High-Availability Decentralized Cryptographic Multi-Agent Key Recovery

Kanokwan Kanyamee and Chanboon Sathitwiriyawong
Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Thailand

Abstract: *This paper proposes two versions for the implementation of a novel High-Availability Decentralized cryptographic Multi-agent Key Recovery System (HADM-KRS) that do not require a key recovery centre: HADM-KRSv1 and HADM-KRSv2. They have been enhanced from our previous work and entirely comply with the latest key recovery system in the National Institute of Standards and Technologies (NIST's) framework. System administrators can specify the minimum number of Key Recovery Agents (KRAs) according to security policies and requirements while maintaining compliance with legal requirements. This feature is achieved by applying the concept of secret sharing and power set to distribute the session key to participating KRAs. It uses the principle of secure session key management with an appropriate design of key recovery function. The system is designed to achieve high availability despite the failure of some KRAs. The performance evaluation results show that the proposed systems incur little processing times. They provide a security platform with good performance, fault tolerance, and robustness in terms of secrecy and availability.*

Keywords: *Cryptographic key management, secret sharing, key recovery, KRAs.*

Received November 5, 2011; accepted May 22, 2012; published online January 29, 2013

1. Introduction

Cryptographic function is an essential component established and implemented to meet security and privacy requirements of network services. However, the dishonest ones can use cryptosystems to conceal their illegal activities. Therefore, it is necessary to develop a key recovery system that meets the requirements of social security by helping to reveal the secret while preserving user privacy.

Starting in 1993, the US government proposed a technology that provides the reconstruction of secret keys, called Key Escrow System (KES) [5]. It provides user privacy and legitimate investigation of any suspected message by government authorities. In February 1994, the US government proposed a standard for KES called the Escrow Encryption Standard (EES) [7, 19]. It specifies the use of the SKIPJACK encryption algorithm, which uses a Law Enforcement Access Field (LEAF) creation method to be implemented in a tamper-resistant chip called Clipper. An authorized government official can obtain the secret encryption key and gain access to the communications. In 1996, the US government announced a regulation on the schemes of commercial key recovery system [26]. The system recovers the secret encryption key by a Key Recovery Agent (KRA) [6, 11]. The session key is encapsulated in the Key Recovery Field (KRF) by the sender for later key recovery as needed. Therefore, it can ensure the protection of user privacy.

Later in 1998, the National Institute of Standards and Technology (NIST) informed an international standard of Key Recovery System (KRS) [25]. It specifies requirements for key recovery products to be used by federal government agencies. These products provide for the recovery of decryption keys when they are not available or the investigation of any suspected message by government authorities is needed.

The key recovery method has been continuous improved in various areas such as trust [3, 20], authentication [1], key management [18], legal access of data but still of user privacy [14], enhancement of system security [15, 16], key recovery for long-term encrypted documents [27], encryption key recovery [8], implementation key recovery in IPSec [23], key recovery attack on the High-Bandwidth Digital Content Protection Protocol [12], and recent year, the number of patents in key recovery system is increasing steadily. Recently, NIST published an important KRS in the framework for designing cryptographic key management system [2]. It proposed a key management function that the key can be recovered by its owner or by an authorized third party after all the rules for recovery have been fulfilled and verified. Key recovery can be achieved by either a single key recovery agent or multiple key recovery agents. A Single-Agent Key Recovery System (S-KRS) can be easily attacked and colluded. Therefore, many researchers resort to designing Multi-Agent Key Recovery System (M-KRS) [15, 16, 22] that refer to a key recovery system that requires the collaboration of at least two KRAs. It can resist various threats such as brute-force attack and collusion of KRAs and enhance the security by reducing any risk of falsification and counterfeiting.

A secret sharing is a typical scheme for distributing a key portion amongst a group of KRAs, each of which is allocated a share of the key or secret. The key can only be reconstructed when the shares are combined together while individual shares are of no use on their own. The M-KRS can provide service by the collaboration of participating KRAs with or without the need of Key Recovery Centre (KRC). The latter can reduce the cost of KRC administration and management and the risk of single point of failure from the unavailability of KRC. The problem of system bottleneck can also be avoided due to the decentralized approach.

The M-KRS researches [14, 16] presented fork and join function for key recovery. The collaboration of at least two KRAs is required to recover the session key. A KRC will act as the coordinating centre for all KRAs within the group. A KRF is created for all KRAs. It contains portions of the session key for later key recovery. The sender can choose two or more KRAs among a pool of KRAs and generates a KRF. When the session key recovery service is requested, the KRF is sent to the KRC. Finally, the KRC joins all portions of the session key to obtain the usable session key. However, many weaknesses persist as follows:

1. The disclosure of the session key to third parties.
2. The risk of single point of failure from the unavailability of KRC and some KRAs since KRC and all KRAs have to participate in the session key recovery process.
3. The lack of a feature to set the minimum number of KRAs for successful session key recovery according to security policies and requirements.
4. The absence of an attack detection function on group authentication of KRAs.

This paper proposes two versions for implementing a novel High Availability Decentralized M-KRS (HADM-KRS) called HADM-KRSv1 and HADM-KRSv2. They have been enhanced from our previous work HADM-KRS [13]. They can work without the involvement of a KRC while retaining the need of law enforcement and resolving the problems incurred in the previous M-KRS's. This work eliminates the cost of KRC administration and management, and reduces the communication time which is suitable for slow-speed networks. Our contribution is fourfold:

1. HADM-KRSv1 and HADM-KRSv2 can circumvent the problem of single point of failure of a certain number of KRAs. This feature is achieved by applying the concept of secret sharing [17] and power set [10].
2. HADM-KRSv1 and HADM-KRSv2 are able to detect attacks against the group authentication of KRAs. Any other KRA that is not assigned as a member of the key recovery group will be detected by the group verification function. This function generates a set of random numbers, each of which is specific to a particular KRA in the key recovery group.
3. HADM-KRSv1 and HADM-KRSv2 are a high-availability system that requires the collaboration of participating KRAs without the need of KRC, as an alternative approach to avoid the involvement of third parties.
4. HADM-KRSv2 has the ability to specify the minimum number of KRAs (*mr*) for successful session key recovery, where $mr \geq 2$. Therefore, it can be tailored to meet security policies and requirements. This feature is achieved by applying the concept of power set to distribute the secrecy of the session key to all participating KRAs.

The rest of the paper is organised into four main sections. Section 2 proposes new session key recovery systems. Section 3 analyses the capabilities of the proposed systems. Section 4 presents performance evaluations of the proposed systems. The last section concludes the manuscript.

2. Proposed HADM-KRS's

There are two versions for the implementation of the proposed HADM-KRS as follows:

1. HADM-KRSv1: Simple high availability model.
2. HADM-KRSv2: Secure high availability model that can specify security level.

The proposed systems have been enhanced from our previous work HADM-KRS [13] and fully comply with the latest NIST's framework [2]. The common features of HADM-KRSv1 and HADM-KRSv2 are as follows:

- *High Secrecy of the Session Key:* The session key is known only to the two communicating parties.
- *High System Availability:* The problem of single point of failure and system bottleneck can be avoided due to the decentralized approach. Also, the failure or the absent of some KRAs does not effect the key recovery service.
- *Detection of Attacks on Group Authentication:* Any KRA that is not in the group can be detected.
- *Cost Effectiveness:* The cost of KRC management and maintenance can be eliminated.

The overview of the proposed HADM-KRS's is shown in Figure 1. It comprises a sender, a requester either receiver or government agency, and a certain number of KRAs. There are two main processes. The first process is the generation of a KRF by the sender in each session of message encryption. The second process is the recovery of the session key requested by the requester as needed. Every participating KRA is requested for a partial session key recovery. The usable session key is finally disclosed by the requester.

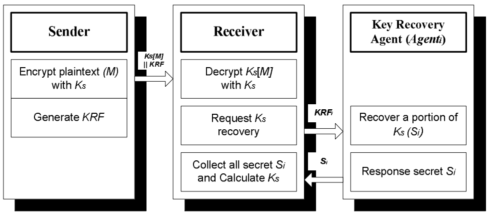
Sender
Encrypt plaintext (M) with Ks
Generate KRF
Ks[M] || KRF
Receiver
Decrypt Ks[M] with Ks
Request Ks recovery
Collect all secret Si and Calculate Ks
KRFi
Si
Key Recovery Agent (Agenti)
Recover a portion of Ks (Si)
Response secret Si

Figure 1. Overview of proposed HADM-KRS's.

The session Key (K_S) that is used to encrypt the message, is requested from the Key Distribution Centre (KDC) [4], or Kerberos [21]. The certificates for all parties are issued by the Certificate Authority (CA) under the PKI environment. It contains a public key (*Ku*) and information for identification and authentication. Each KRA can use a trust model based on Gateway CA's (GWCA) [9], that is designed to allow certification to other different kinds of CA located anywhere in the global trust network. The two main processes and their sub-processes are described as follows.

2.1. Generation of the KRF

The generation of a KRF comprises two sub-processes:

1. Initial provision of KRF components and
2. Generation and formation of the KRF. They can be described as follows.

2.1.1. Initial Provision of KRF Components

A KRF is created by the sender during each encryption operation for the readiness of future key recovery if needed. It is sent together with the standard encrypted message ($K_S[M]$). It comprises portions of KRF (KRF_i*'s*), a hash value of the Share Group Number (*h*(*SGN*)) and Unique Secret Numbers (TT_i*'s*). Each KRF_i comprises a portion of the session key (S_i), an *SGN* and other information. The S_i is used to recover K_S, while the *SGN* is used for the authentication of the key recovery group. The *SGN* is specific to a certain group of KRAs at an instantaneous time. The TT_i*'s* are reserved to recover K_S in case some participating KRAs cannot provide the service. Other information is the identification and authentication information stored in the certificate. The initial procedures to construct a KRF are shown in Figure 2, and can be described as follows:

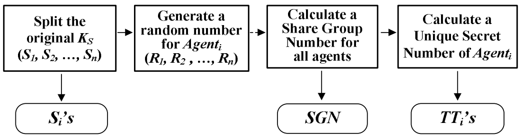
Split the original K_S ($S_1, S_2, \ldots, S_n$)
Generate a random number for $Agent_i$ ($R_1, R_2, \ldots, R_n$)
Calculate a Share Group Number for all agents
Calculate a Unique Secret Number of $Agent_i$
S_i's
SGN
TT_i's

Figure 2. Initial procedures for KRF construction.

1. Secret sharing is used to split K_S into a number of S_i*'s* equalling the number of participating KRAs (*n*). The K_S can then be recovered by articulating all S_i*'s* by the exclusive-OR (XOR) operation. This step can strengthen the security of the system and it can always securely recover K_S through the concept of secret sharing to circumvent the collusion of unfaithful KRAs. The procedures are described as follows:
   - Generate *n-1* random strings, $S_1, S_2, \ldots, S_{n-1}$, for $Agent_1$ to $Agent_{n-1}$, respectively.
   - Calculate S_n for $Agent_n$ as follows:

$$S_n = S_1 \oplus S_2 \oplus \ldots \oplus S_{n-1} \oplus K_S \quad (1)$$

2. A random number (R_i) for $Agent_i$, for *i*=1 to *n*, are generated. All R_i*'s* are distributed to $Agent_i$, for *i*=1 to *n*.
3. A *SGN* is obtained by the XOR operation of all R_i*'s* as follows:

$$SGN = R_1 \oplus R_2 \oplus \ldots \oplus R_n \quad (2)$$

   The benefit of creating *SGN* is to detect non-participating KRAs disguising into the key recovery group to secure K_S. When the requestor detects an unknown *SGN*, the system is regarded as compromised.
4. A Unique Secret Number of $Agent_i$ (TT_i) will be used for the recovery of S_i in case $Agent_i$ cannot provide the session key recovery service. TT_i is calculated as follows:

$$TT_i = S_i \oplus SGN \quad (3)$$

A *h(SGN)* is obtained from the calculation of *SGN* value by using a hash function. Although, it has been proved and accepted as secured, a recent research [24], has further improved its strength. When the requester detects an unequal *h(SGN)*, the system is regarded as compromised.

2.1.2. Generation and Formation of the KRF

The proposed system focuses on the security of the session key, the user's privacy, and the ability to recover the session key despite the failure of some KRAs. These processes can be described as follows:

- *HADM-KRSv1:* The formation of the KRF is improved to enhance the system performance and support law enforcement for message investigation by removing the excess process of the final KRF encryption from our previous work [13]. There are two sub-processes as follows:
  1. S_i, *SGN*, and *other information* are combined to form KRF_i for $Agent_i$. KRF_i is then encrypted with the public key of $Agent_i$ (Ku_{agi}) as follows: $Ku_{agi}[KRF_i]=Ku_{agi}[S_i||SGN||other\ information]$.
  2. $Ku_{agi}[KRF_i]$ for every $Agent_i$, *h(SGN)*, and TT_i*'s* are combined to form *KRF* as follows:

$$KRF=\{Ku_{agi}[KRF_i]'s||h(SGN)||TT_i's\} \quad (4)$$

- *HADM-KRSv2:* An additional feature of HADM-KRSv2 is the ability to specify the minimum number of KRAs for successful session key recovery to meet security policies and requirements. For example, if a high security level and high availability are required, the minimum number of KRAs or active KRAs should be at least 4 agents and the number of failed KRAs should be at least 2 agents. Therefore, the number of participating KRAs should be at least 6 agents (*n*=6) to meet the security requirements. The sub-processes are described as follows:

1. Collect a Unique Secret Number (TT_i) of every neighbour agent of $Agent_i$, for all $Agent_i$*'s*. This method applies the basic mathematical concept of power set and can be described as follows:
    - Choose the number of participating KRAs (*n*) from the key recovery group required to recover the session key.
    - Choose the minimum number of KRAs (*mr*) required for successful session key recovery, where *mr* is at least two KRAs to preserve M-KRS.
    - Calculate the number of KRAs (*t*) required to distribute TT_i as follows:

    $$t = n - mr \tag{5}$$

    - Distribute TT_i to the *t* consecutive nearest neighbour agents of $Agent_i(A_i)$, for all KRAs or for *i*=1 to *n*, as follows. Since KRAs are arranged in sequence and in a circular manner, $Agent_n$ is therefore the next neighbour of $Agent_1$.

    $$A_i \rightarrow \begin{cases} A_{i+1}, A_{i+2}, \ldots, A_{i+t} & where\ i < mr\ \ and\ \ i = m \\ A_{i+1}, A_{i+2}, \ldots, A_n, A_1, A_2, \ldots, A_j & where\ i > mr\ \ and\ \ j = i - mr \\ A_1, A_2, \ldots, A_t & where\ i = n \end{cases}$$

2. S_i, *SGN*, TT_i*'s*, and *other information* are combined to form KRF_i. Every KRF_i for $Agent_i$ is formed and encrypted with the public key of $Agent_i$ (Ku_{agi}) as follows: $Ku_{agi}[KRF_i]=Ku_{agi}[S_i \,||SGN\,||TT_i\text{'s}||other\ information]$. $Ku_{agi}[KRF_i]$ for all $Agent_i$*'s* are combined to form a *KRF* as follows:

    $$KRF=Ku_{agi}[KRF_i]\text{'s} \tag{6}$$

    The *KRF* is then sent together with $K_S[M]$ to the receiver as usual.

2.2. Recovery of the Session Key

When the recovery of the session key is required, a partial session key recovery is requested to every participating KRA. The session key recovery process is separated into two phases. The first phase is a partial session key recovery performed by individual KRA to make sure that the session key is secure and private. The second phase is a full session key recovery in which the session key is finally constructed by the requester. They are described as follows.

2.2.1. Partial Session Key Recovery

Every KRA perform a partial session key recovery using its partial KRF (KRF_i) to recover a portion of the session key (S_i). The overview of the partial session key recovery process is shown in Figure 3. The sub-processes of the partial session key recovery can be described as follows:

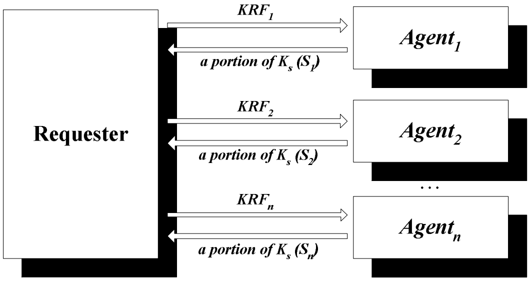

Figure 3. Overview of partial session key recovery.

1. Extraction of the *KRF* to get *n* partial *KRFs*.
    - The requester extracts the *KRF* to get all encrypted partial *KRFs* ($Ku_{agi}[KRF_i]$*'s*), for *i*=1 to *n*.
    - $Ku_{agi}[KRF_i]$ is sent to $Agent_i$, for *i*=1 to *n*.
2. Partial session key recovery at each *KRA*.
    - $Agent_i$ decrypts $Ku_{agi}[KRF_i]$ with its private key (Kr_{agi}) to get KRF_i and reveal S_i, *SGN*, and other information.
    - $Agent_i$ verifies *SGN* and the public key certificate of the requester.
    - $Agent_i$ encrypts S_i and *SGN* with the public key of the requester (Ku_{req}).
    - $Agent_i$ sends $Ku_{req}[S_i \,||\, SGN]$ to the requester for the compilation and construction of K_S.

In case some KRAs are out of service, all S_i*'s* cannot be collected. In this case, the requester will collect the lost portions of the session key from the associated active KRAs as follows:

- *HADM-KRSv1:*
    1. The requester calculates for the lost S_i*'s* by adopting TT_i of $Agent_i$ that cannot deliver S_i from *KRF*. Each S_i is calculated as follows:

    $$S_i = TT_i \oplus SGN \tag{7}$$

    2. Upon completing the collection of all S_i*'s*, the requester can now construct K_S.
- *HADM-KRSv2:*
    1. The requester encrypts the request ($req\text{-}S_i$) and *SGN* with the public key of the next neighbour agent (Ku_{nxt} of $Agent_{nxt}$) that can provide key recovery service to obtain $Ku_{nxt}[req\text{-}S_i||SGN||$ *other information]*.
    2. The requester sends $Ku_{nxt}[req\text{-}S_i||SGN]$ to $Agent_{nxt}$ that can provide key recovery service.

3. $Agent_{nxt}$ decrypts $Ku_{nxt}[req\text{-}S_i||SGN||$ *other information]* with its private key (Kr_{nxt}).
4. $Agent_{nxt}$ verifies *SGN*, public key certificate of the requester, and calculates S_i as equation 5.
5. $Agent_{nxt}$ encrypts S_i and *SGN* with Ku_{req} to obtain $Ku_{req}[S_i, SGN]$.
6. $Agent_{nxt}$ forwards $Ku_{req}[S_i, SGN]$ to the requester.

2.2.2. Full Session Key Recovery

In this phase, the session key is constructed by the requester as shown in Figure 4 and can be described as follows:

- The requester decrypts $Ku_{req}[S_i||SGN]$ with its private key (Kr_{req}) to obtain S_i and *SGN*.
- S_i of $Agent_i$ is verified using *SGN*.
- K_S is calculated as follows:

$$K_S = S_1 \oplus S_2 \oplus .... \oplus S_n \tag{8}$$

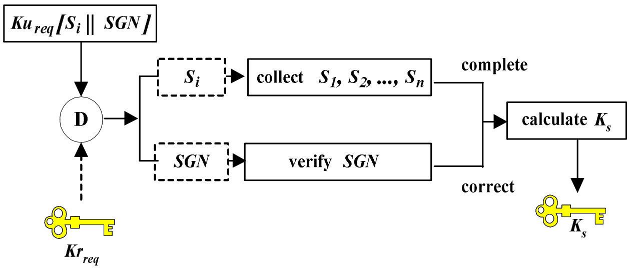

Figure 4. Session key recovery by the requester.

3. Capabilities Comparison with the Typical M-KRS

The capabilities comparison of the proposed HADM-KRS's with the typical M-KRS is shown in Table 1. Compared to the typical M-KRS, the proposed HADM-KRSv1 and HADM-KRSv2 are able to avoid the single point of failure of KRAs, detect attacks on group authentication of KRAs, and manage the minimum number of KRAs to set security requirements, while maintaining law enforcement support. They are described in details as follows:

- HADM-KRSv1 and HADM-KRSv2 have an appropriate function of high secrecy of the K_S without the need of KRC by using a suitable algorithm.
- HADM-KRSv1 and HADM-KRSv2 can avoid the single point of failure since K_S can be recovered even the failure of some KRAs by using the concept of secret sharing and power set.
- HADM-KRSv1 and HADM-KRSv2 have strong authentication function to detect attacks from non-participating KRAs.
- Finally, HADM-KRSv2 can specify the minimum number of KRAs for successful session key recovery to set the security requirements that do not exist in the typical M-KRS.

Table 1. Capabilities comparison of HADM-KRS's with the typical M-KRS.

Capabilities	Typical M-KRS	HADM-KRSv1	HADM-KRSv2
High Secrecy of K_S	Yes	Yes	Yes
Can Recover K_S Despite the Failure of Some KRAs	No	Yes	Yes
Group Authentication of KRAs	No	Yes	Yes
Can Specify the Minimum Number of KRAs	No	No	Yes
Law Enforcement Support	Low	High	High

4. Performance Evaluation

This paper compares the performance of three M-KRS's: Typical M-KRS, HADM-KRSv1, and HADM-KRSv2. A number of measurement experiments were conducted to determine the processing time (in seconds) during the generation of a KRF, the recovery of partial session keys, the recovery of the full session key, and the recovery of lost partial session keys in terms of the number of KRAs (n). The results are shown in Figures 5 to 8, respectively.

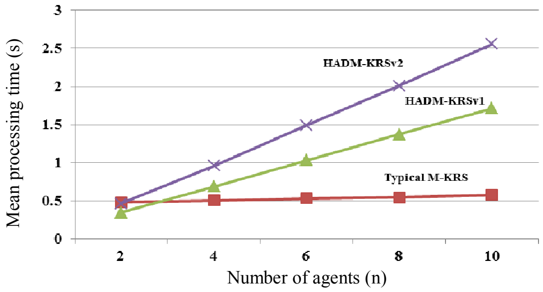

Figure 5. Performance of KRF generation.

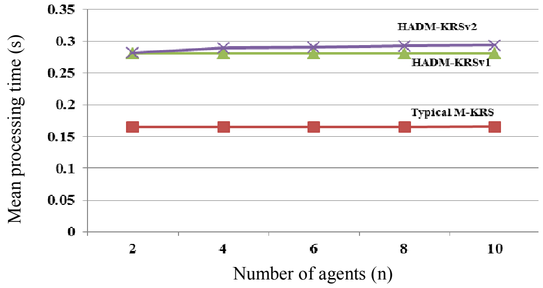

Figure 6. Performance of partial session key recovery.

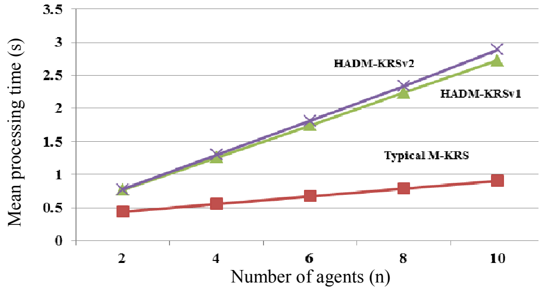

Figure 7. Performance of full session key recovery.

For the generation of a KRF, the recovery of partial session key, and the recovery of full session key, HADM-KRSv2 requires more processing time than HADM-KRSv1 and typical M-KRS because the KRF contains the distribution of unique secret numbers function to manage the minimum number of KRAs and recover the session key in case of the failure of some KRAs. The processing times of HADM-KRSv1 and HADM-KRSv2 increase more rapidly when the number of agents is more than 10. They incur little processing times and the difference is negligible. The extra time added is not significant for many applications, knowing that the proposed HADM-KRS's are more robust than the typical M-KRS in terms of secrecy, availability, authenticity, security setting, and legal support.

The last measurement experiment was conducted to determine the processing time (in millisecond: ms) required for the recovery of some lost partial session keys due to the failure of the associated KRAs. The number of participating KRAs (*n*) was set to 4, 6, 8, and 10, respectively. The number of failed KRAs was set to *n*-2 so that the system remains two KRAs.

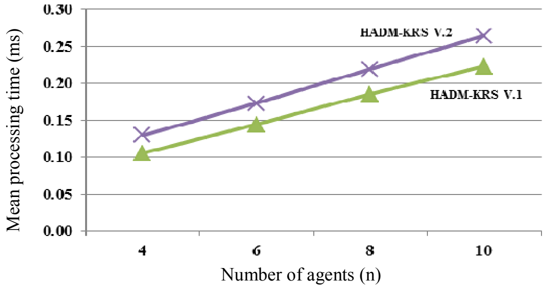

Figure 8. Performance of lost partial session keys recovery.

Figure 8 shows the recovery times of lost partial session keys when *n*-2 agents fail to provide their partial session keys for session key recovery service. The result shows that the processing time increases slowly in relation to the number of agents.

5. Conclusions

This paper presents two versions for implementing the new HADM-KRS: HADM-KRSv1 and HADM-KRSv2. All aspects of secrecy, security, and performance are considered, including confidentiality, integrity, availability, and response time. The proposed system provides a high-performance security platform, as well as robustness and fault tolerance. This work applies the fundamental concept of secret sharing and power set to distribute the secrecy of session key to all participating KRAs. The proposed system can avoid the single point of failure of some KRAs since it can work despite the failure of some KRAs. In conclusion, the new scheme clearly relies on the group collaboration of KRAs to provide the high secrecy of session key. It provides the ability to detect KRAs that are not legitimate participants, complies with legal requirements, and is based on a well-defined Public Key Infrastructure (PKI). The HADM-KRSv2 provides an additional feature to specify the minimum number of KRAs for successful key recovery.

References

[1] Al-Salqan Y., "Cryptographic Key Recovery," *in Proceedings of the Computer Society Workshop on Future Trends of Distributed Computing Systems*, pp. 34-37, 1997.

[2] Barker E., Branstad D., Chokhani S., and Smid M., *A Framework for Designing Cryptographic Key Management Systems*, *Draft Special Publication 800-130*, National Institute of Standards and Technology, 2010.

[3] Cylink Corporation, CyKey^{TM}: Cylink's Key Recovery Solution, available at: http://www.csm.ornl.gov/~dunigan/cykey.pdf, last visited 2011.

[4] D'Arco P., "On the Distribution of a Key Distribution Center," *in Proceedings of the 7th Italian Conference on Theoretical Computer Science*, Springer, pp. 357-369, 2001.

[5] Denning D., "The US Key Escrow Encryption Technology," *Computer Communications*, vol. 17, no. 7, pp. 453-457, 1994.

[6] Denning D. and Branstad D., "A Taxonomy for Key Recovery Encryption Systems," *Internet Besieged: Countering Cyberspace Scofflaws*, vol. 39, no. 3, pp. 357-371, 1997.

[7] Denning D. and Smid M., "Key Escrowing Today," *IEEE Communications Magazine*, vol. 32, no. 9, pp. 58-68, 1994.

[8] Global Information Assurance Certification, "Encryption Key Recovery," *GSEC Certification Practical Assignment V.1.4b*, 2004.

[9] Guo Z., Okuyama T., and Finley M., "A New Trust Model for PKI Interoperability," *in Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services*, pp. 37, 2005.

[10] Jech T., *Set Theory*, Springer-Verlag, New York, 2006.

[11] Jefferies N., Mitchell C., and Walker M., "A Proposed Architecture for Trusted Third Party Services," *in Proceedings of the International Conference on Cryptography*, Berlin, pp. 98-104, 1996.

[12] Johnson R., Rubnich M., and DelaCruz A., "Implementing a Key Recovery Attack on the High-Bandwidth Digital Content Protection Protocol," *in Proceedings of the IEEE Consumer Communications and Networking Conference*, Las Vegas, pp. 313-317, 2011.

[13] Kanyamee K. and Sathitwiriyawong C., "High-Availability Decentralized Multi-Agent Key Recovery System," *in Proceedings of the International Conference on Computer and*

Information Science, Shanghai, pp. 290-294, 2009.

[14] Lee Y. and Laih C., "On the Key Recovery of the Key Escrow System," *in Proceedings of the Annual Computer Security Applications Conference*, San Diego, pp. 216-220, 1997.

[15] Lim S., Hani H., Kim M., and Kim T., "In Design of Key Recovery System using Multiple Agent Technology for Electronic Commerce," *in Proceedings of the Industrial Electronics*, Pusan, pp. 1351-1356, 2001.

[16] Lim S., Kang S., and Sohn J., "Modeling of Multiple Agent Based Cryptographic Key Recovery Protocol," *in Proceedings of the Annual Computer Security Applications Conference*, pp. 119-128, 2003.

[17] Lv C., Jia X., Tiany L, Jing J., and Suny M., "Efficient Ideal Threshold Secret Sharing Schemes Based on Exclusive-Or Operations," *in Proceedings of the 4th International Conference on Network and System Security*, Melbourne, pp. 136-143, 2010.

[18] McConnell B. and Appel E., "Enabling Privacy, Commerce, Security and Public Safety in the Global Information Infrastructure," available at: https://www.cdt.org/crypto/clipper_III/clipper_III_draft.html, last visited 1996.

[19] National Institute of Standards and Technology, "Escrowed Encryption Standard," *Federal Information Processing Standards Publication 185*, 1994.

[20] National Institute of Standards and Technology. Key Recovery Examples, available at: http://csrc.nist.gov/krdp/exa.html, last visited 2011.

[21] Neuman B. and Ts'o T., "Kerberos: An Authentication Service for Computer Networks," *IEEE Communications Magazine*, vol. 32, no. 9, pp. 32-38, 1994.

[22] Numao M. and Nakayama Y., "Internet Archiving Server with Key Recovery Function," *in Proceedings of the Symposium on Cryptography and Information Security*, Japan 1998.

[23] Su R., Che X., Fu S., Li L., and Zhou L., "Protocol-Based Hidden Key Recovery: IBE Approach and IPSec Case," *in Proceedings of the Conference on Networks Security, Wireless Communications and Trusted Computing*, Wuhan, pp. 719-723, 2009.

[24] Thulasimani L. and Madheswaran M., "A Novel Secure Hash Algorithm for Public Key Digital Signature Schemes," *International Arab Journal of Information Technology*, vol. 9, no. 3, pp. 262-267, 2012.

[25] Wakid S., "Requirements for Key Recovery Products," *Report of the Technical Advisory Committee to Develop a Federal Information Processing Standard for the Federal Key Management Infrastructure*, National Institute of Standards and Technology, 1998.

[26] Walker S., Lipner S., Ellison C., and Balenson D., "Commercial Key Recovery," *Communications of the ACM*, vol. 39, no. 3, pp. 41-47, 1996.

[27] Wang E., Yau J., Hui L., Jiang Z., and Yiu S., "A Key-Recovery System for Long-term Encrypted Documents," *in Proceedings of the International Enterprise Distributed Object Computing Conference Workshops*, China, pp. 52, 2006.

Kanokwan Kanyamee received her BSc in computer science from Rajabhat Institute Uttaradit in 1999, and her MSc and PhD in information technology from Naresuan University and King Mongkut's Institute of Technology Ladkrabang, Thailand in 2003 and 2013, respectively. She is currently a lecturer at Uttaradit Rajabhat University. Her research interests are in cryptography and information security.

Chanboon Sathitwiriyawong received his BEng degree in electrical engineering from Prince of Songkla University, Thailand in 1986. He earned his MSc in data tele-communications and networks in 1993 and his PhD in electronic and electrical engineering from the University of Salford, United Kingdom in 1996. He is an associate professor at the faculty of Information Technology and the dean, King Mongkut's Institute of Information Technology Ladkrabang. His current research interests are in the area of computer network, and network and system security. He is a member of the IEEE Communication Society.