A NOTE ON PUBLIC KEY CRYPTOGRAPHY BASED ON WILLIAMS-GUY FUNCTIONS

E. L. Roettger
Department of General Education, Mount Royal University, Calgary, Alberta, Canada
eroettger@mtroyal.ca

Received: 6/23/23, Revised: 7/11/23, Accepted: 8/17/23, Published: 9/15/23

Abstract

In 2011 Williams and Guy developed a pair of fourth-order sequences and they showed how these sequences possess many of the same properties as the Lucas functions; hence, are a fourth-order generalization of said functions. We show here that these fourth-order Williams-Guy functions can be used to perform public key cryptography.

– *Dedicated to Hugh Williams on the occasion of his* 80^{th} *birthday.*

1. Introduction

It is the goal of this paper to verify that a fourth-order generalization of the Lucas functions developed by Williams and Guy can be employed to perform public key cryptography. That such a public key protocol is possible is mentioned by Granville and Pomerance in [1]; after they introduce the Williams-Guy functions they write:

> "This work led to further understanding, including a series of papers with [Williams'] former doctoral student Eric Roettger that culminated with a solution to Lucas' unsolved problem of generalizing the Lucas sequences to the setting of higher order recurrences, as well as an idea for public key cryptography."

2. The Diffie-Hellman-like Key-Exchange Protocol

The integer sequences (U_n) and (V_n) are introduced in [2, 7] by letting

$$V_n = \alpha_1^n + \beta_1^n + \alpha_2^n + \beta_2^n \quad \text{and} \quad U_n = \frac{\alpha_1^n + \beta_1^n - \alpha_2^n - \beta_2^n}{\alpha_1 + \beta_1 - \alpha_2 - \beta_2},$$

DOI: 10.5281/zenodo.8349074

where α_1, β_1, α_2, and β_2 are the zeroes of $F(x) = x^4 - P_1x^3 + (P_2+2Q)x^2 - QP_1x + Q^2$, where P_1, $P_2, Q \in \mathbb{Z}$, $\Delta = P_1^2 - 4P_2 \neq 0$ and $\gcd(P_1, P_2, Q) = 1$. Also, if we set $E = (P_2 + 4Q)^2 - 4QP_1^2$, then the discriminant D of $F(x)$ is $D = E\Delta^2Q^2$. Further, in [6, p. 1270 Case 4] they show that if the Legendre symbols $(\Delta|p) = (E|p) = -1$, $p \nmid P_1$, and $p \nmid D$, then the splitting field of $F(x)$ considered as a polynomial over $\mathbb{F}_p$ is $\mathbb{F}_{p^4}$. This result assures we are working over $\mathbb{F}_{p^4}$ in the presented cryptosystem.

Many valuable properties of (U_n) and (V_n) are developed in [2, 6, 7]; however, here we only require the following addition formulas from [6, p. 1259]:

$$2V_{n+m} = V_nV_m + \Delta U_nU_m - 2Q^mV_{n-m} \tag{1}$$

and

$$2U_{n+m} = U_nV_m + U_mV_n - 2Q^mU_{n-m}. \tag{2}$$

Making appropriate substitutions for n, m in (1) and (2) yields the formulas

$$2V_{4n} = V_{2n}^2 + \Delta U_{2n}^2 - 8Q^{2n} \quad \text{and} \quad U_{4n} = V_{2n}U_{2n}. \tag{3}$$

Using these few equations above, an efficient double and add method to compute U_m and $V_m \pmod N$ is presented in [2, Section 4] (with the aid of auxiliary functions K_j, L_j and J_j) and it is somewhat repeated here, as it is what makes the Diffie-Hellman-like key-exchange protocol developed in the sequel possible.

We begin by noting that $U_2 = P_1$ and $V_2 = P_1^2 - 2P_2 - 4Q$, and for any fixed integer $t > 0$ we define

$$K_j = U_{2jt}/2Q^{jt} \quad \text{and} \quad L_j = V_{2jt}/2Q^{jt}. \tag{4}$$

Thus, if we substitute nt for n in (3) and divide (3) by $4Q^{2nt}$, we get

$$L_{2n} = L_n^2 + \Delta K_n^2 - 2 \quad \text{and} \quad K_{2n} = 2K_nL_n. \tag{5}$$

Also, if we set n to be $2nt + 2t$ and m to be $2nt$ in (1), (2) and divide (1), (2) by $4Q^{2nt+t}$, we recover

$$L_{2n+1} = L_{n+1}L_n + \Delta K_{n+1}K_n - L_1 \quad \text{and} \quad K_{2n+1} = L_{n+1}K_n + L_nK_{n+1} - K_1. \tag{6}$$

Now let N be any positive integer such that $\gcd(Q, N) = 1$. We will use identities (5), (6) to perform a double and add algorithm to calculate K_m and $L_m \pmod N$ for the case that $t = 1$. We present two similar algorithms based on the cases: $t = 1$ and $t > 1$ in equation (4).

`Case 1:` $(t = 1)$ We will first show how to calculate $K_m \equiv U_{2m}/2Q^m$ and $L_m \equiv V_{2m}/2Q^m \pmod N$. Since $t = 1$, by (4) we assign $K_1 = U_2/2Q$ and $L_1 = V_2/2Q$. If we let $h = \lceil \log_2 m \rceil$, then we can represent the binary expansion of m by $\sum_{i=0}^{h} b_{h-i}2^i$, where $b_0 = 1$ and $b_i \in \{0, 1\}$ for positive $i \leq h$. We begin with

the 4-tuple $\mathscr{W}_0 \equiv \{L_1, K_1, L_2, K_2\} \pmod N$. Now if $\mathscr{W}_i \equiv \{A, B, C, D\} \pmod N$, then

$$\mathscr{W}_{i+1} \equiv \begin{cases} \{A^2 + \Delta B^2 - 2, 2AB, AC + \Delta BD - L_1, \\ \qquad BC + AD - K_1\} \pmod N, & \text{if } b_{i+1} = 0; \\ \{AC + \Delta BD - L_1, BC + AD - K_1, \\ \qquad C^2 + \Delta D^2 - 2, 2CD\} \pmod N, & \text{if } b_{i+1} = 1. \end{cases}$$

Hence, $\mathscr{W}_h \equiv \{L_m, K_m, L_{m+1}, K_{m+1}\} \pmod N$.

`Case 2:` $(t > 1)$ Now given $U_{2t}/2Q^t$ and $V_{2t}/2Q^t$ we can perform a similar process to calculate $U_{2mt}/2Q^{mt}$ and $V_{2mt}/2Q^{mt}$. We will again use K_1 and L_1, however since $t \neq 1$ it is worth stating that in what follows $K_1 = U_{2t}/2Q^t$ and $L_1 = V_{2t}/2Q^t$. Again, following [2] we define $J_j = Q^{-(j-1)t}U_{2jt}/U_{2t}$ and note that since $K_1 = U_{2t}/2Q^t$, we have

$$K_1 J_j = (U_{2t}/2Q^t)(Q^{-(j-1)t}U_{2jt}/U_{2t}) = U_{2jt}/2Q^{jt} = K_j. \tag{7}$$

Hence, we can modify equations (5), (6) by performing substitutions with (7) using appropriate values for j and setting $\tilde{\Delta} = \Delta K_1^2$ to obtain:

$$L_{2n} = L_n^2 + \tilde{\Delta} J_n^2 - 2, \qquad J_{2n} = 2J_n L_n, \tag{8}$$

$$L_{2n+1} = L_{n+1}L_n + \tilde{\Delta} J_{n+1}J_n - L_1 \quad \text{and} \qquad J_{2n+1} = L_{n+1}J_n + L_n J_{n+1} - 1. \tag{9}$$

If we use the same binary expansion of m and let $\mathscr{W}_0 \equiv \{L_1, 1, L_2, J_2\} \pmod N$, then we can compute $\mathscr{W}_h \equiv \{L_m, J_m, L_{m+1}, J_{m+1}\} \pmod N$ as follows. As before, we let $\mathscr{W}_i \equiv \{A, B, C, D\} \pmod N$; then

$$\mathscr{W}_{i+1} \equiv \begin{cases} \{A^2 + \tilde{\Delta} B^2 - 2, 2AB, AC + \tilde{\Delta} BD - L_1, \\ \qquad BC + AD - 1\} \pmod N, & \text{if } b_{i+1} = 0; \\ \{AC + \tilde{\Delta} BD - L_1, BC + AD - 1, \\ \qquad C^2 + \tilde{\Delta} D^2 - 2, 2CD\} \pmod N, & \text{if } b_{i+1} = 1. \end{cases}$$

Thus, at the end of the algorithm we have $J_m \equiv Q^{-(m-1)t}U_{2mt}/U_{2t}$ and $L_m \equiv V_{2mt}/Q^{mt} \pmod N$.

In order to avoid confusion, in the case that $t > 0$ we will write $L_{m,t}$ to denote calculating $L_m \pmod N$ with the initial conditions $K_1 \equiv U_{2t}/2Q^t$, $L_1 \equiv V_{2t}/2Q^t$, and $\tilde{\Delta} \equiv \Delta U_{2t}/2Q^t \pmod N$.

Theorem 1. *For $a, b \in \mathbb{Z}^+$, let $L_{a,b}$ and J_a be the result of the the above algorithm with initial values $K_1 \equiv U_{2b}/2Q^b$, $L_1 \equiv V_{2b}/2Q^b$, and $\tilde{\Delta} \equiv \Delta U_{2b}/2Q^b \pmod N$. Further, let $L_{b,a}$ and J_b be the result of the same algorithm with initial values $K_1 \equiv U_{2a}/2Q^a$, $L_1 \equiv V_{2a}/2Q^a$, and $\tilde{\Delta} \equiv \Delta U_{2a}/2Q^a \pmod N$. Then $L_{a,b} \equiv L_{b,a}$ and $J_a U_{2b}/2Q^b \equiv J_b U_{2a}/2Q^a \pmod N$.*

The above theorem is what makes the following novel Diffie-Hellman-like key-exchange possible and it is easily verified using identity (7).

The Diffie-Hellman-like Algorithm:

1) Alice and Bob agree on a large prime p such that $p-1$ is not smooth and integers P_1, P_2 such that $2 \mid P_1$, $\gcd(P_1, P_2) = 1$, $(\Delta|p) = (E|p) = -1$, $p \nmid P_1$ and $p \nmid D$. Further, it will always be the case that $Q = 1$. The values p, P_1 and P_2 are public.

2) Alice selects some integer a at random such that $1 < a < \mathscr{B}$ ($\mathscr{B}$ is some predetermined bound). Alice computes L_a and $K_a \pmod p$ using the first double and add algorithm with the initial conditions $K_1 \equiv U_2/2 \equiv P_1/2$, $L_1 \equiv V_2/2 \equiv (P_1^2 - 2P_2 - 4Q)/2 \pmod p$ and sends them to Bob. Bob selects some integer b at random such that $1 < b < \mathscr{B}$. Bob computes L_b and $K_b \pmod p$ using the first double and add algorithm (with the same initial conditions as Alice) and sends them to Alice.

3) Alice computes $L_{a,b}$ and J_a modulo p using the second double and add algorithm with the initial conditions $K_1 = K_b$, $L_1 = L_b$ and $\tilde{\Delta} = \Delta K_b^2$. Bob computes $L_{b,a}$ and J_b modulo p using the second double and add algorithm with the initial conditions $K_1 = K_a$, $L_1 = L_a$ and $\tilde{\Delta} = \Delta K_a^2$. Alice and Bob use either $L_{a,b} = L_{b,a}$ or $J_a K_b = J_b K_a$ as their common key.

3. Security and Efficiency of Our Cryptosystem

We can certainly break the system if, in general, we can compute n given K_1, L_1, K_n, and $L_n \pmod p$. Note that if we expand $(x-\alpha_1^m)(x-\beta_1^m)(x-\alpha_2^m)(x-\beta_2^m)$, we get $x^4 - a_1x^3 + a_2x^2 - a_3x + a_4$, where

$$a_1 = V_m,\ a_2 = 2Q^m + W_m^2 + P_2U_m^2 + P_1U_mW_m,\ a_3 = V_mQ^m, \text{ and } a_4 = Q^{2m}.$$

Now if we let $m = 2n$ and note that $W_{2n} + P_2U_{2n} = V_{2n}$, we get

$$a_1 = V_{2n}, \quad a_2 = 2Q^{2n} + V_{2n}^2 + (P_1 - 2P_2)V_{2n}U_{2n} + (P_2^2 - P_1P_2)U_{2n}^2,$$
$$a_3 = V_{2n}Q^{2n}, \quad \text{and} \quad a_4 = Q^{4n}.$$

Thus, upon setting $K_n = U_{2n}/2Q^n$, $L_n = V_{2n}/2Q^n$, we have

$$a_1/Q^n = 2L_n, \quad a_2/Q^{2n} = 2 + 4L_n^2 + 4(P_1 - 2P_2)L_nK_n + 4(P_2^2 - P_1P_2)K_n^2,$$
$$a_3/Q^{3m} = 2L_n, \quad \text{and} \quad a_4/Q^{4m} = 1.$$

If θ is a zero of $a(z) = z^4 - 2L_1z^3 + (2 + 4L_1^2 + 4(P_1 - 2P_2)L_1K_1 + 4(P_2^2 - P_1P_2)K_1^2)z^2 - 2L_1z + 1$ in $\mathbb{F}_{p^4}$ ($a(z)$ is irreducible over $\mathbb{F}_p$ by selection of P_1, P_2,

Q), then $\gamma = \theta^n$ is a zero of $b(z) = z^4 - 2L_n z^3 + (2 + 4L_n^2 + 4(P_1 - 2P_2)L_n K_n + 4(P_2^2 - P_1P_2)K_n^2)z^2 - 2L_n z + 1$. Thus the problem of determining n can be reduced to solving the discrete log problem (DLP) in $\mathbb{F}_{p^4}$.

Let $L_q[\alpha, c] = \exp(c(\log q)^\alpha (\log\log q)^{1-\alpha})$. Schirokauer [3] conjectured that the complexity of solving the DLP in $\mathbb{F}_q$ is $L_q[1/3, (64/9)^{1/3}+o(1)]$. Thus the complexity of solving the DLP in $\mathbb{F}_{p^4}$ is likely greater than $(L_p[1/3, (64/9)^{1/3} + o(1)])^{\sqrt[3]{4}}$.

We now provide a rough performance comparison of our system with Diffie-Hellman and LUCDIF (a similar cryprosystem that relies on the Lucas functions [4]) using parameters designed to provide 80 bits of security (i.e., roughly 2^{80} operations to break). In all three cases a 160 bit exponent or multiplier k will be used. Following the key size suggested in [5, Table 1], let p_1 be a 1024 bit prime, p_2 be a 512 bit prime and p_3 be a 256 bit prime. To compare the speed of calculations of our system versus classic Diffie-Hellman key exchange and LUCDIF, let us recall that due to the compression factor, p_1 used in the Diffie-Hellman key exchange provides equivalent security to using p_2 in LUCDIF or p_3 in our system.

The Diffie-Hellman key exchange performs $(3/2)k$ modular multiplications with modulus p_1, we will denote the cost of these modular multiplications by m_{p_1}. LUCDIF performs $5k$ modular multiplications with modulus p_2, having cost m_{p_2}. In our system we need to perform at most $9k+2$ modular multiplications to compute $\mathscr{W}_m \pmod{p_3}$; denote the cost of these multiplications by m_{p_3}. Therefore, we are interested in $\frac{(9k+2)m_{p_3}}{(3/2)km_{p_1}}$ and $\frac{(9k+2)m_{p_3}}{5km_{p_2}}$ to compare our system to classic Diffie-Hellman and LUCDIF, respectively. However, for ease of comparison we will replace $(9k+2)$ in our system with simply $9k$. This is perhaps appropriate as in [2, p. 525] it is argued that since Δ is often very small in comparison to the modulus N, the cost of computing $\Delta X \pmod N$ is essentially that of $X \pmod N$. Hence, the cost is closer to $9k$ modular multiplications in the first algorithm. A similar argument is made for the cost of the second algorithm being closer to $9k+1$ modular multiplications. Hence, we will proceed by comparing $\frac{9km_{p_3}}{(3/2)km_{p_1}}$ and $\frac{9km_{p_3}}{5km_{p_2}}$.

If we compare our system to the Diffie-Hellman key-exchange, we have $\frac{9km_{p_3}}{(3/2)km_{p_1}} = \frac{6m_{p_3}}{m_{p_1}}$. Under the best case scenario for modular multiplication we can expect m_{p_i} to be $\tilde{O}(\log p_i)$ bit operations and worst case scenario we can expect m_{p_i} to be $O((\log p_i)^2)$. Hence with the fastest possible multiplication we would expect $\frac{m_{p_3}}{m_{p_1}} \approx \frac{\log p_3}{\log p_1} = \frac{256}{1024} = \frac{1}{4}$, or with slower multiplication we have $\frac{m_{p_3}}{m_{p_1}} \approx \frac{(\log p_3)^2}{(\log p_1)^2} = \frac{256^2}{1024^2} = \frac{1}{16}$. Thus we can expect $\frac{3}{8} < \frac{9km_{p_3}}{(3/2)km_{p_1}} < \frac{3}{2}$, which may be favourable for our cryptosystem depending on the speed of the modular multiplication used.

Similarly, to compare our system to LUCDIF we have $\frac{9km_{p_3}}{5km_{p_2}} = \frac{9m_{p_3}}{5m_{p_2}}$. Again, with the fastest possible multiplication we have $\frac{m_{p_3}}{m_{p_2}} \approx \frac{\log p_3}{\log p_2} = \frac{256}{512} = \frac{1}{2}$, or with the slower multiplication $\frac{m_{p_3}}{m_{p_2}} \approx \frac{(\log p_3)^2}{(\log p_2)^2} = \frac{256^2}{512^2} = \frac{1}{4}$. Therefore here we certainly

have a favourable outcome, as we have shown $\frac{9}{20} < \frac{9km_{p3}}{5km_{p2}} < \frac{9}{10}$.

Thus, our cyrptosystem may take fewer bit operations than Diffie-Hellman or LUCDIF. Also, although we double the bandwidth, we get two numbers that can be used for the key: $L_{a,b} = L_{b,a}$, $J_aK_b = J_bK_a$. These numbers have no obvious relationship to each other.

4. Conclusion

Despite this paper providing verification of the existence of public key cryptography using Williams-Guy functions, as observed possible by Granville and Pomerance, it is stressed that the key-exchange developed herein is purely of recreational interest, as it does not compete with modern high powered methods nor does it belong to the class of post-quantum schemes. However, it is of some theoretical interest as an application of the Williams-Guy functions.

Acknowledgment. Much thanks to an anonymous referee, whose careful reading of the original submission of this paper and thoughtful suggestions resulted in a substantial improvement in its exposition.

References

[1] A. Granville and C. Pomerance, The man who loved problems: Richard K. Guy, *Notices Am. Math. Soc.* **69**, (4) (2022), 574–585.

[2] E. L. Roettger, H. C. Williams and R. K. Guy, Some primality tests that eluded Lucas, *Des. Codes Cryptogr.* **77** (2015), 515–539.

[3] O. Schirokauer, Discrete logarithms and local units, *Phil. Trans. Royal Soc.* **345** (1993), 409–423.

[4] P. J. Smith and M. J. J. Lennon, LUC a new public key system, *SEC* (1993), 103–117.

[5] S. S. Wagstaff, Is there a shortage of primes for cryptography?, *Int. J. Netw. Secur.* **3** (2006), 296–299.

[6] H. C. Williams and R. K. Guy, Some fourth-order linear divisibility sequences, *Int. J. Number Theory* **7** (2011), 1255–1277.

[7] H. C. Williams and R. K. Guy, Some monoapparitic fourth-order linear divisibility sequences, *Integers* **12** (2012), 1463–1485.