ON KEY STORAGE IN SECURE NETWORKS

by

Martin Dyer
School of Computer Studies,
University of Leeds, Leeds, U. K.

and

Trevor Fennor
Department of Computer Science, Birkbeck College
University of London,London, U. K.

and

Alan Frieze

Department of Mathematics, Carnegie Mellon University,
Pittsburgh, PA USA

and

Andrew Thomason
Department of Pure Mathematics and Mathematical Statistics
University of Cambridge, Cambridge, U.K.

Research Report No. 92–142 2

January, 1992

On Key Storage In Secure Networks

Martin Dyer*
School of Computer Studies, University of Leeds,
Leeds, U.K.

Trevor Fenner
Department of Computer Science, Birkbeck College
University of London, London, U.K.

Alan Frieze†
Department of Mathematics, Carnegie-Mellon University,
Pittsburgh, U.S.A.

Andrew Thomason
Department of Pure Mathematics and Mathematical Statistics
University of Cambridge, Cambridge, U.K.

January 17, 1992

Abstract

We consider systems where the keys for encrypting messages are derived from the pairwise intersections of sets of private keys issued to the users. We give improved bounds on the storage requirements of systems of this type for secure communication in a large network.

*Supported by NATO grant RG0088/89
†Supported by NSF grant CCR-8900112 and NATO grant RG0088/89

1 Introduction

The problem of secure communication in a network with multiple users has been considered by Blom [2, 3], Mitchell and Piper [6], Li Gong and Wheeler [5] and other authors. These papers consider a solution in which a *key distribution centre* (KDC) must, in principle, issue a unique cryptographic key to each of the pairs of users in a network whenever they choose to begin communication. Suppose N is the set of users, with $|N| = n$. Then, since there are $\binom{n}{2}$ possible pairs, if n is large a direct implementation will involve the storage of about $\frac{1}{2}n^2$ keys. In modern computer networks, it is in fact very likely that n will be rather large, and current trends seem likely to increase the number of users attached to such networks.

In [6] Mitchell and Piper, inspired by a proposal of Blom [2, 3], considered a scheme in which the KDC stores some global set K of keys ($|K| = k$) and issues to each user i ($i = 1, 2, \ldots, n$) a subset $S_i \subseteq K$ of these keys. We will assume these keys are numbered arbitrarily $1, 2, \ldots, k$. There is a directory, which need not be secure, which lists the numbers of the keys held by each user. Now, if user i wishes to communicate with user j, the key to be used is constructed from the set of keys contained in $S_i \cap S_j$. (Information about which numbered keys users i and j hold can be exchanged without secure communication.) We insist that for no other user, t, is it true that $S_i \cap S_j \subseteq S_t$. The required keys are then clearly available to both users but to no other *single* user. These keys can then be transformed, possibly via some one-way function, into the key to be used to encrypt all communication between users i and j. Mitchell and Piper discuss various schemes, some of which require only $O(n)$ keys in total, as opposed to the $\Omega(n^2)$ required by a direct implementation. Li Gong and Wheeler [5] give a different scheme, requiring $O(n)$ keys in total, with each user required only to hold $O(\sqrt{n})$ keys.

We will call schemes of the type discussed above *set intersection schemes.* In this paper, in Section 2, we will demonstrate the existence of set intersection schemes requiring only $O(\log n)$ keys in total, and hence *a fortiori* each user has to hold only $O(\log n)$ keys. We will show that this is optimal to within constant factors. Constant factors are of course, very important in applications, but we will also show that they are of quite reasonable size in our scheme. Our construction is an application of the *probabilistic method* in combinatorics [7]. This is a simple yet robust method for obtaining combinatorial constructions, which perhaps deserves to be more widely known in the cryptography community. In Section 3 we give some experimental evaluations of our proposals to demonstrate their practical feasibility. To counter possible objections to the use of random methods, we will further show, in Section 4, that if necessary, the construction can be "derandomized" by the *method of conditional probabilities* [7].

A known difficulty with set intersection schemes [6, 5] is the problem of *collusion.* If all members of some group $W \subseteq N$ $(|W| = w)$ choose to disclose their keys to one of their number, this user may then possess the subset of keys which two others, not in the group, are using to communicate, i.e. for some i, j and $W \subseteq N$ we have

$$S_i \cap S_j \subseteq \bigcup_{t \in W} S_t.$$

This clearly compromises the network for secure communication between the users i and j. We will examine this problem in section 5 and give upper and lower bounds on the number of keys required to ensure network security against groups of at most w colluders, for any given w. In section 6 we consider a generalization of the problem to communication between groups of (more than two) users, and in Section 7 we make some concluding remarks.

2 A space efficient key storage system

We will use $\lg n$ for $\log_2 n$. Our first simple result also appears, in a different form, in [6].

Lemma 1 *Any set intersection scheme requires each user to hold at least* $\lg n$ *keys, and at least* $2\lg n$ *keys in total* ($n \geq 4$).

Proof Any user who has less than $\lg n$ keys can form less than $n-1$ distinct nonempty subsets from their keys. Thus they cannot have a different intersection with each of the other $n-1$ users. There are $\binom{n}{2}$ pairs of users, and their key intersections must form a set antichain. It follows that we must have

$$\binom{k}{\lfloor k/2 \rfloor} \geq \binom{n}{2}.$$

This implies $k \geq 2\lg n$ for all $n \geq 4$. □

We now show

Theorem 1 *There exists a set intersection requiring only* $\lceil 13\lg n \rceil$ *keys in total.*

Proof Suppose the sets are generated randomly in some way. Let X_{is} be the random variable which is the indicator of the event $s \in S_i$. A "bad triple" is a triple (i, j, t) such that $S_i \cap S_j \subseteq S_t$. Thus the expected number of bad triples is

$$\mathbf{E}\left(\sum_{t=1}^{n}\sum_{i=1}^{n}\sum_{j=1}^{i-1}\prod_{s=1}^{k}(1 - X_{si}X_{sj}(1 - X_{st}))\right). \tag{1}$$

This is an upper bound on the probability of the existence of any bad triple. Thus, if the X_{is} are independent Bernoulli variables with $\Pr(X_{is} = 1) = p$, then

$$\Pr(\text{Any bad triple}) \leq n\binom{n}{2}(1 - p^2(1-p))^k \leq \tfrac{1}{2}n^3(1 - p^2(1-p))^k. \tag{2}$$

This is minimized by choosing $p = 2/3$. Then if $k = 13\lg n$, it is easy to check that

$$\Pr(\text{Any bad triple}) \leq \tfrac{1}{2}n^3(23/27)^{13\lg n} \leq \tfrac{1}{2}n^{-.007} < \tfrac{1}{2}. \tag{3}$$

Thus if we generate the sets randomly in this manner, we have a probability of at least $\frac{1}{2}$ that the scheme we generate will be "good". □

Observe that we may *check* that the generated scheme has the required properties. This can be done by an $O(n^3k)$ computation, by checking all triples to ensure that they are not bad. This is feasible for small n, and could easily be implemented in parallel. In fact, this checking problem is easily seen to be in the class NC of efficiently parallelizable problems. It may be observed that the whole scheme could be generated and checked by a randomized parallel computation. Thus generating and checking is in the parallel complexity class RNC.

3 Practical schemes

From a practical point of view, the moral of the previous section is that set intersection schemes are best generated at random. For example, it is instructive to compare the bound of Theorem 1 with those of [5]. The matrix scheme of [5] requires n keys. Now $13\lg n < n$ for all $n \geq 83$. If $n = 1,000$ say, we have $\lceil 13\lg n \rceil = 130$, less than one-seventh of the key requirement. The total number of keys stored by all users is at least $3n^{3/2}$ for the method of [5], whereas ours is approximately $\frac{26}{3}n\lg n$. This is smaller for $n \geq 800$, say. Lest this should appear a disadvantage of our method, it should be observed that these comparisons are not entirely fair to our scheme, since those of [5] are not proven to be free of bad triples. Also, we have based our bound for of k on estimates of probabilities which may be pessimistic for small n.

In practice, we can in fact do rather better than Theorem 1 would imply. We chose a very simple random model to make the theoretical analysis easy; in practice it is more efficient to choose other models. In particular, rather than select, for each user, a random subset of the k keys by selecting each with probability p, it is better to give each user a random subset of size t, for some fixed t. The best value of t to choose would seem to be that which minimises the probability that a randomly chosen triple is "bad", in the sense of the proof of Theorem 1. This probability is

$$p_{k,t} = \sum_{u=0}^{t} \binom{t}{u} \binom{k-t}{t-u} \binom{k-u}{t-u} \binom{k}{t}^{-2},$$

u being the size of the intersection $S_i \cap S_j$. The value of $p_{k,t}$ is usually less than $(23/27)^k$, and can be substituted for $(23/27)^k$ in (3) to demonstrate the existence of a scheme for n users with fewer keys than required by Theorem 1. For small values of n the gain is striking. For 100 users, Theorem 1 (or its proof) shows the existence of a scheme with 86 keys, the number of keys rising to 116 for 500 users, 129 for 1000 users and 258 for a million users. This is already dramatically less than [5], but the method of choosing t-sets allows these figures to be reduced to 56 keys, 75 keys, 83 keys and 165 keys respectively. Table 1 gives further data; in this table the column labelled "Thm. 1" gives the upper bound on the number of keys required by the proof of Theorem 1, and the column labelled "t-sets" gives the improved upper bound given by the method of choosing t-sets.

Table 1: The existence of set intersection schemes

	Number of keys required			
Users	Thm. 1	t-sets	risk 10^{-10}	risk 10^{-20}
50	73	48		
75	80	53		
100	86	56		
200	99	64	152	
500	116	75	163	254
10^3	129	83	171	262
10^4	172	111	199	290
10^5	215	138	226	317
10^6	258	165	253	344
10^8	344	220	308	399

The method is genuinely practical. For 500 users the schemes generated can be tested on a SUN SPARCstation 1 in two minutes, rising to 16 minutes for 1000 users, with an eightfold increase in time for every doubling of the number of users. For much larger numbers of users, each user could be asked to check his own keys, thus performing the checking computation in a "distributed" manner; with one hundred thousand users a user can check his own keys in about five hours (sequential) time on the SPARC.

If an astronomically large number of users is envisaged, a value of k can be chosen to ensure the probability of an incorrect scheme being generated is less than, say, 10^{-20}. For 100 million users, $k = 399$ would suffice. This is verified by checking that, with $n = 10^8$ and $k = 399$, $\frac{1}{2}n^3 p_{k,t} < 10^{-20}$ holds. Values for k ensuring that a randomly chosen collection of t-sets will give a set intersection scheme for n users with probability at least $1 - 10^{-20}$ are given in the "risk 10^{-20}" column of Table 1; the "risk 10^{-10}" column has an analogous meaning.

An efficient way to generate schemes in practice is to select a suitable value of k, compute the best value of t as above, and then randomly select, one by one, subsets of size t, stopping when one of them forms a bad triple with two previous choices. In this way quite large schemes can be found with relatively few keys. Sometimes a bad choice is made early on; the method can be made less susceptible to such bad luck by allowing another random choice after a failed choice, and another choice again if necessary, up to a specified limit on the number of successive failures before stopping. This method, which we call the "retry" method, has been tried in practice and the results are summarised in Table 2. For each value of k (in the "keys" column), and number of failures allowed (in the "fails" column), ten attempts were made to generate schemes by this method. The table shows the minimum, mean and maximum number of users in the final schemes. Note that the number of keys needed for a given number of users is dramatically less even than that given by Table 1; for instance a scheme for 3000 users with only 70 keys can

Table 2: Set intersection schemes found by the 'retry' method

		Number of users		
keys	fails	min	mean	max
40	0	14	33	69
40	1	49	68	89
40	2	84	97	111
40	5	106	137	153
50	0	41	84	155
50	1	115	168	237
50	2	190	248	303
50	5	314	414	489
60	0	92	141	201
60	1	201	470	637
60	2	307	711	962
60	5	1111	1260	1387
70	0	186	392	503
70	1	663	1292	1732
70	2	1481	2141	2571
70	5	3236	3787	4408

be found in just minutes of computer time.

4 Derandomization

While in practice, we believe that an efficient randomized scheme is always practically acceptable (see Section 7 below), from a theoretical point of view we might wish to dispense with the requirement for randomization. This can easily be achieved using the method of conditional probabilities [7].

Theorem 2 *The set intersection scheme of Theorem 1 can be generated by an $O(n^3k)$ time deterministic algorithm using $O(n^3)$ storage.*

Proof Suppose we construct the sets by fixing one X_{is} at a time in major order $i = 1, 2, \ldots, n$ and minor order $s = 1, 2, \ldots, k$. We ensure that initially the probability of a bad triple, as given by (2), is less than 1. When we come to X_{is} the "previous" X's will have been fixed (at 0 or 1) in (1). The following "unknown" X's will still be Bernoulli random variables with probability p ($= 2/3$ here). Assume the value of (1) at this stage is $E < 1$. We now, try in turn, both cases for $X_{is} = 0, 1$ in (2) and evaluate the expected value. The reader may check that this can be accomplished by an $O(n^2)$ computation if E and the $O(n^3)$ products of which it is the sum are stored and updated whenever an X_{is} "changes" from p to 0 or 1. Let the two expected values calculated in this way be E_0, E_1. Then clearly

$$E = (1-p)E_0 + pE_1.$$

Thus if $E < 1$, then either $E_0 < 1$ or $E_1 < 1$. We fix $X_{is} = 0$ if $E_0 < 1$, $X_{is} = 1$ if $E_1 < 1$, and proceed to the next variable. (If both $E_0 < 1$ and $E_1 < 1$, we may choose arbitrarily.) Obviously we must terminate with $E = 0$, and hence we will have constructed the desired scheme. The algorithm takes $nk \times O(n^2) = O(n^3k)$ time. □

It may be observed that this is only of the same complexity as checking the randomized construction. However, there is an important difference from a practical point of view. The derandomized algorithm does not appear to be efficiently parallelizable.

5 The problem of collusion

We consider here the collusion problem mentioned in Section 1. Let us first establish a lower bound on k in this context.

Theorem 3 *Any set intersection scheme which is secure against w colluders must have at least $w(2\lg n - \lg w - 1)$ keys.*

Proof Let A, B be any two communicating sets, and $C_1, C_2, \ldots, C_w$ be any possible w colluding sets. Then we must have

$$A \cap B \not\subseteq C_1 \cup C_2 \cup \cdots \cup C_w. \tag{4}$$

Suppose we take all $N = \binom{n}{2}$ possible intersections pairs of sets $S_i \cap S_j$. Then all $\binom{N}{w}$ unions of these intersections must form a set antichain. Otherwise for some sets

$$(A_1 \cap B_1) \cup \cdots \cup (A_w \cap B_w) \subseteq (C_1 \cap D_1) \cup \cdots \cup (C_w \cap D_w).$$

But this contradicts (4). Note that some of the pairs on the left could be the same as some on the right, and there is no contradiction for such pairs. But for *some* i, the pair A_i, B_i is not one of the pairs on the right side. For this pair we may select, from each pair C_j, D_j, one which is neither A_i nor B_i. This set of choices allows us to construct the contradiction of (4) required. Thus we must have

$$\binom{k}{\lfloor k/2 \rfloor} \geq \binom{N}{w}.$$

The bound follows from this by easy manipulations. □

Thus we must have k growing at least linearly with the number of potential colluders. However, it seems difficult to achieve this. By a randomized contruction we can achieve

Theorem 4 *There exists a set intersection scheme which is secure against w colluders and has* $\left\lceil \frac{(w+2)^{w+3}}{4w^w} \ln n \right\rceil$ *keys.*

Proof Suppose we choose the sets randomly as before with probabilities p. In the notation of the proof of Theorem 3, the probability of a successful collusion by fixed sets $C_1, \ldots, C_w$ against A, B is

$$\Pr(A \cap B \subseteq C_1 \cup \cdots \cup C_w) = (1 - p^2(1-p)^w)^k.$$

This is minimized by putting $p = 2/(w+2)$, giving

$$\begin{aligned} \Pr(A \cap B \subseteq C_1 \cup \cdots \cup C_w) &= \left(1 - \frac{4w^w}{(w+2)^{w+2}}\right)^k \\ &\leq \exp\left(-\frac{4kw^w}{(w+2)^2}\right) \end{aligned}$$

Thus the expected number of successful collusions is less than

$$\frac{1}{2w!} n^{w+2} \exp(-4kw^w/(w+2)^{w+2}) \leq \frac{1}{2w!}$$

if $k \geq \frac{(w+2)^{w+3}}{4w^w} \ln n$. □

Corollary 1 *There exists a set intersection scheme which is secure against w colluders and has less than* $\lceil 2(w+2)^3 \ln n \rceil$ *keys.*

Proof

$$\frac{(w+2)^{w+3}}{4w^w} < \frac{e^2(w+2)^3}{4} < 2(w+2)^3.$$

□

The method of Theorem 4 can be derandomized as in Section 4.

From Theorem 4 we can obtain a scheme guaranteed secure against any five colluders, in a network of 4,000 users, with 3,826 keys. Each user would have about 1,093 keys on average. To be guaranteed secure for large n against five colluders, the scheme of [5] requires n keys, with each user holding at least $18\sqrt{n}$ keys. For $n = 4,000$ this would be 4,000 keys in total, with each user holding at least 1,139. Moreover, it is not clear that for $n = 4,000$ the scheme would in fact be secure against five colluders. For $n = 100,000$ the corresponding figures would be 5,310 keys, with each user holding an average of 1,517 keys for the scheme here, and 100,000 keys, with each user holding at least 5,693 for the scheme of [5].

There is one drawback of the randomized construction here (and of its derandomization). If we wish to check that the scheme meets its security specification, the computational burden will grow like $\Omega(n^{w+2}k)$. This would clearly be infeasible for large n, if w is not very small. From a practical point of view, the answer is to make the probabilities sufficiently small. However we can show that, at the expense of more keys, we can construct schemes which are checkable (or derandomizable) in time little more than required by the basic scheme of Section 2.

Theorem 5 *There exists a set intersection scheme, secure against w colluders, requiring less than $\lceil 500w^3 \ln n \rceil$ keys in total, which can be checked, or constructed deterministically, in $O(n^3k)$ time.*

Proof Choose n sets from $k = 500w^3 \ln n$ with probability $p = 1/(5w)$. If A, B are the communicating sets, it is sufficient to ensure that, for any set C,

$$|C \cap A \cap B| < |A \cap B|/w. \tag{5}$$

Clearly this implies (4) for any collection $C_1, \dots, C_w$ $(i = 1, \dots, w)$. We bound the probability of (5) as follows, using a version of Chernoff's bound for the tails of the binomial distribution. (See, for example, [1].)

$$\begin{aligned}\Pr(|A \cap B| \le 0.68p^2k) &\le \exp(-\tfrac{1}{2}(0.32)^2(0.2)^2 500w \ln n) \\ &\le n^{-2.048} \quad (w \ge 2).\end{aligned}$$

Hence,

$$\Pr(\text{Any such } A, B \text{ exist}) \le \binom{n}{2} n^{-2.048} < \tfrac{1}{2}. \tag{6}$$

Now, for fixed C,

$$\begin{aligned}\Pr(|A \cap B \cap C| \ge 0.68p^2k/w) &= \Pr(|A \cap B \cap C| \ge 3.4p^3k) \\ &\le \left(\frac{e}{3.4}\right)^{3.4p^3k} \\ &< n^{-20}.\end{aligned} \tag{7}$$

(For a proof of (7) see e.g. Bollobás [4], Theorem I.7(ii).)

Hence,

$$\Pr(\text{Any such } A, B, C \text{ exist}) \le n\binom{n}{2} n^{-20} < \tfrac{1}{2}. \tag{8}$$

From (6) and (8) it follows that the scheme exists with nonzero probability. The checking or derandomization of the scheme involves establishing the conditions (5) for all triples. This can be achieved by a similar method to that discussed in Theorem 2.

More practical versions of the schemes suggested here could be developed, as in section 3, but we will not do so here. □

6 Groups of users

Mitchell and Piper [6] also consider a generalization of the problem considered here, where every subset of users of size at most g, for some given g, needs to have a key known only to the members of the group. The key is constructed by using intersections of sets taken g at a time, rather than in pairs as we have done so far. Let us call such schemes g-intersection schemes. (Thus a set intersection scheme is a 2-intersection scheme.) Our methods readily generalize to this situation. Thus we may prove a lower bound analogous to Theorem 3.

Theorem 6 *Any g-intersection scheme which is secure against w colluders must have $k \geq w(g \lg n - \lg w - g \lg g)$.*

Proof We use the same method as for Theorem 3, but consider all intersections of sets taken g at a time, and let $N = \binom{n}{g}$. To avoid cumbersome notation, let us introduce some temporary terminology. The word *block* will mean a set of keys assigned to one of the n users, and a *group* an intersection of some g blocks. We use the term *collusion* to mean a set of w groups. Now the $\binom{N}{w}$ sets, formed by taking the unions of groups in each possible collusion, are an antichain. Otherwise, the union of some collusion is contained in some other. To show that this implies a contradiction to the security assumption, suppose the union of collusion A is contained in that of collusion B. Now some group in A clearly does appear in B. Let the blocks forming this group be $C_1, \ldots, C_g$, the group being $\bigcap_{i=1}^{g} C_i$. Now, for each group in B, we can select a block which is not one of the C_i. The union of these w blocks now contains the intersection of the C_i, giving the contradiction. Thus we have

$$\binom{k}{\lfloor k \rfloor} \geq \binom{N}{w},$$

which implies $2^k \geq (N/w)^w \geq ((n/g)^g/w)^w$, giving the bound. □

Corresponding to Theorem 4 we have

Theorem 7 *There exists a g-intersection scheme which is secure against w colluders and has $\left\lceil \frac{(w+g)^{w+g+1}}{g^g w^w} \ln n \right\rceil$ keys.*

Proof The proof is similar to that of Theorem 4. Selecting the sets randomly with probability p, now the probability of successful collusion is

$$\Pr(A_1 \cdots \cap A_g \subseteq C_1 \cup \cdots \cup C_w) = (1 - p^g(1-p)^w)^k.$$

This is minimized by putting $p = g/(w+g)$. Then

$$\begin{aligned}(1 - p^g(1-p)^w)^k &= \left(1 - \frac{g^g w^w}{(w+g)^{w+g}}\right)^k \\ &\leq \exp\left(-\frac{kg^g w^w}{(w+2)^2}\right)\end{aligned}$$

The expected number of successful collusions is at most

$$\frac{1}{g!w!} n^{w+g} \exp(-kg^g w^w/(w+g)^{w+g}) \leq \frac{1}{g!w!}$$

if $k \geq \frac{(w+g)^{w+g+1}}{g^g w^w} \ln n.$ □

Again we can derandomize the construction if required.

' Concluding remarks

'he basic scheme we propose is within a small factor (i.e. less than 6.5 mes) of optimal. In practice, using the method of section 3, we can do ,en better than this. It would be exceedingly difficult to find deterministic ›mbinatorial constructions having the required property which are as small. owever, collusion-proof schemes seem much harder to produce. Note that ıe order of growth in w of the upper bound of Corollary 1 is $\Omega(w^3)$, as ɔposed to $\Omega(w)$ in the lower bound of Theorem 3. Establishing the correct :der of growth remains an open question. This is a crucial question since ıe bound of Theorem 4 is rather large if w is appreciable (and the bound of heorem 5 is larger). When we consider groups, as in Section 6, the situation even worse.

possible criticism of our scheme, by comparison with those based on block ɜsigns [6], might be that the set intersections will vary in size. This is not :ally a practical difficulty however, since if required we can always "pad ıt" all intersections to a standard length using "dummy" keys known to all ɜers.

'e have given methods for checking and derandomization, but these are early practicable only for relatively small n, unless massive parallelism is 'ailable. In practice, the schemes we propose would be generated randomly n is large, ensuring that the failure probabilities remain very small. This ay raise questions about the practical production of large numbers of ran- ›m bits. However, the same question could be raised for the myriad other ıccessful applications of random numbers It is usual practice to employ a ıitably guaranteed pseudo-random source, and at present this seems the path one must follow. However, pseudo-randomness seems to be an unnecessary device when (at least according to the quantum theory) nature itself is subject to continual and unavoidable random fluctuation. Randomized algorithms and constructions are becoming widely used, in addition to the already widespread use of random simulations. Consequently, there seems a clear need for the development of cheap physical devices for generating truly random bits, rather than cleverer ways of generating pseudo-random bits. As a modest example, such devices would allow the schemes proposed here to be generated simply and conveniently, and easily extended whenever desired.

References

[1] D. Angluin and L. G. Valiant, "Fast probabilistic algorithms for Hamiltonian circuits and matchings", *Journal of Computer and System Sciences* **18** (1979), 155–193.

[2] R. Blom, "Non-public key distribution", in *Advances in Cryptology: Proceedings of Eurocrypt 82*, Plenum Press, New York, 1983, pp 231–236.

[3] R. Blom, "An optimal class of symmetric key generation systems", in *Advances in Cryptology: Proceedings of Eurocrypt 84*, Lecture Notes in Computer Science 209, Springer-Verlag, Berlin, 1984, pp 335–338.

[4] B.Bollobás, "Random Graphs", Academic press, London 1985.

[5] Li Gong and D. H. Wheeler, "A matrix key storage scheme", *Journal of Cryptology* **2** (1990), 51–59.

[6] C. J. Mitchell and F. C. Piper, "Key storage in secure networks", *Discrete Applied Mathematics* **21** (1988), 215–228.

[7] J. Spencer, *Ten lectures on the probabilistic method*, SIAM, Philadelphia, 1987.