Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Cryptanalysis of Boolean permutation-based key escrow scheme

Amr M. Youssef [*]

Concordia Institute for Information Systems Engineering, Concordia University, 1515 St. Catherine Street West, Ev7.638, Montreal, Quebec, Canada H3G 1M8

A R T I C L E I N F O

Article history:
Received 15 March 2008
Received in revised form 16 January 2009
Accepted 23 April 2009
Available online 30 May 2009

Keywords:
Boolean functions
Public key encryption systems
Trapdoor permutations

A B S T R A C T

Wu and Varadharajan [Computers and Electrical Engineering 25(4) 1999] proposed a fast public key escrow scheme. The security of this system is based on a special class of trapdoor Boolean permutations that can be constructed efficiently. In this paper, we show that this proposed class of Boolean permutations can be easily inverted without the knowledge of the secret key parameters. This allows the cryptanlyst to efficiently recover the session key using the known public key parameters.

1. Introduction

Public key cryptosystems based on the integer factorization and the discrete log problems, such as RSA and ElGamal [2], require a large number of arithmetic operations which make them less efficient compared to symmetric key systems that are based on Boolean functions operations. Several attempts were made to construct fast public key systems based on Boolean functions (e.g., [3–6]). However, the majority of these systems were broken (e.g., [7–9]). For a comprehensive survey of several other related schemes, the reader is referred to [1].

In general, inverting a nonlinear n-variable mapping (Boolean permutation with n variables) is equivalent to solving a set of nonlinear Boolean equations which is an NP-complete problem. Designing trapdoor Boolean permutations aims to finding a class of efficiently computable random-looking permutations, with compact representation, that can be easily inverted given the knowledge of some trapdoor information and, at the same time, are hard to invert without this information. If such a trapdoor Boolean permutation is constructed, then it can be easily used to design public key systems in which the trapdoor information represents the user's private key.

In [10], Wu and Varadharajan proposed a key escrow scheme using a public key system constructed from a family of Boolean permutations. The security of this scheme is based on the trapdoor property of this class of permutations. An obvious advantage of this cryptosystems is that both the encryption and decryption operations can be made very efficient using a simple hardware which makes it attractive in several applications, especially in resource constrained environments such as smart cards, RFIDs and sensor networks. Similar schemes were proposed by the same authors in [11].

In this paper, we present a practical attack that allows the cryptanalyst to efficiently recover the session key from the known public key parameters. In particular, we show that finding the input, corresponding to a given output of the trapdoor

* Tel.: +1 514 848 2424.
E-mail address: youssef@ciise.concordia.ca

0045-7906/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compeleceng.2009.04.004

permutations proposed by Wu and Varadharajan, is equivalent to solving a system of linear equations over Z_2 which renders this class of permutation unsuitable for use in any public key system.

2. The proposed trapdoor permutation

A trapdoor function, f, is a function that is easy to compute in one direction, yet believed to be difficult to compute in the opposite direction without the knowledge of its trapdoor information. In other words, given x, it is easy to evaluate $y = f(x)$, but given y, it is hard to evaluate $x = f^{-1}(y)$ without the trapdoor information. Trapdoor functions are widely used in public key cryptography where several function classes have been proposed. It should be noted that trapdoor functions which can be computed efficiently are harder to find than was initially thought. For example, an early suggestion was to use schemes based on the NP-hard knapsack subset sum problem but it was soon realized that the underlying knapsacks often have a low density and hence they are vulnerable to lattice reduction attacks [2].

Since the security of Wu and Varadharajan key escrow scheme is based only on the difficulty of inverting their proposed trapdoor permutations, we focus only on the generation and properties of these permutations. The other features of the cryptosystem that are not relevant to our attack are not discussed in here.

Lemma 1. *Let*

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_1 \\ \boldsymbol{a}_2 \\ \vdots \\ \boldsymbol{a}_n \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

be a non singular $n \times n$ matrix. Let $l_i(\boldsymbol{x}) = \boldsymbol{a}_i \cdot \boldsymbol{x}_i = a_{i1}x_1 \oplus \cdots \oplus a_{in}x_n$. Then $L = [l_1, \cdots_n]$ forms a linear Boolean permutation.

It is clear that finding the inverse permutation of L is equivalent to finding the inverse of the matrix $\boldsymbol{A}$. We will use the notation $L \leftrightarrow \boldsymbol{A}$ to denote the fact that the linear permutation L is derived from the matrix A.

The following algorithm [10] gives an iterative method for constructing Boolean permutations based on old ones.

Algorithm 1. Let $P(\boldsymbol{x}) = [f_1(\boldsymbol{x}), \cdots, f_n(\boldsymbol{x})], \boldsymbol{x} = (x_1, \cdots, x_n) \in Z_2^n$ be a Boolean permutation of order n. Set $q_i(\boldsymbol{x}, x_{n+1}) = f_i(\boldsymbol{x}) \oplus x_{n+1} \oplus g(\boldsymbol{x}), i = 1, \cdots, n$, and $q_{n+1}(\boldsymbol{x}, x_{n+1}) = x_{n+1} \oplus g(\boldsymbol{x}), x_{n+1} \in Z_2$. Then $Q = [q_1, \cdots, q_{n+1}]$ is a Boolean permutation of order $n + 1$.

In general, given the algebraic description of the output coordinates of a random Boolean permutation, R, finding its inverse, R^{-1}, is a hard problem. In [10], it is shown how to efficiently evaluate the inverse permutation Q^{-1} using the knowledge of g and P^{-1}.

It should be noted that any cryptographic weakness (such as linearity) in the output coordinates of P is inherited by all nonzero linear combinations of even number of output coordinates of Q, since the same function g is used to mask all the coordinates of P.

Algorithm 2. Construction of trapdoor Boolean permutations

- Construct two linear Boolean permutations $L_1 \leftrightarrow A$ and $L_2 \leftrightarrow B$ of order $n - 2$ and compute their inverses.
- Run Algorithm 1 twice using L_1 to have a Boolean permutation Y of order n and compute its inverse Y^{-1}
- Repeat the above step using L_2 to get another Boolean permutation Z of order n and compute its inverse Z^{-1}.

The target trapdoor Boolean permutation is $P(\mathbf{x}) = Z(Y(\mathbf{x}))$ and its inverse is also available as $P^{-1}(\mathbf{x}) = Y^{-1}(Z^{-1}(\mathbf{x}))$. In [10], the authors discussed some techniques for selecting the function g used by Algorithm 1 so that the resulting permutation P has a compact algebraic representation. In here, we do not discuss these techniques since our attack can be applied irrespective of any choice of g.

Wu and Varadharajan [10] argued that there is no efficient algorithm to evaluate $P^{-1}(\mathbf{x})$ without the knowledge of the secret parameters used by Algorithms 1 and 2 (i.e., the L's and g's). Based on this argument, they suggested that the above algorithm can be used by establish a key escrow scheme based on a public key system in which user U generates P during the key setup procedure using the above Algorithms. The public key is the algebraic description of the Boolean permutation P and the private key is the algebraic description of P^{-1}. When another user, say Alice, wants to communicate with U, Alice selects a random binary string $\mathbf{k}$ of length n and sends $\mathbf{e} = P(\mathbf{k})$ to user U. The session key $\mathbf{k}$ can be recovered by U using her private key, P^{-1}. For further details about the system, the reader is referred to [10].

3. The proposed attack

Let $P(\mathbf{x}) = Z(Y(\mathbf{x})) = [p_1(\mathbf{x}), \cdots, p_n(\mathbf{x})]$ denote a Boolean permutation constructed using Algorithm 2 above. Let $\mathbf{x}' = (x_1 \cdots x_{n-2}), \mathbf{y}' = (y_1 \cdots y_{n-2}), \mathbf{x}'' = (\mathbf{x}', x_{n-1}), \mathbf{y}'' = (\mathbf{y}', y_{n-1}), \mathbf{x} = (\mathbf{x}'', x_n)$, and $\mathbf{y} = (\mathbf{y}'', y_n)$. Then we have

$$
\begin{array}{lcl}
y_1(\mathbf{x}) & = & \mathbf{a}_1 \cdot \mathbf{x}' \oplus x_{n-1} \oplus g_1(\mathbf{x}') \oplus x_n \oplus g_2(\mathbf{x}'') \\
y_2(\mathbf{x}) & = & \mathbf{a}_2 \cdot \mathbf{x}' \oplus x_{n-1} \oplus g_1(\mathbf{x}') \oplus x_n \oplus g_2(\mathbf{x}'') \\
\vdots & \vdots & \vdots \\
y_{n-2}(\mathbf{x}) & = & \mathbf{a}_{n-2} \cdot \mathbf{x}' \oplus x_{n-1} \oplus g_1(\mathbf{x}') \oplus x_n \oplus g_2(\mathbf{x}'') \\
y_{n-1}(\mathbf{x}) & = & x_{n-1} \oplus g_1(\mathbf{x}') \oplus x_n \oplus g_2(\mathbf{x}'') \\
y_n(\mathbf{x}) & = & x_n \oplus g_2(\mathbf{x}'')
\end{array} \tag{1}
$$

$$
\begin{array}{lcl}
z_1(\mathbf{y}) & = & \mathbf{b}_1 \cdot \mathbf{y}' \oplus y_{n-1} \oplus g_3(\mathbf{y}') \oplus y_n \oplus g_4(\mathbf{y}'') \\
z_2(\mathbf{y}) & = & \mathbf{b}_2 \cdot \mathbf{y}' \oplus y_{n-1} \oplus g_3(\mathbf{y}') \oplus y_n \oplus g_4(\mathbf{y}'') \\
\vdots & \vdots & \vdots \\
z_{n-2}(\mathbf{y}) & = & \mathbf{b}_{n-2} \cdot \mathbf{y}' \oplus y_{n-1} \oplus g_3(\mathbf{y}') \oplus y_n \oplus g_4(\mathbf{y}'') \\
z_{n-1}(\mathbf{y}) & = & y_{n-1} \oplus g_3(\mathbf{y}') \oplus y_n \oplus g_4(\mathbf{y}'') \\
z_n(\mathbf{y}) & = & y_n \oplus g_4(\mathbf{y}'')
\end{array} \tag{2}
$$

We start by explaining how one can find the input $\mathbf{e}$ corresponding to a given known output $\mathbf{o} = (o_1 \cdots o_n) = Y(\mathbf{e})$, where the coordinates of Y are given by Eq. (1). Given the algebraic description of $y_i(\mathbf{x}), 1 \leqslant i \leqslant n-1$, one can form the following system of $n-2$ equations

$$o_1 \oplus o_j = y_1(\mathbf{x}) \oplus y_i(\mathbf{x}), i = 2, \cdots n-1.$$

It should be noted that $y_1(\mathbf{x}) \oplus y_i(\mathbf{x}) = (\mathbf{a}_1 \oplus \mathbf{a}_i) \cdot \mathbf{x}'$. Thus the above system of equations is linear over Z_2 and hence it can be solved in $O(n^3)$ operations. Once the solution $\mathbf{e}' = (e_1 \cdots e_{n-2})$ is determined, one can easily determine e_{n-1} and e_n by computing $Y(\mathbf{e}', 0, 0)$ and $Y(\mathbf{e}', 0, 1), Y(\mathbf{e}', 1, 0)$ and $Y(\mathbf{e}', 1, 1)$ and then compare the result to $\mathbf{o}$. We emphasis the fact that the above system of equations is formed without the need to know any of the secret parameters used in the generation process of the Boolean permutation Y. Our attack is a generalization of the above idea.

Given the encrypted session key $\mathbf{e} = (e_1, \cdots e_n) = P(\mathbf{k})$, the attacker executes the following steps to recover $\mathbf{k}$

– Construct the set of $\binom{n-1}{2}$ equations in the form

$$e_i \oplus e_j = p_i(\mathbf{x}) \oplus p_j(\mathbf{x}),$$

where $1 \leqslant i \leqslant n-1, 1 \leqslant j \leqslant n-1, i \neq j$.
For consistency of notation, let $\mathbf{b}_{n-1} = \mathbf{0}$. Note that $p_i(\mathbf{x}) \oplus p_j(\mathbf{x}) = (\mathbf{b}_i \oplus \mathbf{b}_j) \cdot y'(\mathbf{x})$. Let $w = wt(\mathbf{b}_i \oplus \mathbf{b}_j)$, where $wt(\cdot)$ denote the Hamming weight of the enclosed argument. If w is odd, the resulting equation is likely to be nonlinear, i.e., it will contain terms with algebraic degree higher than 1. On the other hand, if w is even, then, from (1) and (2), we have

$$e_i \oplus e_j = p_i(\mathbf{x}) \oplus p_j(\mathbf{x}) \Rightarrow e_i \oplus e_j = (\mathbf{a}_{i_1} \oplus \cdots \oplus \mathbf{a}_{i_w}) \cdot \mathbf{x}.$$

– Remove all the nonlinear equations from the above set.
– Solve the remaining set of linear equations. Let $K = \{\mathbf{k}_1, \cdots, \mathbf{k}_M\}$ denote the set of obtained solutions. Note that the session key $\mathbf{k} \in K$ because $\mathbf{k}$ also must satisfy this set of linear equations.
– Using the known public key, P, the session key can be uniquely determined by noting that $P(\mathbf{k}_i) = \mathbf{e} \Rightarrow \mathbf{k} = \mathbf{k}_i$.

The size of the set K is determined by the rank, r, of the obtained system of linear equations. For, $6 \leqslant n \leqslant 16$, our experimental results, using 1000 random permutations generated using Algorithm 2, the rank of the obtained system of equations was consistently equal to $r = n-3$. In other words, $|K| = 2^{n-r} = 2^3 = 8$ solutions. Finding an analytical expression for the expected value of r does not seem to be an easy combinatorial problem. Furthermore, it is not going to change the conclusion of the paper given the consistency of the obtained experimental results above.

Because of its simplicity, this attack allows the cryptanalyst to recover the session key, from its encrypted value, in few seconds for any practical choice of n. A toy example that illustrates the steps described in the above attack is given in the Appendix.

4. Conclusions

The class of permutations proposed by Wu and Varadharajan can be easily inverted without the knowledge of its secret trapdoor parameters and hence it is not suitable for use as a trapdoor permutation.

Despite that fact that almost all similar previous attempts to construct trapdoor permutations using Boolean functions have failed, it is somewhat premature to claim that Boolean permutations and Boolean functions are not suitable for constructing public key systems, especially since each one of the previous proposals have failed for a different reason.

The main problem in Wu and Varadharajan's approach is the reliance on linear functions, L_1 and L_2 in Algorithm 2, to construct their trapdoor permutations. Also, as discussed above, using the same nonlinear function, g, in Algorithm 1, to hide this inherent linearity was not enough. As shown by our attack, simple algebraic manipulation of the coordinate functions of Wu and Varadharajan's Boolean permutations allows the cryptanalyst to get rid of all the nonlinear terms introduced by this nonlinear Boolean function and easily invert the permutation by solving a set of linear equations over Z_2. After all, linearity was and will remain the curse of cryptographers.

Finally, we believe that finding an efficient Boolean function based trapdoor permutation still remains a very interesting and challenging practical research problem.

Appendix.

Example 1. Consider the permutation P of order $n = 5$ constructed using Algorithm 2 with the following parameters: $g_1(\mathbf{x}') = x_1 \oplus x_1x_3, g_2(\mathbf{x}'') = x_2 \oplus x_1x_2 \oplus x_3 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_4 \oplus x_2x_4 \oplus x_1x_2x_4 \oplus x_3x_4 \oplus x_1x_3x_4, g_3(\mathbf{x}') = x_2 \oplus x_1x_3, g_4(\mathbf{x}'') = x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_4 \oplus x_1x_2x_4 \oplus x_3x_4 \oplus x_1x_3x_4 \oplus x_2x_3x_4 \oplus x_1x_2x_3x_4$,

$$A1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \text{ and } A2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Then the public key $P = [p_1(\mathbf{x}), p_2(\mathbf{x}), p_3(\mathbf{x}), p_4(\mathbf{x}), p_4(\mathbf{x})]$ will be given by:

$$\begin{aligned}
p_1(\mathbf{x}) &= x_1 \oplus x_2 \oplus x_1x_2 \oplus x_3 \oplus x_1x_3 \oplus x_4 \oplus x_2x_4 \oplus x_1x_2x_4 \oplus x_2x_3x_4 \oplus \\
&\quad x_1x_2x_3x_4 \oplus x_5 \oplus x_1x_5 \oplus x_1x_2x_5 \oplus x_3x_5 \oplus x_1x_3x_5 \oplus x_1x_2x_3x_5 \\
p_2(\mathbf{x}) &= x_2 \oplus x_1x_2 \oplus x_3 \oplus x_1x_3 \oplus x_4 \oplus x_2x_4 \oplus x_1x_2x_4 \oplus x_2x_3x_4 \oplus \\
&\quad x_1x_2x_3x_4 \oplus x_5 \oplus x_1x_5 \oplus x_1x_2x_5 \oplus x_3x_5 \oplus x_1x_3x_5 \oplus x_1x_2x_3x_5 \\
p_3(\mathbf{x}) &= x_1x_2 \oplus x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_4 \oplus x_1x_4 \oplus x_1x_2x_4 \oplus x_3x_4 \oplus \\
&\quad x_2x_3x_4 \oplus x_1x_2x_3x_4 \oplus x_1x_5 \oplus x_1x_2x_5 \oplus x_3x_5 \oplus x_1x_3x_5 \oplus x_1x_2x_3x_5 \\
p_4(\mathbf{x}) &= x_2 \oplus x_1x_2 \oplus x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_4 \oplus x_1x_4 \oplus x_1x_2x_4 \oplus x_3x_4 \oplus \\
&\quad x_2x_3x_4 \oplus x_1x_2x_3x_4 \oplus \oplus x_1x_5 \oplus x_1x_2x_5 \oplus x_3x_5 \oplus x_1x_3x_5 \oplus x_1x_2x_3x_5 \\
p_5(\mathbf{x}) &= x_1 \oplus x_2 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_2x_3 \oplus x_4 \oplus x_2x_4 \oplus x_1x_2x_4 \oplus \\
&\quad x_2x_3x_4 \oplus x_5 \oplus x_1x_5 \oplus x_3x_5 \oplus x_1x_3x_5 \oplus x_1x_2x_3x_5
\end{aligned}$$

Given the following encrypted key $\mathbf{e} = P(\mathbf{k} = (1\,1\,0\,1\,1)) = (0\,1\,0\,1\,1)$, the cryptanalyst can construct the following system of equations

$$\begin{aligned}
&e_1 \oplus e_2 = 1 = p_1(\mathbf{x}) \oplus p_2(\mathbf{x}) = x_1 \\
&e_1 \oplus e_3 = 0 = p_1(\mathbf{x}) \oplus p_3(\mathbf{x}) = x_1 \oplus x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_5 \\
&e_1 \oplus e_4 = 1 = p_1(\mathbf{x}) \oplus p_4(\mathbf{x}) = x_1 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_5 \\
&e_2 \oplus e_3 = 1 = p_2(\mathbf{x}) \oplus p_3(\mathbf{x}) = x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_5 \\
&e_2 \oplus e_4 = 0 = p_2(\mathbf{x}) \oplus p_4(\mathbf{x}) = x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_4 \oplus x_3x_4 \oplus x_5 \\
&e_3 \oplus e_4 = 1 = p_3(\mathbf{x}) \oplus p_4(\mathbf{x}) = x_2
\end{aligned}$$

By removing the nonlinear equations from the above set, the remaining linear equations are $x_1 = 1 \Rightarrow k_1 = 1$ and $x_2 = 1 \Rightarrow k_2 = 1$. Note that the rank of this linear system of equations is $2 = n - 3$. Then the cryptanalyst evaluate $P(1, 1, k_3, k_4, k_5)$ for all the 8 choices of (k_3, k_4, k_5) to obtain

$$\begin{aligned}
&P(1\,1\,0\,0\,0) = 01100 \\
&P(1\,1\,1\,0\,0) = 01010 \\
&P(1\,1\,0\,1\,0) = 11101 \\
&P(1\,1\,1\,1\,0) = 10010 \\
&P(1\,1\,0\,0\,1) = 10111 \\
&P(1\,1\,1\,0\,1) = 01101 \\
&P(1\,1\,0\,1\,1) = 01011 \\
&P(1\,1\,1\,1\,1) = 10101
\end{aligned}$$

Note that $P(1\,1\,0\,1\,1) = 01011 = \mathbf{e} \Rightarrow \mathbf{k} = 11011$.

References

[1] Ding J, Gower JE, Schmidt D. Multivariate public key cryptosystems (advances in information security). 1st ed. Springer-Verlag; 2006.
[2] Menezes AJ, van Oorschot PC, Vanstone SA. Handbook of applied cryptography. CRC Press; 1996.
[3] Imai H, Matsumoto T. Algebraic methods for constructing asymmetric cryptosystems. In: Proceedings of algebraic algorithms and error correcting codes (AAECC-3). LNCS, vol. 229. Springer-Verlag; 1986. p. 108–19.
[4] Patarin J, Goubin L. Asymmetric cryptography with S-Boxes. In: Proceedings of ICICS 97. LNCS, vol. 1334. Springer; 1997. p. 369–80.

[5] Patarin J. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms, advances in cryptology. In: Proceedings of Eurocrypt 96. LNCS, vol. 1070. Springer-Verlag; 1996. p. 33–48.
[6] Patarin J, Goubin L. Trapdoor one-way permutations and multivariate polynomials, advances in cryptology. In: Proceedings of ICICS 97. LNCS, vol. 1334. Springer-Veralg; 1997. p. 356–68.
[7] Biham E. Cryptanalysis of patarins 2-round public key system with S Boxes(2R), advances in cryptology. In: Proceedings of Eurocrypt 2000, LNCS, vol. 1807, Springer-Verlag; 2000. p. 408–16.
[8] Feng Y, Yan L, Duo D. Cryptanalysis of 2R schemes, advances in cryptology. In: Proceedings of crypto 99, LNCS, vol. 1666, Springer-Verlag; 1999. p. 315–25.
[9] Patarin J. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt88, advances in cryptology. In: Proceedings of crypto 95, LNCS, vol. 963; 1995, pp. 248–61.
[10] Wu C, Varadharajan V. Boolean permutation-based key escrow. Comput Electr Eng 1999;25(4):291–304.
[11] Wu C, Varadharajan V. Public key cryptosystems based on boolean permutations and their applications. Int J Comput Math 2000;74(2):167–84.