Cryptanalysis of the Knapsack Generator

Simon Knellwolf Willi Meier

FHNW, Switzerland

FSE 2011, February 14-16, Lyngby, Denmark.

Knapsack Generator

n-bit integers $w_0, \ldots, w_{n-1}$ (**weights**)

n-bit LFSR sequence $u_0, u_1, u_2, \ldots$ (**control bits**)

Keystream generation

- Addition $v_i = \sum_{j=0}^{n-1} u_{i+j} w_j \mod 2^n$
- Truncation $z_i = v_i \gg \ell$
- Output $n - \ell$ bits of z_i

Secret key: weights + initial state of LFSR $= n^2 + n$ bits

Background

Introduced by Rueppel and Massey in 1985

Alternative to boolean filter / combining function

Security is not related to the hardness of the knapsack problem

Previous Cryptanalysis

Rueppel, 1986:

- LSBs of v_i have low linear complexity: choose $\ell = \lceil \log n \rceil$
- Effective key length $\geq n(\lfloor \log n \rfloor - 1)$ bits

Von zur Gathen and Shparlinski, SAC 2004:

- Attacks based on lattice basis reduction
- Known control bits: only for $\ell \geq \log(n^2 + n)$, $n^2 - n$ outputs
- Guess and Determine: complexity difficult to estimate, no empirical results

Von zur Gathen and Shparlinski, J. Math. Crypt. 2009:

- Fast variant of the Knapsack Generator
- Analysis of output distribution

A System of Modular Equations

Generation of s outputs (without truncation):

$$\mathbf{v} = U\mathbf{w} \mod 2^n$$

where U is a $s \times n$ matrix containing the control bits.

- U has full rank modulo 2^n.
- $\mathbf{w} = U^{-1}\mathbf{v} \mod 2^n$ if U is known and $s = n$.
- U is determined by n bits: Guess and Determine.

Challenge: Output is truncated, we only get $\mathbf{z} = \mathbf{v} \gg \ell$.

Weight Approximation Matrix

Direct approach: Don't care about the discarded bits

$$\begin{aligned}\tilde{\mathbf{w}} &= U^{-1}(\mathbf{z} \ll \ell) \\ &\approx U^{-1}(\mathbf{z} \ll \ell) + U^{-1}\mathbf{d} = \mathbf{w}\end{aligned}$$

where $\mathbf{d} = \mathbf{v} - (\mathbf{z} \ll \ell)$.

- $s = n$: bad approximation, because $U^{-1}\mathbf{d}$ is large.
- $s > n$: not a unique U^{-1}, but many choices for T such that $TU = I_n$.

  T is called *approximation matrix* and $\tilde{\mathbf{w}} = T(\mathbf{z} \ll \ell)$.

Prediction with Approximate Weights

Prediction of a subsequent sum:

$$\begin{aligned}\tilde{v}_s = \mathbf{u}_s \tilde{\mathbf{w}} &= \mathbf{u}_s T(\mathbf{z} \ll \ell) \\ &\approx \mathbf{u}_s T(\mathbf{z} \ll \ell) + \mathbf{u}_s T \mathbf{d} = v_s\end{aligned}$$

Sufficient condition for prediction (at least one bit with $p > 0.5$):

$$\lceil \log \|T\| \rceil \leq n - \ell - 1,$$

where $\|T\| = \sum_{i,j} |t_{ij}|$.

Finding Good Approximation Matrices

Task: Find T such that $TU = I_n$ with small coefficients.

Row by row, this is a special case of the following problem:

Problem: Find a short vector $\mathbf{x}$ such that $\mathbf{x}A = \mathbf{b}$.

Solving strategy

1. Find some solution $\mathbf{x}'$.
2. Find a close vector $\mathbf{x}''$ in the kernel of A.
3. Set $\mathbf{x} = \mathbf{x}' - \mathbf{x}''$.

At step 2: Use a variant of Babai's algorithm on a LLL reduced kernel basis. The basis must be reduced only once for all rows.

Empirical Results: Approximation Matrix

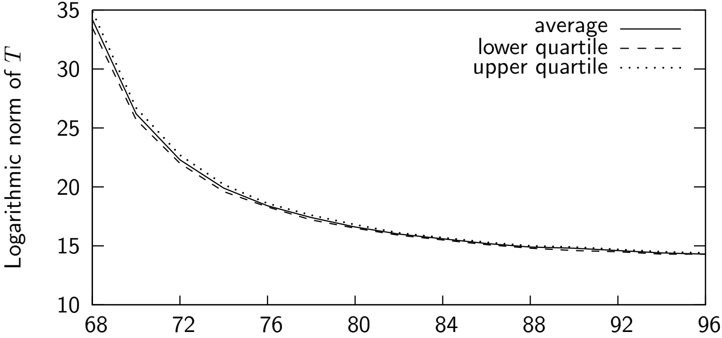

Figure: Average logarithmic norm of T for $n = 64$ in function of s.

Empirical Results: Prediction

Scenario: known control bits

$s-n$	$n=32$	$n=64$	$n=128$	$n=256$
8	20.6	42.9	85.3	164.6
16	22.2	48.7	100.9	203.4
24	22.6	50.3	105.9	216.4
32	22.7	50.8	108.1	222.4

Table: Average number of correctly predicted bits per output for $\ell = \log n$.

The Full Attack (Guess and Determine)

Scenario: known keystream

1. Guess $u_0, \ldots, u_{n-1}$ and derive $s \times n$ matrix U.
2. Find T based on U.
3. Use T and $\mathbf{z}$ to compute $\tilde{\mathbf{w}}$.
4. Compute t predictions and check their λ most significant bits. If almost all of them are correct, the control bits have been guessed correctly. Otherwise, go back to step 1.

Empirical Results: Attack for $n = 32$

Recall: key length $= 32^2 + 32 = 1056$ bits

The full attack is practical on a Desktop Computer:

- Approximation parameter: $s = 40$.
- Checking parameter: $t = 20, \lambda = 5$.

In about three days:

- Correct initial control bits identified (32 bits).
- 85% of the weight bits recovered (about 870 bits).
- 22 bits/output can be predicted (output = 27 bits).

Fast Knapsack Generator

R an arbitrary ring

- Choose $a, b \in R$.
- Compute the n weights as $w_i = ab^{n-i}$.

The v_i can be computed recursively:

$$v_{i+1} = bv_i - ab^{n+1}u_i + abu_{i+n}$$

$R = \mathbb{F}_p$: provable results for uniformity of output distribution.

Fast Knapsack Generator

The v_i can be computed recursively:

$$v_{i+1} = bv_i - ab^{n+1}u_i + abu_{i+n}$$

Basic attack strategy (for $R = \mathbb{F}_p$)

1. Find i such that $u_i = 0$ and $u_{i+n} = 0$.
2. Guess the discarded bits of v_i and v_{i+1} (2ℓ bits).
3. Compute $b = v_{i+1}/v_i$ and $a = v_i / \sum_{j=0}^{n-1} u_{i+j} b^{n-j}$.
4. Check the guess.

Maximum number of guesses: $2^{2\ell}$.

Conclusion

The concept of the weight approximation matrix leads to an effective guess and determine attack. The use of LLL in this context gives striking results:

- All attacks work for relevant parameters n and ℓ:

n	32	64	128
ℓ up to	≈ 25	≈ 42	≈ 98

- Known control bits: weights can be approximated from no more than $n+8$ outputs.
- Known keystream: security is not higher than n bits (at the prize of a n^2+n bit key).