International Journal of Electrical, Electronics and Computer Engineering **3**(2): 132-137(2014)

ISSN No. (Online): 2277-2626

Efficient Re-Keyed Encryption Schemes for Secure Communications

Md Raghib Jamal* and Mohd Ahtesham Farooqui*

**Department of Computer Science,*
Jazan University, KSA

(Corresponding author: Mohd Ahtesham Farooqui)

(Received 05 October, 2014 Accepted 22 November, 2014)

ABSTRUCT: There are many aspects to security and many applications, ranging from secure commerce and payments to private communications and protecting passwords. One essential aspect for secure communications is that of cryptography. A fundamental problem in cryptography is how to communicate securely over an insecure channel, which might be controlled by an adversary. This problem has for a long time fascinated people and has become even more important with the proliferation of computers and communication networks such as the Internet. From simple message exchanges among friends to purchases made with a credit card, to the transmission of sensitive documents, these communication systems are now being used for many different purposes. In this paper we present a new encryption schemes based on re-keyed. If encryption is performed under a single key there are a certain maximum threshold number of messages that can be safely encrypted. It thus effectively extends the lifetime of the key increasing the threshold number of encryptions that can be performed without requiring a new exchange of keys.

Keywords: Encryption, Authentication, re-keyed, communicate, message, signature.

I. INTRODUCTION

Encryption is certainly one of the most fundamental problems in cryptography and it is usually what it comes to mind when one talks about cryptography. The latter however encompasses many other areas including but not limited to message authentication, digital signatures, identification protocols, and zero knowledge protocols. Re-keying can be used in conjunction with any shared-key based cryptographic data processing. This might be data encryption, under any of the common modes of operation; it might be data authentication using some MAC. It might be something else. We wish to provide tools that enable the analysis of any of these situations. In Sect. II we present the need and significance of our approach. The section III we give a literature review of notions. Then, we describe in section IV our new Re-keyed Encryption Schemes in section V we provide the algorithms and proof of our construction while in section VI we analysis its efficiency in practice.

II. NEED AND SIGNIFICANCE

The common attacks base their success on the ability to get lots of encryptions under a single key. For example differential or linear cryptanalysis will recover a DES key once a certain threshold number of encryptions have been performed using it. Furthermore the most modes of operation are subject to birthday attack leading to compromise of the privacy of a scheme based on a block cipher with block size k once 2 k/2 encryptions are performed under the same key.

III. LITERATURE REVIEW

There are several ways of classifying cryptographic algorithms. The three types of algorithms.(i). Secret key cryptography (SKC): Uses a single key for both encryption and decryption.(ii). Public key cryptography (PKC): Uses one key for encryption and another for decryption. (iii). Hash functions: Uses a mathematical transformation to irreversibly "encrypt" information. Over the last decade several schemes for public-key encryption have been proposed in the literature.

A. Diffie-Hellman Integrated Encryption Scheme

Diffie-Hellman based encryption is designed to be a natural extension of the ElGamal scheme. This is suitable in a variety of groups and which enhanced ElGamal in a couple of ways important to cryptographic practice. (i). The scheme needs to provide the capability of encrypting arbitrary bit strings. (ii). The scheme should be secure against chosen-ciphertext attack (ElGamal is not). The above two goals have to be realized without increasing the number of group operations for encryption and decryption and without increasing key sizes relative to ElGamal. The approach above is somewhat in contrast to related schemes in the literature. Diffie-Hellman works like this. Alice and Bob start by agreeing on a large prime number, N. They also have to choose some number G so that G<N. There is actually another constraint on G, namely that it must be primitive with respect to N. Primitive is a definition that is a little beyond the scope of our discussion but basically G is primitive to N if we can find integers i so that Gi mod N = j for all values of j from 1 to N-1.

As an example, 2 is not primitive to 7 because the set of powers of 2 from 1 to 6, mod 7 (i.e., 21 mod 7, 22 mod 7 ... 26 mod 7) = {2,4,1,2,4,1}. On the other hand, 3 is primitive to 7 because the set of powers of 3 from 1 to 6, mod 7 = {3,2,6,4,5,1}. DHIES is a very natural scheme. The method follows standard ideas and practice. Naturally it is secure. But it seems difficult to prove security under existing assumptions about the Diffie-Hellman problem. More typical is to fix an assumption and then strive to find the lowest cost scheme which can be proven secure under that assumption.

B. RSA Public-Key Cryptography

Unlike Diffie-Hellman, RSA can be used for key exchange as well as digital signatures and the encryption of small blocks of data. Today, RSA is primarily used to encrypt the session key used for secret key encryption (message integrity) or the message's hash value (digital signature). RSA's mathematical hardness comes from the ease in calculating large numbers and the difficulty in finding the prime factors of those large numbers. Although employed with numbers using hundreds of digits, the math behind RSA is relatively straight-forward. To create an RSA public/private key pair, here are the basic steps: Choose two prime numbers, p and q. From these numbers you can calculate the modulus, n = pq. Select a third number, e, that is relatively prime to (i.e., it does not divide evenly into) the product (p-1)(q-1). The number e is the public exponent. Calculate an integer d from the quotient (ed-1)/[(p-1)(q-1)]. The number d is the private exponent. The public key is the number pair (n,e). Although these values are publicly known, it is computationally infeasible to determine d from n and e if p and q are large enough. To encrypt a message, M, with the public key, create the ciphertext, C, using the equation:$C = M^e \bmod n$
The receiver then decrypts the ciphertext with the private key using the equation: $M = C^d \bmod n$

C. Cramer and Shoup Scheme

Cramer and Shoup and that of Shoup who start from the decisional Diffie-Hellman assumption, and then try to find the best scheme they can that will resist chosen-ciphertext attack under this assumption. In fact, the latter can also be proved secure in the random oracle model based on the weaker computational Diffie-Hellman assumption. These schemes are remarkable but their costs are about double that of ElGamal, which is already enough to dampen some practical interest. Their scheme is more costly than ours in terms of key sizes, encryption time, and decryption time (in particular, encryption takes five exponentiations), but the scheme is still practical. The notions of in distinguish ability and semantic security, and their equivalence under chosen-plaintext attack is due to. The notion of chosen-cipher-text security that we use is due to.

D. Zheng and Seberry Scheme

They have proposed an ElGamal-based scheme that uses universal one-way hash functions. Security of their scheme is not supported by proofs in the reductionist sense of modern cryptography. Lim and Lee have pointed out that in some of the cryptosystems proposed in , the method of adding authentication capability may fail just under known plaintext attacks. A submission to IEEE P1363a based on has been made by Zheng. Another contemporaneous suggestion was proposed by Johnson, Matyas and Peyravian. Assume that the message M already contains some redundancy and unpredictability.

IV. PROPOSED APPROACH

A Re-keying is a commonly employed paradigm in computer security systems about whose security benefits users appear to have various expectations. Say two parties share a key K, and want to encrypt data they send to each other. They will use some block cipher based mode of operation, say CBC. The straightforward approach is to use K directly to encrypt the data. An often employed alternative is re-keyed encryption. The key K is not used to encrypt data but rather viewed as a master key. Sub-keys K1, K2, K3 . . . are derived from K, by some process called the re-keying process. A certain number l of messages is encrypted using K1 and then the parties switch to K2. Once l messages have been encrypted under K2 they switch to K3 and so on. As a sample of our results we discuss CBC encryption. Suppose we CBC encrypt with a block cipher F having key length and block-length k. Let's define the encryption threshold as the number Q of k-bit messages that can be safely encrypted. We know from that this value is Q ≈ 2 k/2 for the single-key scheme. We now consider re-keyed CBC encryption under the parallel or serial re-keying methods discussed above where we use the same block cipher F as the re-keying function. We show that by re-keying every 2k/3 encryptions i.e. set the subkey lifetime l = 2 k/3 the encryption threshold increases to Q ≈ 2 2k/3. That is one can safely encrypt significantly more data by using re-keying. Let SE = (Ke, E, D) be the base (symmetric) encryption scheme, specified by its key generation, encryption and decryption algorithms. Let G = (Kg, N) be a stateful generator with block size k, where k is the length of the key of the base scheme. Let l > 0 be a subkey lifetime parameter. We now specify two particular generators, the parallel and serial ones. We fix a PRF F: {0, 1} k × {0, 1} k → {0, 1} k. (As the notation indicates, we are making the simplifying assumption that the key length, as well as the input and output lengths of each individual function F(K, ·) are all equal to k.) In practice this might be instantiated via a block cipher or via a keyed hash function such as HMAC. (For example, if DES is used, then we set k = 64 and define F(K) to be DES(K).

V. ALGORITHMS

A stateful generator G = (K, N) is a pair of algorithms. The probabilistic key generation algorithm K produces the initial state or seed of the generator.

The probabilistic key generation algorithm K produces the initial state or seed of the generator. The F-based parallel generator PG[F] = (K, N) is defined by:

```
algorithm $\mathcal{K}$
    $K \xleftarrow{R} \{0,1\}^k$
    Return $\langle 0, K \rangle$

algorithm $\mathcal{N}(\langle i, K \rangle)$
    $Out \leftarrow F(K, i)$
    Return $(Out, \langle i+1, K \rangle)$
```

```
algorithm $\mathcal{K}$
    $K \xleftarrow{R} \{0,1\}^k$
    Return $K$

algorithm $\mathcal{N}(K)$
    $Out \leftarrow F(K, 0)$
    $K \leftarrow F(K, 1)$
    Return $(Out, K)$
```

The state has the form hi, Ki where K is the initial seed and i is a counter, initially zero. In the i-th stage, the output block is obtained by applying the K-keyed PRF to the (k-bit binary representation of the integer) i, and the counter is updated. This generator has block length k.

The standard desired attribute of a (stateful) generator is pseudorandomness of the output sequence. We adopt the notion of which formalizes this by asking that the output of the generator on a random seed be computationally indistinguishable from a random string of the same length. Below, we concretize this notion by associating to any generator an advantage function which measures the probability that an adversary can detect deviation in pseudo randomness as a function of the amount of time invested by the adversary.

Let G = (K, N) be a stateful generator with block length k, let n be an integer, and let A be an adversary. Consider the experiments:

```
experiment $\mathbf{Exp}^{\text{prg-real}}_{\mathcal{G},n,A}$
    $St_0 \leftarrow \mathcal{K}$ ; $s \leftarrow \varepsilon$
    for $i = 1, \ldots, n$ do
        $(Out_i, St_i) \leftarrow \mathcal{N}(St_{i-1})$ ; $s \leftarrow s \,\|\, Out_i$
    $g \leftarrow A(s)$
    return $g$

experiment $\mathbf{Exp}^{\text{prg-rand}}_{\mathcal{G},n,A}$
    $s \leftarrow \{0,1\}^{n \cdot k}$
    $g \leftarrow A(s)$
    return $g$
```

Now define the advantage of A and the advantage function of the generator, respectively, as follows:

$$\boxed{\begin{aligned}\mathbf{Adv}^{\text{prg}}_{\mathcal{G},n,A} &= \Pr[\,\mathbf{Exp}^{\text{prg-real}}_{\mathcal{G},n,A} = 1\,] - \Pr[\,\mathbf{Exp}^{\text{prg-rand}}_{\mathcal{G},n,A} = 1\,] \\ \mathbf{Adv}^{\text{prg}}_{\mathcal{G},n}(t) &= \max_{A}\,\{\,\mathbf{Adv}^{\text{prg}}_{\mathcal{G},n,A}\,\}\,,\end{aligned}}$$

Where the maximum is over all A with time-complexity t. If D is a distinguisher having an oracle then:

$$\boxed{\mathrm{Adv}^{\mathrm{prf}}_{F,D} = \Pr[\,D^{F(K,\cdot)} = 1 \;:\; K \overset{R}{\leftarrow} \{0,1\}^k\,] - \Pr[\,D^{f(\cdot)} = 1 \;:\; f \overset{R}{\leftarrow} R^k\,]}$$

is the advantage of D. The advantage function of F is:

$$\boxed{\mathbf{Adv}^{\text{prf}}_{F}(t,q) = \max_{D}\,\{\,\mathbf{Adv}^{\text{prf}}_{F,D}\,\}\,,}$$

Where the maximum is over all A with time-complexity t and making at most q oracle queries. The time-complexity is the execution time of the experiment K R← {0, 1} k ; v ← DF(K,·) plus the size of the code of D, and, in particular, includes the time to compute FK(·) and reply to oracle queries of D. Let us now specify our construction more formally.

The Fig. 1.1 show for a pictorial representation. Let Gl = (Kl , Nl) be a stateful pseudorandom number generator used at level l. Let al be the arity of the tree at level l and kl be the key length of Gl . Then we define our overall stateful pseudorandom number generator G = (K, N), whose key length is k = k1, as follows:

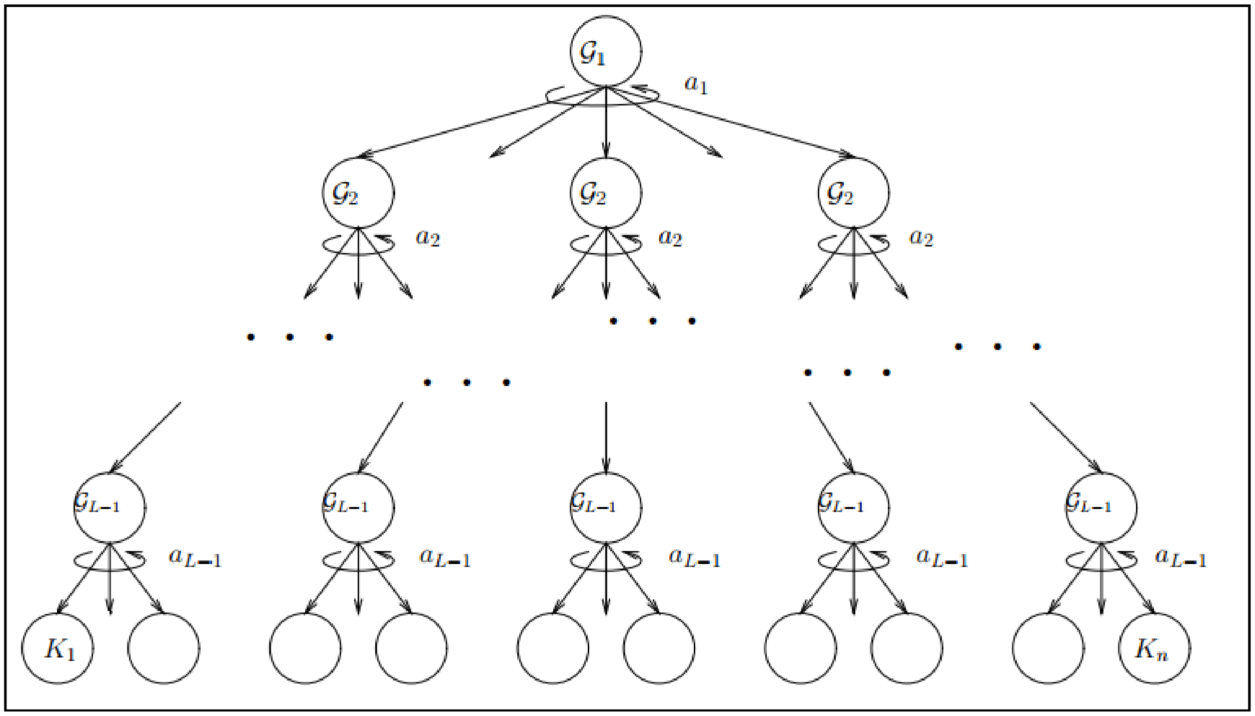

Fig. 1. Diagram of our tree-based construction.

```
algorithm $\mathcal{K}$
    $l \leftarrow 1$
    $St_l \leftarrow \mathcal{K}_l$
    while $l < L - 1$ do
        $(Out_l, St_l) \leftarrow \mathcal{N}_l(St_l)$
        $St_{l+1} \leftarrow \mathcal{K}_{l+1}(Out_l)$
        $l \leftarrow l + 1$
    $St \leftarrow \langle St_1, \ldots, St_{L-1}, 0 \rangle$
    return $St$
```

```
algorithm $\mathcal{N}(St)$
    parse $St$ as $\langle St_1, \ldots, St_{L-1}, i \rangle$
    $l \leftarrow L - 1$ ; $d \leftarrow i + 1$
    while $d \bmod a_l = 0$ do
        $d \leftarrow \lfloor d/a_l \rfloor$ ; $l \leftarrow l - 1$
    $(Out_l, St_l) \leftarrow \mathcal{N}_l(St_l)$
    while $l < L - 1$ do
        $l \leftarrow l + 1$ ; $St_l \leftarrow \mathcal{K}_l(Out_{l-1})$
        $(Out_l, St_l) \leftarrow \mathcal{N}_l(St_l)$
    $St \leftarrow \langle St_1, \ldots, St_{L-1}, i + 1 \rangle$
    return $(Out_{L-1}, St)$
```

VI. ANALYSIS

Our analysis takes a modular approach. Rather than analyzing separately the re-keyed encryption schemes corresponding to different choices of generators, we first analyze the security of a re-keyed encryption scheme with an arbitrary generator, showing how the advantage of the encryption scheme can be bounded in terms of that of the generator and the base scheme. We associate to them a re-keyed encryption scheme SE[SE, G, l] = (K, E, D). The initial state of the encryption scheme includes the initial state of the generator, given by St 0 R← Kg. Encryption is divided into stages i = 1, 2, ... Stage i begins with the generation of a new key Ki using the generator: (Ki , Sti) ← N (Sti−1). In stage i encryption is done using the encryption algorithm of the base scheme with key Ki . An encryption counter is maintained, and when l encryptions have been performed, this stage ends. The encryption counter is then reset, the stage counter is incremented, and the key for the next stage is generated. If the base scheme is stateful, its state is reset whenever the key changes. Our construction of the distinguishing algorithm exploits the fact that collisions in the tree-based construction can be detected at the output. For instance if two output blocks at the same level are the same, then both subtrees rooted at these output blocks will also be the same and this can be easily detected solely by looking at the output string. However, collisions between data blocks at different levels can also be exploited. For example, if there is a collision between a data block at the last level and one at the level before that, we can find that out by using each of the data blocks in the output string as a seed to a generator and checking whether a group of a data blocks matches. The resulting advantage of such distinguisher is O(nL · nL−1)/2k (this is the probability of having a collision between these two levels) but it would require O(a · nL) many queries.
The resulting advantage, though smaller, is still O(nL' nL−1)/2 k while only O(nL) number of queries is now required. Thus:

$$\mathrm{Adv}^{\mathrm{prg}}_{RF^{\star},\mathcal{G}}(n_L, O(n_L)) \geq \frac{c' \cdot n_L \cdot n_{L-1}}{2^k}$$

$$\boxed{\mathbf{Adv}^{\mathrm{prg}}_{RF,\mathcal{G}^{RF^k}}(a,\tau) \leq \frac{\tilde{c}\cdot\tau}{2^k}\ ,}$$

$$\boxed{\mathbf{Adv}^{\mathrm{prg}}_{RF^\star,\mathcal{G}}(n_L, O(n_L)) \geq c \cdot n_{L-1} \cdot \mathbf{Adv}^{\mathrm{prg}}_{RF,\mathcal{G}^{RF^k}}(a, O(n_L))\ ,}$$

For some constant c > 0, which concludes our proof. It is worth to say that an even tighter result can be obtained by making the distinguisher check for collisions between any two levels and within each level. Although this would result in a larger advantage, the order of total number of queries is still kept the same.

VII. CONCLUSION

We saw in that the parallel generator offered greater security than the serial one. We note that this did not materialize in the application to re-keyed CBC encryption: here, the advantage functions arising from re-keying under the two generators are the same. This is because the term corresponding to the security of the base scheme in Corollaries and dominates when the base scheme is CBC. In conclusion we wish to stress two in this paper that security increases are possible, and that our results provide general tools to estimate security in a variety of re-keyed schemes and to choose parameters to minimize the advantage functions of the re-keyed schemes.

REFERENCE

[1]. Ferguson, N., Schneier, B., & Kohno, T. (2013). "*Cryptography Engineering: Design Principles and Practical Applications*". New York: John Wiley & Sons.

[2]. M. Abdalla and M. Bellare(2002). "*Increasing the lifetime of a key: a comparative analysis of the security of re-keying techniques*". In T. Okamoto, editor, Advances in Cryptology – ASIACRYPT 2000, volume **1976** of Lecture Notes in Computer Science, pages 546–559.

[3]. M. Bellare, J. Kilian, and P. Rogaway (1994). *"The security of the cipher blocks chaining message authentication code"*. In Y. Desmedt, editor, Advances in Cryptology – CRYPTO'94, volume **839** of Lecture Notes in Computer Science, Santa Barbara, CA, USA.

[4]. Ford, W., & Baum, M.S. (2001). "*Secure Electronic Commerce: Building the Infrastructure for Digital Signatures and Encryption*", 2nd ed. Englewood Cliffs, NJ: Prentice Hall.

[5]. M. Bellare and P. Rogaway(1997). "*Minimizing the use of random oracles in authenticated encryption schemes*". In Information and Communications Security, volume **1334** of Lecture Notes in Computer Science, pages 1–16.

[6]. Belfield, R. (2007). "*The Six Unsolved Ciphers: Inside the Mysterious Codes That Have Confounded the World's Greatest Cryptographers. Berkeley*", CA: Ulysses Press.

[7]. Stallings, W. (2013). "*Cryptography and Network Security: Principles and Practice, 4th ed.*" Englewood Cliffs, NJ: Prentice Hall.

[8]. Trappe, W., & Washington, L.C. (2012). "*Introduction to Cryptography with Coding Theory, 2nd ed.*" Upper Saddle River, NJ: Pearson Prentice Hall.