Proceedings of the International Multiconference on Computer Science and Information Technology pp. 775–780
ISBN 978-83-60810-27-9
ISSN 1896-7094

Obfuscation Methods with Controlled Calculation Amounts and Table Function

YuanYu Wei
Shibaura Institute of Technology
Toyosu, Koutou-ku Tokyo , 135-8548 Japan
Email: m710101@shibaura-it.ac.jp

Kazuo Ohzeki
Shibaura Institute of Technology
Toyosu, Koutou-ku Tokyo , 135-8548 Japan
Email: ohzeki@sic.shibaura-it.ac.jp

***Abstract*—This paper describes a new obfuscation method with two techniques by which both computational complexity can be controlled and semantic obfuscation can be achieved. The computational complexity can be strictly controlled by using the technique of encryption. The computational complexity can be arbitrarily specified by the impossibility of factorization of prime numbers by length from one second to about one year. Semantic obfuscation is achieved by transforming a function into a table function. A nonlinear, arbitrary function can be incorporated into the functions, while only linear functions are used in the conventional methods. Because the explicit function form is hidden, it is thought that analysis takes time. The computational complexity technique and semantic technique can be used at the same time, and the effect of integrated obfuscation with both techniques is great.Introduction.**

I. Introduction

OBFUSCATION is used for protecting software copyright and hiding ID and passwords when you disclose the software program. Barak showed the impossibility of obfuscation with theoretical proof [1]. Using a virtual black--box, the proof was carried out for an unnatural function. The idea was restricted to such an unnatural function. It is still meaningful to consider obfuscation of software programs beyond the impossibility discussion. Moreover, obfuscation requirements have been upgraded to keep the calculation process secret in public areas. This is called the white-box scheme.Chow et al presented the white-box obfuscation scheme in [3]. document template contains different styles for appropriate text elements. Most of styles are divided into two classes: paragraph styles and symbol styles.

Pa In this paper, we will present both controlling accurate calculation amounts and creating semantic difficulty in conversion from a plain software program to an obfuscated version. For evaluating the calculation amounts of a obfuscated program, we will introduce a cipher method to obtain an accurate evaluation. As for semantic obfuscation, we will introduce a table for a function used in the program. The table function does not present the method of calculation explicitly

Monden [4] presented obfuscation methods according to loops, branches and control of orders. Hachez [5] and Myles [6] presented a wide variety of obfuscation methods including opaque predicates, computation, quality evaluation and applications to watermarking. We studied the loops and branches and then transformed these into complex functions [3]. But these methods were not suitable for reverse-engineering analyses. In this paper, we provide two different methods. One is to control calculation amounts of a software program. The other is to produce semantic obfuscation. We can use both methods in one software program because the two methods correspond to different parts of programs. The former is used for constant values in the program, while the latter is used for numerical functions such as Fourier transform etc. A basic discussion of the former method for calculation amounts has been carried out in technical reports in Japanese [7][8]. Brief results of the former method were presented in [9]. The basic idea of the latter method of semantic obfuscation was presented in [3] but it was a basic method with the name of the ROM function. ROM means 'read only memory'. This means that the function output values are all pre-calculated for a limited number of input discrete variables. Therefore the data of the function operation can be written in read-only-memory (ROM). As a result, the explicit function description disappears and only numerical values can be seen in the program. If the function is a simple linear one, reconstruction of the function formula is easy. But if the function is of second order or more, reconstruction of the function is done by numerical calculation. We can also add fake data into the ROM at the function values of unused input variables. In this paper, the basic method in [3] is implemented by giving a specific function and making part of a table.

In the following section of this paper, we present the obfuscation method by controlling calculation amounts in section 2. Then semantic obfuscation by using table function is presented in section 3.

II. Controlling calculation amounts

A. Encrypting Essential Number (EEN)

It is important to measure the number of calculations needed to analyze and break an obfuscated program. If we can prove the required number of calculations, which is the minimum number, it would be very useful as a component of obfuscation. The method to realize this is that using an intermediate result for a decoding key of RSA encryption, an essential number needed for processing the following calculation is encrypted by an encoding key that corresponds to the decoding key. This method is effective because it cuts the serial calculation flow. No parallel analyses are effective. Consequently, only after obtaining the intermediate value, can the program proceed to the next step by decoding the en-

This work was supported by Universal Robot Corporation.

crypted essential number. The method is called Encrypting Essential Number (EEN).

Fig. 1 shows a block diagram of the ENN software program flow. At first, an appropriate intermediate number "y" is selected. Then decoding key "Y" is decided in relation to "y". If the value "y" is a large prime number, Then Y=y is sufficient. If "y" is not a large prime number, we search for a large prime number Y and get the difference between Y and y as Bias, and decide Y=y+Bias. Then, encoding key "e", which corresponds to "Y", is obtained. To obtain the encoding key, we should decide p, q N in the same way as the RSA encrypting method in the background. Then the essential number "a" is encrypted using encoding key "e", and we get E(a). Next, the essential number - usually it is a constant in the program - is removed from the program, and instead encrypted value E(a) is written where the essential number "a" existed.

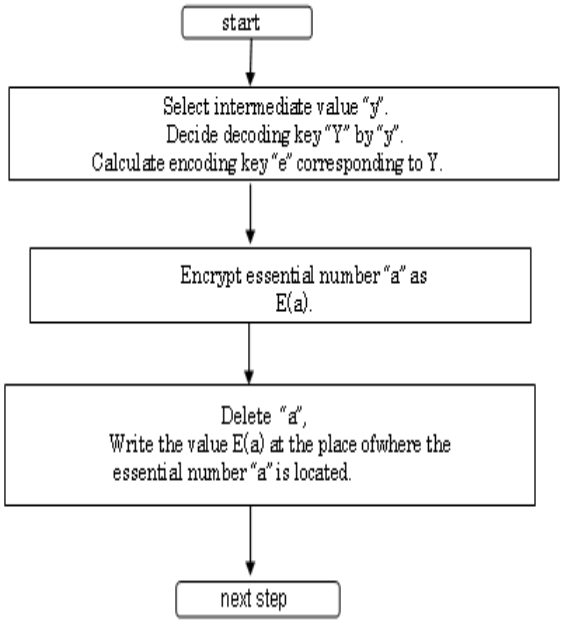

Fig.1 EEN obfuscation block diagram

Fig. 2 shows a transformation from the original program to the proposed method of encrypting essential number (EEN). The numbers of k, z and x are given. The value "y=12" is an intermediate number, which is calculated from the values in the upper part. The value "a=38" is an essential number. To transform the original program first, a prime number 13 related to the value y=12 is selected. The difference between y and 13 is described as Bias, and "Bias=1" is written in the transformed program. Based on the prime number 13, P=38 q=23 are selected in the same manner as the RSA method. Also, the least common multiplier of p-1 and q-1 is calculated as n=396. From N=p*q=851, encrypting key Y=e=61 is obtained. The essential number a=38 is encrypted as a message. In fact, the 61-th power of "a=38" yields an encrypted value as E(a)=3861 (mod 851) =815. In an operation after transforming, the intermediate value y=12 is first obtained. Then the bias value "Bias=1" is added to the intermediate value, and decoding key "d=13" is obtained. The encrypted value "N=815" is raised to 13th power, yielding "a=38".

Before deformation	after deformation
z=k;	z=k;
x=4;	x=4;
y=2*x+4;	y=2*x+4;
a=38;	(a=38;) //hiding the numbers required
B=a*y+3;	Bias=1; // add
	d=y+Bais; //d=Y, add
	N=851; // modulo, add
	$a=815^{d} \pmod{851}$ // $Enc(a)^{d}$ add

Fig.2 Transform from the original program to EEN form

In this way, the hidden essential number can only be obtained by proceeding to the intermediate value position. The minimum required number of calculations is equivalent to calculating E(a)e raised to the d-th power for hidden essential number E(a)e. This kind of exponential calculation can be reduced to a known level by Montgomery multiplication etc.

B. Evaluation of Calculation Amounts

Evaluation of the obfuscation degree in a quantitative way is very difficult. Hachez said that there were several evaluation methods of software complexity, but they were qualitative methods [5]. In the proposed method in this paper, the number of calculations is specified. RSA is used by the fact that there is no way at this moment to decompose the arbitrary number into a product of prime numbers. So the evaluation of this EEN can be described by the number of multiplications. In this EEN method, encoding key "e" need not be disclosed, on the contrary, RSA does not disclose an encoding key. This means that this method is more difficult than RSA. At this moment the excess part is not considered and put forward for further study.

Table I
Examples of decoding key values

intermediate value	digit number of power r	after being raised to the r-th power	decoding key	Bias	index
3.19584	5	319584	319733	-149	i
	5.1	351542	355753	-4211	ii
	5.5	479376	479371	5	iii
	6	3195840	3195869	-29	iv

The evaluation of EEN is carried out by the following rule;

"Amounts of calculation required to decode the encrypted value" (i)

Table 1 shows examples for several specific numbers of large decoding key values. In these examples is a real fractional value of the intermediate number. The intermediate number can be such a real and fractional number. The intermediate number value "3.19584 is shifted to several large

numbers by digit number "r" by an exponential operation. For the case of r=5 at the top row, 3.19584 is raised to the 5-th power to 319584. The exponent values can be fractional as 5.1 or 5.5 as seen in the second and third rows. Next, a decoding key value is selected from an associated value of 319584. In this example, 319733 is selected as an associated value. But it is easy to understand that this value can be freely selected from the intermediate value because they are related to each other by the Bias value. The Bias value is the difference between 319584 and the decoding key value 319733.

Figs. 3(a),(b) show examples of transforming EEN methods of (i) and (ii) in Table 1. Based on these examples, we confirmed actual encoding keys derived from decoding keys, intermediate values in excecution.

For small values of decoding keys, the obtained results are shown in Table 2. A relation between an encoding key and a decoding key is,

$$e \cdot d = 1 \pmod{N}. \tag{2}$$

Obtaining an encoding key from a decoding key is done using the same process as obtaining a decoding key from an encoding key in the RSA method. In obtaining the encoding key, two prime numbers p and q are arbitrary selected. There are many choices in selecting p and q. Also after deciding p and q and the least common multiplier N of (p-1,q-1), there are many choices for selecting an encoding key.

The calculation amounts for this part of the program are decided according to the size of decoding key "d=Y" and the size of module value N. To design the system, it is sufficient to adjust these two sizes.

A detailed transform flowchart of EEN obfuscation from the original software program is shown in Fig. 4. That expression above was described for one unit of obfuscation. Within the frequency of occurrence of the intermediate numbers, requisite times of units can be arranged.

Figure 5 shows a block diagram of the digital watermark system with the type of detector that is disclosed to the public. As for the digital watermark of the image, there are various attacks, and it is difficult to insist on authenticity.

```
z=k;
x=3;
y=x+0.19584;
(a=38;) //Hiding essential number
pw=5; //Exponent multiplier
Bias=149; //added
d=y*10^pw+Bias;
N=modulo; //modulo added
a=Enc(a)^d (mod N); // Enc(a)^d added
B=a*y+3*z;
```

Fig. 3 (a) Detailed program sentences transformed by EEN method with index (i) in Table 1.

```
z=k;
x=3;
y=x+0.19584;
(a=38; //Hiding essential number
pw=5.1; //Exponent multiplier
Bias=4211; //added
d=y*10^5.1.1+Bias; // added
N=modulo; //modulo added
a=Enc(a)^d (mod N); // Enc(a)^d added
B=a*y+3*z;
```

Fig. 3 (b) Detailed program sentences transformed by EEN method with index (ii) in Table 1

Ref [3] proposed by the idea that credibility is sure to be improved in this method to the extent that it opens the detector to the public compared with copyright owners secretly verifying it by themselves. The execution module is obfuscated, though the detector of the digital watermark is a software program. The operation verification is made more difficult by increasing the load in the computational complexity corresponding to an attack that executes the operation analysis coping with slow down in the execution speed. Moreover, semantic obfuscation is being examined as the method replacing the complicated calculation method and the table by ROM function [3]. This uses the input value as is at used positions and uses fake functions at not-used positions. This is different from the DES calculations of Chow [2] with a table realizing as an affine-transformation.

Table Ⅱ
Calculated encoding keys and decoding keys

d	e	p	q	N	lcm
13	61	23	37	851	396
17	233	23	37	851	396
19	667	23	37	851	396
29	437	23	37	851	396
31	511	23	37	851	396
41	425	23	37	851	396
43	571	23	37	851	396
47	455	23	37	851	396

Finally, even if a minute error margin is included, it is a thing that the device such as becoming a correct value can be included because the output of the function of the input value not used is adjusted to the value of the imitation, and it will quantize the result where a discrete calculation is done.

Estimation from the calculation and the example of the execution time of the calculation frequency in obfuscation that provides for the computational complexity, and the design data of the length of the decoding key matched at the required operation time are made. Fig. 6 shows the design time when searching for the encoding key versus the length of the designed decoding key.

Searches for the prime number by the calculation, enable to assume the further estimation for the larger required length of calculation. The value of encoding key e, prime numbers of p and q are selected as each of resembling length comparable to the decoding key length. They are not neces-

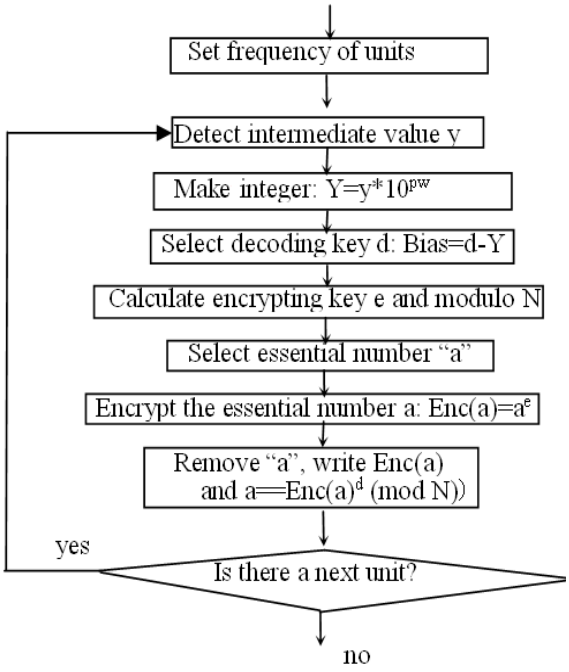

Fig. 4 Detailed transform flowchart of EEN obfuscation from the original software program.

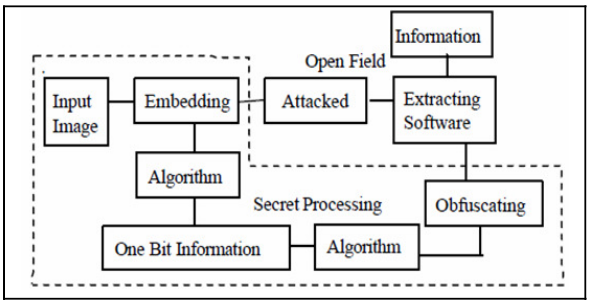

Fig. 5 Watermarking system disclosing a detector.

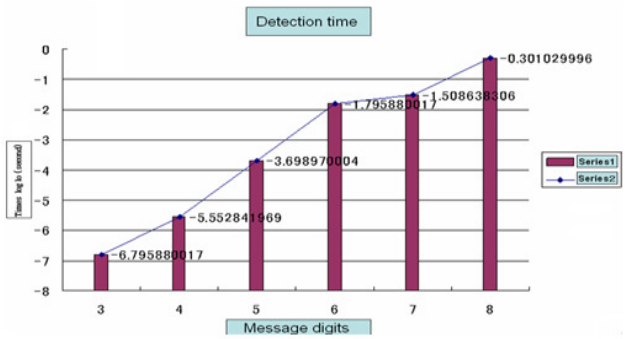

Fig. 6 Time for decoding vs. digit number of decoding key

sarily united because they are made as a design procedure for which it searches from the small number.

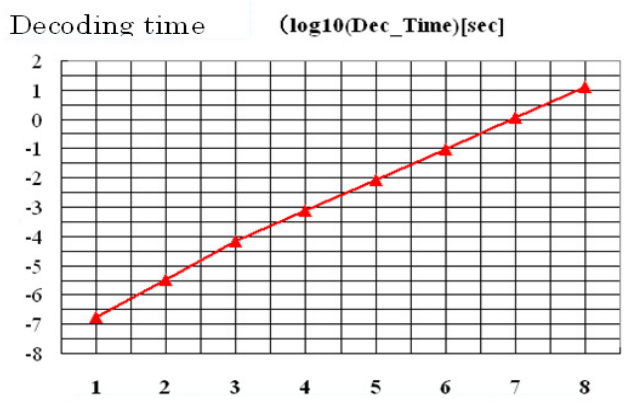

Fig.7 Decoding time vs. length of key number

Fig. 7 shows the decoding times versus the lengths of the decoding keys. The message length had the same length as the decoding key though the message length also influenced the decoding time.

The decoding time for a number composed of eight digits is about one second. The computer specification used was a Pentium 4, 3.2GHz with C language. Modulo operation was used for each multiplication and speed-up processing such as the Montgomery multiplication etc. was not used. The vertical axis was a logarithm at time, and the result for a longer digit number can be worked out by extending the graph if it is almost considered as a straight line.

When Fig. 7 was calculated for 1-3 of the horizontal axis, the digit numbers of decoding key d, encoding key e, and p, q, lcm, N, and the message were calculated as shown in Table 3. For one of four or more of the horizontal axes, almost maximum prime number was used for the digit number of p, q, and decoding key d.

It is on a regular, actual measuring evaluation because the computational complexity depends on the number of digits and the number of multiplies, divisions of modulo.

It is possible to contribute to the improvement of credibility by making the detector of the digital watermark available to the public instead of processing the secret by the closed-door method. It focused only on the increase of computational complexity as the technique for carrying out obfuscation when the detector was disclosed to the public, and the amount was estimated. The program cannot be analyzed in parallel by dividing it, which provides computational complexity of the decoding key.

In this construction, an accurate estimate can be given for computational complexity with exponent power and the modulo calculation. For a computer that uses it to experiment, about one year will cost 13-14 digits since it will cost a day and one month every 12 digits every 10-11 digits from about 10 seconds by eight digits and seconds it about ten times a digit when fast computation is not used.

III. Semantic Method by Table Function

Table 4 shows a comparison of the methods using the table function. Method (i) is a method of [3], and obfuscates the DES operation processing. The DES operation consists of permutation and xor. Because the method of permutation

Table Ⅲ
Calculating data for obtaining decoding time of an EEN unit in a detector. (The values in a bald frame really exist. The others are provisional with lengths of digit numbers

Number Of digit	1	2	3	4	5	6	7	8
p	7	97	997	9973	99991	999983	9999991	99999989
q	5	89	991	9967	99989	999979	9999979	99999971
N	35	8633	988027	99400891	9998000099	999962000357	99999640000243	9999996000109989
n	24	8448	986040	99380952	9997800120	999960000396	99999620000280	9998998800120000
d	5	79	907	9931	99961	999931	9999937	25956377
e	29	1711	97843	84560371	8325115681	552335018575	86684306711833	3826082737342313
a	4	78	900	600	6080	600800	6008000	60080000
E(a)	9	6157	306326	30121083	5305700101	953843280291	86311048530626	5357971538660731

processing is known beforehand, it is possible to decipher it by examining the concealed relation of the I/O and examining the law of permutation. Method (ii) obfuscation is for general, arbitrary functions [3]. An example of the shape of the function is shown in [3]. Moreover, the relation of I/O is not understood for those not used only from observing input and output values because of the error incorporated. Method (iii) is proposed in this paper, basically the same as case of (ii). The example of the function is concretely given, and the shape of the table function for obfuscation is considered in this paper.

To transform a linear function to a table function is not effective obfuscation because the transform unknown coefficients can be obtained from as many numbers of relations between the input and output as those of the unknown coefficients. To transform quadratic polynomial functions or general non-linear functions to table functions can be effective obfuscation because the calculation amounts become huge even though input and output relations are obtained.

In this paper, the function of the Fourier transform etc. is considered for embedding digital watermarks. Because the Fourier transform is a linear transformation, a modified version of a nonlinear transformation that provides pseudo frequency components is newly developed. Examples of the nonlinear transformations are (1) RGB-HSV color conversion, and (2) a non-linear pseudo frequency transform with exponent multiplication.

First, the original RGB-HSV color conversion:

Max=MAX(R,G,B), Min=MIN(R,G,B)

H=undefined if Max=0

$$H=\begin{cases} C_{60}\times\dfrac{G-B}{Max-Min}+C_0, & \text{if } Max=R \\ C_{60}\times\dfrac{B-R}{Max-Min}+C_{120}, & \text{if } Max=G \\ C_{60}\times\dfrac{R-G}{Max-Min}+C_{240}, & \text{if } Max=B \end{cases}$$

H=H+360 if H<0

$$S=\frac{Max-Min}{Max}$$

$$V=Max$$

H=H+360 if H<0

$$S=\frac{Max-Min}{Max}$$

$$V=Max$$

C_* are constants.

Next, a pseudo-frequency transform with exponent multiplication for the case of the least number of I/Os, which has two inputs (x,y) and two outputs (p,q) is introduced;

$$p=a\cdot x^e+b\cdot y^f$$

$$q=c\cdot x^g+d\cdot y^h$$

Here, a, b, c, d, e, f, g, and h are hidden constants. Examples of these are a=1.01, b=0.98, c=1.02, d=-0.99, e=1.01, f=1.02, g=0.997, and h=0.996, as are likewise as the Hadamard transform.

This transform converts signals into a kind of pseudo frequency region. As the exponent part is set to a number near one, the behavior of this transform resembles the Fourier transform. While, the analysis of this transform is still difficult because it is non-linear, the amount of the operation of the analysis of this transform increases remarkably compared with a linear case. For the use of a digital watermark, it need not be an accurate Fourier transform, and quantizing can use pseudo frequency components.

IV. Conclusion

This paper presents a new obfuscation method with two techniques by which both the computational complexity can be controlled and the semantic obfuscation can be achieved. Decoding keys and encrypting keys are actually calculated by using intermediate values and essential values in a software program. Among many choices of selecting encrypting keys from the decoding key, the one whose length is the same as the decoding key is selected. Actual computing times are obtained for the small sizes of encrypting key numbers. For the large sizes of them are obtained from the tendency to a extended graph for smaller sizes. The computational complexity can be arbitrarily specified by the impossibility of factorization on prime numbers by lengths from 1 second to about 1 year. Semantic obfuscation is achieved by transforming a function into a table function. A nonlinear, arbitrary function can be incorporated into the functions, while

Table Ⅳ
Comparison of table functions

method	(i) [2]	(ii) [3]	(iii)
type	permutation XOR	General	Linear Non-linear
error data	non	incorporable	incorporable
object	DES operation with Key	General functions	Matrix Fourier Transform Higher order functions

only linear functions are used in conventional methods. Two functions of the form with exponential power terms are shown. These proposed table functions effectively provide quasi-frequency components like the Fourier transform though these are nonlinear functions. These obfuscation processing methods can be applied also to many applications in signal processing such as embedding digital watermarks etc.

Acknowledgment

The authors thank Eizaburo Iwata for his dicussion to this paper.

References

[1] Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S. and Yang, K.: On the (Im)possibility of Obfuscating Programs, pp. 1-18, CRYPTO (2001).

[2] Chow, S., Eisen, P., Johnson, H., Van Oorschot, P.C.,: A White-Box DES Implementation for DRM Applications, Proceedings of ACM CCS-9 Workshop DRM pp.1-15, Nov., (2002)

[3] Kazuo Ohzeki and Cong Li, “Fingerprinting System Depending On An Anonymous Third Party Authentication Using An Assumption of Computationally Measurable Obfuscation”, IPSJ Tech Rept. CSEC-32, pp.61-66, Mar. 2006. in Japanese

[4] Akito Monden, Yoshihiro Takada and Koji Torii, “Methods for Scrambling Programs Containing Loops”, IEICE Trans D-I Vol. J80-D-I, No.7 pp.1-11 July 1997. in Japanese

[5] Gael Hachez, "A Comparative Study of Software Protection Tools Suited for E-Commerce with Contributions to Software Watermarking and Smart Cards", Thesis submitted to Belgian Catholic University, UCL, March 2003.

[6] Ginger Myles and Christian Collberg, "SoftwareWatermarking via Opaque Predicates: Implementation, Analysis, and Attacks" Proc. 7th International Conference on Electronic Commerce Research, (ICECR-7) Dallas, Texas, June 2004

[7] Engyoku Gi(YuaYu Wei), and Kazuo Ohzeki,” Evaluation of Methods and Computational Amount of Watermarking with Disclosing Obfuscated Detector”, ITE 2009 Winter Symposium 9-6, in Japanese

Engyoku Gi(YuaYu Wei), and Kazuo Ohzeki,”Computational Effort of Watermarking System with Disclosing Obfuscated Detector”, Proc. IMPS 2009 I-6-14, pp.147-148, in Japanese.

[8] Kazuo Ohzeki and Engyoku Gi (YuanYu Wei),” An Obfuscation Method for a Detector of Watermarking “, IPSJ Tech Rept. 2010-DPS142CSEC48-(30) Vol.142, pp.1-7, in Japanese.

[9] Yuanyu Wei and Kazuo Ohzeki,“A New Obfuscation Method Using Random Functions ” proceedings of Telecommumications,Networks and Systems, IADIS International conferences. P263-265,July 2010.