BREAKING ITERATED KNAPSACKS*

Ernest F. Brickell

Sandia National Laboratories
Albuquerque, New Mexico 87185

ABSTRACT

This paper presents an outline of an attack that we have used successfully to break iterated knapsacks. Although we do not provide a proof that the attack almost always works, we do provide some heuristic arguments. We also give a detailed description of the examples we have broken.

INTRODUCTION

R. Merkle and M. Hellman [10] devised the first knapsack based cryptosystem. In this paper we will deal only with the Merkle-Hellman knapsack, although similar techniques will work on Graham-Shamir iterated knapsacks also. We will say that a set of positive integers $a_1,\ldots,a_n$ is an ordered Y-times iterated knapsack if there exists a superincreasing sequence $s_1,\ldots,s_n$ (i.e., $s_i > \sum_{j<i} s_j$), and integers W_j^*, M_j, $a_{j,i}$ for $1 \leq j \leq Y$ and $1 \leq i \leq n$ such that

$$a_{0,i} = s_i \qquad \text{for} \quad 1 \leq i \leq n \ , \tag{1.1}$$

$$a_{j,i} = a_{j-1,i} W_j^* \bmod M_j \qquad \text{for} \quad 1 \leq j \leq Y \quad \text{and} \quad 1 \leq i \leq n \ , \tag{1.2}$$

$$M_j > \sum_{i=1}^{n} a_{j-1,i} \qquad \text{for} \quad 1 \leq j \leq Y \ , \tag{1.3}$$

$$a_i = a_{Y,i} \qquad \text{for} \quad 1 \leq i \leq n \ . \tag{1.4}$$

---

* This work performed at Sandia National Laboratories supported by the U. S. Department of Energy under Contract Number DE-AC04-76DP00789.

G.R. Blakley and D. Chaum (Eds.): Advances in Cryptology - CRYPTO '84, LNCS 196, pp. 342-358, 1985.

We use a = b mod M to mean that a is the least nonnegative residue. Let $W_j = W_j^{*-1} \bmod M_j$ for $1 \leq j \leq Y$. We will define integers $k_{j,i}$ for $1 \leq j \leq Y$ and $1 \leq i \leq n$ to be those integers satisfying

$$a_{j,i}W_j - k_{j,i}M_j = a_{j-1,i} \quad \text{for} \quad 1 \leq j \leq Y \quad \text{and} \quad 1 \leq i \leq n . \qquad (1.5)$$

We will say that a set of positive integers $a_1,\ldots,a_n$ is an unordered Y-times iterated knapsack if there is some permutation of the $a_1,\ldots,a_n$ that is an ordered Y-times iterated knapsack.

To cryptanalyze this system, one must solve the knapsack (or subset sum) problem for the integers $a_1,\ldots,a_n$ and any subset sum s. That is given s, one must find a 0-1 vector$(\alpha_1,\ldots,\alpha_n)$ such that

$$s = \sum_{i=1}^{n} \alpha_i a_i$$

if such a 0-1 vector exists.

In fact it is sufficient to be able to solve the knapsack problem for a 0-1 vector $(\alpha_1,\ldots,\alpha_n)$ which has $\leq \frac{1}{2}$ n ones, because one can consider the subset sum s and the subset sum $\sum_{i=1}^{n} a_i - s$.

In this paper we will describe an algorithm for breaking Y-times iterated knapsacks in polynomial time. We have successfully demonstrated this algorithm on examples with n = 100 and Y = 5, 10, and 20.

The first attack on knapsack based cryptosystems was found by Adi Shamir [13]. He discovered an algorithm for cryptanalyzing the single iteration Merkle-Hellman knapsack in polynomial time. Len Adleman [1] found a method for breaking the single iteration Graham-Shamir knapsack in polynomial time. His attack used the Lenstra, Lenstra, Lovász (L^3) lattice basis reduction algorithm [9], which could also be used to greatly speed up the attack on the Merkle-Hellman knapsack. Len Adleman [1] and Jeff Lagarias [7] have both developed attacks for the doubly iterated Merkle-Hellman knapsack. Adleman [1] also proposed an attack on multiply iterated knapsacks, but Brickell, Lagarias, and Odlyzko [3] showed that there were some problems with it. However the lattice

that we use in our new attack is the same as the first lattice that Adleman used in his attack.

By using a different approach, E. Brickell [2] and J. Lagarias and A. Odlyzko [8] have developed algorithms which will cryptanalyze MH or GS cryptosystems in polynomial time if the information rate is low enough. The information rate of a knapsack based cryptosystem is roughly $n/\log_2(\max a_{Y,i})$. It is not known exactly how low the information rate must be for these algorithms to work, but the information rate must at least be less than .645 before these methods can possibly be successful. Since each iteration lowers the information rate, these methods will break a knapsack cryptosystem if it has been iterated too many times.

THE USE OF LATTICE REDUCTION

The L^3 lattice basis reduction algorithm [9] is used in all of the attacks on knapsack based cryptosystems. A set of points, L, in $\mathbf{R}^n$ is a lattice if there exists a set of independent vectors $v_1,\ldots,v_m$ such that

$$L = \{z_1v_1 + \ldots + z_nv_n : z_i \in \mathbf{Z} \quad \text{for} \quad 1 \leq i \leq m\} \ .$$

Such a set of vectors, $v_1,\ldots,v_n$ is called a basis for the lattice. The L^3 algorithm finds a reduced (or short) basis for the lattice. We will not give a precise definition of a reduced basis. We will only note that in a reduced basis, all of the basis vectors are relatively short in the Euclidean norm.

Let $a_i,\ldots,a_n$ be an unordered Y-times iterated knapsack. Let m be an integer $< n$. Later we will put conditions on how small m can be. Let L be the m-dimensional lattice generated by the following vectors.

$$\begin{aligned}
\mathbf{b}_1 &= (a_2, a_3, a_4, \ldots, a_m, n^{-1}) \\
\mathbf{b}_2 &= (a_1, 0, 0, \ldots, 0, 0) \\
\mathbf{b}_3 &= (0, a_1, 0, \ldots, 0, 0) \\
&\vdots \\
\mathbf{b}_m &= (0, 0, 0, \ldots, a_1, 0)
\end{aligned}$$

The first step in the algorithm is to find a reduced basis for L by using the L^3 algorithm. We now must go into a rather detailed discussion to describe the vectors we expect to see in the reduced basis.

Let D be the smallest integer such that $2^D > \max\{a_1,\ldots,a_n\}$. Then M_Y, W_Y, $a_1,\ldots,a_n$ should be $O(2^D)$. Since $M_j > \sum_{i=1}^{n} a_{j-1,i}$, we will assume that M_{Y-k}, W_{Y-k}, $a_{Y-k,i}$ are $O(2^{D-k \log n})$.

The norm of the vector

$$w = -a_1b_1 + \sum_{i=2}^{m} a_ib_i = (0,0,\ldots,0,\frac{-a_1}{n})$$

is $O(2^{D-\log n})$. If we take $m > \frac{D}{\log n - 1}$, then w would probably be the shortest vector in L if the integers $a_1,\ldots,a_n$ were chosen from the uniform distribution on $(0,2^D)$. However, because $a_1,\ldots,a_n$ are an unordered Y-times iterated knapsack, there are other vectors in L that are about the same length as w. For this discussion we will refer to any vector in L with norm $\leq O(2^{D-\log n})$ as a short vector.

Adleman found that there was a short vector that was a result of the last iteration. From (1.5)

$$a_iW_Y - k_{Y,i}M_Y = a_{Y-1,i} \quad .$$

Divide by M_Ya_i

$$\frac{W_Y}{M_Y} - \frac{k_{Y,i}}{a_i} = \frac{1}{M_Y}\left(\frac{a_{Y-1,i}}{a_i}\right) .$$

Subtract equation i from equation 1

$$\frac{k_{Y,i}}{a_i} - \frac{k_{Y,1}}{a_1} = \frac{1}{M_Y}\left(\frac{a_{Y-1,1}}{a_1} - \frac{a_{Y-1,i}}{a_i}\right) .$$

Multiply by a_1a_i

$$k_{Y,i}a_1 - k_{Y,1}a_i = \frac{1}{M_Y}\left(a_ia_{Y-1,1} - a_1a_{Y-1,i}\right) \quad . \qquad (2.1)$$

$$k_{Y,i}a_1 - k_{Y,1}a_i = 0(2^{D-\log n}) \quad .$$

In the vector

$$x = -k_{Y,1}b_1 + \sum_{i=2}^{m} k_{Y,i}b_i$$

each coordinate is $0(2^{D-\log n})$. So

$$|x| = 0(2^{D-\log n}) \quad .$$

Lagarias [7] found a description for short vectors in L that are the result of many iterations. Let t be an integer with $1 < t < Y$. By applying equation (1.5) repeatedly we get

$$a_i W_Y \cdots W_t - k_{Y,k} M_Y W_{Y-1} \cdots W_t - k_{Y-1,i} M_{Y-1} W_{Y-2} \cdots W_t - \cdots - k_{t,i} M_t = a_{t-1,i} \cdot$$

We divide by $a_i M_Y W_{Y-1} W_{Y-2} \ldots W_t$ to get

$$\frac{W_Y}{M_Y} - \frac{1}{a_i}\left[k_{Y,i} + k_{Y-1,i}\frac{M_{Y-1}}{M_Y W_{Y-1}} + k_{Y-2,i}\frac{M_{Y-2}}{M_Y W_{Y-1} W_{Y-2}} + \ldots + k_{t,i}\frac{M_t}{M_Y W_{Y-1} W_{Y-2} \ldots W_t}\right] = \frac{a_{t-1,i}}{a_i M_Y W_{Y-1} W_{Y-2} \cdots W_t} \quad .$$

Using the theory of simultaneous Diophantine approximation, there exists many sets of integers $(r_1, \ldots, r_Y)$ such that for $1 < j < Y$

$$r_j = 0$$

or (2.2)

$$\frac{r_j}{r_Y} \approx \frac{M_j}{M_Y W_{Y-1} \ldots W_j} \quad .$$

Each of these vectors will give rise to a short vector in the lattice. For if we let

$$h_i = \sum_{j=1}^{Y} k_{j,i} r_j - r_0 a_i \quad \text{for} \quad 1 < i < m \quad , \quad (2.3)$$

then the vector

$$-hb_1 + \sum_{i=2}^{m} h_i b_i \tag{2.4}$$

will be a short vector in the lattice. We will call such a vector a desirable vector.

We would like to have Y desirable vectors in the reduced basis. Let π be a permutation on the first n integers such that $a_{\pi(1)},\ldots,a_{\pi(n)}$ is an ordered Y-times iterated knapsack. We can pick any m of the weights $a_1,\ldots,a_n$. If we pick these weights so that we do not pick $a_{\pi(n)}, a_{\pi(n-1)}, a_{\pi(n-2)}$, then we expect to get Y desirable vectors in the reduced basis.

The reason that we do not want $a_{\pi(n)}$, $a_{\pi(n-1)}$, or $a_{\pi(n-2)}$ is because the Y short vectors exist because $\frac{a_{j-1,i}}{M_j}$ is small for $j=1,\ldots,Y$ and all i in the weights that we pick. But

$$\frac{a_{0,\pi(n)}}{M_1} \approx \frac{1}{2} \qquad \text{and} \qquad \frac{a_{0,\pi(n-2)}}{M_1} \approx \frac{1}{8}$$

and these ratios are not small enough. In the examples that we tried, if $a_{\pi(n)}$, $a_{\pi(n-1)}$, or $a_{\pi(-2)}$ were in the chosen set, then we only had Y-1 short vectors. But if these three weights were not in the chosen set and $a_{\pi(n-3)}$ was, then we still had Y short vectors.

This condition on the way we must choose an m-set will not seriously affect the running time. The probability of picking a good m-set is

$$P = \frac{\binom{n-3}{m}}{\binom{n}{m}} \approx \left(\frac{n-m}{n}\right)^3 .$$

Thus we expect to make about $\left(\frac{n}{n-m}\right)^3$ choices to get a good m-set.

The information that we obtain from these short vectors are the coefficients h_i which we can recover from (2.4). We only know h_i for $1 \leq i \leq m$, and these h_i satisfy

$$\left|\frac{h_1}{a_1} - \frac{h_i}{a_i}\right| < \frac{1}{n}\left(\frac{1}{M_Y}\right) \qquad \text{for} \qquad 1 < i < m \quad .$$

(We haven't proven this, but it has been true in all of our examples.) We can define h_i for $i > m$ by

$$h_i = \left[\frac{h_1}{a_1} a_i\right] \qquad \text{for} \qquad m < i < n \quad ,$$

where $[x]$ is the closest integer to x. Because the h_i, for $1 < i < m$, have the special form of (2.3) and the r_j have the special form of (2.2),

$$\left|\frac{h_1}{a_1} - \frac{h_i}{a_i}\right| < \frac{1}{n}\left(\frac{1}{M_Y}\right) \qquad \text{for} \qquad 1 < i < n \quad .$$

To get the full impact of the power of these vectors $(h_1,\ldots,h_n)$, let s be a subset sum such that

$$s = \sum_{i=1}^{n} \alpha_i a_i$$

where $(\alpha_1,\ldots,\alpha_n)$ is a 0-1 vector with $< \frac{1}{2} n$ ones. If we then define

$$t = \left[\frac{h_1}{a_1} s\right] \quad , \tag{2.5}$$

then

$$t = \left[\frac{h_1}{a_1} \sum_{i=1}^{n} \alpha_i a_i\right] = \left[\sum_{i=1}^{n} \alpha_i \frac{h_1}{a_1} a_i\right]$$

but

$$\frac{h_1}{a_1} a_i = h_i + \varepsilon_i$$

where

$$|\varepsilon_i| < \frac{1}{n} \quad .$$

So

$$t = \sum_{i=1}^{n} \alpha_i h_i + \sum_{i=1}^{n} \alpha_i \varepsilon_i \quad .$$

But

$$\sum_{i=1}^{n} \alpha_i \epsilon_i < \frac{1}{2} \quad .$$

So

$$t = \sum_{i=1}^{n} \alpha_i h_i \quad .$$

Thus we can find $\sum_{i=1}^{n} \alpha_i h_i$ without knowing what the α_i are.

THE USE OF DESIRABLE VECTORS

In this section we will show how the short vectors in the reduced basis can be used to recover a superincreasing sequence of length n-Y-ε. We will comment about ε later, but for now, assume that ε is a small integer. In this section we will assume that $a_1,\ldots,a_n$ is an ordered Y-times iterated knapsack. Suppose that we have reduced the lattice and we have Y desirable vectors of the form

$$w_\ell = -h_{\ell,1}b_1 + \sum_{i=2}^{n} h_{\ell,i}b_i \qquad 1 \leq \ell \leq Y$$

such that

$$h_{\ell,i} = \sum_{j=1}^{Y} r_{\ell,j}k_{j,i} - r_{\ell,0}a_i \qquad \begin{matrix} 1 \leq \ell \leq Y \\ 1 \leq i \leq n \end{matrix}$$

and $r_{\ell,j} = 0$ or $\frac{r_{\ell,j}}{r_{\ell,Y}}$ is a good approximation to $\frac{M_j}{M_Y W_{Y-1}\cdots W_j}$.

For $1 \leq i \leq n$, let

$$H_i = \begin{pmatrix} a_1 & \cdots & a_Y & a_i \\ h_{1,1} & \cdots & h_{1,Y} & h_{1,i} \\ & \vdots & & \\ h_{Y,1} & \cdots & h_{Y,Y} & h_{Y,i} \end{pmatrix}$$

$$K_i = \begin{pmatrix} a_1 & \cdots & a_Y & a_i \\ k_{1,1} & \cdots & k_{1,Y} & k_{1,i} \\ & \vdots & & \\ k_{Y,1} & \cdots & k_{Y,Y} & k_{Y,i} \end{pmatrix} \quad .$$

Let

$$R = \begin{pmatrix} 1 & 0 & & 0 \\ -r_{1,0} & r_{1,1} & \cdots & r_{1,Y} \\ & & \vdots & \\ -r_{Y,0} & r_{Y,1} & \cdots & r_{Y,Y} \end{pmatrix} .$$

Then

$$RK_i = H_i \quad .$$

The following theorem gives us a way of computing the determinant of K_i.

Theorem:

Let $a_1,\ldots,a_n$ be an unordered Y-times iterated knapsack. Let $1 \leq j \leq Y$. Then

$$M_1 \cdots M_Y \begin{vmatrix} a_{j,1} & \cdots & a_{j,j+1} \\ k_{1,1} & \cdots & k_{1,j+1} \\ & \vdots & \\ k_{j,1} & \cdots & k_{j,j+1} \end{vmatrix} = \begin{vmatrix} a_{0,1} & \cdots & a_{0,j+1} \\ a_{1,1} & \cdots & a_{1,j+1} \\ & \vdots & \\ a_{j,1} & \cdots & a_{j,j+1} \end{vmatrix}$$

Proof:

The case j=1 follows immediately from (2.1). We will complete the proof by induction. Since any permutation of $a_1,\ldots,a_n$ is an unordered Y-times iterated knapsack, we can apply the inductive hypothesis to any permutation of $a_1,\ldots,a_n$. So we get equality in the following statement by expanding about the last row and seeing that the cofactors are all equal.

$$\begin{vmatrix} a_{0,1} & \cdots & a_{0,j+1} \\ & \vdots & \\ a_{j-1,1} & \cdots & a_{j-1,j+1} \\ a_{j,1} & \cdots & a_{j,j+1} \end{vmatrix} = M_1 \cdots M_{j-1} \begin{vmatrix} a_{j-1,1} & \cdots & a_{j-1,j+1} \\ k_{1,1} & \cdots & k_{1,j+1} \\ & \vdots & \\ k_{j-1,1} & \cdots & k_{j-1,j+1} \\ a_{j,1} & \cdots & a_{j,j+1} \end{vmatrix}$$

$$= -M_1 \cdots M_{j-1} \begin{vmatrix} a_{j,1} & \cdots & a_{j,j+1} \\ k_{1,1} & \cdots & k_{1,j+1} \\ & \vdots & \\ k_{j-1,1} & \cdots & k_{j-1,j+1} \\ a_{j-1,1} & \cdots & a_{j-1,j+1} \end{vmatrix}$$

$$= -M_1 \cdots M_{j-1} \begin{vmatrix} a_{j,1} & \cdots & a_{j,j+1} \\ k_{1,1} & \cdots & k_{1,j+1} \\ & \vdots & \\ k_{j-1,1} & \cdots & k_{j-1,j+1} \\ a_{j,1}W_j - k_{j,1}M_j & \cdots & a_{j,j+1}W_j - k_{j,j+1}M_j \end{vmatrix}$$

$$= M_1 \cdots M_j \begin{vmatrix} a_{j,1} & \cdots & a_{j,j+1} \\ k_{1,1} & \cdots & k_{1,j+1} \\ & \vdots & \\ k_{j-1,1} & \cdots & k_{j-1,j+1} \\ k_{j,1} & \cdots & k_{j,j+1} \end{vmatrix} .$$

Once again, let us take $a_1,\ldots,a_n$ to be an ordered Y-times iterated knapsack. For $1 < i < n$, let

$$A_i = \begin{pmatrix} s_1 & \cdots & s_Y & s_i \\ a_{1,1} & \cdots & a_{1,Y} & a_{1,i} \\ & \vdots & & \\ a_{Y,1} & \cdots & a_{Y,Y} & a_{Y,i} \end{pmatrix}$$

and

$$x_i = |\det(A_i)| \quad . \tag{3.1}$$

For $i < Y$, $x_i = 0$ since the last column in A_i is identical to another column. However in all of the examples we have run, the sequence x_i has been superincreasing for $i > Y + \varepsilon$, where $\varepsilon < 2$. Furthermore, since the cryptanalyst can compute $\det(H_i)$, he can compute the sequence cx_i where $c = \det(R)/M_1 \ldots M_Y$.

Let us give a heuristic argument for why we might expect x_i to be superincreasing for $i > Y + \varepsilon$. Expand A_i about the first row to get

$$\det(A_i) = s_1 A^i_{1,1} + \ldots + s_Y A^i_{1,Y} + s_i A^i_{1,Y+1} \tag{3.2}$$

where $A^i_{l,j}$ is the l,j cofactor of A_i. Let $B^i_{l,j}$ be the submatrix of A_i formed by deleting the first row and j^{th} column of A_i. So

$$|A^i_{l,j}| = |\det B^i_{l,j}| \quad .$$

Each entry in the l^{th} row of $B^i_{l,j}$ is less than M_l. We expect that the distribution of the values of the $A^i_{l,j}$ to be independent of i and j since the distribution of the entries in $B^i_{l,j}$ is independent of i and j. But the sequence s_i is superincreasing, so there should be an ε such that for $i > Y + \varepsilon$, the expression for $\det(A_i)$ in (3.2) is dominated by the last term.

RECOVERING THE ORDER

In the previous section, we showed that if a cryptanalyst had an ordered Y-times iterated knapsack, then he could recover a superincreasing sequence. However, a cryptanalyst will usually be faced with an unordered Y-times iterated knapsack. So in this section we will show how we can apply techniques similar to those of the last section to find the order in an unordered Y-times iterated knapsack.

Let $a_1,\ldots,a_n$ be an unordered Y-times iterated knapsack. Let π be a permutation such that $a_{\pi(1)},\ldots,a_{\pi(n)}$ is an ordered Y-times iterated knapsack, i.e., $a_{\pi(n)}$ corresponds to s_n, the largest element in the superincreasing sequence. We will present a method for finding $\pi(i)$ for $i > Y + \varepsilon$, for ε as described in the previous section.

Suppose we have already found $\pi(n),\ldots \pi(n-j)$ for some j, $-1 \leq j \leq n-Y-\varepsilon-2$. (If $j = -1$, we haven't found anything yet.) Let $I_j = \{1,\ldots,n\}/\{\pi(n),\ldots,\pi(n-j)\}$. We will pick many (Y+1)-subsets of I_j. For each subset $j_0,\ldots,j_Y \subseteq I_j$ that we pick, we will form a determinant.

$$X_{j_0,\ldots,j_Y} = \begin{vmatrix} a_{j_0} & \cdots & a_{j_Y} \\ h_{1,j_0} & \cdots & h_{1,j_Y} \\ & \vdots & \\ h_{Y,j_0} & \cdots & h_{Y,j_Y} \end{vmatrix} = c \begin{vmatrix} s_{j_0} & \cdots & s_{j_Y} \\ a_{1,j_0} & \cdots & a_{1,j_Y} \\ & \vdots & \\ a_{Y,j_0} & \cdots & a_{Y,j_Y} \end{vmatrix} \cdot \quad (4.1)$$

The reason we use these determinants is essentially the same as the reason we gave in the last section for expecting the x_i to be superincreasing. When we expand the right hand matrix in (4.1) about the first row, we expect the distribution on the values of the cofactors to be independent of the choice of $j_0,\ldots,j_Y$. So the value of $|X_{j_0,\ldots,j_Y}|$ should be dominated by the largest s_{j_i} in the first row.

After picking many (Y+1)-subsets of I_j, we will keep the subset $j_0,\ldots,j_Y$ that gives the smallest determinant $X_{j_0,\ldots,j_Y}$. We expect that the set $j_0,\ldots,j_Y$ does not contain any of $\pi(n-j-1),\ldots,\pi(n-j-\ell)$ for some value of ℓ which will depend on n, j, and Y. Thus we form the sequence

$$z_i = \begin{vmatrix} a_{j_1} & \cdots & a_{j_Y} & a_i \\ h_{1,j_1} & \cdots & h_{1,j_Y} & h_{1,i} \\ & \vdots & & \\ h_{Y,j_1} & \cdots & h_{Y,j_Y} & h_{Y,i} \end{vmatrix}$$

for $1 \leq i \leq n$. For some small λ, the sequence $z_{\pi(n-j-\ell+\lambda)},\ldots,z_{\pi(n)}$ should be superincreasing, and the other values of z_i should be less than $z_{\pi(n-j-\ell+\lambda)}$. So we should be able to identify $\pi(n-j-\ell+\lambda),\ldots,\pi(n-j-1)$ and also check to see if our choice of $\pi(n-j),\ldots,\pi(n)$ was correct. We will set $j \leftarrow j+\ell-\lambda$ and continue with this iterative process. We continue until j does not change.

This method for recovering the order worked much better in practice than we expected. In our examples, we always were able to iterate until j was less than Y+5. In other words we were able to find an ε-superincreasing sequence with $\varepsilon < 5$.

SOLVING FOR THE α_i

All of the previous analysis has used only the weights $a_1,\ldots,a_n$. In this section we show how to finish the cryptanalysis. That is, given s find a 0-1 vector α_i that has less than $\frac{1}{2}$ n ones such that

$$\sum_{i=1}^{n} \alpha_i a_i = s \quad ,$$

if such a vector exists. We will assume that we have integers $h_{j,i}$ for $1 \leq j \leq Y$ and $1 \leq i \leq n$ such that the sequence

$$x_i = \begin{vmatrix} a_1 & \cdots & a_Y & a_i \\ h_{1,1} & \cdots & h_{1,Y} & h_{1,i} \\ & \vdots & & \\ h_{Y,1} & \cdots & h_{Y,Y} & h_{Y,i} \end{vmatrix}$$

is superincreasing for $i > Y + \varepsilon$ and we can find t_j (from (2.5)) such that

$$t_j = \sum_{i=1}^{n} \alpha_i h_{j,i} \quad .$$

We will find the α_i in three steps.

Step 1: $i > Y + \varepsilon$

Let

$$t = \begin{vmatrix} a_1 & \cdots & a_Y & s \\ h_{1,1} & \cdots & h_{1,Y} & t_1 \\ & \vdots & & \\ h_{Y,1} & \cdots & h_{Y,Y} & t_Y \end{vmatrix}$$

then

$$t = \sum_{i=1}^{n} \alpha_i x_i \quad .$$

Since x_i is superincreasing for $i > T + \varepsilon$, we can easily find α_i for $i > Y + \varepsilon$.

Step 2: $Y < i \leq Y + \varepsilon$

To find α_i, we must solve a knapsack problem with about ε weights. We know α_i for $i > Y + \varepsilon$, and $x_i = 0$ for $i \leq Y$, so we can find

$$t' = \sum_{i=1}^{n} \alpha_i x_i - \sum_{i=Y+\varepsilon+1}^{n} \alpha_i X_i = \sum_{i=Y+1}^{Y+\varepsilon} \alpha_i X_i$$

So we hope that ε is small. We conjecture that ε is bounded by $O(\log n)$. In our examples, we always had $\varepsilon < 5$.

Step 3: $i < Y$

Since we now know α_i for $i > Y$, let

$$s' = s - \sum_{i=Y+1}^{n} \alpha_i a_i$$

$$t_j' = t_j - \sum_{i=Y+1}^{n} \alpha_i h_{j,i} \qquad 1 \leq j \leq Y \ .$$

Let

$$\mathbf{s} = (s', t_1', \ldots, t_Y')$$

and

$$Z = \begin{pmatrix} a_1 & \cdots & a_Y & a_{Y+1} \\ h_{1,1} & \cdots & h_{1,Y} & h_{1,Y+1} \\ & \vdots & & \\ h_{Y,1} & \cdots & h_{Y,Y} & h_{Y,Y+1} \end{pmatrix} .$$

The problem we are left with is to find a 0-1 vector $\mathbf{a} = (\alpha_1, \ldots, \alpha_{Y+1})$, if one exists, such that

$$Z\mathbf{a} = \mathbf{s} \ .$$

If $\det(Z) \neq 0$, then this problem is easily solved. In all of our examples, we have found that $\det(Z) \neq 0$. We can speed up this operation by computing Z^{-1} mod p where p is the smallest prime such that $\det(Z) \neq 0 \bmod p$. Then we are computing using integers less than p instead of integers of size $O(2^D)$.

RUNNING TIME

The worst case running time of the L^3 algorithm is $O(m^6 D^3)$. However in practice, the running time appears to be $O(mD^3)$. Also we

might have to make $O\left(\frac{n}{n-m}\right)^3$ choices of an m-set of weights before we get a good one. To find the order of the weights we must take O(n) determinants. Each determinant takes $O(Y^3)$ multiplications of integers of length D. So the running time for finding the order is $O(nY^2D^2)$. So the total running time on the first part of the algorithm is

$$0\ m\left(\frac{n}{n-m}\right)^3 D^3 + O(nY^3D^2) \quad .$$

The second part of the algorithm is solving for the α_i's after a cypher is received. The running time for this part is

$$O(n^2) + O(2^{\varepsilon/2}) + O(Y^2) \quad .$$

TESTS OF THE ALGORITHM

Table 1 summarizes the examples we have run to test the algorithm.

Type - refers to Merkle-Hellman or Graham-Shamir.

N - the number of weights.

R - the number of random bits used in constructing the superincreasing sequence. For MH knapsacks, all random bits are the low order bits. For GS knapsacks, half of the random bits are the low order bits, and the other half are the high order bits.

Y - the number of iterations.

D - the number of bits in the final (or public) weights.

m - the number of weights used in the lattice reduction.

Time - the running time in seconds of the L^3 algorithm on a Cray 1.

Con - $Time/(mD^3)$

Table 1.
Examples

Type	N	Y	R	D	m	Time	Con
GS	40	4	14	82	18	27.6	2.78E-6
MH	40	5	10	93	28	108.2	4.80E-6
MH	40	5	20	113	30	210.6	4.86E-6
MH	40	7	10	105	30	138.1	3.98E-6
MH	40	10	10	123	32	250.1	4.20E-6
MH	50	5	7	97	28	108.7	4.25E-6
GS	50	5	16	96	24	92.3	4.34E-6
MH	50	10	7	127	32	263.2	4.02E-6
GS	64	5	64	158	37	653.7	4.48E-6
GS	64	10	16	140	32	451.8	5.15E-6
GS	100	5	16	151	30	295.5	2.86E-6
GS	100	10	100	270	53	3542.0	3.40E-6
GS	100	20	20	260	52	3355.3	3.67E-6

CONCLUSION

By extrapolating from the data in Table 1, we project that an iterated knapsack with N = 1000, Y = 40, and R = 100 could be broken in 750 hours on the Cray. Since a knapsack of this size would require a 1.5 Mbit key, it is doubtful that a larger knapsack would ever be seriously considered.

This algorithm will also break Shamir's ultimate knapsack [14]. It appears that the algorithm can be modified to break the knapsack that Brickell presented at Crypto'83 and also the lexicographic knapsack scheme of Petit [12].

REFERENCES

1. L. Adleman, "On Breaking the Iterated Merkle-Hellman Public Key Cryptosystem," Advances in Cryptology, Proceedings of Crypto 82, Plenum Press 1983, 303-308.

2. E. F. Brickell, "Solving Low-Density Knapsacks," to appear in Advances in Cryptology, Proceedings of Crypto 83, Plenun Press.

3. E. F. Brickell, J. C. Lagarias and A. M. Odlyzko, Evaluation of Adleman's Attack on Multiply Iterated Knapsacks (Abstract), to appear in Advances in Cryptology, Proceedings of Crypto 83, Plenum Press.

4. E. F. Brickell and G. J. Simmons, "A Status Report on Knapsack Based Public Key Cryptosystems," Congressus Numerantium 37 (1983), 3-72.

5. Y. Desmedt, J. Vandewalle, R. Govaerts, "A Critical Analysis of the Security of Knapsack Public Key Algorithms, preprint.

6. J. C. Lagarias, "Simultaneous Diophantine Approximation of Rationals by Rationals, preprint.

7. J. C. Lagarias, "Knapsack Public Key Cryptosystems and Diophantine Approximation," to appear in Advances in Cryptology, Proceedings of Crypto 83, Plenum Press.

8. J. C. Lagarias and A. M. Odlyzko, "Solving Low-Density Subset Sum Problems, Proc. 24th Annual IEEE Symposium on Foundations of Computer Science (1983), 1-10.

9. A. K. Lenstra, H. W. Lenstra, Jr. and L. Lovasz, "Factoring Polynomials with Rational Coefficients," Math. Annalen, 261 (1982), 515-534.

10. R. Merkle and M. Hellman, "Hiding Information and Signatures in Trapdoor Knapsacks," IEEE Trans. Information Theory IT-24 (1978), 525-530.

11. A. M. Odlyzko, "Cryptanalytic Attacks on the Multiplicative Knapsack Cryptosystem and on Shamir's Fast Signature Scheme," preprint.

12. M. Petit, "Etude mathematique de certains systemes de cipherement: les sacs a dos," doctor's these, Universite de Rennes, France.

13. A. Shamir, "A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem," Proc. 23rd Annual Symposium on Foundations of Computer Science (1982), 145-152.

14. A. Shamir, "The strongest knapsack-based cryptosystem," presented at Crypto 82.