The Discrete Logarithm Problem Modulo One: Cryptanalysing the Ariffin–Abu cryptosystem

Simon R. Blackburn
Department of Mathematics
Royal Holloway, University of London
Egham, Surrey TW20 0EX
United Kingdom
s.blackburn@rhul.ac.uk

April 23, 2010

Abstract

The paper provides a cryptanalysis the AA_β-cryptosystem recently proposed by Ariffin and Abu. The scheme is in essence a key agreement scheme whose security is based on a discrete logarithm problem in the infinite (additive) group $\mathbb{R}/\mathbb{Z}$ (the reals modulo 1). The paper breaks the AA_β-cryptosystem (in a passive adversary model) by showing that this discrete logarithm problem can be efficiently solved in practice.

1 Introduction

Let $x, y \in [0, 1)$ be real numbers with the property that the fractional part of nx is equal to y for some integer n. We write $nx = y \bmod 1$ to express this relationship. The *discrete logarithm problem modulo* 1 is defined to be the problem of finding n when x and y are given. (So the discrete logarithm problem modulo 1 is just the standard discrete logarithm problem in the additive group $\mathbb{R}/\mathbb{Z}$.)

Ariffin and Abu [1] have recently described a 'chaos-based' public key cryptosystem which they call the AA_β cryptosystem. It turns out that the

security of their scheme is based on the difficulty of the discrete logarithm problem modulo 1. This short note shows how to solve this discrete logarithm problem efficiently in practice, and so the Ariffin–Abu cryptosystem is insecure.

We provide a description of the Ariffin–Abu cryptosystem (which is in essence a key agreement scheme) in Section 2. Our description is rather different from the description in [1], but our approach makes it clear that the scheme's security depends on the discrete logarithm problem modulo 1. In Section 3 we show how to solve this discrete logarithm problem, thus breaking the scheme. (We derive the shared key in polynomial time, even in a passive adversary model.) We provide a short conclusion in Section 4.

We will use some standard material from cryptography and number theory without comment. In particular, we assume knowledge of Diffie–Hellmann key agreement (see, for example, Stinson [3]), and we assume some standard material on convergents and congruences (see, for example, Hardy and Wright [2]).

2 The AA_β-cryptosystem

This section contains a brief description of the AA_β-Cryptosystem. Just as in Diffie–Hellman key agreement, two parties, A and B, aim to interact to produce a common shared key. This shared key can then be used with any symmetric cryptosystem to encrypt and decrypt messages between the two parties.

Let α and β be public integers. Let $x \in [0, 1)$ be a public real number. Let integer matrices A_0 and A_1 be defined by

$$A_0 = \begin{pmatrix} 1 & \alpha \\ 1 & 0 \end{pmatrix} \text{ and } A_1 = \begin{pmatrix} \beta & 1 \\ 1 & 0 \end{pmatrix}.$$

Stage 1: A calculates a private integer n_A, as follows. She chooses a random k bit binary string $b_1 b_2 \cdots b_k$. She then calculates the integer matrix $A_{b_k} A_{b_{k-1}} \cdots A_{b_1}$, and sets n_A to be the top left entry of the resulting matrix.

B calculates an integer n_B in exactly the same way, by choosing his own k-bit binary string.

Stage 2: Recall that $x \in [0, 1)$ is a public real number. A calculates $r_A = n_A x \bmod 1$, and sends r_A to B. Similarly B sends $r_B = n_B x \bmod 1$ to A.

Stage 3: The shared key is $n_A n_B x \bmod 1$. Party A derives the shared key by calculating $n_A r_B \bmod 1$, and Party B derives the same key by calculating $n_B r_A \bmod 1$. The shared key can then be used with any symmetric cryptosystem to send encrypted messages between A and B.

A numerical example taken from [1] is as follows. Let $\alpha = 2$, $\beta = 3$ and $x = 0.78217087686061859\ldots$. Party A chooses the binary string 10110011, and calculates the matrix product

$$A_1A_1A_0A_0A_1A_1A_0A_1 = \begin{pmatrix} 2415 & 533 \\ 734 & 162 \end{pmatrix}.$$

So $n_A = 2415$. Party A calculates $r_A = 0.94266761839389801\ldots$ and sends this value to B.

Similarly, Party B chooses the binary string 10110111. B then calculates $n_B = 3725$ and $r_B = 0.58651630580425262\ldots$, and sends r_B to A.

The shared key $n_A n_B x \bmod 1$ is $0.436878317270082\ldots$.

We end this section with some comments.

- The Ariffin–Abu scheme is essentially a Diffie–Hellman key agreement scheme, but with exponentiation in a discrete group replaced by a multiplication operation in the reals modulo 1.
- In practice (as Ariffin and Abu point out) since real numbers cannot be represented on a computer we would work with approximations in some way (but the approximation method is immaterial to our analysis).
- If we can efficiently solve the discrete logarithm problem modulo 1, the scheme is clearly insecure. For an eavesdropper who sees r_A and r_B can solve the discrete logarithm problem $n_A x = r_A \bmod 1$ to find n_A, and can then calculate the shared key $n_A r_B \bmod 1$.

3 Solving the discrete logarithm problem

This section shows how to solve the discrete logarithm problem modulo 1. For the remainder of this section, we write DLP for the discrete logarithm problem modulo 1. So there exist real numbers $x, y \in [0, 1)$ and an integer n with $nx = y \bmod 1$. We are given x and y, and the DLP asks us to calculate n

(or any other integer $\overline{n}$ with $\overline{n}x = y \bmod 1$). Suppose n is a κ-bit integer. We aim to give an algorithm for the DLP that is polynomial in κ (of low degree).

When $y = 0$ we may choose $n = 0$ as our solution, so we may assume that $y \neq 0$. In particular, this assumption implies that x cannot be written in the form p/n for some integer p (for then $nx = 0 \bmod 1$ and so $y = 0$).

We may assume that n is positive. To see this note that when n is negative the equation $-nx = (1 - y) \mod 1$ may be solved to recover $-n$, provided we can solve the DLP for positive integers.

Our algorithm for the DLP for n positive is as follows:

Input: real numbers x, y.
If $y = 0$, output $n = 0$ and halt.
Compute convergents to x, using a continued fraction algorithm.
For each convergent a/b:
  Compute a', the nearest integer to by.
  Compute the unique integer n' with $n' = a'a^{-1} \bmod b$ and $0 \leq n' < b$.
  If $n'x = y \bmod 1$, output $n = n'$ and halt.
  Otherwise try the next convergent.

Consider, for example, the DLP $y = r_A = n_A x \bmod 1$ from the previous section. We compute the convergents to $x = 0.78217087686061859\ldots$ as:

$$0, 1, \frac{3}{4}, \frac{4}{5}, \frac{7}{9}, \frac{18}{23}, \frac{61}{78}, \frac{79}{101}, \frac{1009}{1290}, \frac{1088}{1391}, \frac{2097}{2681}, \frac{5282}{6753}, \ldots.$$

Once we have computed the convergent a/b with $a = 2097$ and $b = 2681$, we can recover $n = n_A$: by rounding $by = b \cdot 0.94266761839389801\ldots$ to the nearest integer, we find that $a' = 2527$; we then solve $n' = 2527 \cdot 2097^{-1} \bmod 2681$, and find that $n' = 2415 = n_A$, as required. (Similarly, we can recover $n_B = 3725$ by using the convergent $5282/6753$ for x, the value $y = r_B$ and calculating $a' = 3961$.)

We now justify why our algorithm is polynomial time. We first observe that after generating $O(\kappa)$ convergents, we have computed a rational approximation a/b to x such that $b \geq 2n$. This is true since the denominators in the convergents a/b for any real number grow exponentially (at least as

fast as the Fibonnacci numbers). Since each convergent can be generated in polynomial time, we generate a convergent with the properties we need in polynomial time. Note that convergents a/b to x have the property that $|x-(a/b)| < 1/b^2$, so we have found a good rational approximation to x. We define $\epsilon = x - (a/b)$, so $|\epsilon| < 1/b^2$.

Since $nx = y \bmod 1$, there exists an integer m such that $nx = y + m$. We claim that the nearest integer a' to by has the property that $a' = na - mb$. To see this, note that

$$\begin{aligned} na - mb &= nb(a/b) - mb \\ &= nb(x - \epsilon) - mb \\ &= by + bm - nb\epsilon - mb \\ &= by - nb\epsilon. \end{aligned}$$

But $na - mb$ is an integer and $|nb\epsilon| < nb/b^2 = n/b \leq 1/2$, and so the nearest integer a' to by is $na - mb$, as claimed.

We have that $a' = na \bmod b$. Since a/b is a convergent, a and b are coprime and so $n = a'a^{-1} \bmod b$. The integer n' computed by our algorithm therefore satisfies $n = n' \bmod b$. But since $n \leq (1/2)b < b$ and $n' < b$, no modular reduction has taken place and $n = n'$, as required.

We have now shown our algorithm is polynomial time. We end this section with a few comments on possible modifications of the algorithm.

- In the cryptanalysis in the section above, we know a realistic upper bound N on n. So $0 < n \leq N$, where N is known. In this application, we could modify our algorithm to first compute the convergents a/b to x off-line, storing the first convergent a/b such that $b \geq 2N$. Once we receive y, we only need to consider this convergent: it will return the correct value for n.

- There are other possibilities for constructing sequences of rational approximations a/b to x, that might be useful instead of convergents in some circumstances. For example, we could construct the ith such approximation by setting $b = 2^i$ and $a = \lfloor xb \rfloor$ when i is odd, and $a = \lceil xb \rceil$ when i is even. (Note that the denominators of our rational approximations would in general become much larger than when using convergents.) In practical situations, it is quite possible that x is represented on a computer by some rational approximation: in this case, it

makes sense to use this rational approximation rather than computing convergents!

4 Conclusion

We have shown that a cryptosystem due to Ariffin and Abu [1] is insecure, by solving the discrete logarithm problem modulo 1. This cryptanalysis is a good illustration of the author's belief that using chaotic systems for cryptography is a bad idea: chaos implies some notion of points being 'close' or not, and this notion of distance can be exploited by the cryptanalyst.

Acknowledgement The author would like to Fred Piper for first drawing his attention to the AA_β cryptosystem, and to Kenny Paterson for his comments on a draft. The author would also like to thank Francesco Sica for suggesting a simplification of (and correcting) the argument for the complexity of our algorithm given in an earlier version of this paper.

References

[1] M.R.K. Ariffin and N.A. Abu, 'AA_β-Cryptosystem: A chaos based public key cryptosystem', *Int. J. Cryptology Research* **1** (2009) 149-163.

[2] G.H. Hardy and E.M. Wright, *An Introduction to the Theory of Numbers* (6th edition), Oxford University Press, Oxford, 2008.

[3] Douglas R. Stinson, *Cryptography: Theory and Practice* (3rd edition), Chapman and Hall, Boca Raton, 2005.