Cryptanalysis of an ID-based Authenticated Dynamic Group Key Agreement with Optimal Round

Qingfeng Cheng[1,3] and Chunming Tang[2]
(Corresponding author: Qingfeng Cheng)
Department of Language Engineering, Luoyang University of Foreign Languages[1]
Luoyang 471003, P.R. China
(Email: qingfengc2008@sina.com)
School of Mathematics and Information Science, Guangzhou University[2]
Guangzhou 510006, P.R. China
State Key Laboratory of Information Security, Institute of Software,Chinese Academy of Sciences[3]
Beijing 100093, P.R. China
(Received Apr. 28, 2013; revised and accepted Mar. 15 & Apr. 25, 2014)

Abstract

Recently, Teng, Wu and Tang proposed a new ID-based authenticated dynamic group key agreement (DGKA) protocol. They claimed that leaving users cannot calculate subsequent group session keys and joining users cannot calculate previous group session keys. In this paper, we will show that Teng et al.'s protocol cannot provide forward confidentiality or backward confidentiality.

Keywords: Backward confidentiality, bilinear pairing, dynamic group key agreement, forward confidentiality

1 Introduction

Ingemarsson et al. [4] first introduced the concept of group key agreement (GKA). Afterward, many group key agreement protocols have been proposed [1, 2, 3, 5]. In particular, some of them are designed for dynamic groups, which are called dynamic group key agreement (DGKA) protocols. Secure DGKA protocols must provide the fundamental security requirements for general GKA protocols, and also should encompass the following two requirements [6, 8]:

- Forward confidentiality: While a group user leaves from the current group, he should not be able to calculate the new session key.
- Backward confidentiality: While a new user joins into the current group, he should not be able to calculate the previous session key.

Recently, Teng, Wu and Tang [7] proposed a new ID-based authenticated DGKA protocol, called Teng-Wu-Tang protocol. They proved the Teng-Wu-Tang protocol's security in the random oracle model. In addition, they also claimed that the leaving users cannot obtain information about subsequent new group session keys and joining users cannot obtain information about previous group session keys. In this paper, we will demonstrate that the Teng-Wu-Tang protocol is not secure. Though the Teng-Wu-Tang protocol's join algorithm only requires one round communication and its leave algorithm does not require exchange message, the Teng-Wu-Tang protocol cannot provide forward confidentiality or backward confidentiality. It means that the Teng-Wu-Tang protocol is infeasible for real-life implementation.

2 Review of the Teng-Wu-Tang Protocol

In this section, we briefly review the Teng-Wu-Tang protocol, which is composed of the following three stages as well as the join algorithm and the leave algorithm. For more details, refer to [7].

2.1 System Initialization Stage

Let q be a large prime, G_1 and G_2 be two groups with the same order of q. P is a generator of G_1 and Q is randomly chosen from G_1. $\hat{e} : G_1 \times G_1 \to G_2$ is a bilinear pairing and $H : \{0,1\}^* \to G_1^*$ is a hash function. Key generation center (KGC) randomly chooses the master private key $s \in Z_q^*$ and computes $P_{pub} = sP$ as the master public key. The system parameters are $\{q, G_1, G_2, P, Q, \hat{e}, H, P_{pub}\}$.

2.2 Key Extract Stage

This phase is run by the KGC for each user with an identity $ID_i \in \{0,1\}^*$. The KGC first computes $Q_i = H(ID_i)$, and then computes the user's private key $S_i = sQ_i$.

2.3 Key Agreement Stage

Let $\{U_1, \ldots, U_n\}$ be the initial group of n users. The key agreement stage is described below:

1) Each user $U_i (1 \le i \le n)$ define the $(n-1) \times n$ matrix

$$A = \begin{pmatrix} a_2 \\ a_3 \\ \cdot \\ \cdot \\ \cdot \\ a_n \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & \cdot & \cdot & \cdot & 0 \\ 1 & 0 & 1 & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & & \\ 1 & \cdot & \cdot & \cdot & 1 & 0 & \\ 1 & 0 & 0 & \cdot & \cdot & 0 & 1 \end{pmatrix}$$

2) Each user $U_i (1 \le i \le n)$ randomly chooses $r_i \in Z_q^*$ and computes $P_i = r_i P, V_{i_j} = r_i Q'_j (1 \le j \le n, j \ne i)$, where $Q'_j = Q + Q_j$. Then user U_i broadcasts P_i, V_{i_j}.

3) Each user $U_i (1 < i < n)$ sets two vectors $a_{i_1} = (0, ..., 0, 1_i, 0, 0, ..., 0)$, which denotes ith element is 1, and $a'_{i_1} = (0, ..., 0, 1_{i+1}, 0, 0, ..., 0)$ with n elements, which denotes $i+1$th element is 1, and then defines two $n \times n$ matrixes A_i and A'_i as follows:

$$A_i = \begin{pmatrix} a_{i_1} \\ a_2 \\ \vdots \\ a_n \end{pmatrix}, \qquad A'_i = \begin{pmatrix} a'_{i_1} \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

User U_1 sets a vector $a'_{1_1} = (0, 1, 0, ..., 0)$ with n elements. User U_n also sets a vector $a_{n_1} = (0, 0, ..., 1)$ with n elements. Then U_1 and U_n define two matrixes A'_1 and A_n respectively as follows:

$$A'_1 = \begin{pmatrix} a_{1'_1} \\ a_2 \\ \vdots \\ a_n \end{pmatrix}, \qquad A_n = \begin{pmatrix} a_{n_1} \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

Two matrixes A_i and A'_i are nonsingular due to $|A_i| \ne 0$ and $|A'_i| \ne 0$, where $|\cdot|$ denotes the determinant of a matrix. Let $(x_1, x_2, \ldots, x_n)$ be the solution of $X \times A_i = (1, 1, ..., 1)$ and $(x'_1, x'_2, \ldots, x'_n)$ be the solution of $X \times A'_i = (1, 1, ..., 1)$.

Further, U_i defines the $(n-1) \times (n-1)$ matrix M_i as follows:

$$M_i = \begin{pmatrix} V_{1_2} & \cdots & V_{(i-1)_1} & V_{(i+1)_1} & \cdots & V_{n_1} \\ V_{1_3} & \cdots & V_{(i-1)_2} & V_{(i+1)_2} & \cdots & V_{n_2} \\ & \vdots & & \vdots & & \\ V_{1_n} & \cdots & V_{(i-1)_n} & V_{(i+1)_n} & \cdots & V_{n_{(n-1)}} \end{pmatrix}$$

It means that messages V_{c_d} from the cth column is received from user c, when $c < i$, and also messages $V_{(c+1)_d}$ from the cth column is received from user $c+1$, when $c \ge i$.

Next, U_i sets two matrixes $M_{i,1}$ and $M_{i,2}$, which are composed of the first $i-1$ columns of the matrix M_i and the other columns of the matrix M_i respectively.

$$M_{i,1} = \begin{pmatrix} r_1(Q+Q_2) & \cdots & r_{i-1}(Q+Q_1) \\ r_1(Q+Q_3) & \cdots & r_{i-1}(Q+Q_2) \\ \vdots & & \vdots \\ r_1(Q+Q_{i-2}) & \cdots & r_{i-1}(Q+Q_{i-2}) \\ r_1(Q+Q_{i-1}) & \cdots & r_{i-1}(Q+Q_i) \\ \vdots & & \vdots \\ r_1(Q+Q_n) & \cdots & r_{i-1}(Q+Q_n) \end{pmatrix}$$

$$M_{i,2} = \begin{pmatrix} r_{i+1}(Q+Q_1) & \cdots & r_n(Q+Q_1) \\ r_{i+1}(Q+Q_2) & \cdots & r_n(Q+Q_2) \\ \vdots & & \vdots \\ r_{i+1}(Q+Q_i) & \cdots & r_n(Q+Q_i) \\ r_{i+1}(Q+Q_{i+2}) & \cdots & r_n(Q+Q_{i+1}) \\ \vdots & & \vdots \\ r_{i+1}(Q+Q_n) & \cdots & r_n(Q+Q_{n-1}) \end{pmatrix}$$

With $(x_1, x_2, \cdots, x_n)$ and $(x'_1, x'_2, \cdots, x'_n)$, U_i computes

$$\hat{Q}_{i,1} = (x_2, x_3, \cdots, x_n) M_{i,1} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{n-1}$$

$$\hat{Q}_{i,2} = (x'_2, x'_3, \cdots, x'_n) M_{i,2} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{n-1}$$

Finally, $U_i (1 < i < n)$ computes the group session key

$$sk = sk_{i,1} \cdot sk_{i,2} \cdot \hat{e}(P_{pub}, r_i \overline{Q_i}) = \prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i},$$

where

$$sk_{i,1} = \hat{e}(\sum_{j=1}^{i-1} P_j, x_1 S_i) \cdot \hat{e}(P_{pub}, \hat{Q}_{i,1}),$$

$$sk_{i,2} = \hat{e}(\sum_{j=i+1}^{n} P_j, x'_1 S_i) \cdot \hat{e}(P_{pub}, \hat{Q}_{i,2})$$

and $\overline{Q_i} = Q + Q_1 + Q_2 + \cdots + Q_{i-1} + Q_{i+1} + \cdots + Q_n$ $(1 \le i \le n)$.

User U_1 computes the group session key $sk = sk_{1,2} \cdot \hat{e}(P_{pub}, r_1 \overline{Q_1}) = \prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}$ and user U_n computes the group session key $sk = sk_{n,1} \cdot \hat{e}(P_{pub}, r_n \overline{Q_n}) = \prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}$.

2.4 Join Algorithm

Let $\{U_1, \ldots, U_n\}$ be the current group and $\{U_{n+1}, \ldots, U_{n+m}\}$ be the set of joining users. For generating the new group session key, each user $U_i (1 \le i \le n+m)$ first defines a new $(n+m-1) \times (n+m)$ matrix A as follows:

$$A = \begin{pmatrix} a_2 \\ a_3 \\ \cdot \\ \cdot \\ \cdot \\ a_m \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & \cdot & \cdot & \cdot & 0 \\ 1 & 0 & 1 & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & & \\ 1 & \cdot & \cdot & \cdot & 1 & 0 & \\ 1 & 0 & 0 & \cdot & \cdot & 0 & 1 \end{pmatrix}$$

Then each user $U_i (1 \le i \le n)$ computes $r_i(Q + Q_{n+j})(1 \le j \le m)$, where r_i is chosen in the key agreement stage. Then user $U_i (1 \le i \le n)$ broadcasts $r_iP, r_i(Q + Q_{j'})(1 \le j' \le n+m, j' \ne i)$, where r_iP and $r_i(Q + Q_{j'})(1 \le j' \le n, j' \ne i)$ are computed in the key agreement stage. At the same time, user $U_{n+j} (1 \le j \le m)$ randomly chooses $r_{n+j} \in Z_q^*$, computes and broadcasts $P_{n+j} = r_{n+j}P, V_{(n+j)_{j'}} = r_{n+j}(Q + Q_{j'})(1 \le j \le m, 1 \le j' \le n+m, j' \ne n+j)$, where r_{n+j} is kept secretly.

Finally, each user $U_i (1 \le i \le n+m)$ computes the new group session key as $sk = \prod_{i=1}^{i=n+m} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}$, where $\overline{Q_i} = Q + Q_1 + Q_2 + \cdots + Q_{i-1} + Q_{i+1} + \cdots + Q_{n+m}$.

2.5 Leave Algorithm

Let $\{U_{m+1}, \ldots, U_n\}$ be the set of leaving users and $\{U_1, \ldots, U_m\}$ be the current group. For generating the new group session key, each user $U_i (1 \le i \le m)$ first defines a new $(m-1) \times m$ matrix A as follows:

$$A = \begin{pmatrix} a_2 \\ a_3 \\ \cdot \\ \cdot \\ \cdot \\ a_m \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & \cdot & \cdot & \cdot & 0 \\ 1 & 0 & 1 & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & & \\ 1 & \cdot & \cdot & \cdot & 1 & 0 & \\ 1 & 0 & 0 & \cdot & \cdot & 0 & 1 \end{pmatrix}$$

Then each user $U_i (1 \le i \le m)$ defines the new $(m-1) \times (m-1)$ matrix M_i', which includes the first $m-1$ rows and the first $m-1$ columns of matrix M_i.

Finally, each user $U_i (1 \le i \le m)$ computes the new group session key as $sk = \prod_{i=1}^{i=m} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}$, where $\overline{Q_i} = Q + Q_1 + Q_2 + \cdots + Q_{i-1} + Q_{i+1} + \cdots + Q_m$.

3 Cryptanalysis of the Teng-Wu-Tang Protocol

In this section, we show that the Teng-Wu-Tang protocol is not secure. Here, we only consider the simplest case, i.e. a joining user and a leaving user. In the join algorithm, a new joining user can use his private key and ephemeral key to recover the accepted group session key generated by the former group users before he joined the group. In the leave algorithm, a leaving user can use his private key and ephemeral key to compute the new group session key generated by the new group users after he left the group.

3.1 Attack on the Backward Confidentiality

Let $\{U_1, \ldots, U_n\}$ be the set of the current group users and U_{n+1} be the new joining user. From the key agreement stage, we know that $U_1, U_2 \ldots, U_{n-1}$ and U_n have shared the group session key $sk = \prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}$, where $\overline{Q_i} = Q + Q_1 + Q_2 + \cdots + Q_{i-1} + Q_{i+1} + \cdots + Q_n$. In order to keep the previous encrypted messages secretly, $U_1, U_2 \ldots, U_n$ and U_{n+1} must use the join algorithm to generate a new shared group session key $sk' = \prod_{i=1}^{i=n+1} \hat{e}(P_{pub}, \overline{Q_i'})^{r_i}$, where $\overline{Q_i'} = Q + Q_1 + Q_2 + \cdots + Q_{i-1} + Q_{i+1} + \cdots + Q_n + Q_{n+1}$.

If U_{n+1} is a malicious user, he can use his private key S_{n+1}, ephemeral key r_{n+1} and the new group session key sk' to recover the previous group session key.

Since

$$\begin{aligned} sk' &= \prod_{i=1}^{i=n+1} \hat{e}(P_{pub}, \overline{Q_i'})^{r_i} \\ &= \left[\prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i'})^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}} \\ &= \left[\prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i} + Q_{n+1})^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}} \\ &= \left[\prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}\right] \left[\prod_{i=1}^{i=n} \hat{e}(P_{pub}, Q_{n+1})^{r_i}\right] \\ &\qquad \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}} \\ &= sk \left[\prod_{i=1}^{i=n} \hat{e}(P_{pub}, Q_{n+1})^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}} \\ &= sk \left[\prod_{i=1}^{i=n} \hat{e}(P, sQ_{n+1})^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}} \\ &= sk \left[\prod_{i=1}^{i=n} \hat{e}(r_iP, sQ_{n+1})\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}} \\ &= sk \left[\prod_{i=1}^{i=n} \hat{e}(P_i, S_{n+1})\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}}, \end{aligned}$$

the malicious user U_{n+1} can compute $\prod_{i=1}^{i=n} \hat{e}(P_i, S_{n+1})$ and $\hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}}$ with his private key S_{n+1} and ephemeral key r_{n+1}, where $P_i = r_iP (1 \le i \le n)$ is a public message and $\overline{Q_{n+1}'} = Q + Q_1 + Q_2 + \cdots + Q_n$.

Then the malicious user U_{n+1} can compute the previous group session key as follows:

$$sk = \frac{sk'}{\left[\prod_{i=1}^{i=n} \hat{e}(P_i, S_{n+1})\right] \hat{e}(P_{pub}, \overline{Q_{n+1}'})^{r_{n+1}}}$$

Clearly, the new joining user U_{n+1} has successfully computed the previous group session key generated be-

fore he joined the current group. Therefore, the Teng-Wu-Tang protocol cannot provide backward confidentiality.

3.2 Attack on the Forward Confidentiality

Let $\{U_1, \ldots, U_n\}$ be the set of the current group users and U_n be the leaving user. From the key agreement stage, we know that $U_1, U_2 \ldots, U_{n-1}$ and U_n have shared the group session key $sk = \prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}$, where $\overline{Q_i} = Q+Q_1+Q_2+\cdots+Q_{i-1}+Q_{i+1}+\cdots+Q_n$. In order to keep the future encrypted messages secretly, $U_1, U_2 \ldots, U_{n-2}$ and U_{n-1} must use the leave algorithm to generate a new shared group session key $sk'' = \prod_{i=1}^{i=n-1} \hat{e}(P_{pub}, \overline{Q_i''})^{r_i}$, where $\overline{Q_i''} = Q + Q_1 + Q_2 + \cdots + Q_{i-1} + Q_{i+1} + \cdots + Q_{n-2} + Q_{n-1}$.

If U_n is a malicious user, he can write the current group session key sk as follows:

$$
\begin{aligned}
sk &= \prod_{i=1}^{i=n} \hat{e}(P_{pub}, \overline{Q_i})^{r_i} \\
&= \left[\prod_{i=1}^{i=n-1} \hat{e}(P_{pub}, \overline{Q_i})^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= \left[\prod_{i=1}^{i=n-1} \hat{e}(P_{pub}, \overline{Q_i''} + Q_n)^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= \left[\prod_{i=1}^{i=n-1} \left[\hat{e}(P_{pub}, \overline{Q_i''})^{r_i} \hat{e}(P_{pub}, Q_n)^{r_i}\right]\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= \left[\prod_{i=1}^{i=n-1} \hat{e}(P_{pub}, \overline{Q_i''})^{r_i}\right] \left[\prod_{i=1}^{i=n-1} \hat{e}(P_{pub}, Q_n)^{r_i}\right] \\
&\qquad \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= sk'' \left[\prod_{i=1}^{i=n-1} \hat{e}(P_{pub}, Q_n)^{r_i}\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= sk'' \left[\prod_{i=1}^{i=n-1} \hat{e}(r_i P_{pub}, Q_n)\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= sk'' \left[\prod_{i=1}^{i=n-1} \hat{e}(r_i sP, Q_n)\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= sk'' \left[\prod_{i=1}^{i=n-1} \hat{e}(r_i P, sQ_n)\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n} \\
&= sk'' \left[\prod_{i=1}^{i=n-1} \hat{e}(P_i, S_n)\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n}.
\end{aligned}
$$

Since S_n is the private key of user U_n and r_n is selected by user U_n, he can compute $\prod_{i=1}^{i=n-1} \hat{e}(P_i, S_n)$ and $\hat{e}(P_{pub}, \overline{Q_n})^{r_n}$, where $P_i = r_i P (1 \le i \le n-1)$ is a public message and $\overline{Q_n} = Q + Q_1 + Q_2 + \cdots + Q_{n-1}$. Finally, the malicious user U_n can compute the new group session key as follows:

$$
sk'' = \frac{sk}{\left[\prod_{i=1}^{i=n-1} \hat{e}(P_i, S_n)\right] \hat{e}(P_{pub}, \overline{Q_n})^{r_n}}
$$

Clearly, the leaving user U_n has successfully computed the new group session key generated after he left the current group. Therefore, the Teng-Wu-Tang protocol cannot provide forward confidentiality.

4 Conclusion

In this paper, we have pointed out that the Teng-Wu-Tang protocol fails to provide forward confidentiality and backward confidentiality. It means that a leaving user can calculate the future session key and a joining user can calculate the previous session key. So the Teng-Wu-Tang protocol is not suitable for practical applications.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61202317, 11271003), Province Natural Science Foundation of Guangdong (Grant No. S2012010009950), High Level Talents Project of Guangdong, the Young Key Teacher Foundation of Henan Province's Universities (Grant No. 2012-GGJS-157), and the Foundation of State Key Laboratory of Information Security (No. 2014-11). The authors would like to thank the anonymous referees for their helpful comments.

References

[1] A. H. Ahmed, M. Ali, O. B. Luis, "Authenticated group key agreement protocols for Ad hoc wireless networks," *International Journal of Network Security*, vol. 4, no. 1, pp. 90–98, 2007.

[2] T. Y. Chang, M. S. Hwang, W. P. Yang, "Cryptanalysis of the Tseng-Jan anonymous conference key distribution system without using a one-way hash function," *Information & Security: An International Journal*, vol. 15, no. 1, pp. 110–114, 2004.

[3] S. Hong, "Queue-based group key agreement protocol," *International Journal of Network Security*, vol. 9, no. 2, pp. 135–142, 2009.

[4] I. Ingemarsson, D. Tang, C. Wong, "A conference key distribution system," *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 714–720, 1982.

[5] J. Katz, M. Yung, "Scalable protocols for authenticated group key exchange," *Journal of Cryptology*, vol. 20, no. 1, pp. 85–113, 2007.

[6] S. Michael, T. Gene, and W. Michael, "Key agreement in dynamic peer groups," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 8, pp. 769–780, 2000.

[7] J. K. Teng, C. K. Wu, C. M. Tang, "An ID-based authenticated dynamic group key agreement with optimal round," *Science China Information Sciences*, vol. 55, no. 11, pp. 2542–2554, 2012.
[8] Y. M. Tseng, "A communication-efficient and fault-tolerant conference-key agreement protocol with forward secrecy," *Journal of Systems and Software*, vol. 80, no. 7, pp. 1091–1101, 2007.

Qingfeng Cheng received his B.A. degree in 2000 and M.S. degree in 2004 from National University of Defense Technology, and Ph.D. degree in 2011 from Information Engineering University. He is now an Associate Professor with the Department of Language Engineering, Luoyang University of Foreign Languages. His research interests include cryptography and information security.

Chunming Tang who is born in 1972, and is a professor in Guangzhou University in P.R. China. He received his Bachelors degree in Xiangtan Normal University in 1995, then became an assistant in same university. He received his masters degree from Xiangtan University in 2001, and his Mathematics PhD from Chinese Academy of Sciences in 2004. Since then, he works in Guangzhou University, where he became an associate professor in 2005 and a professor in 2009. His research field is cryptology and cloud computing.