MIT/LCS/TM-164

THE CRYPTOGRAPHIC SECURITY OF COMPACT KNAPSACKS
(PRELIMINARY REPORT)

Adi Shamir

April 1980

THE CRYPTOGRAPHIC SECURITY OF COMPACT KNAPSACKS

(Preliminary Report)

Adi Shamir*

Department of Mathematics

Massachusetts Institute of Technology

April, 1980

*Supported by the Office of Naval Research under Contract No. N00014-76-C-0366

Keywords: cryptography, knapsack problems, Merkle-Hellman cryptosystems.

Abstract

In 1978, Merkle and Hellman introduced a knapsack-based public-key cryptosystem, which received widespread attention. The two major open problems concerning this cryptosystem are:

(i) Security: How difficult are the Merkle-Hellman knapsacks?

(ii) Efficiency: Can the huge key size be reduced?

In this paper we analyze the cryptographic security of knapsack problems with small keys, develop a new (non-enumerative) type of algorithm for solving them, and use the algorithm to show that under certain assumptions it is as difficult to find the hidden trapdoors in Merkle-Hellman knapsacks as it is to solve general knapsack problems.

1. Motivation

To introduce our notation, we briefly describe the Merkle-Hellman cryptosystem (more details can be found in Merkle and Hellman [1978]). The published key is a list of n generators a_i, each one of which is a randomly looking q bit nunber (the recommended parameters are $n \geq 100$, $q \geq 200$). To encrypt an n-bit message $X = x_1 x_2 \ldots x_n$, the sender uses the receiver's key to compute the cyphertext $b = \sum_{i=1}^{n} x_i a_i$, and transmits it over the insecure communication channel. To decrypt this cyphertext, the receiver uses a secret structure (trapdoor) embedded in the generators in order to solve this knapsack problem by a shortcut polynomial method. An eavesdropper, who knows b and the a_i's but not the secret trapdoor, is forced to use some general purpose knapsack solving algorithm, and even the best such algorithm (Schroeppel and Shamir [1979]) is currently too slow for problems of this size.

The main practical drawback of the Merkle-Hellman scheme is its huge key size (tens of thousands of bits, compared with hundreds of bits in the Rivest-Shamir-Adleman [1978] scheme and tens of bits in the DES [1976] scheme). The public key directory of large communication networks (telephone users, banks or military installations) can be extremely long, and the many minutes required to exchange such keys over slow telephone lines can severely restrict the usefulness of this public key cryptosystem.

To reduce the size of the key in a knapsack based cryptosystem, we can shorten the generators or decrease their number. The first approach is impossible, since:

(i) When $q < n$, the decryption function becomes ambiguous since there cannot be enough distinct sums to encode all the 2^n possible messages.

(ii) When $q \tilde{\approx} n$, the encryption function is almost a permutation, and knapsacks with this property seem to be cryptographically insecure (see Shamir [1979]).

(iii) When q is sufficiently small, the cryptanalyst can prepare a complete cleartext-cyphertext table by preprocessing the published key.

The second approach (which is mentioned in Merkle and Hellman's original paper) is possible, provided we use multi-bit substrings of the message as coefficients. All the knapsack solving algorithms developed to date are based on the enumeration of potential x_i solutions, and thus their complexity does not change when we replace an equation with one hundred 0-1 coefficients by an equation with four 25-bit coefficients (which are the four quarters of the 100-bit message). The key size, on the other hand, is reduced by a factor of 25, which makes this approach extremely attractive from the cryptographic point of view.

In this paper, we investigate the complexity of compact knapsack problems with a small number of generators and multi-bit coefficients.

In particular, we develop a new kind of knapsack solving algorithm which is not based on the enumeration of potential solutions, and use it to show that compact knapsacks are considerably less secure than their 0-1 counterparts.

2. Preliminaries

Definition: The set of n-generator knapsack problems is the set of equations of the form

$$\sum_{i=1}^{n} x_i a_i = b$$

in which the generators a_i and the target value b are given natural numbers, and the coefficients x_i (which must be integral and non-negative) are the unknowns. The set of compact knapsack problems is the union of these sets for all n.

Remarks: (i) There is a trivial upper bound of $\lfloor b/a_i \rfloor$ on the value of each x_i, and thus the set of compact knapsack problems is in NP. An easy reduction from set covering shows that it is NP-complete.
(ii) In cryptographic applications, it is necessary to publish a limit ℓ as part of the encryption key, and to encrypt only messages in which $0 \leq x_i < \ell$ (without such a bound, the decryption process cannot be unambiguous). This upper bound is assumed to be known to the

cryptanalyst, and can reduce the size of his search space from $\lfloor b/a_1 \rfloor \cdot \lfloor b/a_2 \rfloor \ldots \lfloor b/a_n \rfloor$ to ℓ^n.

Theorem 1: The sets of 1-, 2- and 3-generator knapsack problems are polynomially solvable.

Proof: (1) The 1-generator knapsack problem $x_1 a_1 = b$ is solvable iff a_i divides b.

(2) The most general integral solution of the equation

$$x_1 a_1 + x_2 a_2 = b$$

is

$$x_1 = c_1 b + t(a_2/\gcd(a_1,a_2))$$

$$x_2 = c_2 b - t(a_1/\gcd(a_1,a_2))$$

where t is an arbitrary integral solution and c_1, c_2 are the coefficients derived by Euclid's algorithm from the equation

$$c_1(a_1/\gcd(a_1,a_2)) + c_2(a_2/\gcd(a_1,a_2)) = 1 \quad .$$

The two inequalities $x_1 \geq 0$, $x_2 \geq 0$ define two rays of t values, and the 2-generator knapsack problem is solvable iff the intersection of the rays contains an integral point.

(3) This is a recent result whose proof is beyond the scope of this paper. The interested reader is referred to Kannan and Shamir [1980]. Q.E.D.

The complexity of n-generator knapsack problems for any fixed $n \geq 4$ is still open: to the best of my knowledge, no such set was ever shown to be either NP-complete or polynomially solvable. The best published algorithm for them takes $O(\sqrt{p})$ time both in the worst case and in the average case measures, where p is the number of points in the search space.

3. The New Approach

Definition: Given a compact knapsack problem K with a bound ℓ on the values of the coefficients, max(K) is defined as the largest target value which can be represented by the generators, i.e.,

$$\max(K) = \sum_{i=1}^{n} (\ell-1)a_i \ .$$

Definition: Two compact knapsack problems

$$K: \quad \sum_{i=1}^{n} x_i a_i = b \qquad 0 \leq x_i < \ell$$

$$K': \quad \sum_{i=1}^{n} x_i a'_i = b' \qquad 0 \leq x_i < \ell$$

are similar if there are two relatively prime numbers w (the multiplier) and m (the modulus) such that $m > \max(K)$, $m > \max(K')$, $b' = wb \pmod m$ and for all i, $a'_i = wa_i \pmod m$.

Lemma 2: Similarity is a reflexive and symmetric relation, and it is transitive whenever all the moduli used are the same.

Proof: Immediate from the fact that the multipliers which are relatively prime to m form a multiplicative group. Q.E.D.

Example: The three compact knapsack problems

$$K_1: \quad x_1 \cdot 19 + x_2 \cdot 31 + x_3 \cdot 46 = 50 \qquad 0 \leq x_i < 2$$

$$K_2: \quad x_1 \cdot 32 + x_2 \cdot 15 + x_3 \cdot 19 = 47 \qquad 0 \leq x_i < 2$$

$$K_3: \quad x_1 \cdot 21 + x_2 \cdot 13 + x_3 \cdot 3 = 34 \qquad 0 \leq x_i < 2$$

are similar, since K_2 is obtained from K_1 by multiplying its generators and target value by 7 (mod 101), K_3 is obtained from K_1 by multiplying its generators and target value by 33 (mod 101), and

$$101 = m > \max(K_1) = 19 + 31 + 46 = 96$$

$$101 = m > \max(K_2) = 32 + 15 + 19 = 66$$

$$101 = m > \max(K_3) = 21 + 13 + 3 = 37 \ .$$

□

Given two compact knapsack problems, we do not know how to check their similarity or how to compute the w and m parameters that prove their similarity in polynomial time. However, for our purposes this will not be a problem since we will always know these parameters from previous computations.

The most important property of the similarity relation is:

Theorem 3: If K and K' are similar, they have the same integral and bounded solutions.

Proof: Let $x_1,\ldots,x_n$ be integers satisfying the equation

$$\sum_{i=1}^{n} x_i a_i = b \quad .$$

Multiplying this equation times w and reducing it mod m, we get

$$\sum_{i=1}^{n} x_i (wa_i) = wb \qquad (\text{mod } m) \quad .$$

Since the x_i's are integers, we can replace wb and each wa_i by b' and a_i' which are their reduction mod m:

$$\sum_{i=1}^{n} x_i a_i' = b' \qquad (\text{mod } m) \quad .$$

If each x_i satisfies $0 \leq x_i \leq \ell-1$ and $m > \sum_{i=1}^{n} (\ell-1)a_i'$, both sides of the equation are integers in the range [0,m), and thus the equation must hold without the (mod m) clause:

$$\sum_{i=1}^{n} x_i a_i' = b \quad .$$

This proves that any integral and bounded solution of the original problem is a solution of the transformed problem, and by symmetry the two compact knapsacks have identical solutions. Note, however, that over the real numbers or over unbounded integers the two equations can have very different sets of solutions. Q.E.D.

The basic idea behind the new algorithm is quite simple: Given an n-generator knapsack problem K_1, we search for n-1 additional n-generator problems $K_2,\ldots,K_n$ which are all similar to K_1. These n problems form a system of n linear equations in n unknowns x_i, which can be easily solved over the rationals or the integers mod ℓ. If the generated system is non-singular and its unique solution is integral and properly bounded, we are done. In fact, this approach is advantageous whenever the rank (mod ℓ) of the system is larger than n/2, since the solution set of such a system contains less than $\ell^{n/2}$ points and their enumeration is faster than the use of the best previously published algorithm.

Example: The three equations in the previous example form a non-singular system over the rational numbers, whose unique solution is $x_1 = 1$, $x_2 = 1$, $x_3 = 0$. Instead of solving the equations over the rationals, we can reduce them mod 2:

$$x_1 \oplus x_2 = 0 \pmod 2$$

$$x_2 \oplus x_3 = 1 \pmod 2$$

$$x_1 \oplus x_2 \oplus x_3 = 0 \pmod 2$$

and solve this simplified system over GF(2). □

The formal analysis of the expected rank of generated systems is not easy. The set of modular multiples of a randomly chosen vector $(a_1,\ldots,a_n)$ form a lattice in the n-dimensional cube of side m, which is usually uniform and isotropic. Extensive experimentation has shown that when the original problem has only one solution (which is always the case in cryptographic knapsacks), the probability of n randomly chosen points in this lattice to span the n-dimensional space is very high. A partial result that supports this claim is:

Theorem 4: Let $(a_1,\ldots,a_n)$ be an integral point and let m be a modulus which is greater than all the a_i's. Then for a randomly chosen integral w in [0,m), the probability of $(a_1,\ldots,a_n)$ and $(wa_1 \pmod m,\ldots,wa_n \pmod m)$ to be linearly dependent over the reals is

$$\gcd(a_1,\ldots,a_n)/\max(a_1,\ldots,a_n) \quad .$$

Proof (sketch): Without loss of generality, we can assume that $a_1 = \max(a_1,\ldots,a_n)$. Let $P_1,\ldots,P_{a_1}$ be the points on the continuous line segment

$$(ta_1,\ldots,ta_n) \qquad 0 \leq t < m$$

defined by

$$P_i: \quad t = (i-1)m/a_1 \quad .$$

For every point $(ta_1,\ldots,ta_n)$ between P_i and P_{i+1}, the point $(ta_1 \pmod m,\ldots, ta_n \pmod m)$ is linearly dependent on $(a_1,\ldots,a_n)$ over the reals if and only if the point P_i is congruent to $(0,\ldots,0)$ modulo m. It is easy to show that exactly $\gcd(a_1,\ldots,a_n)$ of the P_i points have this property, and thus

the probability of linear dependence for randomly chosen t is $\gcd(a_1,\ldots, a_n)/a_1$. Since the points with integral values of t are equally distributed among the various (P_i,P_{i+1}) segments, this probability applies to them as well. Q.E.D.

Corollary: If $\gcd(a_1,\ldots,a_n) = 1$ and the a_i's are sufficiently large, it is extremely unlikely that a randomly chosen transformed equation will be linearly dependent on the original equation.

We were unable to extend this proof technique to the case of n similar equations, but our numerical experiments indicate that the relative frequency of singular systems is similar to that expected from n x n matrices whose entries are chosen at random from [0,m). When m is large, this relative frequency is extremely small and does not have a practical significance in cryptanalysis.

4. The Algorithm

The main problem in applying the method outlined in the previous section is how to choose the m and w parameters that transform the original problem K into a similar problem K'. When m is a fixed prime > max(K) and w varies between 1 and m-1, each generator a'_i in K' (i.e., each $w\cdot a_i \pmod m$) becomes uniformly distributed (in a pseudo-random sense) between 1 and m-1. To satisfy m > max(K'), all these random variables must be simultaneously small. Assuming that their distributions are independent, the probability of this event can be estimated as follows:

Lemma 5: Given n independent and uniformly distributed random variables $a_i' \in [0,m)$, the probability P that $m > \sum_{i=1}^{n} (\ell-1)\, a_i'$ is $O((\ell n/e)^{-n})$.

Proof: The probability of n independent and uniformly distributed random variables $r_i \in [0,1)$ to satisfy $\sum_{i=1}^{n} r_i < d \leq 1$ is equal to the volume cut from the n-dimensional unit cube by the hyperplane $\sum_{i=1}^{n} r_i = d$, which is $d^n/n!$. By scaling up the range of the r_i's to $[0,m)$ and using the bound $d = m/(\ell-1)$, we get $P = 1/(\ell-1)^n \cdot n!$. By Stirling's formula, this probability is $O((\ell n/e)^{-n})$. Q.E.D.

Corollary: The expected number of useful multipliers w is $O(m\cdot(\ell n/e)^{-n})$, and this value is larger than 1 whenever m has more than $O(n \log(\ell n/e))$ bits.

Example: A knapsack problem with ten generators and twenty bit coefficients is likely to have over 2^{80} useful multipliers when the modulus is 300 bits long. However, a simple trial-and-error is not likely to find them, since they are scattered in $[0,2^{300})$ with a relative frequency of less than 2^{-220}. In fact, for any $n \geq 3$ the $O((\ell n/e)^{-n})$ probability of success is even lower than the $O(\ell^{-n})$ probability of guessing the correct x_i solution of the original knapsack problem!

As far as we know, there are no efficient number-theoretic algorithms for the simultaneous minimization (under modular multiplication)of three or more natural numbers. The algorithm presented in this section is based on combinatorial ideas, and it should be viewed as a first attempt at solving this problem. Better algorithms (based on other approaches) undoubtedly exist, and research in this direction is still at a preliminary stage.

Our algorithm is described in terms of a free parameter s, whose exact value will be determined later. It attempts to minimize the various generators in n successive stages. At each stage $1 \leq k \leq n$, it computes a set of s "independent" multipliers $w_1^k, \ldots, w_s^k$ each one of which makes the first k generators small under modular multiplication:

$$\forall\ 1 \leq i \leq k \quad \forall\ 1 \leq j \leq s \ , \ w_j^k a_i \pmod m \text{ is small.}$$

The final s multipliers $w_1^n, \ldots, w_s^n$ have the desired property with respect to all the a_i generators.

An informal description of the algorithm is:

<u>k=0</u> (initialization): Choose a sufficiently large prime modulus m and s random numbers $w_1^0, \ldots, w_s^0$ in [0,m).

<u>1≤k≤n</u> (iteration) : Form the set U of all the 2^s sums of subsets of the s numbers w_j^{k-1}. The new multipliers w_j^k are defined as the s elements of U-{0} that makes a_k smallest under modular multiplication (regardless of what they do to the other generators).

Appealing once more to the pseudo-random behaviour of modular multiplication, we can show:

Theorem 6: For all $1 \leq i \leq n$, $1 \leq j \leq s$ and $1 \leq k \leq n$, the expected value of $w_j^k\ a_i \pmod m$ is

$$\begin{cases} m/2 & \text{when } k<i \\ mj/2^s & \text{when } k=i \\ (m/2^s)(s/2)^{k-i+1} & \text{when } k>i \end{cases} .$$

Proof: The value of the i^{th} generator does not affect the choice of the multipliers at stages k=1,...i-1, and thus $w_j^k\ a_i \pmod m$ fluctuates randomly in [0,m) and its expected value is m/2. At stage k=i, $w_j^k\ a_i \pmod m$ is chosen as the j^{th} smallest element in a pseudo-random set of 2^s points in [0,m), and thus its expected value is $mj/2^s$.

At stage k=i+1, $w_j^{i+1}\ a_i \pmod m$ is by definition the sum of some subset of the s numbers $w_1^i\ a_i \pmod m, \ldots, w_s^i\ a_i \pmod m$, and thus its expected size is approximately

$$1/2 \sum_{j=1}^{s} (mj/2^s) \approx (m/2^s)(s/2)^2 .$$

At any latter stage, the subset addition increases this value by a factor of s/2, and thus at stage k > i the expected value is $(m/2^s)(s/2)^{k-i+1}$. Q.E.D.

The key to the efficiency of the algorithm is the sawtooth behaviour of the expected value of each $w_j^k\ a_i \pmod m$ as a function of the stage k: it drops sharply at stage k=i but increases only moderately at later stages (when the other generators are handled).

Example: Let m be a 300 bit modulus, let n be 4 and let s be 32. Then the expected size (in bits) of $w_1^k a_i \pmod m$ as a function of the stage k and the generator i is:

	i=1	i=2	i=3	i=4
k=1	268	300	300	300
k=2	276	268	300	300
k=3	280	276	268	300
k=4	284	280	276	268 □

For any multiplier w_j^n computed at the last stage of the algorighm, the expected value of the sum of the transformed generators, $\sum_{i=1}^{n} w_j^n a_i \pmod m$, is at most $\sum_{i=1}^{n} (m/2^s)(s/2)^{n-i+1} \approx (m/2^s)(s/2)^n$.

To satisfy the condition $m > \max(K')$, the parameter s must satisfy

$$m > (\ell-1)(m/2^s)(s/2)^n \quad .$$

By taking the logarithm of both sides and rearranging the terms, we get the basic inequality

$$s > n \log s + \log(\ell-1) - n \quad .$$

For any given n and ℓ, we can use numeric methods to solve this implicit inequality to find the smallest s that satisfies it. To estimate the asymptotic growth rate of s, we can consider the single-parameter set of problems in which n is both the number of generators and the length of each coefficient. Since $\log(\ell-1) = n$, the inequality simplifies to $s > n \log s$. The value $s = n \log n$ does not satisfy the inequality, but any ε-improvement in it of the form $s = (1+\varepsilon)\, n \log n$ satisfies it for all sufficiently

large values of n:

$$n \log n + \varepsilon\, n \log n = s > n \log s = n \log n + n \operatorname{loglog} n + n \log (1+\varepsilon).$$

Consequently, the asymptotic behaviour of s in this case if $O(n \log n)$.

A straightforward implementation of the iteration stages requires $O(2^s)$ operations per stage. A better implementation can be obtained by using the Schroeppel-Shamir [1979] algorithm in order to find the smallest sums of subsets (mod m) in $O(2^{s/2})$ time and $O(2^{s/4})$ space. Further optimizations can eliminate the first two stages ($w_1^2,\ldots,w_s^2$ can be directly computed in polynomial time by the "best approximations" algorithm of number theory), and reduce the complexity of the remaining stages by using a decreasing sequence of s values (the final sizes of most of the transformed generators are unnecessarily low - it suffices to make all these sizes roughly equal).

A problem with n generators and n bits per coefficient contains a total of n^2 unknown bits, and thus the best previously published algorithm for solving it requires

$$O(2^{n^2/2})$$

operations. By using the $o(2^{s/2})$ implementation of the new algorithm with $s = n \log n$ we can solve the problem in $O(2^{(n \log n)/2})$ operations, which is a very substantial saving even for moderate values of n.

In practical applications, s must be limited to 80 or less in order to make the $O(2^{s/2})$ time complexity feasible. When ℓ is small and s=80, the inequality

$$s > n \log s + \log (\ell-1) - n$$

yields $n \leq 15$ as the practical upper limit on the number of generators our algorithm can handle. When n is slightly decreased, ℓ can be considerably increased since it occurs only within a log. For example, when n=10 s=60 and the improved algorithm is used, ℓ can be as large as one million. The total number of unknown x_i bits in such a 10-generator knapsack problem is 200, and even with the best previous algorithm and an ultimate 1 picosecond machine, its solution takes longer than the age of the universe. The new algorithm, on the other hand, can solve it in less than 20 minutes on a conventional 1 microsecond machine.

5. Consequences of the Algorithm

The analysis of the expected behaviour of our algorithm in the previous section was based on certain plausible but unproved assumptions about the behaviour of the generators under modular multiplication. So far we were unable to make this analysis rigorous, and thus all the consequences of the algorithm mentioned in this section are somewhat speculative.

For any fixed $m > 3$, the asymptotic complexity of our algorithm (when the sizes of a_i and x_i grow to infinity) is non-polynomial, and thus it does not solve the basic theoretical question of whether n-generator knapsack problems are in P, NP-complete, or somewhere in between. However, the efficiency of the new algorithm for small values of n makes them an unacceptable security risk in cryptographic applications, and thus a large key size seems to be an inherent feautre of knapsack-based cryptosystems.

One of the main cryptanalytic advantages of the new algorithm is that once the appropriate multipliers and moduli are found (by preprocessing the published generators), the decryption of actual cyphertexts b becomes extremely fast -- all the cryptanalyst has to do is to compute a vector of n modular multiples of b and to solve the resultant system of linear equations. This behaviour can justify weeks or even months of preprocessing time, and compares favorably with other knapsack-solving algorithms in which every decryption attempt is independently time consuming.

The algorithm strongly indicates that (unintentional) trapdoors are built into most uniquely decodable knapsack systems, since the knowledge of the n modular multipliers makes them solvable in polynomial time. From the complexity-theoretic point of view, these multipliers form short and easily checkable proofs both for the existence and for the non-existence of solutions - a phenomenon that characterizes problems in $\Delta = NP \cap co\text{-}NP$. Furthermore, the uniformity of these proofs for all the knapsack problems represented by the same generators indicates that the circuit complexity of these collections of problems is polynomial.

Another major cryptographic conclusion is related to the security of the Merkle-Hellman cryptosystem. To decode a cyphertext in this system, the cryptanalyst can either solve the knapsack problem or expose the secret trapdoor embedded in the public key. The NP-completeness of knapsack problems is some indication that the first type of attack is not likely to succeed, but the difficulty of the second type of attack is an open problem

about which almost nothing is currently known. The trapdoor suggested by Merkle and Hellman is based on the repeated transformation of one set of generators into a similar set of generators via modular multiplications (whose m and w parameters are kept secret). When the number of scrambling stages is large, the resultant generators become randomly-looking numbers with no observable structure in them. The main (and probably the only) cryptanalytic attack that can expose the initial set of generators is to undo the similarity transformations one at a time in reverse order. However, any general purpose algorithm for finding the appropriate m and w parameters was shown in this paper to lead to an efficient knapsack-solving algorithm, and thus the detection of the secret trapdoor is not likely to be any easier than the direct solution of the original knapsack problem.

BIBLIOGRAPHY

1. DES [1976], "Data Encryption Standard", FIPS, Pub. 46.

2. R. Kannan and A. Shamir [1980], "Three Generator Knapsacks Are Polynomially Solvable", in preparation.

3. R. Merkle and M. Hellman [1978], "Hiding Information and Receipts in Trapdoor Knapsacks", IEEE Trans. Information Theory, September, 1978.

4. R. Rivest, A. Shamir and L. Adleman [1978], "A Method For Obtaining Digital Signatures and Public-Key Cryptosystems", CACM, February, 1978.

5. R. Schroeppel and A. Shamir, "A $T \cdot S^2 = O(2^n)$ Time/Space Tradeoff for Certain NP-Complete Problems", 20-th Symposium in Foundations of Computer Science, October, 1979.

6. A Shamir [1979], "On the Cryptocomplexity of Knapsack Systems", 11-th Symposium on Theory of Computing, May 1979.