A Brief Retrospective Look at the Cayley-Purser Public-key Cryptosystem, 19 Years Later

Douglas R. Stinson*
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada
dstinson@uwaterloo.ca

March 13, 2018

Abstract

The purpose of this paper is to describe and analyze the Cayley-Purser algorithm, which is a public-key cryptosystem proposed by Flannery in 1999. I will present two attacks on it, one of which is apparently new. I will also examine a variant of the Cayley-Purser algorithm that was patented by Slavin in 2008, and show that it is also insecure.

1 Introduction

When she was only 16 years of age, Sarah Flannery won the *EU Young Scientist of the Year Award* for 1999. Her project consisted of a proposal of a public-key cryptosystem based on 2 by 2 matrices with entries from $\mathbb{Z}_n$, where n is the product of two distinct primes p and q. The cryptosystem she proposed was named the *Cayley-Purser algorithm*.[1]

Because this algorithm was faster than the famous *RSA public-key cryptosystem*, it garnered an incredible amount of press coverage in early 1999; see, for example, the BBC News article [1] published on January 13, 1999. However, at the time of this press coverage, the algorithm had not undergone

*The author's research is supported by NSERC discovery grant RGPIN-03882.

[1]The cryptosystem was named after the mathematicians Arthur Cayley and Michael Purser. Flannery [3] states that the Cayley-Purser algorithm was based in part on ideas in an unpublished paper by Michael Purser.

any kind of serious peer review. Unfortunately, the Cayley-Purser algorithm was shown to be insecure later in 1999, e.g., as reported by Bruce Schneier [5] in December, 1999.

Ms Flannery later wrote an interesting book, entitled *In Code: A Mathematical Journey* [3], which recounts her experiences relating to her work on the algorithm. The technical description and the analysis of the Cayley-Purser algorithm, as well as an attack on it, are found in [3, Appendix A].

In this paper, I will describe the Cayley-Purser algorithm and two attacks on it, one of which is apparently new. I will also comment a bit on the underlying mathematical theory. Finally, I will examine a variant of the Cayley-Purser algorithm, which was patented in 2008 by Slavin, and show that it is also insecure.

2 The Cayley-Purser Algorithm

In this section, we describe the Cayley-Purser algorithm, which is presented in [3, pp. 274–277]. Note that all material in this section is paraphrased from [3].

Setup: Let $n = pq$, where p and q are large distinct primes. (We assume that it is infeasible to factor n.) $\mathbb{GL}(2, n)$ denotes the 2 by 2 invertible matrices with entries from $\mathbb{Z}_n$. Let $A, C \in \mathbb{GL}(2, n)$ be chosen such that $AC \neq CA$. Define $B = C^{-1}A^{-1}C$. Then choose a secret, random positive integer r and let $G = C^r$.

The *public key* consists of A, B, G, n.

The *private key* consists of C, p, q.

Encryption: Let $X \in \mathbb{GL}(2, n)$ be the plaintext to be encrypted. The following computations are performed:

1. choose a secret, random positive integer s
2. compute $D = G^s$
3. compute $E = D^{-1}AD$
4. compute $K = D^{-1}BD$
5. compute $Y = KXK$

6. the ciphertext is (E, Y).

Decryption: Let $(E, Y) \in \mathbb{GL}(2, n) \times \mathbb{GL}(2, n)$ be the ciphertext to be decrypted. The following computations are performed:

1. compute $L = C^{-1}EC$ (note: $L = K^{-1}$)
2. compute $X = LYL$

Observe that the factorization $n = pq$ is not needed in order to decrypt ciphertexts; the matrix C is all that is required.

The correctness of the decryption process is easy to show.

Theorem 1. *[3] If the ciphertext (E, Y) is an encryption of the plaintext X, then the decryption of (E, Y) yields X.*

Proof. First we show that $L = K^{-1}$:

$$\begin{array}{rcll} LK & = & (C^{-1}EC)(D^{-1}BD) & \text{substituting for } L \text{ and } K \\ & = & C^{-1}(D^{-1}AD)CD^{-1}BD & \text{substituting for } E \\ & = & D^{-1}C^{-1}ACDD^{-1}BD & \text{because } C \text{ and } D \text{ commute} \\ & = & D^{-1}C^{-1}ACBD & \text{cancelling } DD^{-1} \\ & = & D^{-1}B^{-1}BD & \text{because } B^{-1} = C^{-1}AC \\ & = & I. & \end{array}$$

Then it is easy to verify that

$$LYL = K^{-1}YK^{-1} = X.$$

□

3 Two Attacks

The basis of the two attacks we will describe is the observation from [3, p. 290] that any scalar multiple μC can be used in place of C in the decryption process. This is easy to see, because

$$(\mu C)^{-1}E(\mu C) = C^{-1}EC. \tag{1}$$

Therefore, using μC in step 1 of the decryption process still results in the correct value of L being computed.

Thus, it is sufficient for an attacker to compute C up to a scalar multiple. This will allow any ciphertext to be decrypted, since the factorization $n = pq$ is not required in order to be able to decrypt ciphertexts.

3.1 Linear Algebra Attack

The attack described in this section is very simple but apparently new. It turns out to be straightforward to construct the private key C (or a scalar multiple μC) directly from the public key by solving a certain system of linear equations in $\mathbb{Z}_n$. We make use of the following two equations involving C:

$$CB = A^{-1}C \tag{2}$$

and

$$CG = GC \tag{3}$$

Note that (2) follows from the formula $B = C^{-1}A^{-1}C$. It is also clear that (3) holds because G is a power of C and hence G and C commute.

We observe that (2) and (3) are sufficient to compute C, up to a scalar multiple, by solving a system of linear equations in $\mathbb{Z}_n$. In these equations, A, B and G are known matrices and we are trying to determine C. Let

$$C = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \tag{4}$$

where $a, b, c, d \in \mathbb{Z}_n$. Then (2) and (3) each yield four homogeneous linear equations (in $\mathbb{Z}_n$) in the four unknowns a, b, c, d. The solution space of (2) is a 2-dimensional subspace of $(\mathbb{Z}_n)^4$, as is the solution space of (3). However, when we solve all eight equations simultaneously, we get precisely the scalar multiples of C (i.e., the solution space is a 1-dimensional subspace of $(\mathbb{Z}_n)^4$).

We will justify the statements made above in the next section. For now, we illustrate the attack with a toy example.

Example 1. *Suppose* $p = 193$ *and* $q = 149$, *so* $n = 28757$. *Suppose we define*

$$A = \begin{pmatrix} 16807 & 19399 \\ 7483 & 18143 \end{pmatrix}$$

and

$$C = \begin{pmatrix} 2910 & 1657 \\ 5341 & 24803 \end{pmatrix}.$$

Then

$$B = \begin{pmatrix} 11947 & 1712 \\ 4630 & 14946 \end{pmatrix}.$$

Finally, suppose $G = C^7$*; then*

$$G = \begin{pmatrix} 1438 & 1433 \\ 20759 & 24068 \end{pmatrix}.$$

The system of linear equation to be solved is

$$\begin{pmatrix} 24034 & 4630 & 19287 & 0 \\ 1712 & 27033 & 0 & 19287 \\ 9570 & 0 & 1724 & 4630 \\ 0 & 9570 & 1712 & 4723 \\ 0 & 20759 & 27324 & 0 \\ 1433 & 22630 & 0 & 27324 \\ 7998 & 0 & 6127 & 20759 \\ 0 & 7998 & 1433 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

The solution to this system is

$$(a, b, c, d) = \mu(28365, 13928, 25231, 28756),$$

$\mu \in \mathbb{Z}_n$. *It is straightforward to verify that this solution space indeed consists of all the scalar multiples of* C.

3.2 Cayley-Hamilton Attack

The other attack I will present is the original attack presented in [3, pp. 290–292]. It is in fact even more efficient than the attack we just described above. We summarize it briefly now.

The Cayley-Hamilton theorem states that every square matrix A over a commutative ring satisfies its own characteristic polynomial. The *characteristic polynomial* of A is the polynomial $\det(xI_n - A)$ in the indeterminate x, where A is an n by n matrix and I_n is the n by n identity matrix. When $n = 2$, the characteristic polynomial is quadratic. In this case, as noted in [3, p. 291], it follows that any power of A can can be expressed as a linear combination of A and I_2.

Recall that G is a power of C and hence C is also a power of G. So the unknown matrix C can be expressed in the form $C = \alpha I_2 + \beta G$, for scalars α and β. Since we only have to determine C up to a scalar multiple, we can WLOG take $\beta = 1$, and write $C = \alpha I_2 + G$ (we are ignoring here the unlikely possibility that $\beta = 0$). Suppose we substitute this expression for C into (2). Then we obtain

$$(\alpha I_2 + G)B = A^{-1}(\alpha I_2 + G).$$

Rearranging this, we have

$$\alpha(B - A^{-1}) = A^{-1}G - GB.$$

If we compute the two matrices $B - A^{-1}$ and $A^{-1}G - GB$, we can compare any two corresponding nonzero entries of these two matrices to determine α.

Example 2. *We use the same parameters as in Example 1. First we compute*

$$B - A^{-1} = \begin{pmatrix} 24034 & 20999 \\ 14200 & 4723 \end{pmatrix}.$$

and

$$A^{-1}G - GB = \begin{pmatrix} 17977 & 4614 \\ 25427 & 10780 \end{pmatrix}.$$

From this, we see that

$$28534(B - A^{-1}) = A^{-1}G - GB,$$

so $\alpha = 28534$. *Hence,*

$$28534I_2 + G = \begin{pmatrix} 1215 & 1433 \\ 20759 & 23845 \end{pmatrix}$$

should be a multiple of C. *In fact, it can be verified that*

$$\begin{pmatrix} 1215 & 1433 \\ 20759 & 23845 \end{pmatrix} = 5485C.$$

4 Discussion and Comments

When the Cayley-Purser algorithm was proposed, there was some mathematical analysis provided to justify its security against certain types of attacks [3, pp. 277–283]. There are some interesting mathematical points related to this that I would like to discuss in this section. I will also look briefly at the efficiency of encryption and decryption.

4.1 Security Analysis from [3]

The main possible attack discussed in [3, pp. 277–283] involves trying to use (2) to compute C (or a scalar multiple of C). The argument given is that the number of solutions (for C) to (2) is so large that it would be infeasible to distinguish the real value of C from the extra "bad" solutions to (2). It is noted that the number of solutions for C is equal to $|\mathsf{C}_{\mathbb{GL}(2,n)}(A^{-1})|$, where $\mathsf{C}_{\mathbb{GL}(2,n)}(A^{-1})$ denotes the centralizer of A^{-1}, i.e., the set of matrices

in $\mathbb{GL}(2,n)$ that commute with A^{-1}. (The actual set of solutions to (2) is a coset of $\mathsf{C}_{\mathbb{GL}(2,n)}(A^{-1})$.)

Then, a lower bound on $|\mathsf{C}_{\mathbb{GL}(2,n)}(A^{-1})|$ is obtained from the observation that every power of A^{-1} (or, equivalently, every power of A) is an element of the set $\mathsf{C}_{\mathbb{GL}(2,n)}(A^{-1})$. Hence, $|\mathsf{C}_{\mathbb{GL}(2,n)}(A^{-1})| \geq ord(A)$. Then, an analysis of the number of group elements of all possible orders is done, and it is shown that most group elements have order that is close to n^2. Since there are only n scalar multiples of the correct C, there are many "bad" solutions remaining.

The above-described analysis is correct. But, more precisely, it turns out that it is fairly straightforward to determine the exact number of solutions to (2) using some standard group theoretic arguments. Note also that the solution space of (2) or (3) contains tuples (a,b,c,d) where the corresponding matrices (4) turn out not be invertible.

We need some definitions to get started. For now, we confine our attention to $\mathbb{GL}(2,q)$ for a prime q. The following results are found in various standard algebra textbooks, such as Dummit and Foote [2]. Details of these calculations are presented in Mathewson [4].

Two matrices A and B are *similar* if $B = C^{-1}AC$ for some matrix C. (Thus, if (2) holds, then A^{-1} and B are similar.) Similarity is an equivalence relation and the equivalence classes under similarity are known as *conjugacy classes*. The conjugacy class containing A is denoted by $\mathsf{conj}(A)$. It follows from the orbit-stabilizer theorem that

$$|\mathbb{GL}(2,q)| = |\mathsf{C}_{\mathbb{GL}(2,n)}(A)| \cdot |\mathsf{conj}(A)| \tag{5}$$

for any $A \in \mathbb{GL}(2,q)$. Further, it is well-known that

$$|\mathbb{GL}(2,q)| = (q^2-1)(q^2-q). \tag{6}$$

Now, it is fairly easy to determine the various conjugacy classes by using the fact that any conjugacy class contains a unique matrix in *rational canonical form*. The rational canonical forms in $\mathbb{GL}(2,q)$ have the following possible structures:

case (1)

$$\begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix}.$$

case (2)

$$\begin{pmatrix} 0 & b \\ 1 & c \end{pmatrix}.$$

Table 1: The number of conjugacy classes in $\mathbb{GL}(2,q)$ of all possible sizes

Case	Size of conjugacy class	Number of conjugacy classes
case (1)	1	$q-1$
case (2a)	q^2-q	$\frac{q^2-q}{2}$
case (2b)	q^2-1	$q-1$
case (2c)	q^2+q	$\frac{(q-1)(q-2)}{2}$

Case 2 further subdivides into three subcases:

case (2a) b^2+4a is not a perfect square in $\mathbb{Z}_q$,

case (2b) $b^2+4a=0$ in $\mathbb{Z}_q$, and

case (2c) b^2+4a is a nonzero perfect square in $\mathbb{Z}_q$.

Further, for a given matrix expressed in rational canonical form, it is relatively straightforward to determine $|\mathsf{C}_{\mathbb{GL}(2,q)}(A)|$. Then $|C_A|$ can also be determined, from (5) and (6). Table 1 lists the number of conjugacy classes of all possible sizes (note that these results are all given in [4]).

The Cayley-Purser algorithm lives in $\mathbb{Z}_n$. So the relevant sizes of conjugacy classes would be obtained by working modulo p and modulo q, and then applying the Chinese remainder theorem to derive the sizes of the conjugacy classes in $\mathbb{GL}(2,n)$. The vast majority of these conjugacy classes in $\mathbb{GL}(2,n)$ have size very close to n^2, which indicates that the solution to (2) will be a two-dimensional subspace of $(\mathbb{Z}_n)^4$.

The second possible attack considered in [3] involves trying to determine the private key C from the public key G. It is known that $G=C^r$, where r is secret. However, r might be chosen from a small range of values (in [3], $r \leq 50$). So we might consider trying various values of r until the equation $G=C^r$ can be solved. However, even if r is known, it is not easy to solve this equation. For example, consider the special case where $r=2$ and G is a scalar multiple of the identity. Solving for C is then equivalent in difficulty to extracting square roots in $\mathbb{Z}_n$, which is equivalent to factoring n. So this particular attack will not succeed.

Of course, these two analyses are not sufficient to establish the security of the Cayley-Purser algorithm. As we saw in the previous section, an attack that utilizes all the public information allows C to be computed up to a scalar multiple, which breaks the cryptosystem.

4.2 Efficiency of Encryption and Decryption

We also have a few comments about the efficiency of encryption and decryption in the Cayley-Purser algorithm. One of the attractive features of the Cayley-Purser algorithm is its speed relative to RSA. It is reported in [3, pp. 284–289] that Cayley-Purser encryption and decryption is roughly 20–30 times faster than the comparable RSA operations.

Clearly Cayley-Purser decryption is much faster than RSA decryption, because Cayley-Purser decryption just requires a few fast matrix operations, whereas RSA decryption uses an exponentiation modulo n. On the other hand, Cayley-Purser encryption involves exponentiating the matrix G, which is an expensive operation. However, there is a trick that can be used to speed up encryption. A careful reading of the Mathematica code that is provided in [3] shows that step 2 of the encryption method is implemented by computing a linear combination of G and the identity. Using the Cayley-Hamilton theorem, it can easily be shown that this is a quicker way of obtaining a matrix D that is actually a power of G. With this modification to the encryption algorithm, no matrix exponentiations are required to encrypt a plaintext.

5 A Variation due to Slavin

In this section, I discuss a variation of the Cayley-Purser algorithm due to Slavin [6]. I am not aware of any analysis of this algorithm in the cryptographic literature. However, it is not difficult to see that it is also insecure.

The following description is from the 2008 U.S. patent [6]. It is clear that this cryptosystem is similar to the Cayley-Purser algorithm in many respects; however, several of the equations have been modified.

Setup: Let $n = pq$, where p and q are distinct primes. Let $A, C \in \mathbb{GL}(2, n)$ be chosen such that $AC \neq CA$. Define $B = CAC$. Then choose a secret, random positive integer r and let $G = C^r$.

The *public key* consists of A, B, G, n.

The *private key* consists of C, p, q.

Encryption: Let X be the plaintext to be encrypted. The following computations are performed:

1. choose a secret, random positive integer s

2. compute $D = G^s$

3. compute $E = DAD$

4. compute $K = DBD$

5. let $Y = e_K(X)$ under some secret-key cryptosystem such as AES.

6. the ciphertext is (E, Y).

Remark: The value K is used as a key in a secret-key cryptosystem. This is different from the Cayley-Purser algorithm, but it does not affect the security of this cryptosystem.

Decryption: Let (E, Y) be the ciphertext to be decrypted. The following computations are performed:

1. compute $L = CEC$

2. compute $X = d_L(Y)$

Using the fact that C and D commute, it is not difficult to verify that $CEC = DBD$ and therefore $L = K$; hence, decryption will succeed.

5.1 The Attack

Our attack is based on the following observation from [6].

Lemma 2. *Define* $M = BGB^{-1}$ *and* $N = AGA^{-1}$. *Then* $M = CNC^{-1}$.

Proof. We compute as follows:

$$\begin{aligned}
CNC^{-1} &= C(AGA^{-1})C^{-1} && \text{substituting for } N \\
&= CACC^{-1}GCC^{-1}A^{-1}C^{-1} && \text{inserting } CC^{-1} \text{ twice} \\
&= BC^{-1}GCB^{-1} && \text{because } B = CAC \\
&= BGC^{-1}CB^{-1} && \text{because } G \text{ and } C \text{ commute} \\
&= BGB^{-1} && \text{cancelling } C^{-1}C \\
&= M.
\end{aligned}$$

□

We now describe our attack on Slavin's cryptosystem. First, note that N and M can both be computed from public information. Using the two equations $M = CNC^{-1}$ and $GC = CG$, we can carry out either of the

attacks described in Section 3 to compute a scalar multiple of the unknown matrix C, say C'. Thus $C = \mu C'$ for some unknown value $\mu \in {\mathbb{Z}_n}^*$.

Slavin [6] argues that, unlike the situation in the Cayley-Purser algorithm, it is not sufficient to compute a scalar multiple of C. In the Cayley-Purser algorithm, equation (1) allows K^{-1} to be computed by an attacker using any scalar multiple of C. On the other hand, in Slavin's cryptosystem, the "key" $K = CEC$. If we replace C by a scalar multiple, then the attacker doesn't obtain the correct value of K.

However, an attacker can compute K by a slightly different approach. Consider the equation $B = CAC$. We can rewrite this as $B = \mu^2 C'AC'$. From this, it is a simple matter to compute μ^2. Computing μ is infeasible unless the factorization of n is known; however, it turns out that we do not need to compute μ.

Finally, consider the equation $K = CEC$. We can rewrite this as $K = \mu^2 C'EC'$. Since C', E and μ^2 are known, the attacker can compute K and use it to decrypt the ciphertext Y.

Thus, the steps in the attack are summarized as follows:

1. Compute M and N from A, B and G.
2. Compute C', where $C = \mu C'$ for some unknown value μ.
3. Use the equation $B = \mu^2 C'AC'$ to compute μ^2.
4. Given a ciphertext (E, Y), compute $K = \mu^2 C'EC'$.
5. Use K to decrypt Y.

Observe that steps 1–3 only involve the public key; they only need to be carried out once. Steps 4–5 then allow the decryption of a specific ciphertext; they can be repeated as often as desired, for various ciphertexts.

Example 3. *Suppose* $p = 223$ *and* $q = 173$*, so* $n = 38579$*. Suppose we define*

$$A = \begin{pmatrix} 16807 & 38390 \\ 17333 & 21788 \end{pmatrix}$$

and

$$C = \begin{pmatrix} 10106 & 10420 \\ 27722 & 27626 \end{pmatrix}.$$

Then

$$B = \begin{pmatrix} 17590 & 36066 \\ 32833 & 33331 \end{pmatrix}.$$

Finally, suppose $G = C^{11}$*; then*

$$G = \begin{pmatrix} 11303 & 17971 \\ 5315 & 18194 \end{pmatrix}.$$

The attack begins by computing M *and* N*:*

$$M = BGB^{-1} = \begin{pmatrix} 18545 & 20365 \\ 25987 & 10952 \end{pmatrix}$$

and

$$N = AGA^{-1} = \begin{pmatrix} 37716 & 5184 \\ 18941 & 30360 \end{pmatrix}.$$

Using the linear algebra attack, the system of linear equation to be solved is

$$\begin{pmatrix} 19171 & 18941 & ,18214 & 0 \\ 5184 & 11815 & 0 & 18214 \\ 12592 & 0 & 26764 & 18941 \\ 0 & 12592 & 5184 & 19408 \\ 0 & 5315 & 20608 & 0 \\ 17971 & 6891 & 0 & 20608 \\ 33264 & 0 & 31688 & 5315 \\ 0 & 33264 & 17971 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

The solution to this system is

$$(a, b, c, d) = \mu(12688, 23061, 22337, 38578),$$

$\mu \in \mathbb{Z}_n$.

Let

$$C' = \begin{pmatrix} 12688 & 23061 \\ 22337 & 38578 \end{pmatrix}.$$

Then C' *is an unknown scalar multiple of* C*. However, the attacker can compute*

$$C'AC' = \begin{pmatrix} 27011 & 27739 \\ 26956 & 8680 \end{pmatrix}$$

By comparing B *to* $C'AC'$*, it is easy to see that* $\mu^2 = 26098$*.*

Now suppose a plaintext is encrypted. First, $D = G^s$ *is computed for a random exponent* s*. Suppose that* $D = G^{129}$*; then*

$$D = \begin{pmatrix} 18776 & 31218 \\ 20617 & 22838 \end{pmatrix}.$$

Then

$$E = DAD = \begin{pmatrix} 33712 & 19745 \\ 30382 & 3658 \end{pmatrix}$$

and

$$K = DBD = \begin{pmatrix} 33935 & 21771 \\ 36280 & 7314 \end{pmatrix}$$

Given E, the attacker can compute

$$\mu^2 C'EC' = \begin{pmatrix} 33935 & 21771 \\ 36280 & 7314 \end{pmatrix},$$

which yields the "key" K.

6 Final Comments

The Cayley-Purser algorithm was a huge news story in early 1999. However, like many other "broken" cryptosystems, it has been forgotten to a certain extent. I hope that this paper serves to highlight some interesting mathematical techniques that can be used to analyze and break this cryptosystem as well as the later, lesser-known variant that was patented by Slain in 2008.

References

[1] *Teenager's email code is a cracker.* BBC News, January 13, 1999, `http://news.bbc.co.uk/2/hi/science/nature/254236.stm`.

[2] David S. Dummit and Richard M. Foote. *Abstract Algebra, Third Edition.* Wiley, 2003.

[3] Sarah Flannery with David Flannery. *In Code: A Mathematical Journey.* Workman Publishing Company, 2001.

[4] Lindsey Mathewson. *The Class Equation of $GL_2(\mathbb{F}_q)$.* Masters Thesis, University of Wisconsin-Milwaukee, 2012,

[5] *Sarah Flannery's public-key algorithm.* Crypto-Gram, December 15, 1999. Schneier on Security, `https://www.schneier.com/crypto-gram/archives/1999/1215.html`.

[6] Keith R. Slavin. *Public Key Cryptography Using Matrices.* United States Patent No. US 7,346,162 B2. March 18, 2008.