VOLUME 2, NUMBER 2 — SUMMER 1996

RSA LABORATORIES'

CryptoBytes

The technical newsletter of RSA Laboratories, a division of RSA Data Security, Inc.

Contents

The Status of MD5 After a Recent Attack

Hans Dobbertin
German Information Security Agency
P.O. Box 20 03 63
D-53133 Bonn, Germany

Hash functions are frequently used cryptographic primitives. In digital signature schemes a message is hashed before signing. To prevent that by this interposition a weakness is generated, the applied hash function has to be collision-resistant. Hash functions also occur as components in various other cryptographic applications with usually much weaker requirements.

The hash function MD4 was introduced by Ron Rivest [15] in 1990. This was the starting point for the development of a series of similar hash functions and for intensive effort in cryptanalyzing these MD4-like hash functions. MD5, the strengthened successor of MD4 introduced by Rivest [16] in 1991, is today the most frequently implemented hash function. Other MD4-derived hash functions are

- the 256-bit extension of MD4 (Rivest [14]),
- the Secure Hash Algorithm (SHA) designed by NIST/NSA, later substituted by the slightly revised SHA-1 [1],
- HAVAL developed by Zheng, Pieprzyk, and Seberry [18],
- RIPEMD designed in the framework of the European project RIPE [2],
- strengthened versions RIPEMD-160 and RIPEMD-128 of RIPEMD, recently published by Bosselaers, Preneel, and the author [12].

Before 1995 collisions for two (of three) rounds of MD4 (den Boer and Bosselaers [5]), almost-collisions of MD4 (Vaudenay [17]), and pseudo-collisions for the compression function of MD5 (den Boer and Bosselaers [6]) had been found. In 1995 the author developed new methods to cryptanalyze MD4-like hash functions. In a series of attacks these techniques were applied on the first two and the last two (of three) rounds of RIPEMD, on MD4 in its entirety and the compression function of the 256-bit extension of MD4 (see [8-10]).

Very recently collisions for the compression function of MD5 have been found as announced at the Eurocrypt '96 rump session. The 1995 autumn issue of *CryptoBytes* [11] already reports on the breaking of MD4. In the present article we shall report on the recent attack on MD5 and discuss its implications.

Designing Hash Functions

The design of hash functions is certainly one of the most difficult goals in cryptography. A hash function maps arbitrary-length messages to fixed-length hash values, which must be fast to compute. But on the other hand a hash function must be collision-resistant, i.e. it must be computationally infeasible to find a collision (that is a pair of different messages with the same hash value), while the *existence* of collisions is of course unavoidable. Note that there are two further well-known requirements for

(continued on page 3)

Professor Hans Dobbertin develops and evaluates cryptographic algorithms. His main research interests are applications of discrete algebra in cryptography. He can be contacted at dobbertin@skom.rhein.de.

Editor's Note

In recent months, the state of hash function analysis has been revolutionized. In a sequence of adaptations and enhancements to a remarkably effective style of cryptanalysis, Hans Dobbertin has made significant advances in the analysis of a variety of hash functions, including the popular MD5.

We encourage any readers with comments, opposite opinions, suggestions or proposals for future issues to contact the CryptoBytes editor.

The work of Dobbertin has had two effects. First there is the direct effect of the cryptanalytic work which gives us a much improved assessment of the security offered by different hash functions. But the second effect has been to force us to look more closely at exactly what it is we might want of a hash function in a particular application. There has been some confusion and perhaps a little misrepresentation about exactly where things stand with regards to Dobbertin's latest work and its implications both for hash functions in general and MD5 in particular. As a consequence we are delighted to lead this issue of *CryptoBytes* with an assessment of the current state of MD5 by Hans Dobbertin who in addition, gives us some perspectives on possible future developments.

In contrast to developments in hash function analysis, there have been few, if any, recent advances in the analysis of DES. Despite this, it is well known that "DES is nearing the end of its useful lifetime" and a computationally cheap way to build on the strengths of DES will always be attractive. Such a mechanism, originally proposed by Ron Rivest, has actually been available for some time and it is called *DESX*. In this issue, Phil Rogaway describes an analysis of DESX and its resistance to exhaustive key-search and shows that in some sense "DESX works." That is, even though DESX is a computationally simple and inexpensive construction, it can be shown to offer a substantial cryptographic advantage over the cipher DES around which it is built.

Our final article in this issue is perhaps a departure from our more usual articles, but it offers the reader a glimpse at one of the more substantial and important cryptographic events of 1996. The Research Program at the Newton Institute at the University of Cambridge, U.K. offered researchers from both academia and industry, including RSA Laboratories' Chief Scientist Burt Kaliski, the opportunity to spend up to six months in a research-conducive atmosphere that included workshops, conferences and the potential for numerous new results and developments. Tom Berson, who was present for four of the six months, offers us his perspective on the pleasures of an extended research trip to Cambridge.

The future success of *CryptoBytes* depends on input from all sectors of the cryptographic community, and as usual we would very much like to thank the writers who have contributed to this second issue of the second volume. We encourage any readers with comments, opposite opinions, suggestions or proposals for future issues to contact the *CryptoBytes* editor at RSA Laboratories or by E-mail to *bytes-ed@rsa.com*.

Newsletter Availability and Contact Information

CryptoBytes is a free publication and all issues, both current and past, are available via the World-Wide Web at *http://www.rsa.com/rsalabs/cryptobytes/*.

For each issue a limited number of copies are printed. They are distributed at major conferences and through direct mailing. While available, additional copies of the newsletter can be requested by contacting RSA Laboratories though a nominal fee to cover handling costs might be charged for individual requests.

RSA Laboratories can be contacted at:

RSA Laboratories
100 Marine Parkway, Suite 500
Redwood City, CA 94065
415/595-7703
415/595-4126 (fax)
rsa-labs@rsa.com

About RSA Laboratories

An academic environment within a commercial organization, RSA Laboratories is the research and consulting division of RSA Data Security, Inc., the company founded by the inventors of the RSA public-key cryptosystem. Through its research program, publications, seminars and consulting services, RSA Laboratories provides state-of-the-art cryptographic expertise for the benefit of RSA Data Security and its customers.

The Status of MD5 After a Recent Attack

Continued from page 1

hash functions, the infeasibility of finding preimages (being one-way) and the infeasibility of finding a second preimage to a given one (weak collision-resistance).

There is a good chance that among the hash values of $2^{n/2}$ different messages a collision occurs. To prevent an attacker from finding collisions simply by computing that many hash values (birthday attack) n should be large enough. While $n = 128$ is still considered to be sufficient today, new results of van Oorschot and Wiener [13] suggest that we could be forced to move to the minimum $n = 160$, the next multiple of 32, sooner than expected previously.

It is an open problem whether collision-resistant hash functions, in an idealized strict sense, can exist at all. There is not much hope that this gap in our knowledge will be filled in the near future, since this would require a dramatic break-through in complexity theory. The construction of hash functions for practical applications has to be approached by combining theoretical investigations with pragmatic methods.

Hashing by Iterated Compression

The hash functions of the MD4-family follow a design principle due to Merkle and Damgård [7]. The basic idea is that hashing, like encryption, should be done blockwise. The Merkle-Damgård principle defines how a hash function h with n-bit hash values can be built up from a *compression function* f. It is assumed that the computation of f is initialized by n-bit vectors, that f compresses input blocks of fixed size, say r bits, and returns n-bit outputs. (For instance, we have $n = 128$ and $r = 512$ for MD4 and MD5.) Suppose that, after appropiate padding (which adds the length of the message), a given message M is split into a sequence of blocks of length r:

$$M = M[1]\, M[2] \dots M[s].$$

The hashing process is initialized with some fixed n-bit initial value IV^*, which is a part of the specification of the hash algorithm. The hash value of M is then computed by an iterative application of f, where the M[i]'s are taken as inputs and each output of f is the inital value for the next application of f:

$$H(0) := IV^*,$$
$$H(i) := f(H(i\text{-}1);\, M[i]), \quad i = 1,\dots,s.$$

The last output of the compression function f is defined to be the hash value of M, i.e.

$$h(M) := H(s).$$

Collisions and Pseudo-Collisions

Suppose an iterated hash function h based on a compression function f is given. A *collision of the compression function* consists of an initial value IV and two different inputs X and $\tilde{X}$ such that

$$f(IV; X) = f(IV; \tilde{X}).$$

It is an important observation that an attack leading to collisions of the compression function is already very close to collisions for the hash function. What remains is to extend the attack in a way that it is possible to prescribe the initial value above as the initial value IV^* of the hash algorithm. In fact, in this case we would have

$$\begin{aligned} h(X) &= f(f(IV^*; X);\, P) \\ &= f(f(IV^*; \tilde{X});\, P) \\ &= h(\tilde{X}). \end{aligned}$$

Here P is the block which has to be appended according to a padding rule like that used for all MD4-like hash functions. This block is identical for both messages X and $\tilde{X}$, since they have the same length.

Loosely speaking collisions of the compression function are collisions of the hash function with a wrong initial value. On the other hand we use the term *pseudo-collision of the compression function* if two different initial values IV, $\tilde{IV}$ and (possibly identical) inputs X, $\tilde{X}$ are given such that

$$f(IV; X) = f(\tilde{IV};\, \tilde{X}).$$

Den Boer and Bosselaers [6] found pseudo-collisions for the compression function of MD5 with $X = \tilde{X}$. (This result implies that the theorem of Merkle-Damgård, which derives the security of a hash function from its underlying compression function, cannot be invoked for MD5.) The finding of pseudo-collisions shows that the compression function, considered as a cryptographic primitive of its own, has a weakness. But their finding alone does not lead us closer to collisions of the hash function.

The construction of hash functions for practical applications has to be approached by combining theoretical investigations with pragmatic methods.

Loosely speaking collisions of the compression function are collisions of the hash function with a wrong initial value.

Compression Function of MD5

The single steps in the compression process of MD5, as of the other hash functions of the MD4-family, are based on the following word operations, where a *word* is a 32-bit quantity:

- bitwise Boolean operations,
- addition modulo 2^{32},
- cyclic shifts.

step	round 1	step	round 3
1	X[0]	33	X[5]
2	X[1]	34	X[8]
3	X[2]	35	X[11]
4	X[3]	36	X[14]
5	X[4]	37	X[1]
6	X[5]	38	X[4]
7	X[6]	39	X[7]
8	X[7]	40	X[10]
9	X[8]	41	X[13]
10	X[9]	42	X[0]
11	X[10]	43	X[3]
12	X[11]	44	X[6]
13	X[12]	45	X[9]
14	X[13]	46	X[12]
15	X[14]	47	X[15]
16	X[15]	48	X[2]

step	round 2	step	round 4
17	X[1]	49	X[0]
18	X[6]	50	X[7]
19	X[11]	51	X[14]
20	X[0]	52	X[5]
21	X[5]	53	X[12]
22	X[10]	54	X[3]
23	X[15]	55	X[10]
24	X[4]	56	X[1]
25	X[9]	57	X[8]
26	X[14]	58	X[15]
27	X[3]	59	X[6]
28	X[8]	60	X[13]
29	X[13]	61	X[4]
30	X[2]	62	X[11]
31	X[7]	63	X[2]
32	X[12]	64	X[9]

Table 1. *The message words X[i] are used in a different order in each round.*

These operations have been chosen, since they can be computed very fast on 32-bit processors and since the mixing of Boolean functions and addition is believed to be cryptographically strong. In the sequel "+" denotes the addition modulo 2^{32} and "$\ll s$" denotes a left circular shift by s positions.

We take a closer look at the MD5 compression function to illustrate its internal structure. The 512-bit input X and 128-bit initial value IV are split into words. The compression process operates on four word registers (chaining registers) A, B, C, D, which are initialized with IV. The compression algorithm has four rounds. A round consists of 16 steps. In each round all input words $X[i]$ ($i<16$) are applied in a different order, as shown in Table 1.

In every step, one of the chaining registers is updated. A typical step operation of the compression in MD5 is

$$A := B + ((A + \Phi(B,C,D) + X[i] + K) \ll s),$$

where Φ is a round-dependent Boolean function (e.g. XOR), K is a step-dependent constant and the rotation amount s is also step-dependent. After processing the four rounds, the compression value is obtained by wordwise addition (modulo 2^{32}) of IV to the chaining registers.

The Attack

Our approach to find collisions of the compression function of MD5 is rather complicated and technical, and here we can only give a rough idea about how it works. It can be described as a taming of the avalanche effect by restriction to collisions of a special form.

First we assume for the message pair $X, \tilde{X}$ that all words coincide, with one exception, say $X[i_0]$ and $\tilde{X}[i_0]$. We choose some word Δ with small Hamming weight and assume that

$$\tilde{X}[i_0] = X[i_0] + \Delta.$$

For the present attack we have $i_0 = 14$, but other choices might be possible. Table 1 shows that $X[14]$ and $\tilde{X}[14]$ are applied in

- step 15 (round 1),
- step 26 (round 2),
- step 36 (round 3),
- step 51 (round 4).

We focus our attention to the segment from the first to the second application and the segment from the third to the fourth application:

(i) steps 15 — 26 (rounds 1/2),
(ii) steps 36 — 51 (rounds 3/4).

The computations of the compression value of X and $\tilde{X}$ run in parallel up to step 14. The first difference occurs at the beginning of segment (i), when $X[14]$ and $\tilde{X}[14]$ are applied in step 15. If we can achieve for segment (i) that in the end, i.e. step 26, where $X[14]$ resp. $\tilde{X}[14]$ are applied the second time, the contents of the chaining registers coincide, then we speak of an *inner collision* of segment (i). In this case everything runs again in parallel from step 27 to 35, since here $X[14]$ and $\tilde{X}[14]$ are not applied. Then we need another inner collision for segment (ii) in order to end with the same compression value for both inputs X and $\tilde{X}$.

Thus our attack consists of three parts:

(1) finding an inner collision for rounds 1/2,
(2) finding an inner collision for rounds 3/4,
(3) connecting these two inner collisions.

To explain at least the basic approach to parts (1) and (2) we note that, based on the definition of the step operations, the occurrence of an inner collision can be expressed as a system of equations, where the contents of the chaining registers are considered as unknowns. The involved input words $X[i]$ can be obtained as functions of the contents of the chaining registers. Thus finding an inner collision is reduced to solving this equational system. After a suitable "specialization", solutions of this system can be found by an iterative solving of equations of the form

$$\Phi(a_1,b_1,z) + k = \Phi(a_2,b_2,z + \delta z),$$

where z is the unknown, Φ is a Boolean function coming from a step operation, and the words a_1, b_1, a_2, b_2, k and δz are given. The collection of all solutions z of such an equation has the structure of a binary tree, which can be computed very fast bitwise by a recursive process, starting with the lowest bit.

However, solutions of a single equation exist only with a probability of about 1/2000. Another problem is that we have to accomplish all parts simultaneously. (Observe that there is a great overlapping between the involved input words.) Here matters are becoming more and more complicated and technical. At this point methods are introduced that can be characterized by terms such as "continuous approximation and improvement", "randomized", and "genetic". Also techniques from differential cryptanalysis are borrowed. The reader who is interested in details can be referred to the related attacks on RIPEMD [8] and MD4 [9].

After parts (1), (2), and (3) are managed simultaneously, the inputs X and $\tilde{X}$ of the collision and also the contents of the chaining registers after step 14 are fixed. It is therefore a routine matter to compute backwards starting with step 14 in order to end at the initial value.

For a possible extension of this attack leading to collisions of the hash function one has to add a fourth part and handle it simultaneously with the three previous parts, namely

(4) matching a prescribed initial value.

This would certainly require hard additional cryptanalytic work. However, we anticipate that it can be done.

For a possible extension of this attack leading to collisions of the hash function one has to add a fourth part [...] matching a prescribed initial value.

An Example

An implementation of the previously described attack allows to find collisions for the compression function of MD5 in about 10 hours on a Pentium-PC. For

$$\Delta = \tilde{X}[14] - X[14] = \texttt{0x00000200}$$

we found the following input X and initial value IV:

$X[0]$ =	0xAA1DDA5E	$X[8]$ =	0x98A1FB19
$X[1]$ =	0xD97ABFF5	$X[9]$ =	0x1FAE44B0
$X[2]$ =	0x55F0E1C1	$X[10]$ =	0x236BB992
$X[3]$ =	0x32774244	$X[11]$ =	0x6B7A669B
$X[4]$ =	0x1006363E	$X[12]$ =	0x1326ED65
$X[5]$ =	0x7218209D	$X[13]$ =	0xD93E0972
$X[6]$ =	0xE01C135D	$X[14]$ =	0xD458C868
$X[7]$ =	0x9DA64D0E	$X[15]$ =	0x6B72746A

$IV[0]$ =	0x12AC2375	$IV[2]$ =	0x5F62B97C
$IV[1]$ =	0x3B341042	$IV[3]$ =	0x4BA763ED

The common compression value C of X and $\tilde{X}$ is

$C[0]$ =	0xBF90E670	$C[2]$ =	0x9CE4E3E1
$C[1]$ =	0x752AF92B	$C[3]$ =	0xB12CF8DE

SHA-1, HAVAL, and RIPEMD-160

The SHA-1 compression function has 80 steps. This is rather impressive compared with the 64 steps for MD5. The most remarkable design idea in SHA-1 is the expansion of the sequence of input words for the compression instead of multiple applications of input words in different rounds. At least in the new version, this expansion seems to make it very difficult to control the effect of an input difference.

Various promising new characteristica are involved in the design of HAVAL. These probably improve the cryptographic strength, but could also be pitfalls introducing unexpected weaknesses. It should be investigated whether there is a suitable modification of the MD4 attack, which could be applied to the 3-round version of HAVAL. As long as such analysis has not been worked out, there is no sufficient base to assess the strength of HAVAL.

The presented attack does not yet threaten practical applications of MD5, but it comes rather close. ....in the future MD5 should no longer be implemented... where a collision-resistant hash function is required.

The design of RIPEMD-160 is directly based on RIPEMD (and the 256-bit extension of MD4). However, conclusions of the recent analytic results are taken into account (how to choose, or not to choose, certain parameters), and the number of rounds is extended from 3 to 5 (for each of two parallel lines), i.e. there are 160 steps. RIPEMD-160 produces 160-bit hash values (nomen est omen).

Conclusions

The presented attack does not yet threaten practical applications of MD5, but it comes rather close. In view of the flexibility of the new analytic techniques it would be unwise to assume that the attack could not be improved. Ron Rivest [16] commented on the status of MD4, after two-round attacks had been found, that it is "at the edge" in terms of risking successful cryptanalytic attack. Today this assessment characterizes the status of MD5.

Therefore we suggest that in the future MD5 should no longer be implemented in applications like signature schemes, where a collision-resistant hash function is required. According to our present knowledge, the best recommendations for alternatives to MD5 are SHA-1 and RIPEMD-160.

MD5 can still be used as a one-way function. The HMAC ... is not touched by the recent analytical progress.

If essentially weaker cryptographic properties than collision-resistance suffice then the use of MD5 might still be secure. MD5 can still be used as a one-way function. The HMAC due to Bellare, Canetti, and Krawczyk [3, 4] is not touched by the recent analytic progress. Future research should analyse hash functions with respect to properties like pseudo-random behaviour, which are required in message authentication constructions.

References

[1] FIPS 180-1. Secure hash standard. NIST, US Department of Commerce, Washington D.C., April 1995.

[2] RIPE Consortium: *Ripe Integrity Primitives — Final report of RACE Integrity Primitives Evaluation (R1040)*. Lecture Notes in Computer Science, vol. 1007, Springer-Verlag, 1995.

[3] M. Bellare, R. Canetti, and H. Krawczyk. The HMAC construction. *CryptoBytes*, 2(1), 1996, pp. 12-15.

[4] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. *Advances in Cryptology — Crypto '96*, Lecture Notes in Computer Science, Springer-Verlag, 1996, pp. 1-15.

[5] B. den Boer and A. Bosselaers. An attack on the last two rounds of MD4. *Advances in Cryptology — Crypto '91*, Lecture Notes in Computer Science, vol. 576, Springer-Verlag, 1992, pp. 194-203.

[6] B. den Boer and A. Bosselaers. Collisions for the compression function of MD5. *Advances in Cryptology — Eurocrypt '93*, Lecture Notes in Computer Science, vol. 773, Springer-Verlag, 1994, pp. 293-304.

[7] I.B. Damgård. A design principle for hash functions. *Advances in Cryptology — Crypto '89*, Lecture Notes in Computer Science, vol. 435, Springer-Verlag, 1990, pp. 416-427.

[8] H. Dobbertin. RIPEMD with two-round compress function is not collision-free. *Journal of Cryptology*, to appear.

[9] H. Dobbertin. Cryptanalysis of MD4. *Fast Software Encryption — Cambridge Workshop*, Lecture Notes in Computer Science, vol. 1039, Springer-Verlag, 1996, pp. 53-69.

[10] H. Dobbertin. Extended MD4 compress is not collision-free. Unpublished abstract, October 1995.

[11] H. Dobbertin. Alf swindles Ann. *CryptoBytes*, 1(3), 1995, p. 5.

[12] H. Dobbertin, A. Bosselaers, and B. Preneel. RIPEMD-160: A strengthened version of RIPEMD. *Fast Software Encryption — Cambridge Workshop*, Lecture Notes in Computer Science, vol. 1039, Springer-Verlag, 1996, pp. 71-82. (An updated and corrected version is available at *ftp.esat.kuleuven.ac.be*, directory */pub/COSIC/bosselae/ripemd/*.)

[13] P.C. van Oorschot and M.J. Wiener: Parallel collision search with applications to hash functions and discrete logarithms. *Proc. of the 2nd ACM Conference on Computer and Communications Security*, ACM, 1994, pp. 210-218.

[14] R. Rivest. The MD4 message digest algorithm. *Advances in Cryptology — Crypto '90*, Lecture Notes in Computer Science, vol. 537, pp. 303-311, Springer-Verlag, 1991.

[15] R. Rivest. The MD4 message-digest algorithm. Request for Comments (RFC) 1320, Internet Activities Board, Internet Privacy Task Force, April 1992.

[16] R. Rivest. The MD5 message-digest algorithm. Request for Comments (RFC) 1321, Internet Activities Board, Internet Privacy Task Force, April 1992.

[17] S. Vaudenay. On the need for multipermutations: cryptanalysis of MD4 and SAFER. *Fast Software Encryption — Leuven Workshop*, Lecture Notes in Computer Science, vol. 1008, Springer-Verlag, 1995, pp. 286-297.

[18] Y. Zheng, J. Pieprzyk, and J. Seberry. HAVAL — a one-way hashing algorithm with variable length and output. *Advances in Cryptology — Auscrypt '92*, Lecture Notes in Computer Science, Springer-Verlag, 1993, pp. 83-104.

RSA-130 Factored

As is well known, the security of the RSA cryptosystem and the difficulty of factoring are closely related. To help track progress in factoring technology and to promote interest in factoring as a research problem, RSA Data Security sponsors the *RSA Factoring Challenge* with financial prizes being awarded to successful factorers.

In April of this year RSA-130, a number of 130 decimal digits listed as part of the RSA Data Security Factoring Challenge, was factored using the generalized number field sieve. Demonstrating that improvements to the generalized number field sieve continue, the computational effort required to factor RSA-130 is reported to be roughly one tenth of that required for the famous factorization of the smaller RSA-129 [1].

The earlier factorization of RSA-129 used a variant of the quadratic sieve and so this latest factorization confirms that for very large general-purpose factorizations, the generalized number field sieve is the algorithm of choice. Many people were involved in the factorization of RSA-130, their contribution as reported to the factoring challenge is as follows:

contribution	*factorer*
28.37%	Bruce Dodson
27.77%	Peter L. Montgomery and Marije Elkenbracht-Huizing
19.11%	Arjen K. Lenstra
17.17%	WWW contributors
4.36%	Matt Fante
1.66%	Paul Leyland
1.56%	Damian Weber and Joerg Zayer

At present, four RSA numbers from the original challenge list have been factored and the notable factorization of RSA-129 (the original RSA challenge as described by Martin Gardner in Scientific American [2]) has also been included for completeness. The concept of a MIPS-year is often used as a measure of computational effort and is defined as the number of operations completed in a year by a computer capable of operating at one million operations in each second.

number	*month*	*MIPS-years*	*algorithm*
RSA-100	April 1991	7	quadratic sieve
RSA-110	April 1992	75	quadratic sieve
RSA-120	June 1993	830	quadratic sieve
RSA-129	April 1994	5000	quadratic sieve
RSA-130	April 1996	500	generalized number field sieve

While the dramatic performance of the generalized number field sieve reflected in these figures is notable, such performance has been anticipated for some time. Once again it serves as illustration that a 512-bit RSA modulus (which is less than 160 decimal digits in length) can be anticipated to offer marginal security [3]. Indeed, RSA Laboratories' current recommendation that the minimum length of RSA modulus should be 768 bits (or more than 230 decimal digits) continues to offer users a wide margin of safety against current factoring techniques. By using the factorization of RSA-130 as a data point, we can estimate that current techniques would require a little less than 10^8 MIPS-years to factor a 768-bit RSA modulus.

While the dramatic performance of the generalized number field sieve reflected in these figures is notable, such performance has been anticipated for some time.

While at first sight the continuing factorization of increasingly large numbers might seem worrisome for the security of RSA, this is not the case when the improvements are broadly in line with general expectations. The excellent performance of the generalized number field sieve in this latest factorization broadly confirms predictions based on earlier factorizations and the factorization of RSA-130 gives yet more information with which to calibrate the security offered by an implementation of RSA. Factoring continues to be one of today's best studied cryptanalytic problems.

The reader interested in more information about possible factoring trends is referred to Andrew Odlyzko's article *The Future of Integer Factorization* [3]. More information on the RSA Factoring Challenge can be obtained by sending E-mail to *challenge-info@rsa.com*.

References

[1] D. Atkins, M. Graff, A.K. Lenstra and P.C. Leyland. The magic words are squeamish ossifrage. In *Advances in Cryptology — Asiacrypt '94*, pages 263-277, Springer-Verlag, 1995.

[2] M.Gardner. Mathematical games. *Sci. Amer.*, August 1977, pages 120-124.

[3] A. Odlyzko. The future of integer factorization. *CryptoBytes*, vol.1, no.2, pages 5-12.

The Security of DESX

Phillip Rogaway
Department of Computer Science
Engineering II Building
University of California
Davis, CA 95616 USA

A simple extension of DES, called DESX, has been shown to be virtually immune to exhaustive key search.

With keys of just 56-bits, the susceptibility of DES to exhaustive key search has been a concern since the cipher was first made public. Now a simple extension of DES, called DESX, has been shown to be virtually immune to exhaustive key search. This note describes the DESX construction and the sense in which it is has been proven sound. The construction is due to Ron Rivest [4], while its soundness proof is due to Joe Kilian and myself [3].

Strengthening DES

Despite impressive results on the differential and linear cryptanalysis of DES, the only practical attack described to date remains exhaustive key search. The cost of this attack keeps going down. In 1993 Wiener provided a careful estimate which showed that for \$1 million one could build an engine which, given a single ⟨plaintext, ciphertext⟩ pair, could recover the key in about 3.5 expected hours [5]. It would seem that DES, when used in its most customary manner, has already become a rather marginal choice for meeting commercial data privacy requirements.

The work to DESX encrypt a message block is just two XORs more than the work to DES encrypt a message block.

An attractive way to overcome DES's susceptibility to key search is to build a new block cipher out of DES, leaving alone DES's internal structure. One advantage of this approach is that it can preserve the assurance benefits which DES has gained over the years. Another advantage is that it allows one to gainfully employ existing DES hardware and software. Finally, making a new block cipher which is to be used in standard ways is more general and more conducive to analysis than coming up with entirely new DES-based modes of operation for specific higher-level tasks such as encryption.

The most well-known suggestion to strengthen DES is "triple DES," one version of this being defined by $\mathrm{EDE3}_{k1.k2.k3}(x) = \mathrm{DES}_{k3}(\mathrm{DES}^{-1}_{k2}(\mathrm{DES}_{k1}(x)))$. That is, the key for EDE3 is $56 \times 3 = 168$ bits, and one enciphers a 64-bit block by enciphering under one 56-bit subkey, deciphering under a second, then enciphering under a third. (The reason the second step is DES^{-1}_{k2} and not DES_{k2} is for DES-compatibility: set $K = k.k.k$ to make $\mathrm{EDE3}_K = \mathrm{DES}_k$. The reason for using DES three times instead of two is the existence of "meet-in-the-middle" attacks on double DES.)

Phillip Rogaway is an assistant professor at UC Davis. His research has focused on using a more practical provable-security approach to cryptographic protocol design and analysis. He can be contacted at rogaway@cs.ucdavis.edu.

The problem with triple DES is that it is much slower than DES itself—roughly a third the speed. When cipher block chaining EDE3, this slowdown will occur in hardware (even if one tries to compensate with additional hardware) as well as in software. In many situations, these performance penalties are unacceptable.

There is, then, a need for a *cheap* way to strengthen DES against key-search attack, injecting extra key bits without impacting the cipher's internal structure. Back in 1984, Ron Rivest came up with just such an extension of DES. It is called DESX.

The DESX Construction

DESX is defined by

$$\mathrm{DESX}_{k.k1.k2}(x) = k2 \oplus \mathrm{DES}_k(k1 \oplus x).$$

That is, a DESX key $K{=}k.k1.k2$ consists of $56 + 64 + 64 = 184$ bits comprising three different subkeys: a "DES key" k, a "pre-whitening key" $k1$, and a "post-whitening key" $k2$.

To encrypt a message block we XOR it with $k1$, DES encrypt under k, then XOR with $k2$. Thus the work to DESX encrypt a message block is just two XORs more than the work to DES encrypt a message block.

The amazing thing about DESX is that these two XOR operations render the cipher much less susceptible to key-search attack. In the sequel we will quantify just how much the "DESX trick" really buys. Here we won't try to impart any intuition as to *why* DESX works, except to suggest that the pre- and post-whitening make it difficult for the attacker to single out even one valid $\langle x_i, \mathrm{DES}_k(x_i)\rangle$ pair when the attacker mounts a chosen-plaintext attack to get many $\langle P_j, \mathrm{DESX}_K(P_j)\rangle$ pairs.

What is Key Search?

We want to emphasize that what we show in [3] is that there is no feasible *key-search* attack on DESX; we don't know that there's not a feasible attack of some other type. Thus it is essential to understand what exactly we mean by a key-search attack on DESX. Understanding this may take a careful reading or two.

Sometimes people think of a key-search attack as one in which the adversary systematically explores some universe of possible keys. But we mean something broader: we claim that the essence of key search is that the adversary carries out her mission in a way which doesn't exploit the internal structure of the underlying cipher—she uses the underlying cipher (e.g., DES) as a *black box.*

Now it's a very nebulous thing what we just said—what does it *mean* to use DES only as a black box? How can we make this idea formal? The first step to an answer, oddly enough, is to generalize the DESX construction.

For *any* block cipher F we can define block cipher FX by $FX_{k.k1.k2}(x) = k2 \oplus F_k(k1 \oplus x)$. Of course DESX = FX when F = DES.

Now if you've come up with a way to break DESX and it doesn't use anything structural about DES (that is, it uses DES as a "black box"), then your method ought to break FX for any F with a 56-bit key and a 64-bit block size. In other words, we can recast the problem *analyze how well* DESX *resists key-search attack* to the problem *analyze how well FX resists key-search attack, for some other function F.* But if we just switch from looking at FX with F = DES to looking at FX with F being some other particular function, then we haven't gained anything: whatever function F we choose, it also might have structure the adversary could exploit.

To overcome this difficulty we will demand that the adversary attack FX for a *random* block cipher F. An adversary can't exploit structure in a random block cipher because there is no structure to exploit! The *thesis* underlying our analysis is that for any particular function F, we can measure how well a *key-search* adversary can attack FX by measuring how well an *arbitrary* adversary can attack the FX-construction for a *random* block cipher F. In particular, it is our thesis that one can measure how well a key-search adversary can attack DESX by measuring how well an arbitrary adversary can attack the FX-construction, when F is a random block cipher with a key length and block length to match DES.

The Formal Model

Not using the structure of DES is one thing, but even a key-search adversary on DESX deserves the ability to compute DES or DES^{-1} at points of her choosing. Analogously, if we're going to ask an adversary to attack FX it's only fair to give her the capability to compute F and F^{-1}. So we will provide our adversary with "black boxes" to compute these. The F black box, on input (k,x), computes $F_k(x)$. The F^{-1} black box, on input (k,y), computes $F_k^{-1}(y)$.

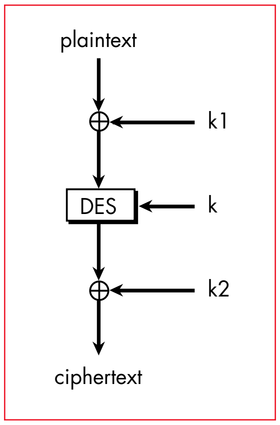

Figure 1. *Encryption using* DESX$_{k.k1.k2}$

Now we're ready to make quantitative the security of the FX construction: we will define the *advantage* an adversary A obtains in attacking it.

Fix parameters κ (the key length) and n (the block length). Choose a random block cipher F which uses a κ-bit key and an n-bit block length. This means to choose a random permutation for each κ-bit key k. We start off by giving adversary A black boxes for F and F^{-1}. This affords A the ability to compute the underlying block cipher without letting her exploit its structure. Next we need to present A with a "test" to see how well she can attack FX.

Here is that test. We present A with one of two types of encryption oracles. Which type A gets is chosen at random.

- A **good** encryption oracle is one which chooses a random $(\kappa + 2n)$-bit FX-key, K, and then answers each n-bit question P by $FX_K(P)$.
- A **bogus** encryption oracle is one which chooses a random permutation π on the space of n-bit blocks and then answers each n-bit question P by $\pi(P)$.

To win the above "FX_K-or-π" game, A is supposed to correctly identify if she was given a **good** encryption oracle or a **bogus** encryption oracle. The *ad-*

It is our thesis that one can measure how well a key-search adversary can attack DESX by measuring how well an arbitrary adversary can attack the FX-construction, when F is a random block cipher.

vantage of A is defined as the probability that A answers that she has a **good** encryption oracle given that we give her a **good** encryption oracle, minus the probability that A answers that she has a **good** encryption oracle given that we give her a **bogus** encryption oracle. An advantage of 1 indicates the the adversary is doing a great job of distinguishing FX_K from a random permutation π unrelated to F; she always answers correctly. An advantage of 0 indicates that A is doing a worthless job of making this distinction; she could just as well answer by flipping a coin. An advantage of -1 indicates that A is doing a terrible job of guessing; in this case, she'd do well to just swap when she says "**good**" and when she says "**bogus**."

Taking account of the DES key-complementation property and then applying our thesis, the effective key length of DESX, with respect to key search, is at least 118-lg m bits, where m is the number of DESX-encrypted message blocks the adversary can obtain.

We have given the adversary what would seem to be a very easy game—much easier, say, then asking her to try to recover the key K when presented with a bunch of FX_K-encrypted points. Indeed if there were an adversary A who could recover K from FX_K-encrypted points, then there would be another adversary A' who, with resources comparable to A's, could get an advantage of ≈ 1 in the FX_K-or-π game. In general, we use the FX_K-or-π game because adopting this liberal notion of adversarial success makes our results stronger: if no adversary can distinguish FX_K from a random permutation, then no adversary can break FX in any useful way.

Like Having 118-lg *m* Bits

Having defined the FX_K-or-π game and the advantage an adversary achieves in playing it, it becomes possible to analyze the maximal advantage that *any* adversary can obtain, assuming that the adversary expends only some specified computational effort. We won't describe any details of this analysis; we'll just state the final answer. See [3] for the proof. That proof and its setup builds, in turn, on [1].

Consider the maximal advantage that any adversary can achieve in the FX_K-or-π game when the adversary asks a total of t questions of her F/F^{-1} black boxes and a total of m questions of her FX_K-or-π encryption oracle. This value, *MaxAdv*, depends on m and t, as well as on F's block length n and key length κ. The main result of [3] says that $MaxAdv(m,t,n,\kappa) \le mt \cdot 2^{-n-\kappa+1}$.

Since each query to an F/F^{-1} black box takes at least one unit of time, we see that an adversary which runs in time T can get advantage which is at most $T \cdot 2^{-n-\kappa+1+\lg m}$. So another way to state the result of [3] is that the effective key length of the FX-construction, with respect to key search, is at least $n+\kappa-1-\lg m$ bits.

Let's apply the above result. Assume that you use DESX on a given key to encrypt at most 2^{30} blocks. Use the thesis stated earlier. Then a key-search adversary attacking DESX which runs in time T could gain an advantage of at most $T \cdot 2^{-56-64+1+30} = T \cdot 2^{-89}$. For example, if $T < 2^{70}$ then the adversary's advantage is less than 2^{-19}.

Now the truth is that there is one structural property of DES which attacks sometimes do exploit: it is the DES key-complementation property: $DES_k(x) = \overline{DES_{\bar{k}}(\bar{x})}$. We can "factor out" this property (that is, allow the adversary to exploit this particular structural property) by fixing some particular bit of the DES key. This reduces the DES key length by 1. Thus, taking account of the DES key-complementation property and then applying our thesis, we get that the effective key length of DESX, with respect to key search, is at least $55 + 64 - 1 - \lg m = 118 - \lg m$ bits.

Concluding Remarks

DESX should be used just as one would use DES. For example, instead of applying cipher block chaining (CBC) to DES, one would apply it to DESX. DESX interacts particularly nicely with CBC, as DESX-CBC encryption amounts to masking the plaintext with key material, then DES-CBC encrypting, then masking what results with more key material. In particular, hardware which performs DES-CBC encryption can be gainfully employed to perform the non-trivial part of DESX-CBC encryption.

A minor inconvenience of DESX is its strange key size. In applications it might be preferable to extend the definition of DESX to use arbitrary-length keys, or else to use keys of some fixed but more convenient length. As an example, in RSA DSI's BSAFE 3.0 implementation of DESX, the underlying key *key* may be 128 bits, in which case the DES key and the pre-whitening key are determined directly from *key*, while the post-whitening key is a complex function of all 16 bytes of *key*.

DESX was intended to *improve* DES's strength against key search, and *preserve* its strength with respect to other possible attacks. But as Kaliski and Robshaw indicate, DESX actually does add security against differential and linear cryptanalysis [2], increasing the required number of known or chosen plaintexts to be in excess of 2^{60}. Further added strength against these attacks would seem to be achieved by replacing the XOR operations of DESX by addition, as in $\text{DES-PEP}_{k.k1.k2}(x) = k1 + \text{DES}_k(k2+x)$, where $L.R+L'.R' = (L \overset{\wedge}{+} L').(R \overset{\wedge}{+} R')$, $|L| = |R| = |L'| = |R'| = 32$, and $\overset{\wedge}{+}$ denotes addition modulo 2^{32}. The $\kappa + n - 1 - \lg m$ bound of [3] also applies to variants such as this one.

As we've explained, our results don't say that it's infeasible to build a machine which would break DESX in a reasonable amount of time. But they do imply that such a machine would have to employ some radically new idea: it couldn't be a machine implementing a key-search attack, in the general sense which we've described.

DESX would seem, in virtually every sense, to be a "better DES than DES." It is simple, DES-compatible, efficiently implementable in hardware, essentially the same speed as single DES in software, can profitably use existing DES hardware, and it has been proven, in a strong sense, to add much strength against exhaustive key search. Emerging standards which specify DES or triple DES would do well to consider DESX.

DESX would seem, in virtually every sense, to be a "better DES than DES."

References

[1] S. Even and Y. Mansour, "A construction of a cipher from a single pseudorandom permutation." Asiacrypt '91.

[2] B. Kaliski and M. Robshaw, "Multiple encryption: weighing security and performance." *Dr. Dobb's Journal*, January 1996, 123—127.

[3] J. Kilian and P. Rogaway, "How to protect DES against exhaustive key search." Crypto '96, and *http://wwwcsif.cs.ucdavis.edu/~rogaway*.

[4] R. Rivest, *personal communication.*

[5] M. Wiener, "Efficient DES key search." Manuscript of August 20, 1993, and Technical Report TR-244, School of Computer Science, Carleton University, May 1994.

S T A N D A R D S U P D A T E

Standardization Efforts for Triple-DES Continue

Progress on the ANSI X9.52 standard continues with the completion of version 6 of a proposed draft standard for the use of triple-DES.

Triple-DES is a natural way to design a block cipher that builds on the strengths of DES, however there are many important considerations in the drafting of such a standard.

In all, nine modes of triple-DES are proposed with four being the natural counterparts of the well known modes of use of single DES. For triple-DES some additional flexibility in specifying some modes is available and in addition to the cipher block chaining (CBC), cipher feedback (CFB) and output feedback (OFB) modes inherited from single DES, three slight variants termed *interleaved* CBC and *pipelined* CFB and *pipelined* OFB are also available.

Two new modes of triple-DES that were developed at IBM are also presented in the current draft. One new mode, termed *cipher block chaining with masking* or CBCM, enhances the conventional CBC triple-DES arrangement with the inclusion of an exclusive-or of the same 64-bit quantity both before and after the middle DES operation. This 64-bit quantity varies for each block of encryption and it is generated by the independent running of a fourth DES operation in OFB mode. The other new mode is the interleaved variant of CBCM.

Two new modes of triple-DES that were developed at IBM are also presented in the current draft.

For more information on the progress of ANSI X9.52 contact the X9.52 editor, Bill Lattin, at *lattin@cylink.com*.

IEEE P1363 Works Toward Integrated Draft

Work continues on the IEEE P1363 project, "*Standard for RSA, Diffie-Hellman, and Related Public-Key Cryptography*" with the integration this summer of material on discrete logarithm, RSA, and elliptic curve cryptosystems toward the working group's first draft standard.

This follows the decision at the May 1996 meeting in Oakland to proceed toward the integrated draft and to appoint Yiqun Lisa Yin of RSA Laboratories as co-editor to facilitate the work. Previously, work on the first draft had been limited to elliptic curve cryptosystems, with RSA and discrete logarithm systems deferred to a second draft.

The P1363 project seeks to provide a basis for interoperable security by specifying cryptographic primitives ... and complete cryptographic schemes.

The P1363 project seeks to provide a basis for interoperable security by specifying cryptographic primitives such as key generation, "raw" encryption, and "raw" signatures, and complete cryptographic schemes based on the primitives, auxiliary functions such as hashing, and certain data formatts.

The integrated draft will be presented and discussed at the working group's meeting August 22-23 in Santa Barbara (following the CRYPTO '96 conference). Balloting is expected sometime in 1997. Further information about the project is available from *http://stdsbbs.ieee.org/*.

PKCS #11 / Cryptoki Workshop Held at MIT

Development of PKCS #11 / Cryptoki, the programming interface for cryptographic devices coordinated by RSA Laboratories, took another step forward with a two-day workshop held at MIT on July 9 and 10.

A general theme ... was that Cryptoki is in the process of becoming a "de facto" standard.

The workshop, attended by 31 people, included implementation reports by Roland Lockhart of Nortel, Bruno Couillard of Chrysalis, and Thi Nguyen of SCI; talks by Phil Mellinger of the Federal Security Infrastructure on that organization's work and by Dave Balenson of TIS on the International Cryptography Experiment; and a tutorial by Mike Matyas of IBM on control vectors and IBM's Common Cryptographic Architecture.

The afternoon of the second day was spent identifying improvements for version 1.1 of PKCS #11, as well as potential topics for consideration in version 2.0.

A general theme echoed by the presenters and the participants was that Cryptoki is in the process of becoming a "de facto" standard at the application / token interface. It was also the general sense that Cryptoki can coexist with interfaces at higher levels, such as GSS-API, GCS-API, and Microsoft Crypto API; it complements and does not compete with them. Thus, Cryptoki seems to be in a stable position in the evolving cryptographic API architecture.

Version 1.1 of PKCS #11 is expected to be completed this fall, with implementations demonstrated in late 1996 and early 1997.

A complete meeting summary, including proposed changes to PKCS #11, is available under Standards at *http://www.rsa.com/rsalabs*.

More Progress on S/MIME

Progress on the S/MIME standard (Secure/Multipurpose Internet Mail Extensions) for secure electronic mail has reached a new level, with the first on-line interoperability test near completion. Five E-mail packages were tested: Frontier Intr@net Genie, OpenSoft ExpressMail, Deming Secure Messenger, ConnectSoft E-Mail Connection, and Raph Levein's Premail.

Due to the growing interest in S/MIME, a second round of interoperability testing is planned to begin in mid-August. A group of four to five new vendors is expected to participate.

The S/MIME message specification was submitted to the IETF as an informational RFC. This is an important step in contributing a well defined and tested design to the standards body.

A second draft of the S/MIME implementation guidelines is currently being written. This will be the main focus of the fourth S/MIME developer's workshop, to be held in San Francisco on September 4.

More detailed information on all of the above is available at *http://www.rsa.com/rsa/S-MIME*.

At the Newton Institute:

Coding Theory, Cryptology, and Computer Security

Thomas A. Berson
Anagram Laboratories
Palo Alto, CA 94301 USA

The Isaac Newton Institute for Mathematical Sciences at the University of Cambridge is housed in a custom-designed building fifteen minute's walk from the center of Cambridge through medieval college courtyards and well-tended gardens. Amongst the treasures it houses are a death mask of Isaac Newton (1642-1727), a machine which at the push of a button will freshly brew a cup of the hot drink of your choice from among twelve different kinds of coffee, tea, or chocolate, and a complete set of Crypto and Eurocrypt proceedings from 1990 to present. From January through June of 1996 this building was temporary home to an aggregate of 250 cryptographers who were there to participate as members or visitors in the Isaac Newton Institute Program on Computer Security, Cryptology, and Coding Theory.

Thanks to my employer's enlightened sabbatical policy I was able to stay at the Institute for four months. In this story I will try to give an idea of what the program was and what it was like to be a member of it. I will also mention one or two of the program's research results which have already been published by their authors.

The Program

Ross Anderson of the Cambridge University Computer Laboratory had the idea in 1992 that a six-month research program on cryptology might be organized at the Newton Institute, a new facility which had not then even opened its doors for scientific work. The program would allow researchers from around the world to work together for an extended period of time away from the distractions of their home departments.

Ross championed the program through four years of proposal writing, revision, fund raising, negotiation and planning. He was joined in this by Paddy Farrell, Peter Landrock, and Roger Needham. The questions of how to organize the scientific work and of whom to invite were addressed by a scientific committee of Tom Berson, Peter Cameron, Whitfield Diffie, John McLean, Jim Massey, Chris Mitchell, Harald Neiderreiter, Andrew Odlyzko, Tatsuaki Okamoto and Jean-Jacques Quisquater.

The scientific committee decided to take the program on an intellectual arc from theoretical toward practical in a sequence of units, each of three or four weeks. These were: coding theory; topics spanning coding theory and cryptography; fast encryption algorithms; computational number theory; public key techniques; formal methods; computer security; and security engineering.

The scientific committee decided to take the program on an intellectual arc from theoretical toward practical in a sequence of units.

The Institute has office space for about forty researchers, but because we were sharing the facility with another program (on Mud Theory, but that's a different story), we could only have about twenty people at a time. We identified key researchers in each of the units and asked them who else should come. Some people we invited didn't come. Others we didn't invite showed up. All in all, about 95 different people were members of the program and spent anywhere from two weeks to six months at the Institute.

In addition to the research program, four information security workshops were held at the Institute during our stay there. These were open to those who submitted papers as well as to members of the Institute program. More about these later.

In addition to the research program, four information security workshops were held at the Institute during our stay there.

The Place

If you had been a member of the program you would have been delivered from the rail station by your taxi to the four-storey yellow brick building which houses the Institute[1]. It is a pagoda designed for mathematicians. Outside there is a sculpture representing inspiration and plenty of room for bicycle parking. Inside, the architecture encourages mixing. This promotes collaboration and allows newly arriving members to assimilate rapidly into the Institute community.

Tom Berson is President of Anagram Laboratories, a cryptographic consultancy in Palo Alto, CA. He is his own enlightened employer. He can be contacted at berson@anagram.com.
More information about the Isaac Newton Institute can be found at http://www.newton.cam.ac.uk.

[1] This assumes you asked to be taken to "the Isaac Newton Institute." One member asked simply for "The Isaac Newton." She was delivered to the pub of that name.

The ground floor contains an entrance area, the library, a large and a small seminar room, a directory showing who is in residence, bulletin boards advertising the week's events in the Institute and in the University, and staff offices. The staff is a staff of angels who take care of planning, housing, meeting arrangements, computer systems, library, catering, cleaning, and accounting. An open stairway leads up to a half floor, open to below, which has offices around the periphery. A few offices are for one person, most are for two, a few are for three, four, or six. All offices have workstations: Suns, HPs, or Macintoshes. In the walkway in front of these offices are the pigeon holes for members' post and a display with mug shots of the staff and members of the Institute.

I do expect that we will see published over the next two years perhaps two hundred papers which give credit to time spent at the Newton Institute.

From here two different open stairways lead to the next level. Here there are again offices around the periphery. The center is taken up with tables, easy chairs, couches, the marvelous hot drink machine, and a catering facility where lunch is served on weekdays. There are also bulletin boards here describing the events in each of the programs. From this level two different open stairways lead up to another half floor, again open to below, again with offices around the periphery. The interior volume of the building is open and pleasantly complex. The stairways give the impression of being in an M.C. Escher drawing.

Did I mention the chalkboards? They are everywhere. In the offices, of course, but also under the stairways, in the public spaces, in the walkways, in the lift (*Am.* elevator), and in the men's and women's loos (*Am.* toilets). The conventionally located chalkboards will typically have a group of three or four people standing around, each with a piece of chalk in hand, engaging in either passionate explanation or hot debate. These discussions often led to collaboration, to seminar presentations, and to papers. I cannot say much about the results of these collaborations in this story; the results belong to the collaborators. I do expect that we will see published over the next two years perhaps two hundred papers which give credit to time spent at the Isaac Newton Institute.

A Week at the Institute

Although the program syllabus progressed from topic to topic, each week's schedule was more or less the same.

Monday. Most new members arrive and get oriented on Monday. In the late afternoon an Institute Seminar, open to the public, on some mathematical topic. My personal favorite of these was *Rational Solution of Diophantine Equations*, by Sir Peter Swinerton-Dyer. Wine and snacks afterward.

Tuesday. In the afternoon members of the program offer formal papers on their work. In the evening we all go to dine at one of the Cambridge colleges. We get dressed up and eat in elegant surroundings: an ancient room, silver on the table, candlelight, portraits on the wall. After dinner we indulge in toasts and speeches.

Wednesday. A slow start. Much reading of journals in the library.

Thursday. A work day, few distractions. Quiet in the offices. Buzz in the walkways. Network printers very busy. Most people gather for breaks at coffee (10AM), lunch (12:30PM) and tea (4:30PM).

Friday. In the morning a seminar with papers by members of the program on their work in progress. These are less formal than those in the Tuesday series. In the afternoon off to the Cambridge Computer Laboratory for their security seminar. Rump session afterward at The Eagle, a pub.

Saturday, Sunday. Days off or work days: your choice. Best network bandwidth. Time to surf the web and download things.

The Workshops

The Third Annual Workshop on Fast Software Encryption (21-23 Feb). Plenty of great stuff on cryptanalysis, hash functions, and algorithms. My favorites: the RIPEMD-160 hash function proposal by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel; and the *Tiger* (a hash function), *BEAR* and *LION* (two block ciphers) proposals by Ross Anderson and Eli Biham. Proceedings are published as D. Gollmann, (Ed.), *Fast Software Encryption.* Springer Lecture Notes in Computer Science, Vol. 1039.

The 1996 Cambridge Workshop on Security Protocols (10-12 Apr). Much about protocol analysis, key escrow, and electronic commerce, especially micropayments. My favorite: Ron Rivest and Adi Shamir's

MicroMint (described in *CryptoBytes* 2(1), Spring 1996).

The First Workshop on Information Hiding (30 May - 1 Jun). An exciting workshop. Keynote talk by David Kahn. Papers on how messages may be hidden in signatures, images, audio, and text. Speakers use message mixing networks as a primitive and build from there. Ross Anderson, Serve Vaudenay, Bart Preneel and Kaisa Nyberg describe *The Newton Channel* (named after where they did the work). They show that many discrete log based signature systems may be insecure as they operate in more than one group at a time and key material may leak through those groups where discrete log is easy. They demonstrate broadcast and narrowcast channels in the ElGamal signature scheme if $p = qm + 1$, where m is smooth. They conclude that DSA is not vulnerable. The papers at the workshop illustrated an urgent need for common terminology to describe information hiding systems. Participants eventually compromised on the following: something to be hidden, the *embedded<type>* is placed into a *cover<type>* under control of a *(stego)key* to form a *stego<type>*. It is understood that *<type>* might be *text*, *message*, *image*, or whatever seems appropriate. The proceedings are to be published soon.

Personal Information – Security, Engineering and Ethics. This workshop was motivated by serious issues arising from the design of a central medical records system in the UK. I had already left the Institute when the workshop was held, so I cannot report on what happened.

The Bottom Line

The Newton Institute program was a one-off. There is not likely to be another cryptology program there within the foreseeable future. But I am certain that the professional collaborations and personal friendships started or furthered at the Institute during the first six months of 1996 will constructively enhance the nature of cryptologic research for many years to come.

I am certain that the professional collaborations and personal friendships started or furthered at the Institute […] will constructively enhance the nature of cryptologic research for many years to come.

N E W S   A N D   I N F O R M A T I O N

Rivest, Lampson Propose Simple Distributed Security Infrastructure

Ron Rivest and Butler Lampson have proposed in a recent paper a new infrastructure for public-key cryptography which may be considered as an simpler, more flexible alternative to the evolving infrastructure based on ITU-T X.509 certificates.

The proposal, called Simple Distributed Security Infrastructure (SDSI), was presented by Rivest at the USENIX Security Symposium in July. The following are some of the notable features of SDSI:

Principals are public keys: Certificates and other information in the system are about keys, rather than about individuals (named objects in the X.500 sense). Of course, names are still part of the system, and a certificate binds a key to a name.

Flexible certification and naming: Certificates can be issued by anyone, not just by particular principals in a hierarchy. Moreover, every principal has an independent name space. There is no single, global name space (though certain principals' name spaces are expected to be widely referenced). X.509 certificates are also flexible as far as certificate hierarchies (at least in principle), but assume names are assigned following X.500 conventions.

Integral groups and properties: Anyone can define a group of principals, either by listing them explicitly, or indirectly by combining other groups using AND, OR, and NOT operators. Groups, which may be named, provide a way of asserting that principals have a certain property.

Simple data structures: SDSI objects are represented as parenthesized lists, rather than ASN.1 / BER encodings.

Other efforts on a public-key infrastructure include the Internet Engineering Task Force's Public-Key Infrastructure (X.509) working group (*http://www.ietf.org*), which is building on the experience with Privacy-Enhanced Mail and with X.509 version 3 certificates; and the U.S. government's Federal Security Infrastructure Program (*http://www.gsa.gov/fsi/*).

A copy of Rivest and Lampson's paper describing SDSI is available on Ron Rivest's home page (*http://theory.lcs.mit.edu/~rivest*).

Ron Rivest and Butler Lampson have proposed … a new infrastructure for public-key cryptography.

RSA Data Security Announces the 1997 RSA Data Security Conference

The annual RSA Conference is the industry's largest and most prestigious computer security event, bringing together the world's cryptography systems experts with encryption policy makers, business people and the development community.

The conference will be held January 28-31, 1997 at the Masonic Auditorium, Nob Hill, San Francisco and has been expanded to four full days, providing unparalleled breadth and depth. There is something here for everyone whose life touches security technology: from the analyst to the CEO; from the developer to the professional cryptographer.

The conference begins with two days of general sessions in the beautiful Masonic Auditorium, at the top of Nob Hill in San Francisco. The gathering continues with two days of classes and breakout sessions at various sites on Nob Hill, including the famous Stanford Court, Mark Hopkins Intercontinental, and Fairmont Hotels.

In addition to general sessions, tutorials and breakouts, conference attendees will have the opportunity to see demos of hundreds of RSA-secured products in our Partner Fair expo floor.

Breakfast, lunch and breaks are included all four days, and several cocktail receptions will be held to provide attendees with ample time to network and trade ideas.

The RSA Conference always sells out, so register now by calling LKE Productions at 415/544-9300.

For contact and distribution information, see page 2 of this newsletter.

RSA LABORATORIES
CRYPTOGRAPHIC RESEARCH AND CONSULTATION
100 MARINE PARKWAY
REDWOOD CITY
CA 94065-1031
TEL 415/595-7703
FAX 415/595-4126
rsa-labs@rsa.com

PRESORT
FIRST CLASS
U.S. POSTAGE
PAID
MMS, INC