20. L. Y. Ren, Study on Scheduling Optimization in Construction Project of Lagerstroemia Hope City. Xi'an University of architecture & technology. 2011,6,pp.12-17.
21. Y. Yona, M. Feder, "Efficient parametric decoder of low-density lattice codes," IEEE International Symposium on Information Theory: June 28-July 3, 2009, Seoul, Korea. New York, NY, USA: IEEE, 2009, pp.744-748.
22. B. Kurkoski, J. Dauwels, "Message-passing decoding of lattices using Gaussian mixtures," IEEE International Symposium on Information Theory: June 6-11, 2008, Toronto, Canada. New York, NY, USA: IEEE, 2008, pp.2489-2493.
23. BICKSON D, IHLER A, AVISSAR H et al. A low-density lattice decoder via non-parametric belief propagation. Forty-Seventh Annual Allerton Conference on Communication, Control and Computing: Sep 30-Oct 2, 2009, Illinois, USA. Monticello, IL, USA: IEEE, 2010, pp.439-446.
24. J. He, Y. Geng and K. Pahlavan, Modeling Indoor TOA Ranging Error for Body Mounted Sensors, 2012 IEEE 23nd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Sydney, 2012,pp. 682-686.
25. S. Li, Y. Geng, J. He, K. Pahlavan,Analysis of Three-dimensional Maximum Likelihood Algorithm for Capsule Endoscopy Localization, 2012 5th International Conference on Biomedical Engineering and Informatics (BMEI), Chongqing, 2012 ,pp.721-725

Design of RFID Security Authentication Protocol for E-Commerce Service

Xiaoli He[1], Yu Song[2], Ralf Volker Binsack[3]

1.School of Computer Science, Sichuan University of Science and Engineering, Zigong Sichuan 643000, China;
2.Department of Network Information Management Center, Sichuan University of Science and Engineering, Zigong Sichuan 643000, China
3.Winterstein 14,Niddatal,61194,Germany

Abstract
As for security of RFID network, this paper has proposed a safe, effective and scalable RFID Authentication Protocol (CRFID) with cloud database as server. Firstly, the tree-structure management tag is used to achieve privacy protection, and the time complexity of RFID system in search of cloud database is reduced from O (N) to O (logN); then, the size of each subkey in keys route of the RFID tags is increased from 4 to 60 digits to prevent tracking attacks; finally, reader determines whether the key stored in the tag is updated through the feedback message of the

tag to prevent desynchronization attacks. Simulation results show that this scheme greatly enhances the security level of the key path while reducing the search complexity.
Keywords: RFID, CLOUD DATABASE, TIME COMPLEXITY, DESYNCHRONIZATION ATTACK, TRACKING ATTACK

Introduction

Radio Frequency IDentification (RFID) system has received much concern in recent years, originally used for military purposes, now gradually replacing the optical bar code system. Compared to barcode system, RFID system has several advantages, such as cloud data management, parallel processing, short access time, long-distance non-contact sensor and re-writable et. [1]. RFID system is very easy with great efficiency, and has been applied in numerous dimensions, such as supply chain, car door locks, product sales, as well as e-passports, etc. RFID system consists of three components: electronic tag (transmitter), back-end database server of saving the associated information of tag and RFID reader (receiver), and tag scanning and database query are conducted via wireless connection [2]. Back-end database server allows access to all of the RFID readers with strong parallel computing capabilities in order to meet simultaneous query of all readers, and it is also known as cloud database, also saving hash value or identity code and key form of key while calculating and storing information. The information transmission between tag and reader are open, and if the information stored in the tag is not protected by proper security measures, then there may be a risk [3]. A relatively simple method of safe data transmitting is that the tag sends the meaningless message, that is, hash value or data encryption. When a valid reader acquires meaningless news, the back-end server can be used to search for the identity tag and associated detailed information [4], while invalid reader cannot connect to the back-end server, thus attackers would have no way to get the detailed information of tag through meaningless messages. However, because the tag has response to the same hash value each time, attackers can be able to successfully trace the label if the same hash value is received again.

1. Proposed CRFID Protocol

This paper presents a scalable, secure and efficient RFID authentication protocol, called CRFID, overcoming security vulnerabilities in existing methods, and Table 1 is the symbol used herein.

The program consists of two phases: initialization and reading. In the initialization phase, a unique key pair k_i and s_i is assigned to each tag T_i. Meanwhile,

Table 1. Symbols used in this paper

Symbol	Meaning
R	RFID reader
Tag	RFID tag
n_i	Characteristic random number of i
l_n	The length of n_i
$H()$	One-way hash function
l_h	Length of the output value $H()$
k_i	Key path of T
$k_i[x]$	The x- subitem of key path
l_k	The length of a subitem of key path
s_i	Tag key
δ	Branching factor of key tree
T	Key tree in database
d	Depth of key tree
c	TagJoin run times
N	The total number of tags
V	List of victims

readers construct a pseudo key tree. It is noted that the label itself does not contain any output information to show that the success or failure of the authentication process. Therefore, any reader cannot determine whether the protocol is executed successfully. In order to address this serious shortcoming, a communication process is designed in step 8 of the proposed algorithm, and the CRFID reading protocol is as follows:

$$
\begin{aligned}
&1.R \to T: \text{ask}, n_1; n_1 \xleftarrow[R]{} \{1\}^n \\
&2.T \quad : U = \{n_2, H(n_1 \| n_2 \| k_i[0]), H(n_1 \| n_2 \| k_i[1]), \\
&\quad ..., H(n_1 \| n_2 \| k_i[d-1]), H(n_1 \| n_2 \| s_i)\}; n_2 \xleftarrow[R]{} \{1\}^n \\
&3.T \to R: U \\
&4.R \quad : \{i, k_i, s_i\} \leftarrow Identify(U) \\
&5.R \quad : add\{k_i, s_i\} toList\ L. \\
&\quad : \{k_i', s_i'\} \leftarrow KeyGen(k_i, s_i, i) \\
&6.R \to T: \sigma = \{H(n_1 \| n_2 \| k_i) \oplus k_i', H(n_1 \| n_2 \| s_i \| k_i)\} \\
&7.T \quad : \{k_i', s_i'\} \leftarrow NewVerify(\sigma, n_1, n_2) \\
&\quad : Ends. \\
&\quad : k_i \leftarrow k_i', s_i \leftarrow s_i' \\
&8.T \to R: \alpha = H(n_1 \| n_2 \| k_i' \| k_i) \\
&9.R \quad : Test\ \alpha\ ?= H(n_1 \| n_2 \| k_i' \| k_i) \\
&\quad : Fause \\
&\qquad \text{Test.} \\
&\quad : Or: \\
&\qquad k_i \leftarrow k_i', s_i \leftarrow s_i' \\
&\qquad \text{Add } (k_i'', s_i'') \text{ to } L, \\
&\qquad TagLeave(k_i'', s_i'') \\
&\quad Clear\ L
\end{aligned}
$$

In addition, if the reader does not receive the message confirmed by the tag, it will read the label again (step 9). In addition to initialization and reading process, the program in this paper also considers adding a new label and verifying the updated key (keygen and New Verify is algorithm 1 and 2 respectively).

1.1. Initialization

A number of parameters shall be determined before the running of system, including depth of key tree d, branch coefficient δ, and the bit length l_k of each sub-key. The length of the key path is $d \times l_k$. δ and l_k are mainly chosen based on systems and security issues [14]. The value of selected parameter (d, δ, l_k) in this paper) is (4,32,60).

Supposed there are N labels, the input of KeyGen function is the identity i of each label and two random strings k_i and s_i. KeyGen algorithm is shown in Table 2. k_i' and s_i' are outputted the security tag of label i. After processing all labels, the key tree is formed, and each leaf represents several tags (possibly only one) [15].

Table 2. KeyGen algorithm

Algorithm 1: KeyGen
1: **Input:** tag ID i, key path k_i, key s_i (k_i and s_i are random strings in initialization) 2: **Output:** new key path k_i' and new key s_i'. 3: **Internal variable:** key tree T and the depth d,. victims list V 4: Assuming M is the root of T 5: **for** j←0,1,…,d-1 **do** 6: **if** deg(M)=δ **then** 7: r←{1…δ} 8: M ← Mr, in which Mr is the r-th sub-item of M. 9: **else** 10: Create node M ‘and attach it to M. 11: Define label (M ‘) ←. $R\{1\}^{l_k}$ 12: M←M’ 13: **end if** 14 $k_i'\{k\}$ ← tag (M) 15:**end for** 16: **if** the label (M) is the same as one in list V then 17: Perform **for** loop. 18:**end if** 19: $s_i' \leftarrow H(k_i \| k_i' \| s_i)$ 20:**return** k_i', s_i'.

2. Security

Since CRFID protocol will update critical path and key after each successful reading, the complexity of key number in tag is O (1), and the reader can identify the leaf node only by matching the hash value d from the U. Each hash value needs at most δ times of hash calculation, and each leaf node contains a few labels, complexity of the search overall $O(\delta \times d)^2$. Therefore, CRFID has three properties: recoverable captured label, fixed size of key and efficiency of search. The process of CRFID resisting the attack of two main RFIDs is as follows:

(1) Resist tracking attack: CRFID increases the size of each subitem of k_i from 4 to 60 digits, while maintaining the same δ, namely the same search complexity. This can largely avoid tracking attack, and the attacker cannot extract the value of each subitem from σ. Figure 1 shows the relationship between l_k and number of searches, and it can be seen that if l_k is increased to 30, the value can still forcibly reverse, tag tree N being 0.1, 1 or 10 million.

(2) Resist desynchronization attack: Step 8 in CRFID reading protocol is used to withstand desynchronization attack. When R receives messages generated in step 8, it can be determined that the key stored in T has been updated. If the message fails to send to R, then T does not update the key, and terminates the protocol, or the updating key of T is(k_i' s_i',) (R generates key correspondingly). In some cases, R will reread T, until it receives the message generated in step 8. That is, the previously generated keys are invalid, thus avoiding false label (it does not exist in the record).

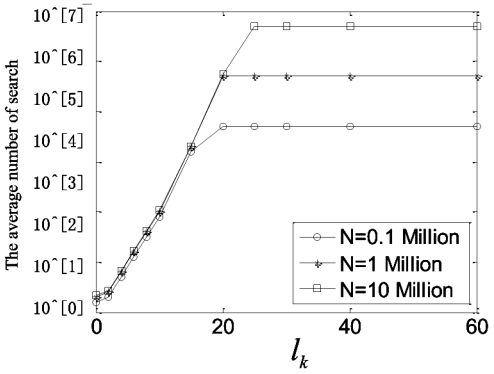

Figure 1. Relation between $k_i[]$, l_k length and search complexity

2.1. Protocol Security

If $U \neq U'$ or $n_1 \neq n_1'$, emulator will reject the message and terminates the reader R. We use the event E_1 to express rejection message of reader error (i.e., the emulator refuses the protocol that attacker destroys successfully). If the message is rejected, the attacker will choose to output σ'.

If $U=U'$ or $n_1=n_1'$, emulator will accept the message and continue implementation of the protocol. Simulator outputs $\sigma = \{\sigma_1, \sigma_2\}$ to the attacker, in which σ_1 and σ_2 are the random numbers with the length of l_h. Then the attacker returns to σ'. If σ'

is different from σ, the emulator will reject σ'. If the emulator refuses U', it will also refuse σ'. According to the algorithm 2, σ' is actually correct, but the emulator rejects as false, using event E_2 to indicate this situation.

Table 3. New Verify algorithm

Algorithm 2: NewVerify
1: **Input:** $\sigma=\{\sigma_1,\sigma_2\}$, n_1, n_2. 2: **Output:** new key path k_i' and new key s_i'. 3: **Internal variable:** the current key path k_i and key s_i.. 4: $k_i' \leftarrow \sigma_1 \oplus H(n_1 \| n_2 \| k_i)$ 5:**if** $\sigma_2 \neq H(n_1 \| n_2 \| s_i \| k_i')$ **then** 6: **return Fail**; 7:**end if** 8: $s_i' \leftarrow H(k_i \| k_i' \| s_i)$ 9:**return** k_i', s_i'

Finally, the simulator sends the random number α of length l_h to the attacker, the attacker replies α', and when the two are unequal, the emulator rejects. The case that α' should not be rejected, and the emulator rejects it is called event E_3.

If the attacker has not changed any news, the emulator assigns key k_i and s_i randomly to the label.

By definition, the overall probability of a successful attack on protocol is $\Pr(E_1)+\Pr(E_2)+\Pr(E_3)$. The following will show that each item of the above formula is very small.

$\Pr(E_1)$: only in the case that U_s' is legitimate, E_1 will occur, and the following situations may be:

(1) The loading of the same n_1 and n_2 has been failed, and the likelihood is, $q_R \times 2^{-l_n}$ in which q_R is the number of all readers allowed by the attacker and l_n is the length of n_1 and n_2. It is noted that although n_2 can be chosen by the attacker, n_1 is randomly selected by the emulator and cannot be predicted.

(2) The attacker has successfully searched for the hash value to the correct key. Since all the news in the protocol are not associated with s_i, the likelihood of this is $q_H \times 2^{-l_k}$, in which q_H is the total number of queries launched by the attacker.

$$U=\{n_2, H(n_1\|n_2\|k_i[0]), H(n_1\|n_2\|k_i[1]),\ldots,$$
$$H(n_1\|n_2\|k_i[d-1]), H(n_1\|n_2\|s_i)\} \quad (1)$$

R identifies Tag through Identify algorithm. Identify algorithm starts from the root, and repeated examination is made whether the tree mark of child node of the target node can meet the equation (2) or not.

$$H(n_1\|n_2\|labelM_r)=H(n_1\|n_2\|k_i[j]) \quad (2)$$

(3) The attacker outputs legitimate U_s', but never gets it in a hash query or in the previous communication. The possibility of such a case is.

Thus, it can be seen from the above that

$$\Pr(E_1)\le q_R\times 2^{-l_n}+q_H\times 2^{-l_k}+2^{-l_h} \quad (3)$$

$\Pr(E_2)$: Only if σ_2 is valid, E_2 could happen. There are three cases that may occur:

(1) The same n_1 and n_2 has been used before, and the same σ can be reused by the attacker, and the probability of its occurrence is $q_R \times 2^{-l_n}$.

(2) If the attacker obtains a hash value in $H(n_1\|n_2\|s_i'\|x)$, a legitimate σ can be created. However, since all the news in the protocol are not associated with s_i, the likelihood of such a case is only $q_H \times 2^{-l_k}$.

(3) The attacker does not generate a legitimate σ_2 through hash function, and the likelihood will not exceed 2^{-l_h}.

Thus, it can be seen from the above that

$$\Pr(E_2)\le q_R\times 2^{-l_n}+q_H\times 2^{-l_k}+2^{-l_h} \quad (4)$$

$\Pr(E_3)$: Only if α is valid, it could happen. There are three cases:

(1) α is the preceding output of the simulator, and the attacker cannot control n_1 and n_2, its possibility of $q_R \times 2^{-2l_n}$.

(2) The attacker obtains a hash value in $H(n_1\|n_2\|k_i'\|k_i)$, and the probability of occurrence is $q_H \times 2^{-2l_k}$.

(3) α is unrelated to previous communications queries, and the probability of occurrence not exceeds 2^{-l_h}.

Thus, it can be seen from the above that

$$\Pr(E_3)\le q_R\times 2^{-2l_n}+q_H\times 2^{-2l_k}+2^{-l_h} \quad (5)$$

According to the above, all the possibilities that the protocol is attacked are

$$\Pr(A)\le q_R\times 2^{-l_n}(2^{-l_n}+2)$$
$$+q_H\times 2^{-l_k}(2^{-l_k}+2)+3\cdot 2^{-l_h} \quad (6)$$

This is the polynomial related to number of hash queries q_H and on-line reading q_R, in which l_n, l_k and l_h are quite large, so the result is very small and can be ignored.

Note that, because each tag has a different key, so Formula (6) is independent of the number of the captured labels. When (l_n, l_k, l_h, q_H) are valued at (60, 60, 60, 216, 230), the likelihood of successful attack on CRFID is less than2^{-29}.

2.2. Privacy

When the label is illegally read, the attacker can get a set of tags ϑ that can generate messages. Privacy function P (c) is used to represent the privacy performance of various programs. P (c) is defined as $1/|\vartheta|$, in which c is the number of the captured label. Ideal privacy protection program can generate function $P(c)=1/(N-c)$ and $\forall c \le N-1$, in which N is the number of all labels in the system.

In legal reading (passive monitoring) and illicit reading (active scanning), the message $U=(n_2,U_0,U_1,...,U_{d-1},U_s)$ only provides information to identify the label. If the attacker does not capture any label, it is difficult to confirm whether the two communication data streams come from the same label, unless the attacker can guess the key path k_i or key s_i, or the failed reading for two times using the same n_1 and n_2 (less likely). Therefore, it is considered $P(0)\approx 1/N$.

The more the captured tag is, the more the information of the leaked path keywords will be. C is set to be set of the captured tag, and K is the set of path keywords subitems obtained from C. Assumed that key pair of tag T_i is k_i and s_i, the output message after receiving the illegal reading is $U_0,U_1,...,U_{d-1}$, defined as $E_{x,y}$: When $k_i[0],k_i[1],...,k_i[x-1]\in K$, but $k_i[x],k_i[x+2],...,k_i[d-1]\notin K$, and y nodes below $k_i[x]$ are captured.

For the event $E_{x,y}$, the average $\bar{N}_{x,y}$ of the rest *N-C-1* is taken for the collection ϑ, calculated as:

$$\bar{N}_{x,y}=1+(\delta-y)(N-c-1)/\delta^x \tag{7}$$

To calculate the likelihood of $E_{x,y}$, we need to make the following analysis. δ^x boxes are filled with c balls. C is set to be a group of δ boxes, and the empty probability of y boxes in C is calculated, and for instance, there is a particular box being empty in C. β is defined to be the number of non-empty boxes. Pr ($E_{x,y}$) can be calculated by the following formula as function $f(c,y,0)$:

$$f(c,y,\beta)=\frac{\delta-\beta-1}{\delta^x}f(c-1,y-1,\beta+1)$$

$$+\frac{\delta^x-\delta+\beta}{\delta^x}f(c-1,y,\beta) \tag{8}$$

$$f(1,1,\beta)=\frac{\delta-\beta-1}{\delta^x} \tag{9}$$

$$f(c,0,\beta)=\left(\frac{\delta^x-\delta+\beta}{\delta^x}\right)^c \tag{10}$$

$$f(u,v,\beta)=0,\forall u<v \tag{11}$$

Formula (8) represents that there is a total of c balls, and β boxes in the C are filled up while y boxes need to be filled, which can be achieved in two ways. In the first case, a ball is put into a box in C, and then there are β+1 boxes filled by c-1 balls, and y-1 boxes need to be filled. In the second case, a ball is not put into the empty box, and then there are c-1 balls and y empty boxes required to be filled. Formula (9) means that there is only one ball, able to be put in any of the boxes except for the prohibited one, and therefore, there are δ-β-1 boxes. Formula (10) means that there are c balls but no empty boxes need to be filled, and all the balls can only be put in the box expect C collection or non-empty box, thus there remain $\delta^x-\delta+\beta$ boxes. Formula (11) means that there are sufficient balls to put into the remaining boxes, and the possibility is 0.

Privacy function can be calculated by combining formulas (7) to (11):

$$P(c)=\frac{1}{\sum_{x=1}^{d-1}\sum_{y=1}^{\delta-1}\Pr(E_{x,y})\bar{N}_{x,y}} \tag{12}$$

Figure 2 shows the privacy function curves, and it shall be noted that tracking attack can be actively scanned only before the label is read again, because once the tags are read again, the key internally stored will be updated, and the attacker's tracking information will be unavailable.

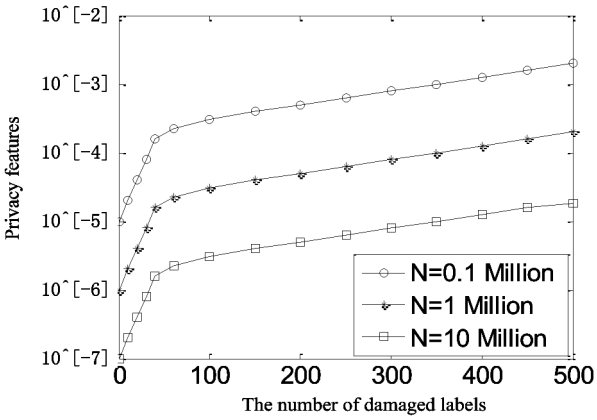

Figure 2. Privacy level curve

3. Discussion

3.1. Parameter setting

According to the program, R will be performing exclusive-or operation and hash operation of the maximum d×δ+ 5; T performs exclusive-or operation and hash operation of d+5; compared with the hash operation, the calculated amount of exclusive-or operation is relatively small, thus only the total of hash operation is considered here. An important question to consider is that based o n hardware features of R

and T, computing power of both is different, and compared to the T, R has stronger computing power. Because in a very short period of time, R, only verifies one label each time, and the validation time for each event should be minimized. To balance the reader and computing time of change as much as possible, we need to calculate the lowest cost of R and T. Assumed that the ratio of different calculating measurements between R and T is r, r = 1 represents the time of the R implementing hash operation is the same as that of and T; r = 100 indicates that R is 100 times faster than T when is hash operation executed. According to the r values, the validation costs are estimated to get the best δ and D values, the results shown in Figure 3. Validation costs depend on δ function of different r values, and the expression is:

$$Auth\_\cos t = (\delta \times d + 5) + (d + 5) \times r \quad (13)$$

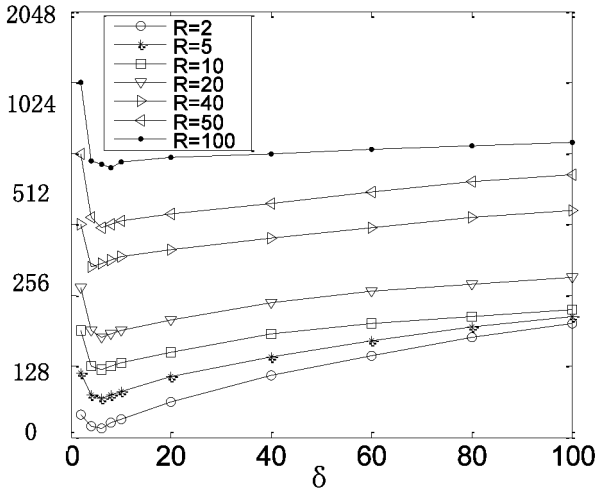

Figure 3. The authentication time cost function curve

3.2. Scalability

The program was originally designed to allow each node associated only with a label. Therefore, for a key tree of (δ, d) = (32,4), There are 220 leaf nodes, the capacity of corresponding label being 220. In fact, the system can accommodate more than 220 labels. That is, a leaf node may be associated with a plurality of labels. The case of the label tolerance in total greater than 220 can be guaranteed.

Although the two labels have the same key path, their keys are different. Furthermore, the updating mechanism of key will refresh key path and the key for logging. Thus, if a label is damaged, then the key path is changed, and the attacker would not be able to track other labels of the same leaf node. In addition, the privacy of label is determined by its unique key. Therefore, the program designed in this paper contains more than 220 labels. For most of the current applications, 220 labels are sufficient to meet demand.

Conclusions

This paper has presented a safe and efficient RFID certification program, and has carried out a detailed security analysis and discussion. In program design, after the key of the reader is updated, it will read the label again, if the previous key has created information, while the label makes response to this, indicating that the key has not been renewed yet, so as to keep away desynchronization attack. The simulation results confirm that the proposed scheme greatly improves the level of security in various parts of the key path while maintaining the same search complexity.

Acknowledgement

The paper is supported by Research on the Intelligent Recognition System of Traffic Signs Based on RFID, No. 2014WYY03.

References

1. Zhihan Lv, Alex Tek, Franck Da Silva, Charly Empereur-Mot, Matthieu Chavent, and Marc Baaden. Game on, science-how video game technology may help biologists tackle visualization challenges. 2013, 3, pp.59-63.
2. Xiaoming Li, et al.. Traffic Management and Forecasting System Based on 3D GIS. 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE, London, 2015, pp.1255-1260.
3. Tianyun Su, et al.. 3D seabed: 3D modeling and visualization platform for the seabed. 2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW). Sydney, 2014, pp. 1-6.
4. Zhang, Xu, et al.. Spike-based indirect training of a spiking neural network-controlled virtual insect. 2013 IEEE 52nd Annual Conference on Decision and Control (CDC). IEEE,Beijing, 2013,pp.235-255.
5. Zhihan Lv, Tengfei Yin, Yong Han, Yong Chen, and Ge Chen. WebVR—web virtual reality engine based on P2P network. Journal of Networks. 2011,6, pp.990-998.
6. Xiaoming Li, et al.. XEarth: A 3D GIS Platform for managing massive city information. IEEE Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA). IEEE, London,2015,pp.124-130.
7. Shuang Zhou, Liang Mi, Hao Chen, Yishuang Geng, Building detection in Digital surface model, 2013 IEEE International Conference

on Imaging Systems and Techniques (IST), New York, 2012,pp.220-229.
8. Jie He, Yishuang Geng, Kaveh Pahlavan, Toward Accurate Human Tracking: Modeling Time-of-Arrival for Wireless Wearable Sensors in Multipath Environment, IEEE Sensor Journal, 2014,11, pp.3996-4006.
9. Na Lu, Caiwu Lu, Zhen Yang, Yishuang Geng, Modeling Framework for Mining Lifecycle Management, Journal of Networks, 2014,9, pp.719-725.
10. Yishuang Geng, Kaveh Pahlavan, On the accuracy of rf and image processing based hybrid localization for wireless capsule endoscopy, IEEE Wireless Communications and Networking Conference (WCNC), Beijing,.2015,pp.352-358.
11. Guanxiong Liu, Yishuang Geng, Kaveh Pahlavan, Effects of calibration RFID tags on performance of inertial navigation in indoor environment, 2015 International Conference on Computing, Networking and Communications (ICNC), London,. 2015,pp.241-249.
12. Liguo Zhang, et al.. Double Image Multi-Encryption Algorithm based on Fractional Chaotic Time Series. Journal of Computational and Theoretical Nanoscience. 2015,4,pp.123-128.
13. Wei Luo, et al.. Method to Acquire a Complete Road Network in High-resolution Remote Sensing Image Based on Tensor Voting Algorithm. EXCLI JOURNAL, 2014,14.pp.215-219.
14. Alex Tek, et al.. Advances in Human-Protein Interaction-Interactive and Immersive Molecular Simulations. InTech, 2012,12, pp. 427-430.
15. Ruina MA, et al.. Research and Implementation of Geocoding Searching and Lambert Projection Transformation Based on WebGIS. Geospatial Information ,2009,5,pp.13-20.
16. Wang Jinping, et al.. 3D Graphic Engine Research Based on Flash. Henan Science, 2010, 4,pp.215-251.

Research of Precise Marketing Strategy Based on Data Mining

Qingqiang Meng[1], Xue Han[2]

1 North China Institute of Aerospace Engineering, Langfang, China
2 Langfang branch of Hebei University of Technology, Langfang, China

Corresponding author is Qingqiang Meng

Abstract
With the rapid growth of the amount of production and management data, precise marketing gradually replaced the traditional extensive marketing. And the rise of Databases and data mining technology provide the best solution for the precise marketing model. Data mining has been gradually into a variety of business applications. This