L'algorithme A5/1 peut-il être utilisé comme générateur pseudo-aléatoire ?

par Alexandre PUKALL

A5/1 utilisé dans les réseaux GSM 2G, utilise trois LFSR pour générer un flux pseudo aléatoire de 114 bits pour chaque trame GSM (228 bits pour la trame montante et la trame descendante).

Les trois LFSR utilisés sont les suivants :

x^19 + x^18 + x^17 + x^14 + 1,
x^22 + x^21 + 1,
x^23 + x^22 + x^21 + x^8 + 1.

Comme les degrés des trois LFSR sont premiers entre eux, la période est égale au produit des périodes des trois LFSR.

19+22+23 = 64 bits. Ainsi la période de A5/1 est de 2^64 bits (2 puissance 64), ce qui est énorme.

On pourrait donc penser qu'A5/1 peut être utilisé comme générateur de nombres aléatoires.

Or il n'en est rien. Dès que la période maximale du plus grand des trois LFSR est atteinte, à savoir 8 Mo, les données produites perdent leur caractère aléatoire (les trois LFSR produisent des périodes maximales de 512 Ko pour le LFSR 19 bits, 4 Mo pour le LFSR 22 bits et 8Mo pour le LFSR 23 bits).

On peut facilement le tester avec le testeur de nombres aléatoires PractRand.

Les tests effectués sur Linux sont les suivants :

La version A5/1 utilisée est celle de Marc Briceno, Ian Goldberg et David Wagner avec les modifications suivantes pour produire un flux pseudo aléatoire infini :

```
void run(byte AtoBkeystream[], byte BtoAkeystream[]) {
        int i;
          unsigned int bit, bits;

   unsigned char byte;

   bits = 0;
   byte = 0;

        /* Zero out the output buffers. */
        for (i=0; i<=113/8; i++)
                AtoBkeystream[i] = BtoAkeystream[i] = 0;

        /* Generate 114 bits of keystream for the
         * A->B direction.  Store it, MSB first. */
        while(1) {
                clock();
                bit=getbit();

                  byte = (byte << 1) | bit;
```

```
            // printf("%d",bit);
    bits++;
    if (bits == 8)
    {
      printf("%c",byte);
      bits = 0;
      byte = 0;
    }
                    //AtoBkeystream[i/8] |= getbit() << (7-(i&7));
          }
```

Ce fichier peut être téléchargé ici : (attention pas de https pour se connecter)

http://pccipher.free.fr/a5-1/a5-random-generator.c

et peut être compilé sur linux avec :

gcc a5-random-generator.c -o a5-random-generator

Practrand peut être téléchargé ici :

https://pracrand.sourceforge.net/

Le test est ensuite lancé par :

./a5-random-generator | rng_test stdin8

```
RNG_test using PractRand version 0.94
RNG = RNG_stdin8, seed = unknown
test set = core, folding = standard (8 bit)

rng=RNG_stdin8, seed=unknown
length= 256 kilobytes (2^18 bytes), time= 2.1 seconds
  no anomalies in 41 test result(s)

rng=RNG_stdin8, seed=unknown
length= 512 kilobytes (2^19 bytes), time= 7.2 seconds
  no anomalies in 50 test result(s)

rng=RNG_stdin8, seed=unknown
length= 1 megabyte (2^20 bytes), time= 14.7 seconds
  no anomalies in 56 test result(s)

rng=RNG_stdin8, seed=unknown
length= 2 megabytes (2^21 bytes), time= 26.6 seconds
  no anomalies in 63 test result(s)

rng=RNG_stdin8, seed=unknown
length= 4 megabytes (2^22 bytes), time= 47.6 seconds
  no anomalies in 71 test result(s)

rng=RNG_stdin8, seed=unknown
```

length= 8 megabytes (2^23 bytes), time= 86.7 seconds

Test Name	*Raw*	*Processed*	*Evaluation*
Gap-16:A		***R= +17.8 p = 1.2e-14***	***FAIL !***

...and 75 test result(s) without anomalies

A5/1 ne peut donc pas être utilisé comme générateur pseudo-aléatoire.