012. Currently, he is a master degree candidate of Zhengzhou Information Science and Technology Institute. His research interest includes software security and protocol analysis.