Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems

Jin-Suk kang, Yong-Sik Choi, Mee Young Sung, Seung-Ho Shin and Taikeong T. Jeong, *Member, KIMICS*

Abstract—Recently, ubiquitous computing in being actively researched and one of the main technology in ubiquitous computing environments is recognized as RFID system. The RFID system has much benefits but simultaneously has some problems such as user's privacy violation. In this paper, in order to improve the survivability of its nodes, it should build available simulation surrounding sensor nodes. Also, In the proposed cryptosystems we use a new hash function for user authentication and a stream cipher based on LFSR(Linear Feedback Shift Register) for message encryption and decryption. Moreover, each algorithm is programmed with C language and simulated on IBM-PC system and we analyze the randomness properties of the proposed algorithms by using statistical tests.

Index Terms—USN(Ubiquitous Sensor Network), HASH LOCK Approach, HASH Function

I. INTRODUCTION

RFID, its application, standardisation, and innovation are constantly changing. Its adoption is still relatively new and hence there are many features of the technology that are not well understood by the general populace. Developments in RFID technology continue to yield larger memory capacities, wider reading ranges, and faster processing. It's highly unlikely that the technology will ultimately replace bar code - even with the inevitable reduction in raw materials coupled with economies of scale, the integrated circuit in an RF tag will never be as cost-effective as a bar code label. However, RFID will continue to grow in its established niches where bar code or other optical technologies aren't effective. If some standards commonality is achieved, whereby RFID equipment from different manufacturers can be used interchangeably, the market will very likely grow exponentially.[1],[2].

A more complex proposal is the "Hash Lock"-approach counteracting unauthorized reads: A tag does not reveal its information unless the reader has sent the right key being the preimage to the hash value sent by the tag. The scheme requires implementing cryptographic hash functions on the tag and managing keys on the backend [8]. This is regarded as economic for the near future [9]. Unfortunately, the scheme offers data privacy but no location privacy since the tag can be uniquely identified by its hash value. Another drawback is that the key is sent in plain text over the forward channel which can be eavesdropped easily from a large distance. The extended scheme called "Randomized Hash Lock" [8] ensures location privacy but. is not scalable for a huge number of tags since many hash-operations must be performed at the back-end and it additionally relies on the implementation of a random number generator in the tags to randomize tag responses. Such devices need sources for physical randomness so that the implementation is rather complex and expensive.

II. THE HASH-LOCK APPROACH

The Hash-Lock approach proposed by Weis et all.[9] uses the concept of locking and unlocking the tag to allow access. The security of the Hash-Lock approach uses the principle based on the difficulty of inverting a one-way hash function. The scheme makes use of a back-end database to provide correct reader to tag identification and the concept of meta-ID stored in each tag. To lock the tag the reader sends a hash of a random key, as the meta-ID, to the tag. i.e. meta-ID<-hash(key). The reader then stores the meta-ID and key in the back end database. While locked, the tag only responds with the meta-ID when queried. As shown in Fig. 1, to unlock the tag, the reader will query the tag for the meta-ID. The reader will then use the meta-ID to lookup a key and ID for the tag in the database. If the meta-ID is found, the reader then sends the key to the tag in an attempt to unlock the tag. The tag hashes the key and compares the results against the meta-ID stored in the tag. [10].

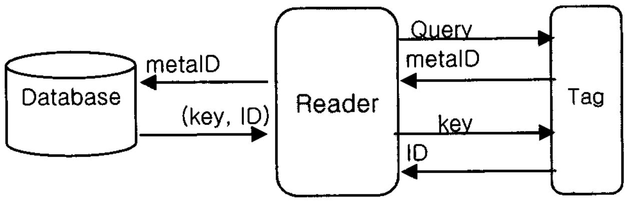


Fig. 1 Hash-Locking : Reader unlock protocol

A. Hash Function Algorithm

The Cryptographic hash functions are playing very important roles in modern cryptology such as checking the integrity of information or increasing efficiency of authentication code and digital signature. While

Manuscript received July 6, 2007.

Jin-Suk Kang is with the Computer Science and Engineering, University of Incheon, Korea, 402-749, Korea (Tel: +82-32-770-4436, Fax: +82-32-766-6894, Email: jskang01@incheon.ac.kr)

compared with general hash functions used in non cryptographic computer applications, although both cases are functions from domain to range, they're different from each other in several important aspects. Also, the hash function outputs the value called has value or has code of fixed length by the input of messages having random length. In more strict words, the hash function h corresponds text alignment of random length as n bit text alignment having fixed length.

When domain is called D and range is called R, the function d $h: D \rightarrow R(|D| > R)$ is a many-to-one corresponding function. Accordingly, the collision exists for the has function in general. For example, assuming function h as the one having input value of t bit and output value of n bit, the number of input values while h has randomness corresponds to each output value. Accordingly, two input values selected at random with probability 2^{-n} gets to have same output value regardless of the t value.

The handling process of most has functions is the repetitive one hashing the input of random length by divided processing of successive fixed blocks. First, the input X becomes padded to become a multiple of block length and divided from X_1 to t number of blocks as X_t. The hash function h is described as follows.

$$\begin{aligned} &H_0 = IV \\ &H_i = f(H_{i-1}, X_i), \\ &1 \le i \le t, \\ &h(X) = H_t \end{aligned} \tag{1}$$

Here, f is the compress function), H_i is the chaining variable between $i-1$ and i, while $.IV$ is the initial value. The general structure of repetitive has function using compressed function is like the (Figure. 2).

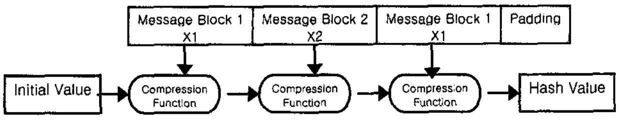


Fig. 2 Structure of the Hash-Function with Recurrent

The calculation of hash value is dependent on the chain variable. While starting the hash calculation, this chain variable gets to have the fixed initial value expressed as the part of algorithm. The compressed function renews this chain variable by getting the message block as input until it becomes hashed. This processes gets repeated in cycles for all message blocks and the last value gets output as hash value on the same message[1.2]. The hash function gets classified into 3 types depending on which structure is used as internal compressed function.

① Hash-Functions based Block Cipher

② Hash-Functions based Modular Calculation

③ The other Hash-Functions

The exclusive hash function has fast processing speed and they're the functions specially designed for hashing regardless of other system sub factors. The exclusive has function proposed until now has the structure based on MD4[3] designed by Rivest in 1990. There are MD5[3], SHA-1[4], RIPEMD-160[5] and HAVAL[6] for hash functions of MD series being widely used at this time.

When a specific hash function is assigned, although it is ideal to verify the lowest limit on complications attacking the hash function for the establishment of safe hash functions, such method isn't known for the most part in reality and the applicable known complication of the attack becomes considered as the security of hash function for the most part. If hash value is assumed as uniform probability variable. The following are well-known facts.

- For the n bit hash function h, the guessing attack to discover preimage and second preimage with 2^n operation.

- For the attacker that is able to select messages, the birthday attack is able to discover the collision message pair M, M' with about $2^{n/2}$ operation.

If n bit hash function satisfies the following two characteristics, it gets to have an ideal security. Once the hash value is given, the discovery of preimage and second preimage require 2^n operation.

B. Suggestion of Algorithm

The exclusive hash function proposed in this thesis has been designed at 32 bit process using addition, subtraction, multiplication and exclusive logical sum operations that are basic operations of the CPU. Although Boolean function has been used in order to raise nonlinearity in case of MD series exclusive hash functions, the x^{-1} operation was used in this thesis. Although the operation of getting inverse elements generally takes long time, the operation was performed in advance to form a reference table because it is an inverse element at $GF(2^2)$. All operations are accomplished as 32bit module and six 32bit registers a, b, c, d, e and f get the final hash value as chain variables. These registers become initialized as h_0 value and its value is as follows.

a = 0x01234567; b = 0xefcdab89; c = 0x98badcef;
d = 0x10325476; e = 0xc3d2e1f0; f = 0x5a3cf01d;

The 256 bit message blocks are divided into 32 bit module and h_i is renewed as h_{i+1} due to the operation being used and connected as initial value of {x0, x1, x2 ...... x7} registers, key scheduling is performed in between and the current value h_{i+1} of registers a, b, c, d, e and f are finally made at the feedforward stage. The expression method of C programming language has been used for the formulae to express algorithm.

1) The Structure of Algorithm

```
save_abcdef
pass(a, b, c, d, e, f, 3)
key_schedule
pass(a, b, c, d, e, f, 5)
key_schedule
pass(a, b, c, d, e, f, 7)
feedforward
```

① save_abcdef is feedforward Step uses h_i

aa = a; bb = b; cc = c; dd = d; ee = e; ff = f;

② construct a pass(a, b, c, d, e, f, mul) is,

```
round(a, b, c, d, e, f, x0, mul);
round(b, c, d, e, f, a, x1, mul);
round(c, d, e, f, a, b, x2, mul);
round(d, e, f, a, b, c, x3, mul);
round(e, f, a, b, c, d, x4, mul);
round(f, a, b, c, d, e, x5, mul);
round(a, b, c, d, e, f, x6, mul);
round(f, b, d, a, c, e, x7, mul);
```

Here, construct a round(a, b, c, d, e, f, X, mul) is,

```
f ^= X;          a -= Gen_32(f, f, f, f);
f ^= a;          b += Gen_32(f, f, f, f);
b *= mul;
f ^= b;          c += Gen_32(f, f, f, f);
c *= mul;
f ^= c;          d += Gen_32(f, f, f, f);
d *= mul;
f ^= d;          e += Gen_32(f, f, f, f);
e *=mul;
```

Here, the Gen_32( ) function is the one which gets four 32 bit registers as input to use first, second, third and fourth 8 bit as the input of S-box and makes 32 bit value with corresponding S-box output.

③ key_schedule is,

```
x0 -= x7 ^ 0xA5A5A5A5;   x1 ^= x0;   x2 += x1;
x3 -= x2; ^ ((~ x1) << 7);   x4 ^= x3;   x5 += x4;
x6 -= x5 ^((~ x4) >> 23);   x7 ^= x6;   x0 += x7;
x1 -= x0 ^((~ x7) << 7);    x2 ^= x1;   x3 += x2;
x4 -= x3 ^((~ x2) >> 23);   x5 ^= x4;   x6 += x5;
x7 ^= x6 ^ 0x01234567;
```

It is like the following. Here the >>, << are left and right logical shift operators.

④ feedforward is,

```
a ^= aa;   b -= bb;   c += cc;
dd ^= dd;  e -= ee;   f += ff;
```

Here the a, b, c, d, e and f registers are h_{i+1} which is the halfway hash value of 192bit and becomes the final hash value after termination of algorithm. Accordingly, the 32bit SRES(Signed Response) and the encryption key 64bit K_c are finally generated by the following formula.

$$SRES = a^\wedge b^\wedge c^\wedge d \,, \tag{2}$$

$$K_C = ef \,, \tag{3}$$

2) S-box

S-box has used the x^{-1} operation in order to raise nonlinearity and the operation of getting inverse elements generally require a lot of operation time. But the operation was performed in advance to form a reference table because it is an inverse element at $GF(2^8)$ Because inverse element of 0 doesn't exist, the value of 0 is corresponded. But because this isn't cryptologically safe, the exclusive-OR has been performed for 0xa5 value .to form a table having 256 eight bit values. The S-box table is shown on <Table 1>.

Table 1 S-box Table

0xa5	oxa4	0x33	0x41	0xee	0xf9	0xd7	0x60	0x16	0xe6	0x8b	0x58	0x9c	0x99	0x51	0x91
0x6a	0x52	0x12	0x97	0xb2	0x4c	0x4d	0xb1	0x2f	0x83	0xbb	0x7f	0xdf	0x15	0xbf	0x7e
0x54	0xec	0x48	0x68	0x68	0x1f	0xbc	0x2e	0x38	0x5b	0x47	0x5c	0xd1	0x08	0xaf	0x59
0xe0	0x44	0xb6	0x13	0xaa	0x50	0xc8	0x04	0x98	0xa9	0xfd	0x20	0xa8	0x9d	0x5e	0x19
0x4b	0x2a	0x17	0xac	0x45	0x95	0x06	0x56	0x55	0x84	0xf8	0xa1	0x3f	0x3d	0x76	0x0e
0x7d	0x6c	0xda	0xfa	0xd4	0x2c	0x4f	0xd8	0x9f	0x21	0x65	0xc3	0xa0	0xef	0xdb	0xf6
0xl1	0x6f	0x43	0x79	0x3a	0x67	0xfe	0x64	0x34	0xdc	0x49	0x86	0x05	0x93	0x63	0x28
0x2d	0xfl	0xa3	0x61	0x89	0x09	0x71	0x25	0x35	0xcc	0xb9	0x14	0x4e	0xf2	0xfb	0xf7
0xd2	0x70	0x74	0x7a	0xfc	0x9e	0x37	0x73	0xd5	0xf0	0xbd	0x82	0x62	0xca	0x4a	0xe4
0xdd	0xcd	0x23	0x72	0x1d	0x02	0xa7	0x40	0xe8	0x3e	0xe9	0x3c	0x5a	0x8d	0x66	0xc1
0xc9	0x92	0x57	0xe3	0x0c	0x0a	0x1c	0x30	0x0b	0x01	0x77	0xea	0xd0	0x88	0x0d	0x00
0xb8	0xde	0xe7	0xad	0xc5	0x6e	0x96	0xb7	0x31	0x03	0x80	0x69	0x9a	0x5f	0x1a	0x1b
0xff	0xc2	0xc0	0x3b	0xd6	0xa2	0xcb	0x29	0x7c	0xd4	0xc4	0x10	0x1e	0x81	0x53	0xb5
0x7b	0x27	0x0f	0xeb	0xd3	0x24	0x22	0x36	0xf5	0x6d	0xbe	0xba	0xc6	0x42	0x75	0x26
0xe1	0x94	0x8f	0x5d	0xa6	0x32	0xc7	0x78	0xb3	0xb0	0xf3	0xd9	0xcf	0x87	0xe5	0x2b
0xed	0x85	0x07	0xe2	0xab	0x90	0x6b	0xb4	0x46	0x8e	0x18	0x9b	0x8a	0xae	0x8c	0x39

IV. DESIGN OF INDEPENDENT AUTHENTI - CATION PROTOCOL

A. Experimental

1) Simulation Environment

In this paper, in order to improve the survivability of its nodes, it should build available simulation surrounding under the surrounding sensor nodes(it mean 4 component; survivability of sensor nodes – available battery, the output of sensor nodes – available area for search, the communication path of sensor nodes – generation of routing table, bandwidth of sensor nodes – the size of data transfer). Visual simulation environment configuration with Figure. 3.

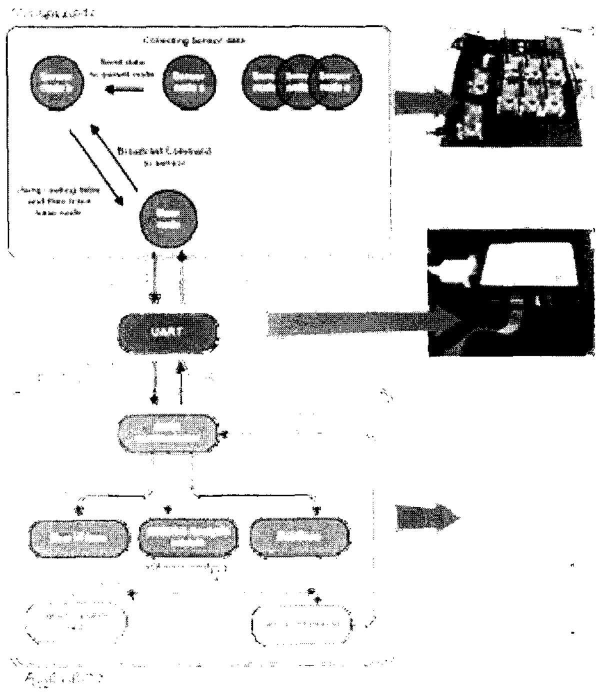


Fig. 3 Simulation Environment architecture.

2) TinyOS

Operating System such as existent UNIX in 32bits computer-on-a-chip a number Megabyte memory need. Sensor node has memory of 10Kbyte degrees of 8-16bit computer-on-a-chip in sensor network. There are TinyOS, MicroC/OS, Nucleus, Nano-X to available Operating System. TinyOS embedded hardware directly and need one physical address space as one Process. Memory is suitable Operating System to sensor network because memory allocates compile dynamically and use Function Call instead of software signaling or exception processing.[11].

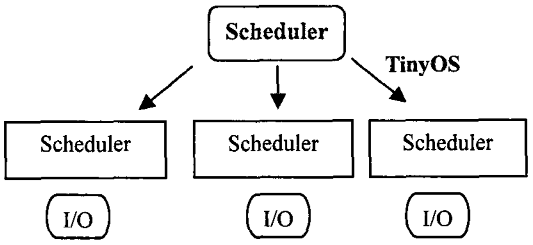


Fig. 4 TinyOS architecture

B. Design of Sense Node Data Modules

1) Design of Message manager

Message manager serial communication data be delivered through UIC(User Interface Command) main form and compose this in a few arrangement form(Array List). Message manager has following function.

- Log Service : Data original save
- Topology service : Paint topology on screen
- Node data history : Data history per node show

2) Module Function

UART is delivered to high position application receiving data of sensor node. Must establish UART for this and serial communication and need connection attribute value setting for this. Data delivered through Database serial communication input database log leave.

V. CONCLUSIONS

The work by Sarma *et al.* [11] predicts that over the next several years, development of low-cost tags in the range of US $0.05 or less will continue to present a challenge to manufacturers. Low-cost tags will remain extremely resource scarce, passively powered, and have limited memory resources comprised of several hundred bytes, as opposed to kilobytes. The range of communications will be a few meters, with a limit on computational power. Using standard cryptographic security mechanisms will exceed the capability of these devices. To meet these challenges, more work must be done to develop new hardware-efficient hash functions within low-cost RFID tags, along with new lightweight cryptographic primitives and protocols. Any new and efficient functions need to take into account the limited resources of low-cost RFID tags.

In this paper the threats to personal privacy and security that exist in low-cost RFID tags have been identified, goals and assumptions defined, and proposed solutions to address these privacy and security risks analyzed. Based on the comparison of these solutions, the selective blocker tag provides the best solution satisfying most requirements.

As RFID technology advances allowing "smarter" tags, the line between RFID devices, smart cards, and general-purpose computers will blur. Today's research benefiting RFID devices will aid in the development of secure ubiquitous computing systems in the future.

ACKNOWLEDGMENT

This work was supported by the 2 Stage Brain Korea 21 Project in 2007.

REFERENCES

[1] T, McConnel, "RFID advances bring fundamental changes to security and personal safety, " http://www.iapplianceweb.com/story/OEG20030604S0047, June 6, 2003.

[2] K. Takaragi, M. Usami, R. Imura, R. Itsuki, T. Satoh, "An ultra small individual recognition security chip," Micro, IEEE Nov/Dec 2001 Pages 43 - 49, Volume 21, Issue 6.

[3] MIT Auto-ID Center. http://www.autoidcenter.org Feb 2004.

[4] S. Sarma, D. Brock, D. Engels, "Radio frequency identification and the electronic product code,"

Micro, IEEE, vol. 21, no. 6, Nov. - Dec. 2001, pp.50 - 54.

[5] D. McCullough. RFID tags: Big Brother in small packages. CNet, 13 January 2003. From http://news.com.com/2010-1069-980325.html.

[6] Steven A. Weis, , "Security and Privacy in Radio-Frequency Identification Devices" MIT Master of Science Thesis, submitted May 2003.

[7] A. Juels, R. Pappu, "Squealing Euros: Privacy Protection in RFID-Enabled Banknotes." In R. Wright, ed., Financial Cryptography ' 03. Pringer-Verlag. 2003.

[8] J. Hoffstein, J. Pipher, J. H. Silverman, "NTRU: A Ring-Based Public Key Cryptosystem." Lecture Notes in Computer Science, 1423:267, 1998.

[9] S.A.Weis, S.E.Sarma, R.L. Revest, D.W. Engels, Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems " accepted for publication to the First International Conference on Security in Pervasive Computing (SPC 2003), March 12-14, 2003.

[10] D. Boneh, P. Golle, "Almost Entirely Correct Mixing With Applications to Voting," Proceedings of the 9th ACM conference on Computer and communications security, 2002, pp. 68 - 77.

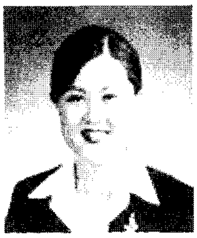
[11] S. Sarma, S. Weis, D. Engels, "Radio-frequency identifiers: Security Risks and Challenges, " CryptoBytes, vol. 6, no. 1, Spring 2003.

Jin-Suk Kang

(M'99–SM'2001–F'2005) and received the M.S. and Ph.D. degrees in Computer Engineering form Cheju National University in 2001 and 2005, respectively. His interests include Image Processing, Computer Vision, Mobile multimedia and Multimedia System. This author became a Member(M) of IEEE Aerospace in 2007. And He is now with University of Incheon(phone: +82-32-770-4436; fax: *82-32-766-6894 and e-mail: jskang01@incheon.ac.kr) in South of Korea.

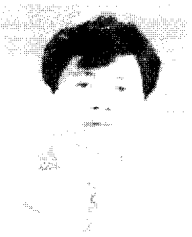
Yong-Sik Choi

Received the M.S. degrees in Computer Science and Engineering from University of Incheon in 2003, respectively. His interests include Computer Communication, Embedded System and Cryptology. He is now Ph.D. degrees candidate with University of Incheon in Korea.



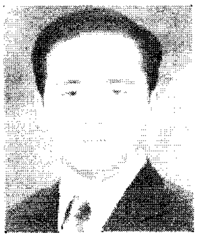
SUNG, Mee Young

Is a professor at the Department of Computer Science & Engineering, University of Incheon. She received her B.S. from Seoul National University, South Korea and her M.S. and Ph.D. in Computer Engineering from INSA (Institut National des Science Appliquees) de Lyon, France. She is Director of Marvelab (Multimedia and Advanced Real Virtual Environment LABoratory). She was a visiting scholar at the Advanced Multimedia Processing (AMP) Laboratory at Carnegie Mellon University from August 2001 until August 2002. From March 2006, She is Director of The Second stage BK21 "Ubiquitous Entertainment" Team. Her research interests are Networked Virtual Environments, Collaborative Computing, Mobile and Embedded Multimedia, etc.



Seung Ho Shin

Is a professor at the Department of Computer Science & Engineering, University of Incheon. He received the M.S. and Ph.D. degrees in Electronics Engineering from Kyung Hee University in 1979 and 1981, respectively. His interests include Computer Communication, Signal Processing and Cryptology.



T. Jeong

Received the Ph.D. degree from the Dept. of Electrical and Computer Eng., the University of Texas at Austin in 2004. He joined the University of Delaware, where he was a research associate under the research grants of NASA (Grant No. NNG05GJ38G), worked on high performance system design. And he is now working as an Assistant Professor at the Dept. of Communications Eng. at Myongji University, conjunction with MITERI. His research interests include VLSI design, computer architecture, embedded system, and high performance circuit design. ph: +82-31-330-6775, e-mail: ttjeong@mju.ac.kr