背包加密和 LLL 算法

1. 背包加密

背包加密基于背包问题，这是一个 NP 完全问题。

背包问题：给定重量为 w_0，w_1，⋯，w_{n-1} 的 n 个物品，能否选取其中若干放入背包使其重量之和等于给定的重量和 S，即能否找到找到 $a_i \in \{0,1\}, 0 \le i \le n-1$ 使得 $S = \sum_{i=0}^{n-1} a_i w_i$ 。

比如重量（62,93,26,52,166,48,91,141），子集和 S=302，有解（1,0,1,0,1,1,0,0）使得 62+26+166+48=303。

一般的背包问题是难解的，但超递增背包可解。超递增背包是指每个物品的重量超过其前面所有物品重量的总和，即 $w_i \ge \sum_{j=0}^{i-1} w_j$ ，比如（2,3,7,14,30,57,120,251），S=186，有解（1,0,1,0,0,1,1,0）使得 120+57+7+2=186。

假设物品数量为 n，子集和为 S，超递增背包求解过程如图所示。

$$
\begin{array}{l}
\textit{input} \quad w = (w_0, w_1, \cdots, w_{n-1}) \;\; \textit{that} \;\; w_i > \sum_{j=0}^{i-1} w_j \quad , \; S \\
\textit{output} \quad \mathbf{a} = (a_0, a_1, \cdots, a_{n-1}) \;\; \textit{or} \;\; \text{无解} \\
\textit{steps}: \\
\quad \mathbf{a} = (0, 0, \cdots, 0) \\
\quad \textit{for} \;\; i = n-1 \;\; \textit{to} \;\; 0 \\
\quad\quad \textit{if} \;\; (S \ge w_i) \;\; \textit{then} \\
\quad\quad\quad \{ \;\; S = S - w_i; \\
\quad\quad\quad\quad a_i = 1; \\
\quad\quad\quad \} \\
\quad \textit{if} \;\; (S == 0) \;\; \textit{then} \;\; \textit{return}(\mathbf{a}); \\
\quad\quad\quad\quad \textit{else} \;\; \textit{return}(\text{"无解"});
\end{array}
$$

背包加密体制：

1.公私钥对：构造一个超递增背包，将超递增背包转换成一个一般的背包作为公钥 pk，超递增背包和转换参数作为私钥 sk；

$$sk = (a_0, a_1, \cdots, a_{n-1})$$

$$\text{选择} m, n \text{使得} (m, n) = 1, n > \sum_{i=0}^{n-1} a_i$$

$$b_i = m a_i \bmod n, 0 \le i \le n-1$$

$$pk = (b_0, b_1, \cdots, b_{n-1})$$

2.加密：将明文比特序列作为一般背包问题的一个解，求出背包重量作为密文；

$$v=(v_0,v_1,\cdots,v_{n-1}),v_i\in\{0,1\}$$
$$c=\sum_{i=0}^{n-1}b_iv_i$$

3.解密：将密文转换成具有相同解的超递增背包的背包重量，求解超递增背包的解，即可解密。

$$c'=m^{-1}c \bmod n$$

举例：假设超递增背包（2,3,7,14,30,57,120,251）为私钥，选择 m=41 和素数 n=491（n 大于私钥中所有物品的重量和）来计算公钥，

$$2\times 41 \bmod 491=82$$
$$3\times 41 \bmod 491=123$$
$$7\times 41 \bmod 491=287$$
$$14\times 41 \bmod 491=83$$
$$30\times 41 \bmod 491=248$$
$$57\times 41 \bmod 491=373$$
$$120\times 41 \bmod 491=10$$
$$251\times 41 \bmod 491=471$$

假设加密明文为 p=10010101，加密为 c=82+83+373+471=1009，得到密文 1009；

解密过程：

$$41^{-1} \bmod 491=12$$
$$1009\times 12 \bmod 491=324$$
$$251+57+14+2=324\Rightarrow 10010101$$

攻击以上背包加密算法的有效方法是 LLL 算法，该算法用于求格中的最短路径，由 Lenstra-Lenstra-Lovasz 发现。

2. LLL 算法破解背包算法

首先介绍格(Lattice)的概念，设 $B=\{\beta_0,\beta_1,\cdots,\beta_{m-1}\}$ 是实向量空间 $\mathbb{R}^{n+1}$ 的一组线性无关的向量，那么由 B 形成的格定义为 $L(B)=\{\sum_{i=0}^{m-1}v_i\beta_i \mid v_i\in\mathbb{Z}\}$。设向量 $\beta=(r_0,r_1,\cdots,r_n)$，其长度用欧氏范数 $\|\beta\|=\sqrt{\sum_{i=0}^{n}r_i^2}$ 来表示，LLL 算法输入向量组 B，可以在多项式时间内，在 $L(B)$ 中找到一组长度“最短“，接近正交的另外一组 m 个线性无关的向量组 B'。

设背包加密的公钥为 $pk=(b_0,b_1,\cdots,b_{n-1})$，密文为 $c=\sum_{i=0}^{n-1}b_iv_i$，下面求向量 v。

构造矩阵 $A=\begin{pmatrix}1&0&\cdots&0&0\\0&1&\cdots&0&0\\\cdots&\cdots&\cdots&\cdots&\cdots\\0&0&\cdots&1&0\\b_0&b_1&\cdots&b_{n-1}&-c\end{pmatrix}$，那么 $A\begin{pmatrix}v_0\\v_1\\\cdots\\v_{n-1}\\1\end{pmatrix}=\begin{pmatrix}v_0\\v_1\\\cdots\\v_{n-1}\\0\end{pmatrix}$，

设矩阵 A 的列向量为 $\beta_0,\beta_1,\cdots,\beta_n$，如果 $c\neq 0$，那么它们是 $n+1$ 个线性无关的向量，可以看成是实向量空间 $\mathbb{R}^{n+1}$ 的一组基底，$(v_0,v_1,\cdots,v_{n-1},0)^T=v_0\beta_0+v_1\beta_1+\cdots+v_{n-1}\beta_{n-1}+\beta_n$，所以 $(v_0,v_1,\cdots,v_{n-1},0)^T\in L(A)$。又因为 $(v_0,v_1,\cdots,v_{n-1},0)^T$ 的每一项为 0 或者 1，所以其长度较小，通过在 $\beta_0,\beta_1,\cdots,\beta_n$ 上运用 LLL 算法，会生成一组新的“最短”的基底，可能在新的基底中出现 $(v_0,v_1,\cdots,v_{n-1},0)^T$，从而得到背包问题的解 v。

构造矩阵 $A=\begin{pmatrix}1&0&\cdots&0&0&0\\0&1&\cdots&0&0&0\\\cdots&\cdots&\cdots&\cdots&\cdots&\cdots\\0&0&\cdots&1&0&0\\b_0&b_1&\cdots&b_{n-1}&-c&0\\1&1&\cdots&1&0&-guess\end{pmatrix}$，那么 $A\begin{pmatrix}v_0\\v_1\\\cdots\\v_{n-1}\\1\\1\end{pmatrix}=\begin{pmatrix}v_0\\v_1\\\cdots\\v_{n-1}\\0\\\sum_{i=0}^{n-1}v_i-guess\end{pmatrix}$

本题刚好是碰到特殊情况，LLL 算法失效。

解决基本思路是再加上一行约束条件，guess 是猜测结果中 1 的个数。