ISSN:2321-1156
International Journal of Innovative Research in Technology & Science(IJIRTS)

A STUDY ON ENCRYPTION KEY MANAGEMENT TECHNOLOGY IN A LIGHT IOT DEVICE ENVIRONMENT

Eun Young Choi, KISA; Haeryong Park, KISA

Abstract

IoT(Internet of Things) technology has been increasing in recent years. Encryption technology is necessary to build an IoT environment that is safe from cyber intrusion, such as hacking, and the encryption key in particular must be securely managed since the safety of encryption technology is critical for the safety of the encryption key. However, there have been few studies on encryption management technologies for the light IoT environment. In this paper, we review the current status of devices related to IoT and open platform technologies, and analyze the recommendation and standard for encryption key management by NIST and ISO/IEC. Then, we propose design considerations for safe key management technology in the IoT environment.

I. Introduction

Interest in the IoT environment, in which many devices such as sensors exchange data with each other through both wired and wireless communication technology, has been increasing lately. According to the market research firm Gartner, the global number of IoT devices - excluding PCs, tablets, and smartphones - will reach 20.8 billion by 2020 [6]. Furthermore, increasing attention is being paid to security vulnerabilities, including hacking and information leakages, since diverse devices ranging from ultra-small devices to high-specification systems are operating in the IoT environment.

Various security technologies such as data encryption and access authority management have already been applied to strengthen security when developing safe Internet-based services and infrastructures. Organizations such as NIST, ISO/IEC, and OASIS are publishing recommendations and standards related to encryption key management in an effort to safely build and manage them [2, 13, 14, 20-22, 25, 29, 31]. However, it is difficult to reflect the requirements of existing encryption key technologies because the IoT uses many low-power and low-specification light devices such as sensors. Moreover, although the establishment of an IoT-based environment and the development of services in such areas as smart home appliances and smart medicine have led to greater interest in using encryption technologies and managing encryption keys that are suitable for the IoT environment, there is a shortage of the relevant technology R&D and guidelines.

This paper is consisted as follows. In Section 2, we introduce IoT open platforms, such as OneM2M, IoTivity and AllJoyn, and the specifications of light devices. In Section 3, we analyze the recommendations and standards for the encryption key as published by international organizations like NIST, ISO/IEC and OASIS. Also, we propose the design of safe key management technology in the light IoT device environment in Section 4. Finally, we conclude in Section 5.

II. Introduction of IoT Technologies

In this section, IoT Open Platform and Device Specifications will be presented.

A. IoT Open Platforms

The currently active IoT platforms include OneM2M, IoTivity and AllJoyn, each of which is described below.

- **AllJoyn :**

AllJoyn is an IoT platform developed by AllSeen Alliance led by Qualcomm [1]. The AllJoyn framework consists of apps and routers. The apps communicate with the routers, and can only communicate with each other only via the routers. This platform currently supports Bluetooth, Wi-Fi, Ethernet, and serial communications, and can use ZigBee and Z-Wave. It uses the PIN-code and RSA certificates of TLS (Transport Layer Security) and establishes the initial communication using the SASL (Simple Authentication and Security Layer).

- **OneM2M** :

OneM2M is the global standards initiative for the M2M common platform established by the world's leading ICT standards bodies, including ETSI (Europe), TTA (Korea), ATIS (North America), TTC (Japan), and CCSA (China) [30]. It provides the framework that supports diverse applications and services, including the smart city and connected cars. This platform supports the overall device services from

the server, uses the AES and SHA encryption algorithms, and supports the ECDHE_ECDSA key exchange technique described in IETF RFC4492 [8].

- **IoTivity :**

The platform is developed by OCEAN (Open allianCE for iot stANdard), a global partnership based on open source and IoT standards in which more than 400 companies, research institutes, and manufacturers, including CISCO and Intel, are participating [7]. The platform is designed to implement the IoT at the device level, and supports DTLS RFC 6347 [10] and TLS RFC 5246 [9] to enhance data safety.

- **Thread :**

The Thread Group is a working group formed by seven companies (including Nest Labs, Samsung, and Silicon Lab) to support diverse home automation devices including home appliances, access control, climate control, energy management, lighting, safety, and security [34]. The protocol uses the AES algorithm to enhance security.

B. IoT Device Specification

One can refer to RFC 7228 [11] by the IETF and ENISA[3] for the classification of IoT devices. The IETF classifies IoT devices into three classes according to constraints of cost, size, weight, and other scaling factors as shown in Table 1. ENSIA classifies them according to RAM capacity and memory storage capacity as shown in Table 2(KiB=1024bytes).

Table 1. IoT Device Classification (IETF)

Classification	Data Size (e.g. RAM)	Code Size (e.g. Flash)	Description
Class0 (C0)	<<10KiB	<<100KiB	Highly constrained (Sensor –like motes) Participate in Internet communications: proxies, gateways, servers, etc. Management purposes: on/off signal, health indications, etc.
Class1 (C1)	~ 10 KiB	~ 100 KiB	Quite constrained (in code space and processing) Participate in Internet communication: No GW, CoAP, sufficient over UDP protocol stack state
			memory, code space, and often power
Class2 (C2)	~ 50KiB	~ 250 KiB	Less constrained Support for protocol stacks such as notebooks or servers

Table 2. IoT Device Classification (ENISA)

Classification		RAM CAPACITY	MEMORY STORAGE CAPACITY	Description
Devices with Limited Resources	Class0	<<10KiB	<<100KiB	It is difficult to provide security support for these devices.
	Class 1	~ 10 KiB	~ 100 KiB	Although these devices cannot use the powerful standard security protocols, it can use such protocols as CoAP for low power and light environment.
	Class 2	~ 50KiB	~ 250KiB	These devices can support the standard security protocols.
High-Performance Devices		>>50KiB	>>250KiB	High-specification devices may have dedicated security hardware or can perform concentrated operations.

According to the IETF standard [11], Class 2 devices can fully use the Internet like a PC, whereas Class 1 devices can communicate with the Internet without a separate gateway, although they have some resource limitations like the smartphone. Class 0 is a very restricted class, and it will be difficult to provide the Internet connection to such devices

without the help of higher-class devices, although they can provide near field communications such as Bluetooth. They usually provide simple measurement or On/Off functions.

Since IoT software security technologies are generally designed to reduce and lighten the functionality of existing security software for application to IoT devices, Class 1 and Class 2 devices may use some parts, but it will not be easy to apply IoT software security technologies to Class 0 devices without applying an additional hardware security technology. The ENISA standard is similar to the IETF standard, except that it further segments Class 2 devices into devices that offer performance similar to conventional servers and high-capacity devices with built-in dedicated security hardware to perform complex operations.

III. Analysis of Standards and Recommendations for Key Management in Encryption Technology

In this section, we analyze the standards and recommendations of ISO/IEC and NIST for encryption key management in IT environments. First, we summarize the state transition for each cycle of the encryption keys used by encryption technologies.

A. Cycle of Each Phase of Encryption Key

The ISO/IEC 11770 standard consists of Parts 1~6, and 11770-1 [13] classifies the encryption key into the three states of Pending Active, Active, and Post Active.

- **Pending Active**: The key is generated but not used immediately.
- **Active**: The key is generated and used for data encryption or decryption.
- **Post Active**: The key is used only for decryption or validation.

The classification defines the state transition processes of Generation, Activation, Deactivation, Reactivation, and Destruction, and specifies the simple service requirements for defined state transition.

- **Generation :** This is the process of generating the key according to the predefined key generation rule.
- **Activation:** This state means the key is available for encryption and decryption.
- **De-activation:** This state limits use of the key and is generated when the key information is leaked or destroyed.
- **Re-activation:** This state means the Post Active key is available for encryption and decryption.
- **Destruction:** This state is the last phase of the life cycle and means the state in which the encryption key is unusable due to physical damage**.**

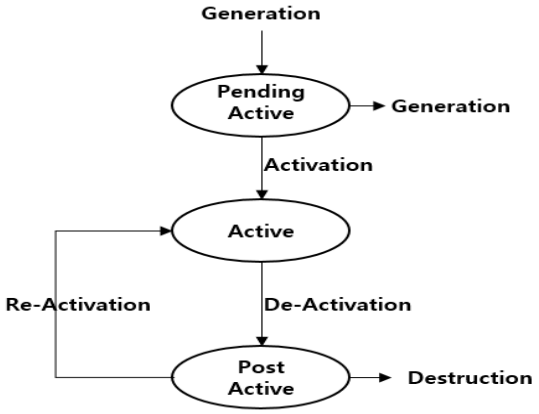

Figure 1. below shows the key life cycle based on the above states.

The NIST SP 800-57 [20, 21, 22] recommendation classifies the states according to the type of encryption key (symmetric, asymmetric cryptography, etc.) and purpose of usage (for data encryption, signature, etc.), and suggests the key operating period according to each type. It presents the key cycle period and defines four phases according to key state and state transition. Figure 2 below shows the simple key management phases and functions.

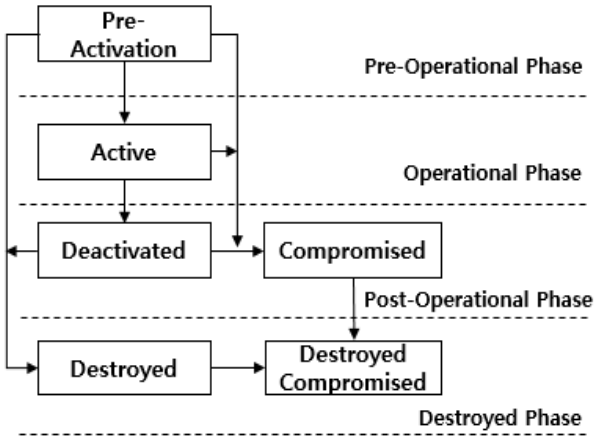

Figure 2. Key Management Phases and State Transition (NIST SP 800-57)

○ Key Management Phases

- **Operational Phase:** The keys are not yet generated or enabled since the keying materials are not normally usable.
- **Operational Phase**: The keying materials are usable, and

the keys are enabled for normal use. The keys can be used to protect the processes or the key and the process.

- **Post-Operational phase:** The keying materials are no longer used normally but are accessible. The keying materials are usable only in some situations, and the key is disabled or compromised.
- **Destroyed Phase:** The keys are no longer usable, and all records of their existence may have been deleted. Although the key is disabled or compromised, the key attributes (key name, type, and encryption cycle) may be available.

After analyzing the standard and recommendation for the encryption key with the ISO/IEC and NIST documents, we defined the encryption key life cycle and state transition for encryption management as shown in Figure 3 below.

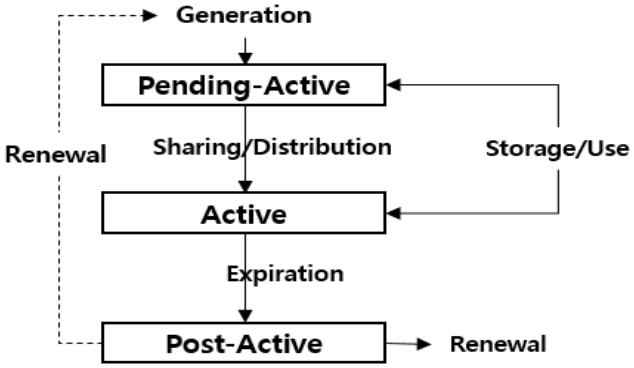

Figure 3. Encryption Key Life Cycle and State Transition

The encryption generation phase consists in 1) generating the key to share or 2) generating the secret parameter (Pending-Active). It is followed by the phase of using the same key through sharing or distribution of the generated key. This phase involves the actual use of the key for data encryption or decryption (Active). It is followed by the disabled phase, in which the key is no longer used for encryption but only for decryption and validation (Post-Active). The renewal of an expired key with the certificate returns the user to the key generation preparation phase to generate a new key

B. Requirement of Key Management for Each Phase of Key Management Life Cycle

As shown in Section 3.A, the life cycle and state transition of the encryption key can be defined in the encryption key management aspect. Table 3 below shows the standard and recommendation [12-29, 33] by ISO/IEC and NIST for each state transition.

Table 3. Standard for Each Life Cycle of Key Management

Phase	Document	Description
Generation	NIST SP 800-90A[23]	Recommendation for generation of a random number to be used for random bit generation
	NIST SP 800-56C[19]	Recommendation for key generation using the key induction function
	NIST SP 800-108[24]	Recommendation of encryption key induction using the pseudorandom function
	NIST SP 800-133[27]	Recommendation of encryption key generation
	NIST SP 800-132[26]	Recommendation for encryption key induction based on password
	NIST SP 800-135[28]	Recommendation for key induction function for each application environment
	PKCS #5[33]	Description of key induction function based on password
	ISO/IEC 14888-3[15]	Standard for key generation by asymmetric key algorithm
Sharing/ Distribution	NIST SP 800-56A[17]	Recommendation for Pair-wise key establishment schemes using discrete logarithm cryptography
	NIST SP 800-56B[18]	Recommendation for Pair-wise key establishment schemes using factorization cryptography
	ISO/IEC 11770-2[12]	Standard for key generation by asymmetric key encryption algorithm
	ISO/IEC 11770-3[14]	Standard for key generation through encryption key consent
Storage	ISO/IEC 19790[16]	Standard for encryption key storage and usage (storage of encryption key in an encryption module)
Full Cycle	NIST SP 800-57[20,21,22]	Recommendation for key management for the key management, key management security policy, and application program
	NIST SP 800-130[25]	Recommendation for design of encryption key management system
	NISTIR 7628[29]	Guideline for smart grid cyber security
	ISO/IEC 11770-1[13]	Standard for key management for each key state and state transition phase

Based on the analysis of the documents related to the encryption key, the requirements for safe generation, distribution (sharing), storage, usage, destruction, and renewal of

encryption can be summarized as follows:

[Key Generation]

- The random number needed to generate the encryption key must be generated safely by the standard SP 800-90A [23], and the products and systems that use the random number generator must comply with the standard.
- Generation of the key for the asymmetric key algorithm must comply with the following standard according to the purpose of using the public or private key:
  - *(Generation of the pair-wise key for electronic signature)* The pair-wise key used by an electronic signature algorithm must be generated safely according to ISO/IEC 14888-3 [15].
  - *(Generation of the pair-wise key for key sharing)* Generation of the key by exchanging the key or through consent must conform to ISO/IEC 11770-2 [12].
- Generation of the key for the symmetric key algorithm must use the following standard for safe key induction.
  - Check whether the method uses NIST SP 800-108 [24] (block encryption/HMAC hash based) or NIST SP 800-132 [26] (password based), i.e. whichever is the safer key induction method.
  - The method must establish the basis for the safest method of key induction.

[Distribution]

- When sharing the key, the safest method of key sharing of the following standards must be used.
  - Key sharing based on the DL/IFC asymmetric encryption conforms to NIST SP 800-56A/56B [17, 18].
  - Sharing of the structure of the sum of the safe symmetric keys using the asymmetric key must conform to ISO/IEC 11770-2, 11770-3 [12, 14].

[Storage/Usage]

- The unexpired key and the related parameters must be stored in both a memory device that is easily accessible when needed and in a backup device.
  - The public key or symmetric key stored in a memory device must be able to protect the data in conformance with ISO/IEC 19790 [16].
  - The backup device may be the hardware (HSM).
- Check if the stored key data are protected for the following situations:
  - The key data must be restored to ensure operability when the key data stored in the encryption key validation agency becomes unusable due to loss or alteration.
  - The system must be designed to back up and restore the key data that must be restored.
- Check if the access to the encryption key is appropriately controlled.

[Destruction (Deletion)]

- Check the time of and mechanism for the immediate destruction of the encryption key when it is no longer needed.
  - All paths (intermediate value, temporary buffer for reproduction, etc.) used for processing (generation, induction, etc.) of the encryption key and the key security parameters must also be deleted.
- Check if the copy of the encryption key is also deleted if it exists(The information of the encryption key may remain even after the encryption key has been destroyed).

[Renewal]

- Check if there is a procedure for renewing an expired key.
- Check if the encryption key is generated according to the encryption key generation procedure without any association with the existing encryption key.

IV. Considerations for Design of Safe Key Management Technology in the Light IoT Device Environment

We analyzed the standards and recommendations for the encryption key and deduced the requirements for the encryption key life cycle and safe encryption key management in Section 3. However, the IoT environment does not have the same computing power as PCs and servers, and thus the existing encryption key management requirements may not be applicable. Therefore, the following matters must be considered in light of the special characteristics of the IoT.

A. Consideration of Encryption Technology for Each Type of Device

An IoT service consists of such devices as a server, PCs, and sensors with various specifications. Therefore, it is necessary to determine the most suitable encryption technology, encryption key management technology and policy according to the performance of these devices. The encryption technologies that are applicable according to the Arduino specification, which is mostly used for devices under the IETF and ENISA standards and the IoT environment described in Section 2.B, are described below. The encryption

technologies and encryption key management must consider these environmental factors.

- **Class 0**
  - **Characteristics:** Architecture formed with minimum communication and configuration files.
  - **Processor**: None.
  - **OS**: Firmware.
  - **Device type**: Low-priced sensor operating with a pre-configured file.
  - **Supported encryption algorithm**: Difficult to support the encryption algorithm.
- **Class 1**
  - **Characteristics**: Use of a light protocol such as CoAP (Constrained Application Protocol); necessary to limit memory and code power consumption.
  - **Processor**: Atmega 128 (8 bits) or MSP430 (16 bits)
  - **OS**: TinyOS, Contiki, or NanoQPlus
  - **Device type**: 8-/16-bit processor-based blood sugar meter, fitness equipment, and smart home devices such as temperature controllers.
  - **Supported encryption algorithm**: AES-128/256/384 [33], RSA 1024/2048 [34], and SAH-1/256/384 [4].
- **Class 2**
  - **Characteristics**: Supports existing protocol stack with almost no limitation on resources.
  - **Processor**: ARM-Cortex A (32 bits).
  - **OS**: Embedded Linux.
  - **Device type**: 32-bit processor-based IP cameras and smart meters.
  - **Supported encryption algorithm**: High-performance algorithms such as AES-128/256/384 [33], RSA 1024/2048 [34], and SAH-1/256/384 [4].
- **Class 3**
  - **Characteristics**: Same performance level as PCs and servers and almost no limitation on resources.
  - **Processor**: All processors currently used in IT environments are available.
  - **OS**: Android, iOS, Embedded Linux, etc.
  - **Device type**: Smartphone, tablet PC, smart TV, smart hub, etc.
  - **Supported encryption algorithm**: High-performance algorithms such as AES-128/256/384 [33], RSA 1024/2048 [34], and SAH-1/256/384 [4].

B. Consideration of Data Characteristics for Each Service Type

The application of encryption technology depends on the specifications of the IoT device in question. Light IoT devices are applicable in diverse environments including the smart home, healthcare, transportation, environment/disaster management, manufacturing, construction and energy areas. The following environments for each area can use the IoT.

- **Smart home**: Smart plugs, smart light bulbs, power sen sors, automatic temperature control systems, home gateways, smart electronic products, etc.
- **Healthcare**: Wearable medical devices, smart watches, smart shoes, medical healthcare, etc.
- **Transportation**: Vehicle sensors and smart automobiles (ECU system), etc.
- **Environment/disaster management**: Gas sensors, image processing modules, etc.
- **Manufacturing**: Control and sensing module, etc.
- **Construction/energy**: Fracture/vibration modules, smart meters, power distribution switch control, etc.

IoT devices exchange detected and acquired data through the Internet in such wide ranging environments. However, it is necessary to develop and apply an optimized encryption technology that is suitable even for devices with low computing and battery performance according to the type of data used in the environment.

For example, healthcare devices that sense and transfer data on the user's health condition may be light, but they process sensitive personal information and thus must transfer the encrypted data. In that case, it is necessary to deploy an optimized encryption function instead of encryption technology supported by high-performance devices. Moreover, the appropriate encryption key management technology and policy must be established and applied in order to use the encryption technology in such an environment. For example, while the existing PC and server environment may renew and exchange the encryption key for each communication, the light IoT environment may renew the key exchange once every several months to extend the battery's life cycle.

C. Consideration of Encryption Technology Support for Each IoT Platform

An IoT environment may use an open platform. As described in Section 2.1, many platforms such as IoTivity,

OneM2M, and Alljoyn have been developed and opened for IoT environments. Since these platforms perform differently and support a different range of IoT environments, the open platform that best supports a given environment must be selected. OneM2M and Alljoyn support the overall service functions of IoT devices, gateways and servers needed to build the IoT environment, whereas IoTivity supports the functions for the operation and Internet connection of light IoT devices. Thus it is advisable to select the encryption technology in consideration of the characteristics and security level of the open platform and to support the encryption key management technology accordingly.

V. Conclusion

In this paper, we reviewed the open platform technologies for the IoT and the status of their development and analyzed the recommendations and standards of NIST and ISO/IEC for encryption management in order to deduce the requirements for encryption key management for each encryption key life cycle. Then, we propose the technology design considerations for safe key management in various light IoT device environments. The result of this study is significant in that it suggests a future direction for study of the encryption key management standard, which is the most crucial, indispensable element of encryption technology safety for protecting data and systems in IoT environments, as few such studies have been conducted to date.

Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2017-0-00267)

References

[1] AllJoyn, https://allseenalliance.org/
[2] CRYPTREC, "List Guide 2010(Key management)", 2012.
[3] ENSIA, "Security and REsilience of Smart Home Environments", https://www.enisa.europa.eu/
[4] FIPS 180-4, "Secure Hash Standard(SHS)", http://dx.doi.org/10.6028/NIST.FIPS.180-4
[5] FIPS 197, "Advanced Encryption Standard(AES)", https://doi.org/10.6028/NIST.FIPS.197
[6] Garnter,Inc, http://www.gartner.com/newsroom/id/ 3165317
[7] Iotivity, https://www.iotivity.org/
[8] IETF RFC 4492, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)",IETF, 2006.
[9] IETF RFC 5246, "The Transport Layer Security (TLS) Protocol Version 1.2", IETF, 2008.
[10] IETF RFC 6347, "Datagram Transport Layer Security Version 1.2", IETF, 2012,
[11] IETF RFC 7228, "Terminology for Constrained-Node Networks", IETF, 2014
[12] ISO/IEC 11770-2, "Information technology-Security techniques-Key management-Part 2: Mechanisms using symmetric techniques", ISO/IEC, 2008.
[13] ISO/IEC 11770-1, "Information technology-Security techniques-Key management", ISO/IEC, 2010.
[14] ISO/IEC 11770-3, "Information technology-Security techniques-Key management-Part 3: Mechanisms using asymmetric techniques", ISO/IEC, 2015.
[15] ISO/IEC 14888-3, "Information technology-Security techniques-Digital signatures with appendix-Part 3: Discrete logarithm based mechanisms", ISO/IEC, 2016.
[16] ISO/IEC 19790, "Information technology-Security techniques-Security requirements for cryptographic modules", ISO/IEC, 2012.
[17] NIST SP 800-56A_Revision1, "Recommendation for Pair-wise key establishment scheme using discrete logarithm cryptography", NIST, 2007.
[18] NIST SP 800-56B, "Recommendation for Pair-wise key establishment schemes using factorization cryptography", NIST, 2009.
[19] NIST SP 800-56C, "Recommendation for key derivation through extraction then expansion", NIST, 2011.
[20] NIST SP 800-57, "Recommendation for Key management- Part 1: General(Revision 3)", NIST, 2012
[21] NIST SP 800-57, "Recommendation for Key management- Part 2: Best Practices for Key Management Organization", NIST, 2016.
[22] NIST SP 800-57, "Recommendation for Key management- Part 3: Application-Specific Key Management Guidance", NIST, 2017.
[23] NIST SP 800-90A, "Recommendation for random number generation using deterministic random bit generators", NIST, 2012.
[24] NIST SP 800-108, "Recommendation for key derivation using pseudorandom functions", NIST, 2009.
[25] NIST SP 800-130, "Framework for Designing Cryptographic Key Management Systems", NIST, 2013.
[26] NIST SP 800-132, "Recommendation for Password-based key derivation _part1_Storage applications", NIST, 2011.
[27] NIST SP 800-133, "Recommendation for cryptographic key generation", NIST, 2012

[28] NIST SP 800-135-rev1, "Recommendation for Existing Application-specific key Derivation functions", NIST, 2006.
[29] NISTIR 7628," Guidelines for Smart Grid Cybersecurity", NIST, 2014.
[30] OneM2M, http://www.onem2m.org/
[31] OASIS, "Key Management Interoperability Protocol Specification Version 1.2", 2015.
[32] RSA Laboratories, "PKCS #1: RSA Cryptography Standard", 2012.
[33] RSA, "Pkcs #5 password-based cryptography standard", 2000.
[34] Thread, https://threadgroup.org

Biographies

EUNYOUNG CHOI received the B.S. degree in Mathematics from Korea University, Korea in 2001, the M.S degree in Information Security from Korea University, Korea, in 2003, and the Ph.D. degree in Information Security from Korea University, Korea, 2009, respectively. Currently, She is a general researcher at Korea Internet & Security Agency in Korea. Her research areas include mobile security, cyber financial security, cryptography and information security. Dr. Eunyoung Choi may be reached at bluecey@kisa.or.kr

HAERYONG PARK received his BS degree in Mathematics from Chonnam National University, Korea, in 1999. In 2001, he received his MS degree in Mathematics from Seoul National University, Korea. In 2006, he received his PhD degree in Information Security from Chonnam National University, Korea. Currently, he is a general researcher at Korea Internet & Security Agency in Korea. His research areas include cryptographic algorithm design & analysis, cyber balckbox technology development and Personal Information Security. Dr. Haeryong Park may be reached at hrpark@kisa.or.kr..