Dynamic Key Cryptography and Applications

Huy Hoang Ngo, Xianping Wu, Phu Dung Le, Campbell Wilson, and Balasubramaniam Srinivasan
(Corresponding author: Huy Hoang Ngo)

Faculty of Information Technology, Monash University
900 Dandenong Road, Caulfield East,Victoria, 3145, Australia (Email: huy.hoang.ngo@infotech.monash.edu.au)

(Received Nov. 24, 2008; revised and accepted Feb. 9, 2009)

Abstract

In modern security models, cryptography plays a fundamental role in protecting data integrity and confidentiality in information systems. However, cryptography itself is subject to cryptanalysis attacks. To reduce the cryptanalysis attack risk, a dynamic key theory is presented and analyzed in this paper. Because these dynamic keys are one-time used symmetric cryptographic keys, they can significantly improve the security of cryptographic systems. The dynamic key theory generation scheme and key update mechanism are formally analyzed to demonstrate balance between security and performance. The theory can be applied to enhance the security and performance of cryptographic systems, especially those used in wireless networks communication. Two case studies using the proposed dynamic key theory are also described and analyzed to illustrate the power of the theory.

Keywords: Authentication, dynamic key cryptography, e-payment, symmetric cryptography

1 Introduction

With the widespread use of wireless network services and applications, security becomes a major concern. From security aspects, data integrity and confidentiality are vital issues for information systems. Confidentiality is concerned with resources being only accessed by authorized users while integrity refers to protection against unauthorized modification. Integrity and confidentiality are often related to authentication, authorization and cryptography. In fact, authentication utilizes strong cryptographic systems in order to secure itself. Thus cryptography plays a crucial part of any security system.

There are two basic techniques in cryptography [25]: symmetric and asymmetric cryptography. In symmetric cryptography, encrypted and decrypted keys are the same. In contrast, cryptography using different encrypted keys from decrypted keys is called asymmetric cryptography. Each of them has pros and cons. Because of its characteristics, asymmetric cryptography is more secure than symmetric in key distribution and exchange. However, symmetric cryptography is significantly faster than asymmetric cryptography. Furthermore, Blaze [2] stated that the asymmetric cryptography key size must be ten times or more that of a symmetric cryptography key in order to have a similar level of security.

In security systems, based on their advantages, symmetric and asymmetric cryptography are often combined together to protect information systems. In TSL/SSL [19] asymmetric cryptography such as Diffie-Hellman [5], ECC [9] operates key exchange between clients and servers in order to distribute session keys. After that, session keys are used as symmetric cryptographic keys for encrypting all messages in one communication session. Each session key can be used within one session only. However, when the session time is too long, the session key becomes more vulnerable. By capturing communication messages, an adversary might be able to detect patterns in the encrypted messages to crack the ciphers. The compromise of one session key exposes all communication data in the session. Furthermore, key exchange protocols rely on permanent asymmetric keys. The more that asymmetric keys are re-used to create sessions, the more cryptographic systems become vulnerable to cryptanalysis attacks. When these keys are compromised, the whole security system becomes vulnerable to adversaries.

In the past, the major solution for enhancing security and reducing the risk of such cryptanalysis attacks was to increase the key size used in the cryptographic systems. However, increasing the cryptographic key size is not always the best solution, since no matter how large the key is, its cryptography is still ultimately breakable. Every cryptographic key is only secure for a certain amount of time. In 2007, Lenstra [12] stated that the 1024 bit RSA encryption used in most banking and e-commerce systems may only be secure for a few more years. In addition, larger keys often require higher computational resources, especially in asymmetric cryptography. In practice, excessively large keys may admit denial of service possibilities whereby adversaries can cause excessive cryptographic processing. Large keys are also clearly unsuitable for mobile devices having slow processing units and/or limited battery powers.

In this paper, a dynamic key theory is described and mathematically analyzed. We discuss the security requirements for the sequence of dynamic keys and how they are used as a guide to build dynamic key generation functions. Based on that guide, we present a family of dynamic key generation functions. The dynamic key sequence created by this family of dynamic key generation functions is examined and analyzed. The analysis shows the advantages of dynamic keys in both security and efficiency. In the security analysis, we show that while one compromised dynamic key exposes one message, the other messages in the session and system are still secure. Although perfect secrecy from one-time pad is impossible, the security of cryptographic system using dynamic key is close to one-time pad. Besides minimizing cryptanalysis attack risks, dynamic keys are also able to prevent replay-attacks on authentication and payment systems. In terms of performance, by storing intermediary keys, dynamic keys used as one-time symmetric cryptographic keys can achieve high levels of security without scarifying performance by increasing key size. Because the dynamic keys are generated offline, there is no key exchange before every encryption. A study is conducted to find the most appropriate sequence size and dynamic key lifetime to balance between security and performance. Hence, the dynamic key generation scheme can adjust to suit different applications requiring different security levels. We also propose a method to reduce the synchronization problem in dynamic key theory. Finally, two applications of dynamic keys in authentication and internet banking payment protocols are discussed and analyzed to demonstrate the power of dynamic keys. In the context of these applications, the security and efficiency advantages of the dynamic keys are presented via SVO analysis on the security of these protocols.

The rest of the paper is structured as follows. In the following section, some previous related works are reviewed. Section 3 describes the dynamic key theory including the definition, security requirements and the dynamic key generation schemes. Comparisons, discussion and analyses are presented in the forth section. Section 5 illustrates and analyzes two applications which demonstrate the dynamic key theory. Finally, we give conclusions on our paper and discuss possible future works.

2 Related Works

This section describes the previous works related to the dynamic key theory, specifically one-time pad, one-time password and a limited-used key generation schemes. Based on the ideas presented in these works, the dynamic key theory is developed.

2.1 One Time Pad

One time pad [10, 20] is a symmetric cryptographic system using randomly generated private keys. Each message is encrypted by a private key that can be used only once. In theory, each encryption is unique and has no relation to the following encryptions. The cryptography no longer relies on long term shared keys which are vulnerable under cryptanalysis attacks. Therefore, it is impossible to detect patterns with which to perform cryptanalysis on the one time pad. The main idea of one time pad is to avoid long term shared cryptographic keys. In other words, when the one-time pad is truly random, it is unbreakable by analyzing successive messages.

In one time pad systems, the pads are shared between senders and receivers. To decrypt the messages, the decrypted pads at the receivers must be the same as the encrypted pads at the senders. Therefore, these pads must be distributed between the parties. Therefore, besides the randomness of the generated pads, the security of one time pad depends on the distribution of the pads among the parties.

In practice, the distribution of pads between parties over networks is the weak point in one time pad systems. Similar to current security systems, symmetric cryptographic keys that are used to secure communication messages require secure key exchange among parties before the communication messages are sent. Normally, the key exchange can be performed via public key algorithms like Diffie-Hellman [5] or MQV [16]. However, the security of these algorithms relies on long term shared keys that contradict the original idea of one time pad.

2.2 One Time Password

The idea of the one-time password was firstly introduced by Lamport [8] in 1981. In a one-time password system, both client and server mutually agree to share a sequence of one-time passwords for authentication. Every authentication request uses a different password in the password sequence. Therefore, the one-time password system can prevent third parties from extracting authentication passwords via eavesdropping. There are two ways to share the sequence of passwords in one-time password systems. The first approach uses a mathematical algorithm to generate the sequence of passwords. A new password is generated from the previous passwords. This approach relies heavily on the synchronization index of the current password in the password sequence. The second approach is based on the synchronous time between client and server to generate the sequence of passwords. Each password has a short life-time. Authentication using the password is valid within that time period. Any attacks launched by re-using the password after the expired time of password are unsuccessful. This approach requires time synchronization between clients and servers. Conversely, the adversary can still gain unauthorized access by re-using password attacks within its life-time.

2.3 Limited-Used Key Generation Scheme

Kungspidan et al. [15] proposed a scheme to generate a sequence of one-time cryptographic keys. These cryptographic keys are an extension of one-time password used for cryptography. In this scheme, a sequence of cryptographic keys is generated offline. Each cryptographic key in the sequence is used to encrypt only one message. Hence, every message is encrypted by a different set of cryptographic keys. The scheme uses a keyed hash function $h(\cdot,\cdot)$ and the pre-shared master key K_{AB}. The dynamic key generation is described in six steps as follows.

1) Authentication Server generates the distributed key DIK and sends it to client via authenticated key exchange protocol.

2) Both client and Authentication Server calculate the set of preference keys:

$$\begin{aligned} K_1 &= h(DIK, K_{AB}) \\ K_2 &= h(DIK, K_1) \\ &\dots \\ K_m &= h(DIK, K_{m-1}). \end{aligned}$$

3) Authentication Server generates a random number r and sends it to client.

4) From the random number r, both Authentication Server and client calculate the middle keys and the SIK as follows: $w = r \bmod m$. K_{Mid1} is the middle key of $K_1 \dots K_w$; K_{Mid2} is the middle key of $K_1 \dots K_{Mid1}$; And $SIK = h(K_{Mid1}, K_{Mid2})$.

5) Both Authentication Server and client generate the set of dynamic keys based on the SIK and DK as follows:

$$\begin{aligned} SK_1 &= h(SIK, DK) \\ SK_2 &= h(SIK, SK_2) \\ &\dots \\ SK_n &= h(SIK, SK_{n-1}). \end{aligned}$$

Although this scheme shows the ability to produce one-time cryptographic keys, it does not show any further analysis as to how secure the dynamic keys are. The initial stage of this dynamic key generation scheme uses two shared initial keys K_{AB}, DIK and a random number r. There is no study that explains the relationship between the initial keys, number r and the security of the dynamic key sequence. When the dynamic key sequence is compromised, a repeat stage is used to generate the dynamic keys. However, there is a lack of investigation as to when the sequence of dynamic keys is no longer secure. There is also no solution to minimize the synchronization problem for these one-time cryptographic keys. In order to implement this scheme for different security requirements, it needs improvements and further investigations.

3 The Dynamic Keys

An explanation of using terminology and notations to describe the dynamic key cryptography and the generation scheme begin this section. Following the terms and notations, a dynamic key definition and a brief discussion of dynamic key theory are presented. The next part will describe and analyze a family of dynamic key generation schemes. The dynamic key generation scheme contains two parts: a function for first time dynamic key generation and another revised version for repeated dynamic key generation.

3.1 Terms and Notations

n	number of dynamic keys in a sequence
m	number of session keys
$DK_i, 1 \le i \le n$	dynamic keys
$\{DK_i\}$	a sequence of dynamic keys
EK	an encrypted key. It is a one-time key cryptographic key
IK	initial key. It is another one-time initial key to generate seed key SK
$TK_1, \dots, TK_m$	temporary keys. They are used as parameters to calculate the beginning dynamic keys in the sequence. During the process to create dynamic key sequence, these parameters are replaced one after another by previous dynamic keys.
SK	a seed key generated from IK and $TK_1, \dots, TK_m$
$f()$	a polynomial function to generate dynamic keys
$f^{-1}()$	invert function of f, supposed to be a non polynomial function
Pr	probability function
NK_1, NK_2	one-time keys computed from IK, EK, DK_{n+1}, and DK_{n+2}
$h(X)$	a one-way hash function h of message X
$\oplus$	bit-wise exclusive or

3.2 Dynamic Key Definition

Dynamic keys are one-time symmetric cryptographic keys forming a sequence of keys. Similar in nature to one-time pad, every message in the system is encrypted by a different cryptographic key. Therefore, any attempts to attack the cryptographic system by re-using a compromised cryptographic key can be easily detected. Instead of distributing the cryptographic keys among the parties, the dynamic keys are generated off-line at participating parties. Unlike session keys which are exchanged among parties in every session, there is no key exchange at every session or transaction. A dynamic key generation scheme is used to produce a sequence of dynamic keys from initial parameters. These parameters can either be pre-shared or exchanged via key exchange protocol only once at the beginning of the session. Mathematically, a sequence of dynamic keys is presented as follows:

$$n \in N, n > 1, \{DK_i\} = \{DK_1, DK_2, ...DK_n\}.$$

The sequence of dynamic keys is required to have minimum risk under cryptanalysis attacks. In other words, compromising one or several cryptographic keys in the sequences should not compromise the whole sequence and therefore the security of entire system. The condition underlying the dynamic generation scheme can be explained in mathematics as: for every algorithm A, the highest probability that A can guess correctly the current dynamic key DK_m from the previous dynamic keys DK_i is, with s being the bits length of dynamic key DK_m.

$$\forall i, m \in N, 1 \leq i \leq m, Pr(A(\{DK_i\}) = DK_m) \leq \frac{1}{2^s}.$$

3.3 Initial Dynamic Key Generation

The dynamic key generation scheme produces a sequence of dynamic keys . The sequence can be unlimited in size. In other words, the number of dynamic keys in a sequence may be infinite. The produced sequence must have the following characteristics:

- The sequence of dynamic keys must be unique at both the sender and receiver. In order to satisfy this requirement, the scheme must produce the same sequence of dynamic keys from the same initial parameters.
- Compromising previous dynamic keys must not create vulnerability for current and future dynamic keys in the sequence. From one or more compromised key $DK_i, 1 \leq i \leq n$, an adversary must not be able to compute any other dynamic keys DK_{n+1} from the previous dynamic keys.

The dynamic key generation can be divided into four following steps as in Figure 1.

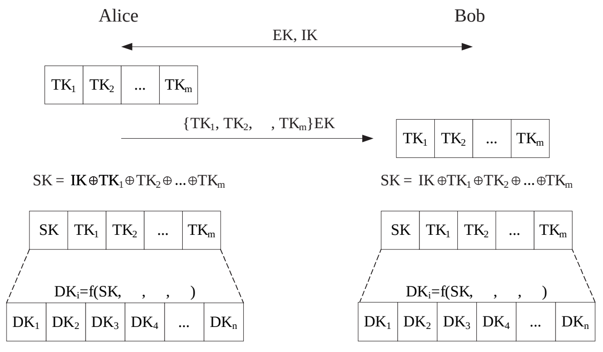

Figure 1: The initial dynamic key generation

Step 1. Alice and Bob exchange two keys EK and IK via a secure channel.

Step 2. Alice randomly generates m initial temporary keys $TK_1, \ldots, TK_m$ and sends the message to Bob, encrypted by EK.

$$A \rightarrow B : \{TK_1, \ldots, TK_m\}EK, \\ h(TK_1 \oplus \ldots \oplus TK_m \oplus EK).$$

The result of the hash function $h(TK_1 \oplus ...TK_m \oplus EK)$ is the digital signature to authenticate the source of the message. It is used to verify that Alice is the one who sends this message.

Step 3. Both Alice and Bob compute a seed key $SK$from the initial key IK and the temporary keys $TK_1, \ldots, TK_m$ using bit-wise exclusive or operation.

$$SK = IK \oplus TK_1 \oplus TK_2 \oplus \ldots \oplus TK_m.$$

Step 4. Generate sequence of dynamic keys. The first dynamic key DK_1 is generated from the seed key SK and the temporary keys $TK_1, \ldots, TK_m$ by using a function $f()$ taking $m + 1$ parameters as follows:

$$DK_1 = f(SK, TK_1, \ldots, TK_{m-2}, TK_{m-1}, TK_m).$$

Assume $n > m$, the other dynamic keys are also generated by function $f()$ but the parameters are replaced on after another by previous dynamic keys.

$$\begin{aligned} DK_2 &= f(SK, TK_2, \ldots, TK_{m-1}, TK_m, DK_1) \\ DK_3 &= f(SK, TK_3, \ldots, TK_m, DK_1, DK_2) \\ &\ldots \\ DK_n &= f(SK, DK_{n-m}, \ldots, DK_{n-3}, DK_{n-2}, \\ &\qquad DK_{n-1}). \end{aligned}$$

Because of the requirements for the dynamic key sequence, function $f(\cdot)$ has two conditions:

- Produce unique outputs: Function $f(\cdot)$ give only one unique value from $m + 1$ input parameters include $SK, parameter_1, \ldots,$ and $parameter_m$.
- Without knowledge of SK, the current dynamic key DK_i cannot be computed from the previous dynamic keys $DK_{i-1}, \ldots, DK_1$.

To analyse the second condition, we assume that in the worst case, an adversary has successfully obtained $(i-1)$ consecutive dynamic keys $DK_1, DK_2, \ldots, DK_{i-1}$. He/she also knows the function $f(\cdot)$ for which $DK_i = f(SK, DK_{i-m}, \ldots, DK_{i-2}, DK_{i-1})$. To break the dynamic key generation scheme, SK is the only parameter of the function $f(\cdot)$ that the adversary does not know. To make the dynamic key generation function secure, guessing SK from $m + 1$ dynamic keys must be equivalent to brute force searching. To satisfy this requirement, the function $f(\cdot)$ must be a one-way function [11] so that guessing SK is infeasible. With $h(\cdot)$ is a one way hash function, the function $f(\cdot)$ can be rewritten as follows:

$$f(SK, param_1, \ldots, param_m) = \\ h(SK \oplus param_1 \oplus \ldots \oplus param_m).$$

In [29], a one-way function $f(\cdot)$ is defined as a mathematical function that is easy to compute but much harder to invert. In other words, there is an algorithm taking polynomial time to compute function $f(\cdot)$. However, there is no probabilistic algorithm to compute the inverse function $f^{-1}(\cdot)$ in polynomial time. In mathematics, the definition of one-way hash function contains two parts:

1) A polynomial algorithm Al exists so that $\forall x, Al(x) = f(x)$.

2) For every probabilistic polynomial time algorithm Al', every positive polynomial statement $p(\cdot)$ and all sufficiently large k.

$$Pr(Al'(f(x), 1^k) \in f^{-1}(f(x)) < \frac{1}{p(k)}.$$

Based on this definition, there are several well-known one-way functions:

- One-way functions based on computational number theory: RSA [23], Rabin [18], Discrete Logarithms [26].
- Trapdoor Permutation functions (RSA Trapdoor) [5], Clawfree Permutations [6].
- Message hash functions[25]: MD2, MD5, SHA, HAVAL [31].

For implementation, any simple and fast one way hash function can be used as the function $f(\cdot)$. Most of the one-way functions based on the idea of public keys require more computational power than simple message hash functions such as MD5 and SHA. Although MD5 and SHA are efficient, they have fixed output size which is not suitable to produce dynamic keys. In our opinion, HAVAL which can produce different output size is the most suitable one-way function as function $f(\cdot)$.

3.4 Repeated Dynamic Key Generation

When the sequence of dynamic keys is used up, a new sequence of dynamic keys is generated by the dynamic key regeneration process. This process is similar to the original dynamic keys generation. Similar to the initial dynamic key generation scheme, it also has two original requirements: the uniqueness of the dynamic key sequence and the security of current dynamic key from compromised previous dynamic keys. Besides, it is also required that the original initial keys EK and IK are not directly re-used to generate a new sequence. The repeated dynamic key generation scheme has four steps in Figure 2.

Step 1. Both Alice and Bob calculate two extra dynamic keys from the old sequence.

$$DK_{n+1} = f(SK, DK_{n-m+1}, \ldots, DK_{n-1}, DK_n)$$
$$DK_{n+2} = f(SK, DK_{n-m+2}, \ldots, DK_n, DK_{n+1}).$$

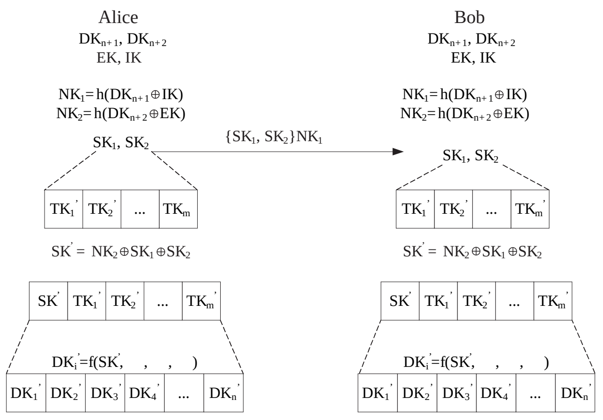

Figure 2: Repeat dynamic key generation

Using them to compute two new initial key using a one way hash function $h(\cdot)$:

$$NK_1 = h(DK_{n+1} \oplus IK)$$
$$NK_2 = h(DK_{n+2} \oplus EK).$$

Step 2. Alice and Bob need a new set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ to generate a new sequence of dynamic keys. To create the new set of temporary keys, Alice randomly generates two session keys SK_1 and SK_2 and sends to Bob encrypted by NK_1.

$$A \rightarrow B : \{SK_1, SK_2\}NK_1, h(SK_1 \oplus SK_2 \oplus NK_1).$$

Both Alice and Bob use these keys to compute the set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ as follows.

$$\begin{aligned} TK_1^{'} &= h(DK_{n-m+4} \oplus SK_1) \\ TK_2^{'} &= h(DK_{n-m+5} \oplus SK_1) \\ &\ldots \\ TK_{m-1}^{'} &= h(DK_{n+2} \oplus SK_1) \\ TK_m^{'} &= h(DK_{SK_2} \oplus SK_1). \end{aligned}$$

Step 3. Both Alice and Bob compute a new seed key $SK^{'}$ from the key NK_2 and the session keys SK_1, SK_2 using bit-wise exclusive or operation.

$$SK^{'} = NK_2 \oplus SK_1 \oplus SK_2.$$

Steps 4. Generate sequence of dynamic keys - this step is the same as Step 4 in the initial dynamic key generation scheme. The new seed key $SK^{'}$ and the new set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ are used to calculate the new sequence of dynamic keys $DK_1^{'}, \ldots, DK_m^{'}$.

$$DK_1^{'} = f(SK^{'}, TK_1^{'}, \ldots, TK_{m-2}^{'}, TK_{m-1}^{'}, TK_m^{'})$$
$$DK_2^{'} = f(SK^{'}, TK_2^{'}, \ldots, TK_{m-1}^{'}, TK_m^{'}, DK_1^{'})$$
$$DK_3^{'} = f(SK^{'}, TK_3^{'}, \ldots, TK_m^{'}, DK_1^{'}, DK_2^{'})$$
$$\ldots$$
$$DK_n^{'} = f(SK^{'}, DK_{n-m}^{'}, \ldots, DK_{n-3}^{'}, DK_{n-2}^{'}, DK_{n-1}^{'}).$$

4 Dynamic Key Discussion and Analysis

This section discusses security and performance issues of the dynamic keys. The first subsection compares the security features between dynamic keys and session keys. The second subsection discusses the ability of dynamic keys to prevent replay attacks on cryptographic protocols. The question of how to find the smallest key size and number of parameters for function f satisfying the security requirements is considered in the third subsection. The forth subsection analyses how to find the most appropriate length of the dynamic key sequence. The matter of the length of the lifetime of a dynamic key be is examined in the fifth subsection. The final subsection discusses the drawback of dynamic key theorems, in particular synchronization problems, how to reduce these problems and re-synchronize dynamic keys.

4.1 Security Comparison Between Dynamic Keys and Session Key in TSL/SSL

The following discussions review and compare security features between dynamic keys and session keys in TLS [22]/SSL [30]. TLS/SSL is currently the most common cryptographic protocol to secure communication over internet. We compare three main features: (a) key exchange, (b) cryptographic keys and cryptanalysis attacks and (c) cryptographic key life time and session hijack.

4.1.1 Key Exchange

At the beginning of TLS/SSL sessions, clients negotiate to share with servers for key exchange protocols available and cryptographic algorithms. The key exchange protocols can be RSA, Diffie-Hellman or ECDH. Kocher [14] found out that RSA, Diffie Hellman protocol can be broken by measuring the amount of time to perform private key operations. Klima et al. [13] also pointed out that the pre-master key can be recovered from inverting RSA encryption. Both of the attacking approaches are based on known plain-text to the server that uses permanent public/private keys. The more key exchange is used to create sessions, the more it exposes vulnerability of TLS/SSL from cryptanalysis attacks on long term public/private keys. The dynamic key cryptography only performs the key exchange/distributing once at the beginning of the initial dynamic key generation. In the repeated dynamic key generation, there is no more key exchange. In opposite of TLS/SSL, the dynamic key cryptography does not have key exchange in every session. Therefore the vulnerability of the key exchange is reduced to minimum in dynamic keys.

4.1.2 Cryptographic Keys and Cryptanalysis Attacks

In TLS/SSL, communication messages within a session are encrypted by a symmetric cryptographic key named session key. Within lifetime of a session, this session key is unchanged. No matter how long a session is, the session key is still valid to encrypt and decrypt messages until the end of the session. The use of a session key within long time may create vulnerability on cryptanalysis attacks on the single session in a weak cryptography. By capturing and analyzing common patterns from an enough number of messages encrypting by the same session key, adversaries may guess correctly the session key. Bard [1] showed a chosen-plaintext cryptanalysis attack on cipher block chaining of SSL 3.0 and TLS 1.0 to break a session key cryptography. If this session key is compromised, communication in the session is vulnerable. In dynamic key cryptography, each dynamic key is used to encrypt only one message. Similar to one-time pad, it is extremely hard to analyze encrypted messages using different dynamic keys to find common patterns to break the cryptography. Even if one or more dynamic keys in the sequence are compromised, only one or a few messages are vulnerable. From these compromised dynamic keys, adversaries cannot guess the next dynamic keys in the sequence to break the cryptographic system. In other words, dynamic keys can reduce the cryptanalysis attack risk.

4.1.3 Cryptographic Key Lifetime and Session Hijack

The longer a session key is used, the more the session is vulnerable under session hijack risks. As on previous discussion, the session key is vulnerable under cryptanalysis attacks. After obtaining the compromised session key, an adversary acting as a proxy can interrupt the connection and hijack the session. From that time, he/she is able to masquerade the authorized user by reading and generating messages with the compromised key without re-authentication. Saito et al. [24] presented two types of attacks to hijack SSL sessions. Because handoff can frequently happens in wireless networks, the risk of session hijacking is even higher than in traditional network. During the hand off, because user's mobile device may change address during reconnecting to other networks, the wireless network connection becomes more vulnerable. Adversaries can perform a fake handoff operation by forcing the client to terminate the connection, and then masquerading this user to take over the session. Long [17] described a common method to hijack session in wireless networks. Because dynamic key cryptography does not use one key for a whole session, a compromised dynamic key cannot be used to hijack a session. Even more than one dynamic key are compromised, adversaries cannot guess the next dynamic key using to encrypt message in the session. The only method to hijack a session is breaking the sequence of dynamic keys. The following subsections discuss how to secure the dynamic key sequence. Table 1 summarizes the security features comparison of cryptographic systems using dynamic key and session key.

Table 1: Comparison between session key and dynamic key

	Dynamic Key	**Session Key**
Key Exchange	Once	Every Session
Life time	Within a message	Within a session
Key Reusable	No	Yes
Vulnerable under man in middle attack	No	Yes
From a compromised cryptographic key, adversary can	Decrypt a message	Decrypt all messages in the session
From a compromised pair of public and private keys of the key exchange protocol	Cryptographic system is still safe	Cryptographic system and session are vulnerable

4.2 Dynamic Key and Replay Attacks

Syverson [27] described replay attacks on cryptography protocols as efforts of using messages captured from previous or current communication to perform unauthorized operations or obtain unauthorized access. Adversaries who perform the replay attacks are supposed to be unable from reading or producing the messages by themselves. They can eavesdrop on communications to capture encrypted requests and then replay them later. Many replay attack scenarios have been analyzed on authentication protocols such as Needham-Schroeder [4], Rees [3]. Even in Kerberos authentication model, Gong [7] also pointed out the possibility of replay attacks when the attacks are performed while the lifetime of the replayed authentication tickets is still valid. These cryptography protocols are vulnerable from replay attacks because they employ reusable authentication keys and session keys. Dynamic keys are able to help to prevent replay attacks. In critical security systems like authentications, a single dynamic key can be used to encrypt only one message. If a dynamic key is used to encrypt two messages, the second message at the receiver will be invalid to decrypt. Because a dynamic key can be used once, a cryptographic message can only be decrypted and validated once. Therefore, authentication servers using dynamic keys can detect replay messages. Without the ability to generate the encrypted messages from correct synchronized dynamic keys, adversaries cannot mount successfully replay attacks on cryptographic protocols using dynamic keys.

4.3 Key Size and Number of Parameters

In this subsection, we analyse how to balance between the efficiency and the security for the dynamic key sequence. Normally, to achieve higher security, cryptographic systems often use longer key sizes. In dynamic key cryptography system, the security of dynamic key sequence also depends of the number of parameters of function $f(\cdot)$ to generate dynamic keys as analysed below. However, the larger of key size and higher number of parameters also require higher storage, computational and communication cost. Instead of aiming for strongly secure dynamic key sequence for all systems, we try to achieve the smallest key size and number of parameters of functions $f(\cdot)$ while maintaining the security level for the cryptographic system. The security level for a dynamic key cryptographic system is presented as the chance to break the sequence of dynamic key. With A being the function to guess a dynamic key, the chance to guess correctly a dynamic key DK_{i-1} is $Pr(A(DK_{i-1}))$. To compute the current and future dynamic keys $DK_i, DK_{i+1}, \ldots$ adversaries must obtain the function $f(\cdot)$ as well as its parameters. From the function $f(\cdot)$ to generate the current dynamic key DK_n in Step 4.

$$DK_n = f(SK, DK_{n-m}, \ldots, DK_{n-2}, DK_{n-1}).$$

Assume that adversary already knows function $f(\cdot)$ and the chances to guess dynamic keys are independent; the chance to break the sequence of dynamic keys can be computed as

$$\begin{aligned} Pr(\{DK_i\}) =& Pr(A(SK)) \times Pr(A(DK_{i-m})) \times \ldots \\ & \times Pr(A(DK_{i-1})). \end{aligned}$$

We can also assume that the cryptographic algorithm is secure so that the adversary has to perform exhaustive search to break it. In other words, the probability of guessing correctly DK_{i-1} is $\frac{1}{2^s}$, with s being the bit length of a dynamic key DK_i. For $m+1$ is the number of parameters of function $f()$, (m is also the number of temporarily key $TK_1, TK_2, \cdots, TK_m$), the chance to break the sequence of dynamic keys can be rewritten as below:

$$Pr(\{DK_i\}) = \frac{1}{2^s} \times \frac{1}{2^s} \times \ldots \times \frac{1}{2^s} = \frac{1}{2^{ms+s}}.$$

This formula shows that the security of symmetric cryptography using dynamic key is improved significantly. Because breaking a cryptographic key does not lead to the vulnerability of the cryptographic system. The probability of breaking dynamic key cryptographic system from a single key $\frac{1}{2^s}$ is reduced into $\frac{1}{2^{ms+s}}$ which is the probability of breaking a whole dynamic key sequence. This formula shows that there are two methods to increase the security of the sequence of dynamic key.

1) To increase the key-size of a dynamic key, or

2) To increase the number of parameters (m) using to create dynamic keys (started with $TK_1 \ldots TK_m$).

However, these solutions also create the following problems:

1) Increasing key-size of the dynamic keys often consumes more computational resource for encrypting and decrypting. It also requires slightly more storage space for the longer key-size in memory.

2) Increasing the number of parameters for function $f(\cdot)$ will use more memory to remember this parameters at all parties, which is not encouraged especially with mobile devices with limited memory and battery power.

From $r(r < \frac{1}{2^s}, r \in [0,1])$ being the highest acceptable probability to break the dynamic key sequence, or $Pr(\{DK_i\}) \leq r$, we can deduce:

$$\frac{1}{2^{ms+s}} \leq r$$
$$\text{or } ms + s \geq log_2 \frac{1}{r} = -log_2 r.$$

Figure 3 describes the relationship between the requirement of value $ms + s$ and the value of r.

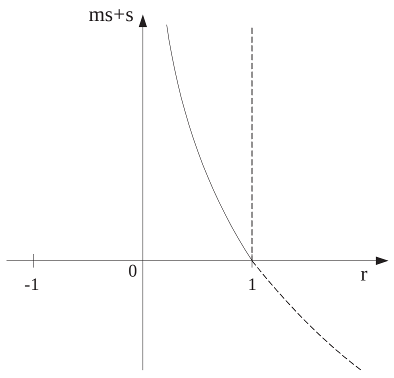

Figure 3: The relationship between the value $ms + s$ and the value of r

For example, with a dynamic key cryptographic system using 256 bits key size and the highest acceptable probability r is $\frac{1}{2^1024}$ (equivalent to 1024 bits key cryptography), the number of parameters m should be $m \geq 3$. In other word, the security of a dynamic key cryptographic system using 256 bits key size and remembering 3 parameters is equivalent to that of a symmetric cryptographic system using 1024 bits key size without sacrificing much computational performance. As the analysis above, the sequence of dynamic key is not forever secure. After a period of time, the probability to break the sequence of dynamic key is increased. A new dynamic key sequence is required to be generated using Repeated Dynamic Key Generation scheme. The following analysis is used to find when it is required to generate a new dynamic key sequence.

4.4 Dynamic Key Sequence Length

A dynamic key sequence is only secure under a limit period of time. As in above analysis, the probability to break the sequence of dynamic key is $Pr(\{DK_i\}) = \frac{1}{2^{ms+s}}$. With enough computational power and amount of time, adversaries are possible to guess the sequence of dynamic keys. To prevent this vulnerability, when the probability to break the dynamic key sequence is higher than the level in security requirement for the system, a new dynamic key sequence is required to be generated. It is assumed that in a unit of time, an adversary can perform x trials to break the dynamic key sequence. At the moment t, he/she may produce xt trials, the probability for the adversary to guess correctly the dynamic key sequence is reduced to be

$$Pr(\{DK_i\}) = \frac{1}{2^{ms+s} - xt}.$$

In security requirement for the system, let r ($\frac{1}{2^{ms+s}} < r < \frac{1}{2^s}$) be the maximum probability for breaking the dynamic key sequence in matter of time, y be the number of used dynamic key in a unit of time, the condition to keep the dynamic key sequence secure is written as follows:

$$Pr(\{DK_i\}) = \frac{1}{2^{ms+s} - xt} \leq r$$
$$\text{or} \quad t \leq \frac{2^{ms+s}}{x} - \frac{1}{xr}$$

with n being the number dynamic keys in the sequence, $n = ty$, we have:

$$n \leq \frac{y2^{m+s}}{x} - \frac{y}{xr}$$

We can conclude the maximum number of keys in the sequence n is $\frac{y2^{m+s}}{x} - \frac{y}{xr}$. For example, with x is $2^{64}, y = 2^{24}, m = 3, s = 256$, r is $\frac{1}{2^{1020}}(\frac{1}{2^{1024}} < r < \frac{1}{2^{256}})$, n should be smaller than 15×2^{980}.

By default, dynamic key system uses only one-time cryptographic key for each message. However, in non-critical communication systems, the analysis in the next subsection shows that the lifetime of dynamic keys can be efficiently extended while the cryptographic systems are still secure.

4.5 Dynamic Key Lifetime

From the beginning of this paper, we assume that every message is encrypted by a different dynamic key. As in the previous subsection, a dynamic key sequence should also be limited in size for being secure. The faster messages are sent in the system, the higher number of dynamic keys in the sequence is used. When the dynamic keys in the sequence are used up, it is required to generate a new sequence. The process to generate new dynamic key sequence creates a considerable overhead in dynamic key cryptography. Instead of using one dynamic key for every message, we propose to re-use the dynamic key until it no longer be secure. Re-using dynamic key does not apply on messages for critical systems (i.e. authentication protocols) which may be vulnerable under replay attacks. Re-using dynamic key may reduce the security of the cryptographic system, thus the dynamic key also has limited lifetime. The secure lifetime for a dynamic key is calculated as follows. It is assumed that in a unit of time, an adversary can perform x' trials to break a dynamic key. At the moment t, he may produce $x't$ trials, the probability for the adversary to guess correctly the dynamic key sequence is reduced to be

$$Pr(DK_i) = \frac{1}{2^s - x't}.$$

In security requirement for the system, $z(z \geq \frac{1}{2^s})$ be the maximum probability for breaking a dynamic key, y' be the number of sending messages in a unit of time, condition to check whether dynamic key is still secure is written as follows.

$$\begin{aligned} Pr(DK_i) = \frac{1}{2^s - x't} &\leq z \\ \text{or} \quad t \leq \frac{2^s}{x'} &- \frac{1}{x'z}. \end{aligned}$$

With o being the number of sent messages, $o = ty'$, we can find the average number of times to use a dynamic key has to satisfy the following condition:

$$o \leq \frac{y'2^s}{x'} - \frac{y'}{x'z}.$$

Or we can conclude that the maximum number of message is $\frac{y'2^s}{x'} - \frac{y'}{x'z}$. For example, with $x' = 2^{64}, z = \frac{1}{2^{250}}, y' = 2^{24}$ and $s = 256, o$ should be smaller than (64×2^{210}).

4.6 Synchronization Problem

Besides security advantages, dynamic key theory has a major drawback: synchronization problem. By using symmetric cryptography, dynamic keys must be identical between senders and receivers. When cryptographic keys between senders and receivers are not the same, communication breaks down because receivers are no longer able to decrypt messages from senders. This issue is called synchronization problem in dynamic keys. There are many reasons causing the synchronization problem which are malicious attacks from adversaries or connection problems. Adversaries can masquerade as senders to send messages encrypted by random cryptographic keys to confuse receivers. In one case, adversaries can play "man in the middle attack" to interfere or intercept into the communication and modify message contents. In another case, because of connection problems, the communication can be interrupted which may happen very frequently in wireless networks. Broken clients or stolen devices may also create synchronization problem. Finally synchronization problem can happen when adversaries break the dynamic key sequence. The synchronization problem by attacks from adversary can be reduced by adding authentication to the communication. By limiting the dynamic key change within authenticated communication, messages from masquerade senders can be detected and ignored. Messages sending before authentication can utilize hash functions as message authentication to verify the authentication of the senders. However, these solutions may decrease the performance and security of dynamic key cryptography. When dynamic keys between senders and receivers are no longer identical, the synchronization process is invoked to generate new dynamic key sequence. Sender Alice notices that communication with receiver Bob is broken because of synchronization problem. Alice sends Bob a re-synchronization message. Bob sends an acknowledgement to Alice and restarts the initial dynamic key generation scheme by re-sending new EK' and IK' keys via a secure channel.

5 Case Study

Two applications are discussed in the case study. An authentication protocol using dynamic keys between two parties is described first. This protocol uses only three messages for secure mutual authentication. There is no session key exchange in this authentication protocol. The security protocol is verified by SVO formal logical method. The second application is a payment protocol using smart cards. In this application, dynamic keys are computed by smart cards. The protocol uses dynamic keys to encrypt and decrypt the messages sending between users, merchants and banks. The analysis under SVO confirms that security requirements of the protocol including integrity, authentication and non-repudiation are fulfilled. To describe and analyze the two applications, the first subsection describes the notation using in this case study. The notations of the case study are listed in Table 2.

5.1 Authentication Protocol

In [21], a dynamic key authentication protocol is presented using third party server. This subsection presents a dynamic key direct authentication between two parties A and B. The main purpose of the protocol is B verifies the identity claiming by A. By proving the ownership of dynamic keys in the sequence, A creates a trust on his claiming identity at B. Figure 4 shows the message flow of the protocol.

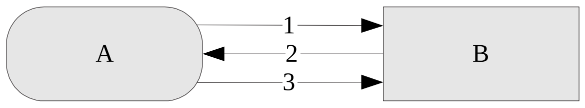

Figure 4: Authentication protocol

Table 2: Notation of the case study

N_A, N_B	nonces (random numbers)
K_t, K_{t+1}, K_{t+2}	dynamic keys ($1 \leq t \leq n-2$)
C	client
M	merchant
B	bank
CID	client's Card ID
$N_{C1}, N_{M1}, N_{B1}, N_{B2}, N_{B3}$	nonces (random numbers
X	money amount to be transfered from client's account to merchant's account
TID	Bank transaction ID
R	result of the transfer operation
$K_{Mt}, K_{Mt+1}, K_{Mt+2}, K_{Mt+3}$	dynamic keys of merchant (K_{Mi})
$K_{Ct}, K_{Ct+1}, K_{Ct+2}, K_{Ct+3}$	dynamic keys of client (K_{Ci})

The protocol is formally described as follows:

1. $A \rightarrow B$: $A, N_A, h(N_A \oplus K_t)$;

2. $B \rightarrow A$: $\{N_A + 1, N_B\}K_{t+1}$;

3. $A \rightarrow B$: $\{N_B + 1\}K_{t+2}$.

In Message 1, A starts an authentication request with a nonce N_A and a message authentication code $h(N_A, K_t)$ to authenticate the source of the authentication request. This message authentication code is used to reduce dynamic key synchronization problem in the analysis above. The rest of the messages in authentication protocol is similar to traditional authentication protocols but without session key exchange. In Message 2, B responses to the challenge from A by encrypting N_A with K_{t+1}. In turn, A also responses to B's challenge by encrypting N_A with K_{t+2} in Message 3.

The security of the authentication protocol is analyzed using SVO [28] logic. The goals of the protocol in SVO are:

G1. P believes Q says X;

G2. P believes (Q says $F(X, N_P) \wedge fresh(N_P)$;

G3. P believes $P \overset{K_{PQ}^{-}}{\longleftrightarrow} Q$;

G4. P believes $fresh(K_{PQ})$;

G5. P believes Q says $Q \overset{K_{PQ}^{-}}{\longleftrightarrow} P$;

G6. P believes $(P \overset{K_{PQ}^{-}}{\longleftrightarrow} Q \ \wedge \ Q$ says $F(K_{PQ}))$.

In this authentication protocol, there is no session key K_{AB}. Dynamic keys $K_i, \forall i > t+2$ are assumed to be communication keys between A and B after authentication is completed. There is one extra requirement in the authentication to reduce the synchronization problem.

G7. B believes A says (N_A).

Initial State Assumptions

P1. A believes $A \overset{K_i}{\longleftrightarrow} B$, $\forall i = 1, \cdots, n$;

P2. B believes $A \overset{K_i}{\longleftrightarrow} B$, $\forall i = 1, \cdots, n$;

P3. A believes $fresh(K_j)$, $\forall j = 1, \cdots, n$;

P4. B believes $fresh(K_j)$, $\forall j = 1, \cdots, n$;

P5. A believes $fresh(N_A)$;

P6. B believes $fresh(N_B)$.

P1 to P2 note that both A and B are assumed to believe in the dynamic keys. P3 and P4 note that they also believe in the freshness of the dynamic keys because they generate these dynamic keys offline by themselves.

Received Message Assumptions

P7. B received $(A, N_A, h(N_A \oplus K_t))$;

P8. A received $\{N_A + 1, N_B\}K_{t+1}$;

P9. B received $\{N_B + 1\}K_{t+2}$.

Comprehension Assumptions

P10. B believes B received $(A, \langle N_A \rangle_{*B}$ from A, $h(\langle N_A \rangle_{*B}, K_t))$;

P11. A believes A received $\{N_A + 1, \langle N_B \rangle_{*A}$ from B $\}Kt + 1$;

P12. B believes B received $\{N_B + 1\}K_{t+2}$.

Interpretation Assumptions

P13. B believes B received $(A, \langle N_A \rangle_{*B}, h(\langle N_A \rangle_{*B} \oplus K_t)) \wedge$ B believes A $\overset{K_t}{\longleftrightarrow} B \ \longrightarrow$ B believes B received $(A, \langle N_A \rangle_{*B}, A \overset{K_t}{\longleftrightarrow} B)$;

P14. A believes A received $\{N_A + 1, \langle N_B \rangle_{*A}\}K_{t+1} \wedge$ A believes A $\overset{K_{t+1}}{\longleftrightarrow} B \ \longrightarrow$ A believes A received $(N_A + 1, \langle N_B \rangle_{*A}\}K_{t+1}, A \overset{K_t}{\longleftrightarrow} B)$;

P15. B believes B received $\{N_A + 1, \langle N_B\rangle_{*A}\}K_{t+2}\wedge$ B believes A $\overset{K_{t+2}}{\longleftrightarrow}$ B $\longrightarrow$ B believes B received $(N_B, A \overset{K_{t+2}}{\longleftrightarrow} B)$.

Derivations for A

i. A believes B says N_B by Source of Association Axioms, P14, and P1.

ii. A believes (B says $\{N_A + 1, N_B\}K_{t+1})\wedge$ fresh(N_A)) by Sources of Association Axioms, P14, P1, P5, and Belief Axioms.

iii. A believes B says $A \overset{K_{t+1}}{\longleftrightarrow} B$ by Sources of Association Axioms, Belief Axioms, Saying Axioms and P14.

From the analysis above, we can derive the following conclusion. For A, G1 is derived in (i), G2 in (ii), G3 in P1, G4 in P3, G5 in (iii). Finally G6 is derived from Belief Axioms, P1, P14.

Derivation for B

i. B believes A says $N_B + 1$ by Sources of Association Axioms, P15 and P1.

ii. B believes (A says $\{N_B + 1\}K_{t+2}\wedge$ fresh(N_B)) by Sources of Association Axioms, P15, P2, P6, and Belief Axioms.

iii. B believes A says $A \overset{K_{t+2}}{\longleftrightarrow} B$ by Sources of Association Axioms, Belief Axioms, Saying Axioms and P15.

iv. B believes (A says N_A) by Source of Axioms, Saying Axioms and P13.

Similar to derivation for A, from the analysis above, we can derive the following conclusion. For B, G1 is derived in (i), G2 in (ii), G3 in P1, G4 in P3, G5 in (iii), G7 in (iv). G6 is derived from Belief Axioms, P1, P14.

5.2 Payment Protocol

Payment protocol is the protocol communicating between three parties: Client(C), Merchant(M) and Bank(B). The main purpose of the payment protocol includes:

- M and C authenticate to B to prove their identities.
- M requests B to conduct a money transfer of amount X from account C to B for payment transaction TID.
- B requests the payment confirmation from C. When the payment is confirmed, the transaction is performed and noticed to the participated parties.

In the payment protocol, C does not trust M. Hence M is not allowed to act on behalf of M to complete the transaction. In other words, M cannot obtain authentication key of C. For privacy reason, C does not reveal his identity to M. However for non-repudiation, Client uses his/her CID (card ID) instead of his/her identity C. In the payment protocol, both C and M must authenticate to B. Besides, C can decide to confirm the request to complete the transaction which transfer amount X from his account to M's account. C also requires mutual authentication from B so that C can trust that he/she is dealing with his/her bank.

The following is the formal description of the protocol.

1. $C \rightarrow M : CID, N_{C1}, h(N_{C1} \oplus K_{Ct})$;

2. $M \rightarrow B$: $M, N_{M1}, \{N_{C1}, N_{M1}, h(N_{C1} \oplus K_{Ct}), CID, X\}K_M t$;

3. $B \rightarrow M$: $\{N_{C1}, N_{B1}, M, X\}K_{Ct+1}$, $\{N_{M1} + 1, N_{B2}\}K_{Mt+1}$;

4. $M \rightarrow C : \{N_{C1}, N_{B1}, M, X\}K_{Ct+1}$;

5. $C \rightarrow M : \{C, CID, N_{B1} + 1, M, X\}K_{Ct+2}$;

6. $M \rightarrow B : \{C, CID, N_{B1} + 1, M, X\}K_{Ct+2}$, $\{N_{B2}, X\}K_{Mt+2}$;

7. $B \rightarrow M : \{R, TID\}DK_{Mt+3}, \{R, N_{B3}, TID\}DK_{Ct+3}$;

8. $M \rightarrow C : \{R, N_{N3}, TID\}DK_{Ct+3}$.

In the first message, client uses smart card to send a random number N_{C1} and card identity CID to merchant. In the second message, the merchant M combines a message starting a request to transfer X from client Cs account to Ms account encrypting by the dynamic key of merchant. In the third message, the bank receives the request message and starts the authorization request message encrypted by the dynamic key of client, in the message, bank also request merchant to authenticate as well. After receiving the Message 3, merchant forwards the authorization request to client C in Message 4. Client receives the request message and asks user for confirm to transfer amount X from his account to M. When user agrees and confirms the authorization, client combines the authorization message encrypted by his new dynamic key to Message 5 and sends it to the merchant. The authorization is forwarded to the bank in Message 6 including the authentication from merchant. After receiving the authorization and authentication from merchant, the bank performs the money transferring and completes the transaction by sending the result of the transferring and transaction identity back to merchant in Message 7. Finally, the result is forwarded again to client in Message 8. Figure 5 depicts the protocol.

Again, SVO is used to verify the security of the payment protocol. The initial State Assumptions are:

P1. C believes $C \overset{K_{Ci}}{\leftrightarrow} B$, $\forall i = 1, \cdots, n$;

P2. B believes $C \overset{K_{Ci}}{\leftrightarrow} B$, $\forall i = 1, \cdots, n$;

P3. M believes $M \overset{K_{Mj}}{\leftrightarrow} B$, $\forall j = 1, \cdots, n$;

P4. B believes $M \overset{K_{Mj}}{\leftrightarrow} B$, $\forall j = 1, \cdots, n$;

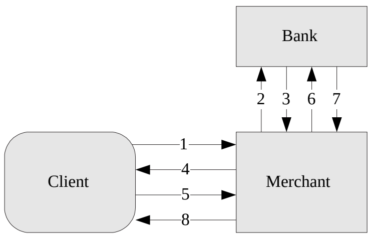

Figure 5: Payment protocol

P5. C believes $fresh(K_{Ci})$, $\forall i = 1, \cdots, n$;

P6. B believes $fresh(K_{Ci})$, $\forall i = 1, \cdots, n$;

P7. M believes $fresh(K_{Mj})$, $\forall j = 1, \cdots, n$;

P8. B believes $fresh(K_{Mj})$, $\forall j = 1, \cdots, n$;

P9. C believes $fresh(N_{C1})$;

P10. M believes $fresh(N_{M1})$;

P11. B believes $fresh(N_{B1})$;

P12. B believes $fresh(N_{B2})$.

P1 to P4 note that C, B and M are assumed to believe in the dynamic keys. P5 and P8 note that they also believe in the freshness of the dynamic keys because they generate these dynamic keys offline by themselves.

Received Message Assumptions

P13. M received $(CID, N_{C1}, h(N_{C1} \oplus K_{Ct}))$;

P14. B received $(M, N_{M1}, \{N_{C1}, N_{M1}, h(N_{C1} \oplus K_{Ct}), CID, X\}K_Mt)$;

P15. M received $(\{N_{C1}, N_{B1}, M, X\}K_{Ct+1}, \{N_{M1}+1, N_{B2}\}K_{Mt+1})$;

P16. C received $(\{N_{C1}, N_{B1}, M, X\}K_{Ct+1})$;

P17. M received $\{C, CID, N_{B1}+1, M, X\}K_{Ct+2}$;

P18. B received $(\{C, CID, N_{B1}+1, M, X\}K_{Ct+2}, \{N_{B2}, X\}K_{Mt+2})$;

P19. M received $(\{R, TID\}DK_{Mt+3}, \{R, N_{B3}, TID\}DK_{Ct+3})$;

P20. C received $(\{R, N_{N3}, TID\}DK_{Ct+3})$.

Messages 7 and 8 are the acknowledgement of the protocol containing the result. These messages are left out of the analysis because they are not involved in the authentication and authorization of the payment protocol.

Comprehension Assumptions

P21. M believes M received $(CID, \langle N_{C1}\rangle_{*M}$ from C, $\langle h(N_{C1} \oplus K_{Ct})\rangle_{*M})$;

P22. B believes B received $(M, \langle N_{M1}\rangle_{*B}$ from M, $\{\langle N_{C1}\rangle_{*B}$ from $C, \langle N_{M1}\rangle_{*B}$ from $M, CID, X\}K_{Mt})$;

P23. M believes M received $(\langle\{N_{C1}, N_{B1}, M, X\}K_{Ct+1}\rangle_{*M}$, $\{N_{M1}+1, \langle N_{B2}\rangle_{*M}$ from $B\}K_{Mt+1})$;

P24. C believes C received $\{N_{C1}, \langle N_{B1}\rangle_{*C}$ from B, M, $X\}K_{Ct+1}$;

P25. M believes M received $\langle\{C, CID, N_{B1}+1, M$, $X\}K_{Ct+2}\rangle_{*M}$;

P26. B believes B received $(\{C, CID, N_{B1}+1, M$, $X\}K_{Ct+2}.\{N_{B2}, X\}K_{Mt+2})$.

Interpretation Assumptions

P27. B believes B received $(M, \langle N_{M1}\rangle_{*B}, \{\langle N_{C1}\rangle_{*B}$, $CID, X\}K_{Mt}) \wedge B$ believes $M \overset{K_{Mt}}{\longleftrightarrow} B \longrightarrow B$ believes B received $(\langle N_{C1}\rangle_{*B}, \langle N_{M1}\rangle_{*B}, CID, X$, $M \overset{K_{Mt}}{\longleftrightarrow} B)$;

P28. M believes M received $(\langle N_{C1}, N_{B1}, M$, $X\}K_{Ct+1}\rangle_{*M}, \{N_{M1}+1, \{\langle N_{B2}\rangle_{*M}\}K_{Mt+1}) \wedge M$ believes $M \overset{K_{Mt+1}}{\longleftrightarrow} B \longrightarrow M$ believes M received $(\langle\{N_{C1}, N_{B1}, M, X\}K_{Ct+1}\rangle_{*M}, N_{M1}, \langle N_{B2}\rangle_{*M}$, $M \overset{K_{Mt+1}}{\longleftrightarrow} B)$;

P29. C believes C received $\{N_{C1}, \langle N_{B1}\rangle_{*C}$ from B, M, $X\}K_{Ct+1} \wedge C$ believes$C \overset{K_{Ct+1}}{\longleftrightarrow} B \longrightarrow C$ believes C received $(N_{C1}, \langle\{N_{B1}\rangle_{*C}, M, X, C \overset{K_{Ct+1}}{\longleftrightarrow} B)$;

P30. B believes B received $(\{C, CID, N_{B1}+1, M$, $X\}K_{Ct+2}, \{N_{B2}, X\}K_{Mt+2} \wedge B$ believes $M \overset{K_{Mt+2}}{\longleftrightarrow} B \wedge$ B believes $C \overset{K_{Ct+2}}{\longleftrightarrow} B \longrightarrow B$ believes B received $(N_{B1}+1, C \overset{K_{Ct+2}}{\longleftrightarrow} B, N_{B2}, M \overset{K_{Mt+2}}{\longleftrightarrow} B)$.

Derivations for C

i. C believes B says N_{C1} by Source of Association Axioms, P29, and Saying Axioms.

ii. C believes $(B$ says $\{N_{C1}, N_{B1}, M, X\}K_{Ct+1}) \wedge$ fresh(N_{C1})) by Sources of Association Axioms, P29, P1, P9, and Belief Axioms.

iii. C believes B says $C \overset{K_{Ct+1}}{\leftrightarrow} B$ by Sources of Association Axioms, Belief Axioms, Saying Axioms and P29.

From the analysis above, we can derive the following conclusion. For C, G1 is derived in (i), G2 in (ii), G3 in P1, G4 in P5, G5 in (iii). G6 is derived from Belief Axioms, P1, P29.

Derivations for M

i. M believes B says N_{M1} by Source of Association Axioms, P28, and Saying Axioms.

ii. M believes (B says $\{N_{M1}, N_{B2}\}K_{Mt+1}$) $\wedge$ fresh(N_{M1})) by Sources of Association Axioms, P28, P3, P10, and Belief Axioms.

iii. M believes B says $M \overset{K_{Mt+1}}{\leftrightarrow} B$ by Sources of Association Axioms, Belief Axioms, Saying Axioms and P28.

From the analysis above, we can derive the following conclusion. For M, G1 is derived in (i), G2 in (ii), G3 in P1, G4 in P5, G5 in (iii). G6 is derived from Belief Axioms, P3, P28.

Derivations for B in Authentication of C

i. B believes C says $N_{B1} + 1$ by Source of Association Axioms, P30, and Saying Axioms.

ii. B believes (C says $\{C, CID, N_{B1}+1, M, X\}K_{Ct+2}$) $\wedge$ fresh(N_{B1})) by Sources of Association Axioms, P30, P2, P11, and Belief Axioms.

iii. B believes C says $C \overset{K_{Ct+2}}{\leftrightarrow} B$ by Sources of Association Axioms, Belief Axioms, Saying Axioms and P30.

For B, G1 is derived in (i), G2 in (ii), G3 in P1, G4 in P5, G5 in (iii). G6 is derived from Belief Axioms, P2, P30.

Derivations for B in Authentication of M

i. B believes M says N_{B2} by Source of Association Axioms, P30, and Saying Axioms.

ii. B believes (M says $\{N_{B2}, X\}K_{Mt+2}$) $\wedge$ fresh(N_{B2})) by Sources of Association Axioms, P30, P4, P12, and Belief Axioms.

iii. B believes M says $M \overset{K_{Mt+2}}{\leftrightarrow} B$ by Sources of Association Axioms, Belief Axioms, Saying Axioms and P30.

For B, G1 is derived in (i), G2 in (ii), G3 in P1, G4 in P5, G5 in (iii). G6 is derived from Belief Axioms, P4, P30.

6 Conclusions

In this paper, we present a dynamic key theory to improve security and efficiency of symmetric cryptography. The advantages and disadvantages of the dynamic key theory have been analyzed to achieve the highest performance and securing the cryptography. Key size, sequence length and synchronization problem are investigated to improve the theory. The analysis shows that the security of symmetric cryptography using dynamic key is significantly improved. The probability to break cryptographic system is reduced from $\frac{1}{2^s}$ to $\frac{1}{2^{ms+s}}$, with s being the bit length of cryptographic key and m being the number of remembered parameters. In the storage aspect, we find out the higher secure cryptography requires the more memory space for more parameters for function $f()$. Two case studies, authentication and payment protocols, are examined and analyzed to demonstrate the power of dynamic keys to secure transaction in internet banking and mobile payment. In future work, a synchronization problem will be investigated further to achieve better security in re-synchronization. We are also developing a logical formalization method for dynamic key protocols based on the SVO Logic.

Acknowledgments

The authors are grateful to the anonymous reviewers for valuable comments.

References

[1] G. V. Bard, "A challenging but feasible blockwise-adaptive chosen-plaintext attack on SSL," *IEEE International Conference on Security and Cryptography*, INSTICC Press, 2006.

[2] M. Blaze, W. Diffie, R. L. Rivest, B. Schneier, T. Shimomura, E. Thompson, and M. Wiener, *Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security*, Report of Ad Hoc Panel of Cryptographers and Computer Scientists, Jan. 1996. (http://www.crypto.com/papers/)

[3] C. Boyd and W. Mao, "On a limitation of BAN logic," *Eurocrypt'93*, LNCS 765, pp. 240-247, Springer-Verlag, 1993.

[4] M. Burrows, M. Abadi, R. Needham , "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18-36, 1990.

[5] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Infomation Theory*, vol. 22, no. 6, pp. 644-654, 1976.

[6] O. Goldreich, *Foundations of Cryptography: Vol. 1, Basic Tools*, Cambridge University Press, 2007.

[7] L. Gong, "A security risk of depending on synchronized clocked," *ACM SIGOPS Operating Systems Review*, vol. 26, no. 1, pp. 49-53, 1992.

[8] N. M. Haller, "The S/KEY TM one-time password system", *Proceedings of the Internet Society Symposium on Network and Distributed Systems*, pp. 151-157, 1994.

[9] D. Hankerson, A. J. Menezes, and S Vanstone, *Guide to Elliptic Curve Cryptography*, Springer, Jan. 2004.

[10] D. Kahn, *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*, Scribner, 1996.

[11] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*, Chapman & Hall/CRC, 2007.

[12] J. Kirk, "Researcher: RSA 1024-bit encryption not enough", *PC World, IDG News Service*, 2007. (http://www.pcworld.com/article/132184/researcher_rsa_1024bit_encryption_not_enough.html)

[13] V. Klma, O. Pokorn, T. Rosa, "Attacking RSA-based sessions in SSL/TLS," *Proceeding of Cryptographic Hardware and Embedded Systems,* LNCS 2779, pp. 426-440, Springer Berlin, 2003.

[14] P. C. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems," *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, LNCS 1109, pp. 104-113, Springler Verlag, 1996.

[15] S. Kungpisdan, P. D. Le, B. Srinivasan, "A limited-used key generation scheme for Internet transactions," *Proceeding of Workshop on Information Security Applications*, LNCS 3325, pp. 302-316, Springer Berlin, 2005.

[16] L. Law, A. Menezes, M. Qu, J. Solinas, S. Vanstone, "An efficient protocol for authenticated key agreement," *Designs, Codes and Cryptography*, vol. 28, no. 2, pp. 119-134, 2003.

[17] X. Long and B. Sikdar, "Wavelet based detection of session hijacking attacks in wireless networks," *Proceeding of IEEE Global Telecommunications Conference, IEEE GLOBECOM*, pp. 1-5, 2008.

[18] A. J. Menezes, P. C. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*, CRC Press, 1996.

[19] J. C. Mitchell, V. Shmatikov, and U. Stern, "Finite-state analysis of SSL 3.0," *Proceeding of the 7th Conference on USENIX Security Symposium*, pp. 201-206, San Antonio, Texas, 1998.

[20] N. Nagaraj, V. Vaidya, and P. G. Vaidya, "Revisiting the one-time pad," *International Journal of Network Security*, vol. 6, no. 1, pp. 94-102, 2008.

[21] H. H. Ngo, X. Wu, P. D. Le and C. Wilson, "A method for Authentication services in wireless networks," *The Proceeding of 14th Americas Conference on Information Systems*, pp. 1-9, Toronto, Canada, 2008.

[22] L. C. Paulson, "Inductive analysis of the Internet protocol TLS," *ACM Transactions on Information and System Security*, vol. 2, no. 3, pp. 332-351, 1999.

[23] R. L. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[24] T. Saito , R. Hatsugai, T. Kito, "On compromising password-based authentication over HTTPS," *Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06)*, pp. 869-874, 2006.

[25] B. Schneier, *Applied Cryptography*, John Wiley & Sons, 1996.

[26] D. Stinson, *Cryptography: Theory and Practice*, CRC-Press, 1995.

[27] P. Syverson, "A taxonomy of replay attacks," *Proceedings of the 7th IEEE Computer Security Foundations Workshop*, pp. 187-191, 1994.

[28] P. Syverson and I. Cervesato, "The logic of authentication protocols," *Processing of Foundations of Security Analysis and Design*, LNCS 2171, pp. 63-136, Springer-Verlag, 2001.

[29] J. Talbot and D. Welsh, *Complexity and Cryptography: An Introduction*, Cambridge University Press, 2006.

[30] D. Wagner and B. Schneier, "Analysis of the SSL 3.0 protocol," *Proceeding of the Second USENIX Workshop on Electronic Commerce*, pp. 29-40, 1996.

[31] Y. Zheng, J. Pieprzyk, J. Seberry, "HAVAL - a one-way hashing algorithm with variable length of output," *Proceeding of International Conference on Cryptology: Advances in Cryptology, AUSCRYPT*, LNCS 718, pp. 81-104, Springer-Verlag, 1993.

Huy Hoang Ngo is a Ph.D. Candidate at Monash University. He received his MS degree in Information Technology from Queensland University, Australia, 2002. He also holds a Bachelor of Information Technology from HCMC University of Technology, Vietnam. His research interests include authentication, network security, cryptographic protocol, and cryptanalysis.

Xianping Wu is a Ph.D. Candidate at Monash University. He holds two master degrees in network computing minor thesis and MBA from Monash and Notre Dame Universities Melbourne, Australia. He also holds a Bsc from the YanShan University, China. His research interests include network security, cryptographic algorithms and sensitive information retrieval.

Phu Dung Le is a lecturer in Monash University, Melbourne, Australia. His main research interests are including: Image and Video Quality Measure and Compression, Mobile Agents and Security in the Quantum Computing Age. He has taught Data Communication, Operating System, Computer Architecture, Information Retrieval and Unix Programming. Dr. Le has also researched Mobile Computing, Distributed Migration.

Campbell Wilson is a senior lecturer in Monash University, Melbourne, Australia. He received his Ph.D. degree from Monash Univeristy in 2002. He is chairman of Technical Services Committee of the Faculty of Information Technology. His research interests include Information Retrieval, Image Retrieval and Data Warehouse.

Bala Srinivasan is a Professor in Monash University, Melbourne, Australia. He has authored and jointly edited 6 technical books and authored and co-authored more than 200 international refereed publications in journals and conferences in the areas of Databases, Multimedia Retrieval Systems, Distributed and Mobile Computing and Data Mining.