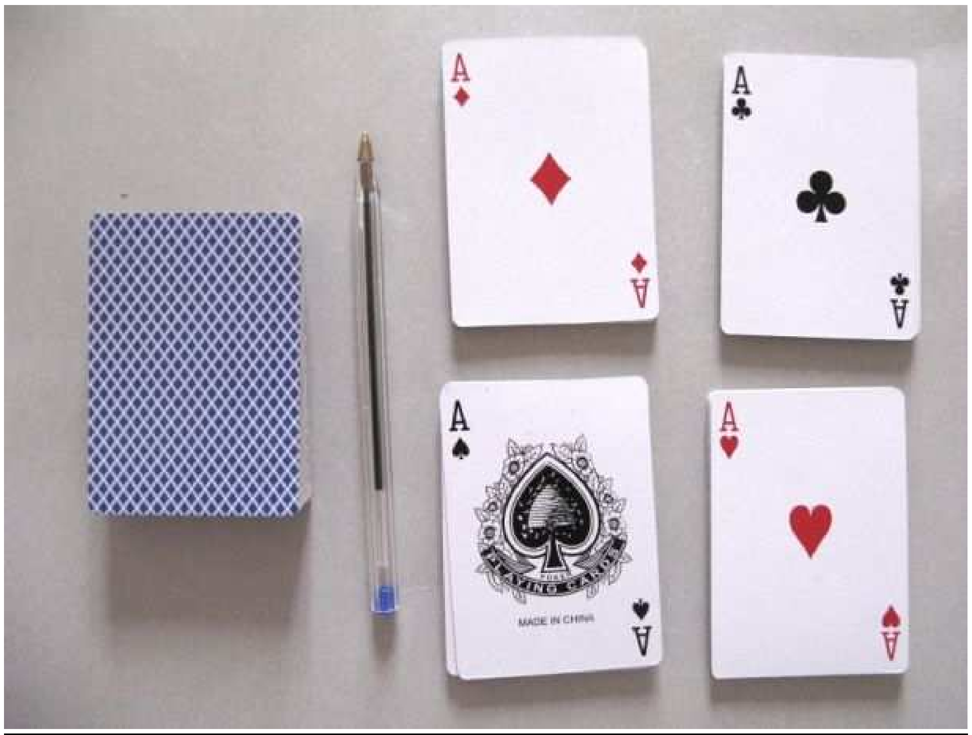
SOLITAIRE PLAYING CARD CIPHERS



To accompany the recent novel *Cryptonomicon*, Bruce Schneier, author of *Applied Cryptography*, developed a cipher using the 52 playing cards and two jokers called **Solitare**, which is described on the Counterpane web site.

My Playing Card Cipher

I, too, have now succumbed to the temptation to construct an alternative to *Solitare*. The basic cycle that takes a deck that has already been scrambled to produce a keystream operates as follows:

Step 1: From the prepared deck (the order of the cards in it is the key), turn up, and deal out face up in a row, successive cards until the total of the cards (A=1, J=11, Q=12, K=13) is 8 or more.

Step 2: If the last card dealt out in Step 1 is a J, Q, or K, take its value, otherwise take the total of the values of the cards dealt out in Step 1. (This gives a number from 8 to 17.) In the next row, deal out that many cards from the top of the deck.

Step 3: Deal out the rest of the deck under that row, in successive rows that begin on the left, and end under the lowest card in the top row, the next lowest card in the top row, and so on, in rotation. A red card is lower than a black card of the same denomination, and when there are two cards of the same color and denomination, the first one in the row is considered lower.

These first three steps may lead to a layout which looks like this:

```
7S  QH
3D  6S  3C  2D  QD  4S  8S  JS  JD  10H QC  8H
5H  AC  6D  KS
10C
QS  9D  5D
2H  3S  KD  JH  2S  9C
9H  6H
8C  2C  7H  JC  4C  8D  3H  KC  7D  6C  AH  4H
5C  10D 10S 7C  9S  KH  4D
AD  5S  AS
```

Step 4: Take the cards dealt out in Step 3, and pick them up by columns, starting with those under the lowest card in the row dealt out in Step 2. The top card in the column is to be on the bottom of a pile of face up cards, and the first column picked up is to be on the bottom of the re-assembled deck.

Step 5: Place the cards dealt out in Step 2 (the last one on the bottom) in face-up form on top of the re-assembled deck, and then the cards dealt out in Step 1, again, the last one on the bottom of a face up pile put on top of the re-assembled deck.

Step 6: Turn the deck of cards over to face-down position to repeat Step 1.

Thus, these steps would cause the following new order of the deck to result from the layout above:

```
KS  JH  JC  7C  5H  10C QS  2H  9H  8C  5C  AD  6D
5D  KD  7H  10S AS  9C  8D  KH  AC  9D  3S  6H  2C
10D 5S  4H  3H  4D  6C  7D  KC  2S  4C  9S  AH  8H
QC  10H JD  JS  8S  4S  QD  2D  3C  6S  3D  QH  7S
```

And the cards that were at the beginning of the deck, and thus controlled the transposition, are now at the end of the deck, and will be subject to the next transposition instead of controlling it.

After doing this three times, obtain a keystream digit as follows:

Looking at the cards from the top of the deck, ignore all J, Q, and K cards; take the first other card, from A to 10, and count down that many cards to find a card from A to 10. Do the same from the bottom of the deck. The last digit of the sum of the values of those two cards is the keystream digit.

Applying this to the scrambled deck obtained above (which is cheating a bit, since in practice the transposition has to be done three times), it works this way:

```
KS  JH  JC  7C  5H  10C QS  2H  9H  8C  5C  AD  6D
            (7) 1   2       3   4   5   6   7*
5D  KD  7H  10S AS  9C  8D  KH  AC  9D  3S  6H  2C
10D 5S  4H  3H  4D  6C  7D  KC  2S  4C  9S  AH  8H
QC  10H JD  JS  8S  4S  QD  2D  3C  6S  3D  QH  7S
    7*              6   5       4   3   2   1   (7)
```

the two selected cards are the ace of diamonds and the ten of hearts, so the keystream digit is a 1.

This way, people don't have to memorize a bridge ordering of the suits, and they use a straddling checkerboard to allow false addition to apply the key, instead of trying to do Vigenere or modulo-26 arithmetic in their heads.

I have received notification, in a recent E-mail, of a serious weakness in this design. Although the two cards selected are selected independently from the two halves of the deck, since they are two different cards, there is a small bias against their having the same value. Since their values are added together, rather than combined by exclusive-OR or subtraction, this bias is spread over all the even numbers.

- One way to eliminate this would be to count only red cards from the front of the deck, and only black cards from the back.
- Another would be to take any card from the front, and then take a card in any of the other three suits to that of that card from the back.
- A third way would simply be to subtract the keystream digit from 9, thus changing even digits to odd, whenever the first card in the deck is red.

Since at least the first two possibilities might conceivably weaken the cipher, perhaps what should be done is to find the first face card from the end of the deck, and if it is a J, Q, or K, respecively, then use the first, second, or third method of removing the bias. But this cipher is already too complicated without such an addition.

Two other ways to eliminate the bias, which are in some ways even more complicated, but which might also seem simpler from other perspectives, and safer than those so far examined, might also be considered:

- From the beginning of the deck, take the first face card, treating J as 1, Q as 2, and K as 3, and then count face cards to find a face card. If that face card is red, add 1 to the keystream digit, otherwise, do not change it.
- Take one card from the end of the deck as previously described, but from the beginning of the deck, use the first red card from A to 10 to find a red card from A to 10, and the first black card from A to 10 to find a black card from A to 10. Add the three cards thus found to form the keystream digit.

By limiting the mental arithmetic required, I'm trying to make my method simpler. However, the way in which the cards are rearranged is more complex; the cards are dealt out in a layout, not merely manipulated in a straight line, and thus the result looks somewhat more like a game of solitare. In the novel, the cipher Solitare was based on a computer stream cipher; my method for using playing cards is instead based on an old pencil-and-paper cipher.

The method of transposition used is the one given by General Luigi Sacco, that breaks up a block into uneven units, and which perhaps has some advantages over ordinary columnar transposition.

Of course, some of the rules used mean that there are biases in the transposition; if every card had a distinct value, the order of the columns would be slightly more random, and the rule intended to limit the row size to 17 instead of 21 makes 11, 12, and 13 more likely row lengths. Note that Step 4 is set up to make the scrambling invertible, so I did accept some good advice from Bruce Schneier's Solitare. The well-known reason for this is noted elsewhere on this site: a non-invertible transformation risks shrinking the *state space* of the thing transformed. That is: the fact that the transformation is invertible is no guarantee of a long or maximal period. But if every possible ordering of the deck is possible at the start, then every possible ordering of the deck remains possible after 20, 30, or 2000 iterations of an invertible transformation. If two orderings of the deck both transformed to the same ordering of the deck, then the transformation would not be invertible. On the other hand, with a non-invertible transformation, the number of possible orderings can continue to shrink as the transformation is repeated.

The Keying Procedure

Starting with a deck in a fixed order, say AS 2S ... KS AH 2H ... KH AD 2D ... KD AC 2C ... KC, the procedure to obtain a scrambled deck order from a keyphrase is as follows:

Divide the keyphrase into parts that are eight or more letters in length as follows: first, use all the words that are eight or more letters long in the phrase, then, go through the phrase, and, using only the shorter words, take as many words as needed at a time to reach eight or more letters. When the last part is formed, and there are less than eight unused letters in the key phrase, include them in the last part.

Then, take these parts of the keyphrase, and use them in pairs.

First, for each part, imagine the word as standing above the columns of cards, and then perform Step 3 and Step 4 of the normal cycle, but on the entire deck.

Example:

Phrase: THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG

This phrase has the parts:
THEQUICK
BROWNFOX
JUMPEDOVER
THELAZYDOG

So, the first two parts lead to the deck being scrambled as follows:

First, the deck is laid out like this:

```
 T  H  E  Q  U  I  C  K
 7  4  2  6  8  3  1  5
 AS  2S  3S  4S  5S  6S  7S
 8S  9S 10S
 JS  QS  KS  AH  2H  3H
 4H  5H
 6H  7H  8H  9H 10H  JH  QH  KH
 AD  2D  3D  4D
 5D
 6D  7D  8D  9D 10D
 JD  QD  KD  AC  2C  3C  4C
 5C  6C  7C
 8C  9C 10C  JC  QC  KC
```

Since the first card of a column is placed on the bottom when the cards are face up, and the first column picked up is at the bottom of the cards when they are face up, they will be on the top when the deck is in normal face-down order, and so this step leads to the cards being in the order:

```
7S  QH  4C  3S 10S  KS  8H  3D  8D  KD  7C 10C  6S
3H  JH  3C  KC  2S  9S  QS  5H  7H  2D  7D  QD  6C
9C  KH  4S  AH  9H  4D  9D  AC  JC  AS  8S  JS  4H
6H  AD  5D  6D  JD  5C  8C  5S  2H 10H 10D  2C  QC
```

Now, the deck is then laid out like this for the second part:

```
 B  R  O  W  N  F  O  X
 1  6  4  7  3  2  5  8
 7S
 QH  4C  3S 10S  KS  8H
 3D  8D  KD  7C 10C
 6S  3H  JH
 3C  KC  2S  9S  QS  5H  7H
 2D  7D
 QD  6C  9C  KH
 4S  AH  9H  4D  9D  AC  JC  AS
 8S
 JS  4H  6H  AD  5D  6D
 JD  5C  8C  5S  2H
10H 10D  2C
 QC
```

which, when picked up, places the deck in this order:

```
7S  QH  3D  6S  3C  2D  QD  4S  8S  JS  JD 10H  QC
8H  5H  AC  6D  KS 10C  QS  9D  5D  2H  3S  KD  JH
2S  9C  9H  6H  8C  2C  7H  JC  4C  8D  3H  KC  7D
6C  AH  4H  5C 10D 10S  7C  9S  KH  4D  AD  5S  AS
```

Then, each of the two parts is used to scramble half of the deck again; the transpositions above depended on the order of letters in each part, but this step will instead depend on which letters are present.

Go through the alphabet, from A through Z, as you take cards from the top of the deck. When you reach a letter that is part of the current part of the keyphrase, that card completes the current pile you are making. The next card starts a new pile. Z always completes the last pile, even if it is not present.

Then put the piles back together, but in the reverse order in which they were obtained.

Thus, the first part, THEQUICK, divides the first half of the deck like this:

A B C
7S QH 3D
D E
6S 3C
F G H I
2D QD 4S 8S
J K
JS KD
L M N O P Q
10H QC 8H 5H AC 6D
R S T
KS 10C QS
U
9D
V W X Y Z
5D 2H 3S KD JH

causing that half of the deck to be placed in the order:

5D 2H 3S KD JH 9D KS 10C QS 10H QC 8H 5H
AC 6D JS KD 2D QD 4S 8S 6S 3C 7S QH 3D

Then, the second half, BROWNFOX, is applied to the second half of the deck.

When there is an odd part of the key phrase, then the deck is transposed with that part, and only its first half is mixed again.

Once the entire keyphrase is applied to the deck of cards, the deck is then subjected to a *non-invertible* triple cut, as follows:

From each end of the deck, a card with a value from A to 10 is located, by the procedure used to find the keystream numbers in normal encipherment. Then, starting from the top of the deck when it is face down, which we will assume is placed on the left, additional cards are counted from that card according to its value: one more card if it is an ace, two more if it is a deuce, and so on, but this time, face cards are *not* ignored.

This part of the deck is then placed on the right-hand side.

The procedure is repeated from the other keystream card, again counting inwards. If cards are left, these stay in the middle, and those from the bottom of the deck to the end of the count are placed on the left-hand side.

Using the previous example of obtaining a keystream digit to illustrate how this works:

```
 KS  JH  JC  AH  5H 10C  QS  2H  9H  8C//KC  AD  6D
        (1) 1*  1  2  3  4  5
 5D  KD  7H  6C  AS  9C  8D  5C  AC  9D  3S  6H  2C
 7D  5S  3H 10D  4D  9S 10S//2S  4C  4H  KH  7C  8H
                     7  6  5  4  3  2
 QC 10H  JD  JS  8S  4S  QD  2D  3C  6S  3D  QH  7S
 1  7*        6  5      4  3  2  1      (7)
```

the pairs of slashes indicate the points at which the deck will be cut, ending up in the order 2S...7S, KC...10S, KS...8C from the order above.

Finally, the cards are laid out according to the word spacing of the keyphrase:

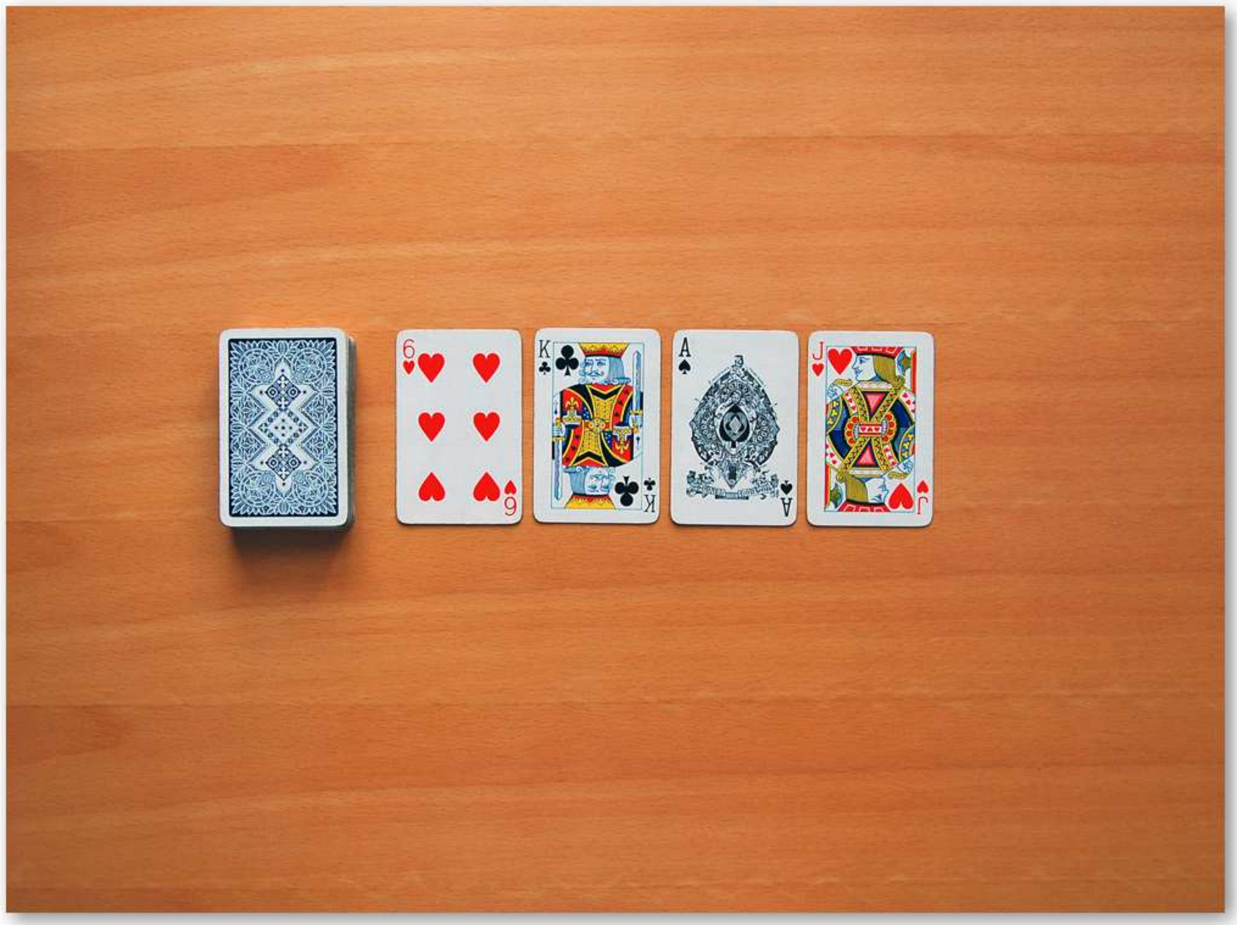
```
 T   H   E
 2S  4C  4H
 Q   U   I   C   K
 KH  7C  8H  QC 10H
 B   R   O   W   N
 JD  JS  8S  4S  QD
 F   O   X
 2D  3C  6S
 J   U   M   P   E   D
 3D  QH  7S  KC  AD  6D
 O   V   E   R
 5D  KD  7H  6C
 T   H   E
 AS  9C  8D
 L   A   Z   Y
 5C  AC  9D  3S
 D   O   G
 6H  2C  7D
 T   H   E
 5S  3H 10D
 Q   U   I   C   K
 4D  9S 10S  KS  JH
 B   R   O   W   N
 JC  AH  5H 10C  QS
 F   O   X
 2H  9H  8C
```

repeated until the deck is all laid out, and then they are picked up in face up form with the last column, and its top card, on the bottom.

In the example, that leads to the cards ending up in this order when turned face down:

```
 6D 10H  QD  AD  JH  QS  QC  4S  KC  6C  3S  KS 10C
 4H  8H  8S  6S  7S  7H  8D  9D  7D 10D 10S  5H  8C
 4C  7C  JS  3C  QH  KD  9C  AC  2C  3H  9S  AH  9H
 2S  KH  JD  2D  3D  5D  AS  5C  6H  5S  4D  JC  2H
```

The Solitaire Encryption Algorithm



---

version 1.2, 5/26/99

Designed by Bruce Schneier
Featured in Neal Stephenson's *Cryptonomicon*

In Neal Stephenson's novel *Cryptonomicon*, the character Enoch Root describes a cryptosystem code-named "Pontifex" to another character named Randy Waterhouse, and later reveals that the steps of the algorithm are intended to be carried out using a deck of playing cards. These two characters go on to exchange several encrypted messages using this system. The system is called "Solitaire" (in the novel, "Pontifex" is a code name intended to temporarily conceal the fact that it employs a deck of cards) and I designed it to allow field agents to communicate securely without having to rely on electronics or having to carry incriminating tools. An agent might be in a situation where he just does not have access to a computer, or may be prosecuted if he has tools for secret communication. But a deck of cards...what harm is that?

Solitaire gets its security from the inherent randomness in a shuffled deck of cards. By manipulating this deck, a communicant can create a string of "random" letters that he then combines with his message. Of course Solitaire can be simulated on a computer, but it is designed to be implemented by hand.

Solitaire may be low-tech, but its security is intended to be high-tech. I designed Solitaire to be secure even against the most well-funded military adversaries with the biggest computers and the smartest cryptanalysts. Of course, there is no guarantee that someone won't find a clever attack against Solitaire (watch this space for updates), but the algorithm is certainly better than any other pencil-and-paper cipher I've ever seen.

It's not fast, though. It can take an evening to encrypt or decrypt a reasonably long message. In David Kahn's book *Kahn on Codes,* he describes a real pencil-and-paper cipher used by a Soviet spy. Both the Soviet algorithm and Solitaire take about the same amount of time to encrypt a message: most of an evening.

Encrypting with Solitaire

Solitaire is an output-feedback mode stream cipher. Sometimes this is called key-generator (KG in U.S. military speak). The basic idea is that Solitaire generates a stream, often called a ``keystream,'' of numbers between 1 and 26. To encrypt, generate the same number of keystream letters as plaintext letters. Then add them modulo 26 to plaintext letters, one at a time, to create the ciphertext. To decrypt, generate the same keystream and subtract, modulo 26 from the ciphertext to recover the plaintext. (Don't worry, I'll explain "modulo" in a minute.)

For example, to encrypt the first Solitaire message mentioned in Stephenson's novel, "DO NOT USE PC":

1. Split the plaintext message into five-character groups. (There is nothing magical about five-character groups; it's just tradition.) Use X's to fill in the last group. So if the message is "DO NOT USE PC" then the plaintext is:

```
DONOT  USEPC
```

2. Use Solitaire to generate ten keystream letters. (Details are below.) Assume they are:

```
KDWUP  ONOWT
```

3. Convert the plaintext message from letters into numbers, A=1, B=2, etc:

```
4 15 14 15 20   21 19 5 16 3
```

4. Convert the keystream letters similarly:

```
11 4 23 21 16   15 14 15 23 20
```

5. Add the plaintext number stream to the keystream numbers, modulo 26. (All this means is, if the sum is more than 26, subtract 26 from the result.) For example, 1+1=2, 26+1=27, and 27-26=1, so 26+1=1.

    15 19 11 10 10   10 7 20 13 23

6. Convert the numbers back to letters.

    OSKJJ JGTMW

If you're really good at this, you can learn to add letters in your head, and just add the letters from steps (1) and (2). It just takes practice. It's easy to remember that A+A=B; remembering that T+Q=K is harder.

Decrypting with Solitaire

The basic idea is that the receiver generates the same keystream, and then subtracts the keystream letters from the ciphertext letters.

1. Take the ciphertext message and put it in five-character groups. (It should already be in this form.)

    OSKJJ JGTMW

2. Use Solitaire to generate ten keystream letters. If the receiver uses the same key as the sender, the keystream letters will be the same:

    KDWUP ONOWT

3. Convert the ciphertext message from letters into numbers:

    15 19 11 10 10   10 7 20 13 23

4. Convert the keystream letters similarly:

    11 4 23 21 16   15 14 15 23 20

5. Subtract the keystream numbers from the ciphertext numbers, modulo 26. For example, 22-1=21, 1-22=5. (It's easy. If the first number is less than or equal to the second number, add 26 to the first number before subtracting. So 1-22=? becomes 27-22=5.)

    4 15 14 15 20   21 19 5 16 3

6. Convert the numbers back to letters.

    DONOT USEPC

As you can see, decryption is the same as encryption, except that you subtract the keystream from the ciphertext message.

Generating the Keystream Letters

This is the heart of Solitaire. The above descriptions of encryption and decryption work for any output-feedback mode stream cipher. It's the way RC4 works. It's the way OFB mode for DES works. This section is specific to Solitaire, and explains how Solitaire generates those keystream letters.

Solitaire generates its keystream using a deck of cards. You can think of a 54-card deck (remember the two jokers) as a 54-element permutation. There are 54!, or about 2.31 * 10^71, possible different orderings of a deck. Even better, there are 52 cards in a deck (without the jokers), and 26 letters in the alphabet. That kind of coincidence is just too good to pass up.

To be used for Solitaire, a deck needs a full set of 52 cards and two jokers. The jokers must be different in some way. (This is common. The deck I'm looking at as I write this has stars on its jokers: one has a little star and the other has a big star.) Call one joker A and the other B. Generally, there is a graphical element on the jokers that is the same, but different size. Make the "B" joker the one that is "bigger." If it's easier, you can write a big "A" and "B" on the two jokers, but remember that you will have to explain that to the secret police if you ever get caught.

To initialize the deck, take the deck in your hand, face up. Then arrange the cards in the initial configuration that is the key. (I'll talk about the key later, but it's different than the keystream.) Now you're ready to produce a string of keystream letters.

Here's how to produce a single output character. This is Solitaire:

1. Find the A joker. Move it one card down. (That is, swap it with the card beneath it.) If the joker is the bottom card of the deck, move it just below the top card.

2. Find the B joker. Move it two cards down. If the joker is the bottom card of the deck, move it just below the second card. If the joker is one up from the bottom card, move it just below the top card. (Basically, assume the deck is a loop...you get the idea.)

It's important to do these two steps in order. It's tempting to get lazy and just move the jokers as you find them. This is okay, unless they are very close to each other.

So if the deck looks like this before step 1:

A 7 2 B 9 4 1

at the end of step 2 it should look like:

7 A 2 9 4 B 1

And if the deck looks like this before step 1:

3 A B 8 9 6

at the end of step 2 it should look like:

    3 A 8 B 9 6

If you have any doubt, remember to move the A joker before the B joker. And be careful when the jokers are at the bottom of the deck. If the joker is the last card, think of it as the first card before you start counting.

3. Perform a triple cut. That is, swap the cards above the first joker with the cards below the second joker. If the deck used to look like:

    2 4 6 B 5 8 7 1 A 3 9

then after the triple cut operation it will look like:

    3 9 B 5 8 7 1 A 2 4 6

"First" and "second" jokers refer to whatever joker is nearest to, and furthest from, the top of the deck. Ignore the "A" and "B" designations for this step.

Remember that the jokers and the cards between them don't move; the other cards move around them. This is easy to do in your hands. If there are no cards in one of the three sections (either the jokers are adjacent, or one is on top or the bottom), just treat that section as empty and move it anyway. If the deck used to look like:

    B 5 8 7 1 A 3 9

then after the triple cut operation it will look like:

    3 9 B 5 8 7 1 A

A deck that looks like:

    B 5 8 7 1 A

will remain unchanged by this step.

4. Perform a count cut. Look at the bottom card. Convert it into a number from 1 through 53. (Use the bridge order of suits: clubs, diamonds, hearts, and spades. If the card is a club, it is the value shown. If the card is a diamond, it is the value plus 13. If it is a heart, it is the value plus 26. If it is a spade, it is the value plus 39. Either joker is a 53.) Count down from the top card that number. (I generally count 1 through 13 again and again if I have to; it's easier than counting to high numbers sequentially.) Cut after the card that you counted down to, leaving the bottom card on the bottom. If the deck used to look like:

    7 ... cards .. 4 5
    ... cards ... 8 9

and the ninth card was the 4, the cut would result in:

```
5 ... cards ... 8 7
... cards ... 4 9
```

The reason the last card is left in place is to make the step reversible. This is important for mathematical analysis of its security.

A deck with a joker as the bottom card will remain unchanged by this step.

Be sure not to reverse the order when counting cards off the top. The correct way to count is to pass the cards, one at a time, from one hand to another. Don't make piles on the table.

5. Find the output card. To do this, look at the top card. Convert it into a number from 1 through 53 in the same manner as step 4. Count down that many cards. (Count the top card as number one.) Write the card after the one you counted to on a piece of paper; don't remove it from the deck. (If you hit a joker, don't write anything down and start over again with step 1.) This is the first output card. Note that this step does not modify the state of the deck.

6. Convert the output card to a number. As before, use the bridge suits to order them. From lowest to highest, we have clubs, diamonds, hearts, and spades. Hence, A-clubs through K-clubs is 1 through 13, A-diamonds through K-diamonds is 14 though 26, A-hearts through K-hearts is 1 through 13, and A-spades through K-spades is 14 through 26. (We need 1 through 26, and not 1 through 52, so we can get to letters.)

That's how to use Solitaire to encrypt a single character. You can use it to create as many keystream numbers as you need; just go through the same six steps once for each output character. (Don't rekey the deck). And remember, you'll need one per message character.

I know that there are regional differences in decks of cards, depending on the country. In general, it does not matter what suit ordering you use, or how you convert cards to numbers. What matters is that the sender and the receiver agree on the rules. If you're not consistent you won't be able to communicate.

Keying the Deck
Before you start producing output cards, you have to key the deck. This is probably the most important part of the whole operation, and the one that the entire security of the system hinges upon. Solitaire is only as secure as the key. That is, the easiest way to break Solitaire is to figure out what key the communicants are using. If you don't have a good key, none of the rest of this matters. Here are some suggestions for exchanging a key.

1. Use identically shuffled decks. A random key is the best. One of the communicants can shuffle up a random deck and then create another, identical deck. One goes to the sender and the other to the receiver. Most people are not good shufflers, so shuffle the deck at least six times. And both parties should keep an additional spare deck in the same keyed order, otherwise if you make a mistake you'll never be able to decrypt the message. Also remember that the key is at risk as long as it exists; the secret police could find the deck and copy down its order.

2. Use a bridge ordering. A description of a set of bridge hands that you might see in a newspaper or a bridge book is about a 95-bit key. Agree on a way to take the bridge-hand diagram and convert it into an ordering of the deck. Then agree on a way to put the two jokers into the deck. (One obvious one is to put the A joker after the first card mentioned in the text, and the B joker after the second card mentioned in the text.)

Be warned, though: the secret police can find your bridge column and copy down the order. You can try setting up some repeatable convention for which bridge column to use; for example, "use the bridge column in your hometown newspaper for the day on which you encrypt the message," or something like that. Or use a list of keywords to the *New York Times* website, and use the bridge column for the day of the article that comes up when you search on those words. If the keywords are found or intercepted, they look like a passphrase. And pick your own convention for converting bridge columns into deck orderings; remember that the secret police read Neal Stephenson's books, too.

3. Use a passphrase to order the deck. This method uses the Solitaire algorithm to create an initial deck ordering. Both the sender and receiver share a passphrase. (For example, "SECRET KEY.") Start with the deck in a fixed order; lowest card to highest card, in bridge suits, followed by the A and then the B joker. Perform the Solitaire operation, but instead of Step 5, do another count cut based on the first character of the passphrase (19, in this example). In other words, do step 4 a second time, using 19 as the cut number instead of the last card. Remember to put the top cards just above the bottom card in the deck, as before.

Repeat the five steps of the Solitaire algorithm once for each character of the key. That is, the second time through the Solitaire steps use the second character of the key, the third time through use the third character, etc.

Optional step: (This is NOT used in the examples below.) Use the final two characters to set the positions of the jokers. If the second to last character is a G (a 7), put the A joker after the seventh card. If the last character is a T (a 20), put the B joker after the twentieth card.

Remember, though, that there are only about 1.4 bits of randomness per character in standard English. You're going to want at least an 64-character passphrase to make this secure; I recommend at least 80 characters, just in case. Sorry; you just can't get good security with a shorter key.

Sample Output
Here's some sample data to practice your Solitaire skills with:

Sample 1: Start with an unkeyed deck: A-clubs to K-clubs, A-diamonds to K-diamonds, A-hearts to K-hearts, A-spades to K-spades, A joker, B joker. (You can think of this as 1 ... 52, A, B.)

Here's how to generate the first two outputs. The initial deck is:

1 2 3 4 ... 52 A B

After the first step (moving the A joker):

1 2 3 4 ... 52 B A

After the second step (moving the B joker):

1 B 2 3 4 ... 52 A

After the third step (the triple cut):

B 2 3 4 ... 52 A 1

After the fourth step (the count cut):

2 3 4 ... 52 A B 1

The last card is a 1, which means cut one card. Remember that the 1 stays where it is, so the one card (the B, moves to the bottom of the deck just above the 1.

The fifth step does not change the deck, but produces an output card. The top card is a 2, so count down two cards to the 4. The first Solitaire output is 4. (Of course, you're not supposed to remove this card from the deck. Keep the 4 where it is; just write it down somewhere.)

To produce the second Solitaire output, go through the five steps again.

Step 1:

2 3 4 ... 49 50 51 52 B A 1

Step 2:

2 3 4 ... 49 50 51 52 A 1 B

Step 3:

A 1 B 2 3 4 ... 49 50 51 52

Step 4:

51 A 1 B 2 3 4 ... 49 50 52

The last card is a 52, so count 52 cards down to the 51. Cut the single card, the 51 with the rest of the deck. Remember that the 52 remains unmoved.

Step 5 produces the output card. The first card is a 51. Counting down fifty-one cards gets to the 49, which is the second output card. (Again, don't remove the 49 from the deck.)

The first ten outputs are:

4 49 10 (53) 24 8 51 44 6 4 33

The 53 is skipped, of course. I just put it there for demonstration.

If the plaintext is

AAAAA AAAAA

then the ciphertext is:

EXKYI ZSGEH

Sample 2: Using keying method 3 and the key "FOO," (remember that the optional keying step is not used in these examples) the first fifteen outputs are:

8 19 7 25 20 (53) 9 8 22
32 43 5 26 17 (53) 38 48

If the plaintext is all As, then the ciphertext is:

ITHZU JIWGR FARMW

Sample 3: Using keying method 3 and the key "CRYPTONOMICON," the message "SOLITAIRE" encrypts to:

KIRAK SFJAN

Remember that Xs are used to fill out the last five-character group.

Of course you should use a longer key. These samples are for test purposes only. There are more samples on the website, and you can use the PERL script to create your own.

Real Security, Not Security Through Obscurity

Solitaire is designed to be secure even if the enemy knows how the algorithm works. I have assumed that "Cryptonomicon" will be a best seller, and that copies will be available everywhere. I assume that the NSA and everyone else will study the algorithm and will watch for it. I assume that the only secret is the key.

That's why keeping the key secret is so important. If you have a deck of cards in a safe place, you should assume the enemy will at least entertain the thought that you are using Solitaire. If you have a bridge column in your safe deposit box, you should expect to raise a few eyebrows. If any group is known to be using the algorithm, expect the secret police to maintain a database of bridge columns to use in cracking attempts. Solitaire is strong even if the enemy knows you are using it, and a simple deck of playing cards is still much less incriminating than a software encryption program running on your laptop, but the algorithm is no substitute for street smarts.

Operational Notes

1. The first rule of an output-feedback mode stream cipher, any of them, is that you should never use the same key to encrypt two different messages. Repeat after me: NEVER USE THE SAME KEY TO ENCRYPT TWO DIFFERENT MESSAGES. If you do, you completely break the security of the system. Here's why: if you have two ciphertext streams, A+K and B+K, and you subtract

one from the other, you get (A+K)-(B+K) = A+K-B-K = A-B. That's two plaintext streams combined with each other with no key involved, and is very easy to break. Trust me on this one: you might not be able to recover A and B from A-B, but a professional cryptanalyst can. This is vitally important: never use the same key to encrypt two different messages.

2. Keep your messages short. This algorithm is designed to be used with small messages: a couple of thousand characters at most. Use shorthand, abbreviations, and slang in your messages. Don't be chatty. If you have to encrypt a 100,000-word novel, use a computer algorithm.

3. Like all output-feedback stream ciphers, this system has the unfortunate feature of never recovering from a mistake. If you're encrypting a message, and you make a mistake in one of the operations, every letter afterwards will be encrypted wrong. You won't be able to decrypt it, even with the key. And you'll never know. So if you're encrypting a message, go through the encryption process twice to make sure they agree. If you're decrypting, check to make sure the message makes sense as you decrypt it. And if you're keying from a random deck, keep a spare copy of the ordered deck for this reason.

4. Solitaire is reversible. This means that if you leave the deck lying around after you've encrypted your message, the secret police can find it and work the algorithm backwards using the deck. This process can recover all of the output cards and decrypt a message. It is important that you shuffle the deck completely, six times, after you finish encrypting a message.

5. For maximum security, try to do everything in your hands and head. If the secret police starts breaking down your door, just calmly shuffle the deck. (Don't throw it up in the air; you'd be surprised how much of the deck ordering is maintained during a game of 52-Pickup.) Remember to shuffle the backup deck, if you have one.

6. Be careful about worksheets, if you have to write things down. They will have sensitive information on them.

Burning is probably the best method of data destruction available, but think about the paper. Ungummed, rice cigarette papers seem ideally suited to this role. A colleague did some tests with Club Cabaret Width papers, and they burn completely.

It's not as difficult to write on cigarette papers as you might think. Using a No. 2 pencil with a fine but blunt tip works well. A No. 3 pencil works better, but it is a bit weirder to be carrying. Pens have a number of problems. First the extremely hard tip of the pen is more likely to leave impressions in the surface below the paper. Also, anything with ink has the possibility to bleed through to the surface below.

And good cigarette papers are made to burn cleanly and completely. The Club papers burned best when allowed to burn in the free air. That is, lit and released at about chest level. These papers also have the advantage of having very low volume and could be easily eaten if required.

They are also extremely thin. These three-inch square papers can be folded six times into a 1 cm square which is about 1 mm thick. One paper can comfortably hold 80 characters in 8 rows of two five letter blocks each. It seems quite possible a reasonably careful writer could fit 120 characters.

7. Solitaire can work on computers. Often only one end of the communications has to use the deck of cards; the other is safe enough to use a computer. Use a computer when you can: it's faster and the computer never makes mistakes.

8. Most card games do not include jokers, so carrying a deck around with them may be suspicious. Be prepared with a story.

9. The security of Solitaire does not depend on the secrecy of the method. I assume that the secret police know that you're using it.

Security Analysis
Properties of the Transformation Semigroup of the Solitaire Stream Cipher (B. Pogorelov and M. Pudovkina)
Problems with Bruce Schneier's "Solitaire" (Paul Crowley)

Learning More

I recommend my own book, *Applied Cryptography* (John Wiley & Sons, 1996), as a good place to start. Then read *The Codebreakers*, by David Kahn (Scribner, 1996). After that, there are several books on computer cryptography, and a few others on manual cryptography. It's a fun field; good luck.

Downloads
test vectors - Perl - Ada - C (#1) - C (#2) - C (#3) - C++ - C++ GUI - C# (#1) - C# (#2) - Clojure - Delphi (#1) - Delphi (#2) - Erlang - Forth (#1) - Forth (#2) - Java - Javascript - K - Palm OS - Pascal - Perl - Perl CGI - PHP - Python (#1) - Python (#2) - Ruby - TCL

Note: only the Perl implementation has been tested by Counterpane.

The **Solitaire** cryptographic algorithm was designed by Bruce Schneier "to allow field agents to communicate securely without having to rely on electronics or having to carry incriminating tools",[1] at the request of Neal Stephenson for use in his novel *Cryptonomicon*. It was designed to be a manual cryptosystem calculated with an ordinary deck of playing cards. In *Cryptonomicon*, this algorithm was originally called **Pontifex** to hide the fact that it involved playing cards.

One of the motivations behind Solitaire's creation is that in totalitarian environments, a deck of cards is far more affordable (and less incriminating) than a personal computer with an array of cryptological utilities. However, as Schneier warns in the appendix of *Cryptonomicon*, just about everyone with an interest in cryptanalysis will know about this algorithm.

Encryption and decryption

The algorithm generates a stream of values which are combined with the message to encrypt and decrypt it. Each value of the keystream is to be used for one value of the message, thus the keystream will need to be the same length as the message.

1. Remove all punctuation and convert the characters to the same case.
2. Convert all the characters to their natural numerical values, A = 1, B = 2, etc., Z = 26.
3. To encrypt a message, add each keystream value to its corresponding character in the plaintext, rolling over back to 1 if the resulting value exceeds 26 (modulo 26 arithmetic). To decrypt, subtract each keystream value from its corresponding character in the ciphertext, rolling back up to 26 if the resulting value should be lower than 1. (In mathematics this is called modular arithmetic.)

Keystream Algorithm

This algorithm assumes that the user has a deck of cards and two jokers which are distinguishable from each other. For simplicity's sake, only two suits will be used in this example. Each card will be assigned a numerical value: the first suit of cards will be numbered from 1 to 13 (Ace through King) and the second suit will be numbered 14 through 26 in the same manner. The jokers will be assigned the values of 27 and 28. Thus, a 5 from the first suit would have the value 5 in our combined deck, the value 1 in the second suit would have the value 14 in the combined deck.

The deck will be assumed to be a circular array, meaning that should a card ever need to advance below the bottom card in the deck, it will simply rotate back to the top (in other words, the first card follows the last card).

1. Arrange the deck of cards face-up according to a specific key. This is the most important part as anyone who knows the deck's starting value can easily generate the same values from it. How the deck is initialized is up to the recipients, shuffling the deck perfectly randomly is preferable, although there are many other methods. For this example, the deck will simply start at 1 and count up by 3's, modulo 28. Thus the starting deck will look like this:
   - 1 4 7 10 13 16 19 22 25 28 3 6 9 12 15 18 21 24 27 2 5 8 11 14 17 20 23 26
2. Locate the first joker (value 27) and move it down the deck by one place, basically just exchanging with the card below it. Notice that if it is the last card, it becomes the second card. There is no way to become the first card. The deck now looks like this:
   - 1 4 7 10 13 16 19 22 25 28 3 6 9 12 15 18 21 24 2 **27** 5 8 11 14 17 20 23 26
3. Locate the second joker (value 28) and move it down the deck by two places. Notice that if it is the second to last card, it becomes the second card by wrapping around. If it is the last card, it becomes the third card. There is no way to become the first card.
   - 1 4 7 10 13 16 19 22 25 3 6 **28** 9 12 15 18 21 24 2 27 5 8 11 14 17 20 23 26
4. Perform a triple-cut on the deck. That is, split the deck into three sections. Everything above the top joker (which, after several repetitions, may not necessarily be the first

joker) and everything below the bottom joker will be exchanged. The jokers themselves, and the cards between them, are left untouched.

   - **5 8 11 14 17 20 23 26** 28 9 12 15 18 21 24 2 27 **1 4 7 10 13 16 19 22 25 3 6**

5. Observe the value of the card at the bottom of the deck, if the card is either joker let the value just be 27. Take that number of cards from the top of the deck and insert them back to the bottom of the deck just above the last card.
   - 23 26 28 9 12 15 18 21 24 2 27 1 4 7 10 13 16 19 22 25 3 **5 8 11 14 17 20** 6
6. Note the value of the top card. Count this many places below that card and take the value of the card there. This value is the next value in the keystream, in this example it would be 8. (Note that no cards are changing places in this step, this step simply determines the value).
7. Repeat steps 2 through 6 for as many keystream values as required.

It is worth noting that because steps 2 and 3 have the wrap around feature, there are two configurations that can lead the same configuration on a step. For instance, when the little joker is either on the bottom of the deck or on the top of the deck it will become the second card after step 2. This means the algorithm is not always reversible.

Securely Encrypt Messages with a Deck of Cards and Solitaire

Neal Stephenson's cypherpunk novel Cryptonomicon contains a cryptosystem called Pontifex. This low-tech cryptographic algorithm uses a deck of playing cards to encrypt and decrypt messages.

Outside of the book, this algorithm is actually called Solitaire. It was designed by cryptographer and security expert Bruce Schneier at the request of Neal Stephenson. Solitaire allows secure communications without having to rely on computers or other tools that might indicate that cover channels are being used, or where access to a computer is not possible. It was designed to be secure even against the most well-funded adversaries with the biggest computers and the smartest cryptanalysts.

Solitaire gets its security from the inherent randomness of a shuffled deck of cards. Using this deck, keyed in a special way, two people can create a set of random letters that will be use to encrypt the messages. The process is somewhat slow, but it's hard to spot that a deck of cards is being used to encrypt information.

Using Solitaire

For the cypherpunks out there, or crypto aficionados, Solitaire is an output-feedback mode stream cipher. As Schneier explains:

The basic idea is that Solitaire generates a stream, often called a "keystream", of numbers between 1 and 26. To encrypt, generate the same number of keystream letters as plaintext letters. Then add them modulo 26 to plaintext letters, one at a time, to create the ciphertext. To decrypt, generate the same keystream and subtract, modulo 26 from the ciphertext to recover the plaintext.

Encryption

Using Schneier's own example, this is how you encrypt a message using Solitaire. For this example, our plaintext message will be the same message used in Cryptonomicon: DO NOT USE PC.

1. Split the plaintext message into five-character groups. Use X's to fill in the last group if necessary.

   DONOT USEPC

2. Use Solitaire to generate ten keystream letters. For example:

   KDWUP ONOWT

3. Convert the plaintext message from letters into numbers, A=1, B=2, etc:

   4 15 14 15 20 21 19 5 16 3

4. Convert the keystream letters similarly:

   11 4 23 21 16 15 14 15 23 20

5. Add the plaintext number stream to the keystream numbers, modulo 26. Modulo 26 means that if the sum is more than 26, subtract 26 from the result. For example, 1+1=2, 26+1=27, and 27-26=1, so 26+1=1.

   15 19 11 10 10 10 7 20 13 23

6. Convert the numbers back to letters to get your ciphertext.

   OSKJJ JGTMW

So DO NOT USE PC just became OSKJJJGTMW.

Decryption

The receiver of the message has to key the same keystream as the person encrypting. That is your encryption key. Then the receiver subtracts the keystream letters from the ciphertext letters.

1. Take the ciphertext message and put it in five-character groups.

   OSKJJ JGTMW

2. Use Solitaire to generate ten keystream letters. If the receiver has the same key as the sender, the keystream letters will be the same:

   KDWUP ONOWT

3. Convert the ciphertext message from letters into numbers:

   15 19 11 10 10   10 7 20 13 23

4. Convert the keystream letters similarly:

   11 4 23 21 16   15 14 15 23 20

5. Subtract the keystream numbers from the ciphertext numbers, modulo 26. For example, 22-1=21, 1-22=5. If the first number is less than or equal to the second number, add 26 to the first number before subtracting. So 1-22=? becomes 27-22=5.

   4 15 14 15 20   21 19 5 16 3

6. Convert the numbers back to letters to get your plaintext.

   DONOT USEPC

Generating the Keystream Letters

Generating the keystream letters is the heart of Solitaire. The keystream is generated using a deck of cards. In a 54-card deck (52 + 2 jokers) there are 54!, or about 2.31 * 10^71, possible different orderings of a deck.

We need a deck set of 52 cards and two jokers. The jokers must be visually different. One will be called “joker A” and the other “joker B”. To initialize the deck, take it in your hand face up. Then arrange the cards in the initial configuration that is the key. You’re now ready to generate keystream letters.

Here’s how to produce a single output character:

1. Find the A joker. Move it one card down, swapping it with the card beneath it. If the joker is the bottom card of the deck, move it just below the top card.

2. Find the B joker. Move it two cards down. If the joker is the bottom card of the deck, move it just below the second card. If the joker is one up from the bottom card, move it just below the top card. (Basically, assume the deck is a loop.)It's important to do these two steps in order. It's tempting to get lazy and just move the jokers as you find them. This is okay, unless they are very close to each other.So if the deck looks like this before step 1:

   A 7 2 B 9 4 1

   at the end of step 2 it should look like:
   7 A 2 9 4 B 1

   And if the deck looks like this before step 1:
   3 A B 8 9 6

   at the end of step 2 it should look like:
   3 A 8 B 9 6

3. Perform a triple cut. That is, swap the cards above the first joker with the cards below the second joker. If the deck used to look like:

   2 4 6 B 5 8 7 1 A 3 9

   then after the triple cut operation it will look like:
   3 9 B 5 8 7 1 A 2 4 6

   "First" and "second" jokers refer to whatever joker is nearest to and furthest from the top of the deck, respectively. Ignore the "A" and "B" designations for this step. Remember that the jokers and the cards between them don't move; the other cards move around them. This is easy to do in your hands. If there are no cards in one of the three sections (either the jokers are adjacent, or one is on top or the bottom), just treat that section as empty and move it anyway.

4. Perform a count cut. Look at the bottom card. Convert it into a number from 1 through 53. (Use the bridge order of suits: clubs, diamonds, hearts, and spades. If the card is a club, it is the value shown. If the card is a diamond, it is the value plus 13. If it is a heart, it is the value plus 26. If it is a spade, it is the value plus 39. Either joker is a 53.) Use that number to count down from the top card. Cut after the card that you counted down to, leaving the bottom card on the bottom. If the deck used to look like:
5. 7 ... cards .. 4 5
   ... cards ... 8 9

   and the ninth card was the 4, the cut would result in:
   5 ... cards ... 8 7
   ... cards ... 4 9

The reason the last card is left in place is to make the step reversible. This is important for mathematical analysis of its security. A deck with a joker as the bottom card will remain unchanged by this step.

Be sure not to reverse the order when counting cards off the top. The correct way to count is to pass the cards, one at a time, from one hand to another. Don't make piles on the table.

6. Find the output card. To do this, look at the top card. Convert it into a number from 1 through 53 in the same manner as in the previous step. Count down that many cards. (Count the top card as number one.) Write the card after the one you counted to on a piece of paper; don't remove it from the deck. (If you hit a joker, don't write anything down and start over again with step 1.) This is the first output card. Note that this step does not modify the state of the deck.
7. Convert the output card to a number. As before, use the bridge suits to order them. From lowest to highest, we have clubs, diamonds, hearts, and spades. Hence, A-clubs through K-clubs is 1 through 13, A-diamonds through K-diamonds is 14 though 26, A-hearts through K-hearts is 1 through 13, and A-spades through K-spades is 14 through 26. (We need 1 through 26, and not 1 through 52, so we can get to letters.)

That's how to use Solitaire to encrypt a single character. You can use it to create as many keystream numbers as you need; just go through the same six steps once for each output character. (Don't rekey the deck). And remember, you'll need one per message character.

Keying the Deck

Before you start producing output cards, you have to key the deck. This is probably the most important part of the whole operation, and the one that the entire security of the system hinges upon. Solitaire is only as secure as the key. That is, the easiest way to break Solitaire is to figure out what key the communicants are using. If you don't have a good key, none of the rest of this matters. Here are some suggestions for exchanging a key.

Use identically shuffled decks

A random key is the best. One of the communicants can shuffle up a random deck and then create another, identical deck. One goes to the sender and the other to the receiver. Most people are not good shufflers, so shuffle the deck at least six times. Both parties should keep an additional spare deck in the same keyed order, otherwise if you make a mistake you'll never be able to decrypt the message.

Use a bridge ordering

A description of a set of bridge hands that you might see in a newspaper or a bridge book is about a 95-bit key. Agree on a way to take the bridge-hand diagram and convert it into an ordering of the deck. Then agree on a way to put the two jokers into the deck. (One obvious one is to put the A joker after the first card mentioned in the text, and the B joker after the second card mentioned in the text.)

Use a passphrase to order the deck

This method uses the Solitaire algorithm to create an initial deck ordering. Both the sender and receiver share a passphrase. (For example, “SECRET KEY.”) Start with the deck in a fixed order; lowest card to highest card, in bridge suits, followed by the A and then the B joker. Perform the Solitaire operation, but instead of Step 5, do another count cut based on the first character of the passphrase (19, in this example). In other words, do step 4 a second time, using 19 as the cut number instead of the last card. Remember to put the top cards just above the bottom card in the deck, as before.

Repeat the five steps of the Solitaire algorithm once for each character of the key. That is, the second time through the Solitaire steps use the second character of the key, the third time through use the third character, etc.

Remember, though, that there are only about 1.4 bits of randomness per character in standard English. You’re going to want at least an 64-character passphrase to make this secure; Bruce Schneier recommend at least 80 characters.

Low-Tech Security

The Solitaire algorithm offers a secure and low-tech, if slow, solution to those who wish to communicate privately. Bruce Schneier has explained that he expected Cryptonomicon to be a best-seller, and that everyone would know the intricacies of how the algorithm works. As with any cryptographic algorithm, the success lies with the secrecy of the key. As long as the key remains secret, it is unlikely that any third-party will be able to decrypt ciphertext that has been encrypted with Solitaire.

For more information regarding cryptanalysis, operational notes and security visit to Bruce Schneier’s Solitaire site.

David in Israel on Practical "Pocket" Cryptography
Permalink | Print

In the absence of computing power if we are reduced to using tiny QRP [low power] transmitters for communication, then there may come a time where some messages require heavy duty encryption. This is the easiest method I know of the Solitaire card deck encryption method. A group could even generate one time pads which would be starting order for a deck and store them in a secure location. See: http://www.schneier.com/solitaire.html Here is a snip from this site:

"In Neal Stephenson's novel Cryptonomicon, the character Enoch Root describes a cryptosystem code-named "Pontifex" to another character named Randy Waterhouse, and later reveals that the steps of the algorithm are intended to be carried out using a deck of playing cards. These two characters go on to exchange several encrypted messages using this system. The system is called "Solitaire" (in the novel, "Pontifex" is a code name intended to temporarily conceal the fact that it employs a deck of cards) and I designed it to allow field agents to communicate securely without having to rely on electronics or having to carry incriminating tools. An agent might be in a situation where he just does not have access to a computer, or may be prosecuted if he has tools for secret communication. But a deck of cards...what harm is that?"

[See the URL cited above, for the details on this enciphering system]

JWR Replies: Thanks for sending that, David. In the near future I plan to post a brief article about "book codes" --using two identical books as one-time pads. This method is called a *Buchspiel* ("book game") by the German spymasters that perfected it

Problems with Bruce Schneier's "Solitaire"

Readers of Neal Stephenson's "Cryptonomicon" will be familiar with the cipher "Solitaire" (called "Pontifex" in the book), which was designed by cryptologist Bruce Schneier specifically for the purposes of the book. It is intended to be the first truly secure hand cipher, and requires only a pack of cards for encryption and decryption.

As yet, the technical information about the design of the cipher has not been made available, but I decided to go investigating, and I've now written a fast "C" implementation of the cipher suitable for collecting statistics about the CPRNG at its heart. I have found two interesting facts:

- The CPRNG state machine is *not* reversible, contrary to what the operational notes claim: the initial step in which a joker is moved to the top if it is on the bottom cannot be reversed. This is surprising since non-reversible CPRNGs tend to have shorter periods and can more easily exhibit bias.
- And indeed, the output of the CPRNG is very biased. The output of each step of the CPRNG is a number from 0 to 25; you would expect successive outputs to be the same around one time in 26, but my experiments show that the frequency is closer to 1/22.5.

I'm making the source code for the tests I ran available to the public domain. Note that the problems reported in earlier versions are not present in this version; it turns out that it was only my development version that had the problems anyway and the one available for download was fine.

- C implementation of Solitaire, optimised to perform both cuts in a single copying step; on my machine I can generate 10^7 samples in 20 seconds.
- Perl implementation of Solitaire modified to do coincidence counting and take the same user interface as the C version, though of course the C version is about 250 times faster. This is useful for debugging and verifying the C version.
- Normal distribution probability calculator, for when your statistical significance gets so high you fall off the end of your normal tables. Unfortunately, the significance of the bias I've detected in Solitaire (passphrase CRYPTONOMICON, 9999999 samples, 444745 coincidences) falls off the end of what this program can calculate, at 98.87 standard deviations from the mean...

I believe I now know the main reasons why this bias arises: when the value of the top card is the same in two successive rounds, an event you would expect to occur with probability just under

2%, the probability that the output letter is also the same is very high, around 34%. This is I assume because when this happens it's likely that many other cards are also in the same positions across the rounds. This isn't the sole source of the bias but it's by far the largest component. Below is the output from my latest version of c-sol, instrumented to measure the correlations between "coincidences" (two successive outputs being the same) and "top matches" (the value of the top card being the same on two successive rounds); everything it measures shows some bias, so there's more to be found than what I've discovered so far.

```
$./c-sol BIASED 26000001
  24509422    334473
    983536    172569
Top matches overall:
Sample: 507042 / 26000000
Measured p = 0.0192044 +0.000297226 = 0.0195016 (+1.5477 %)
+11.0429 SDs from mean
Coincidences overall:
Sample: 1156105 / 26000000
Measured p = 0.0384615 +0.00600404 = 0.0444656 (+15.6105 %)
+159.196 SDs from mean
Coincidences where top matches:
Sample: 172569 / 507042
Measured p = 0.0384615 +0.301883 = 0.340345 (+784.896 %)
+1117.8 SDs from mean
Coincidences where top doesn't match:
Sample: 983536 / 25492958
Measured p = 0.0384615 +0.000119155 = 0.0385807 (+0.309803 %)
+3.12843 SDs from mean
```

Update 2001 August 13: Frans Lategan made an interesting observation on the movement of the jokers in Solitaire, which I hadn't spotted until now.

I'm interested in designing a secure hand cipher; my first attempt, Mirdek turns out to be very insecure, but I return to the problem from time to time.

Description of the Mirdek card cipher

Mirdek is a cipher you can carry out using only a pack of cards, pen and paper, and a surface to work on; it's intended to provide strong crypto to resist even the best-funded of eavesdroppers.

Cards as letters, letters as numbers

Each card corresponds to a letter of the alphabet: black Ace-King correspond to A-M, and red Ace-King to N-Z; thus, the 5 of Spades and the 5 of Clubs both represent "E", and the Jack of Hearts and of Diamonds both represent "X". While learning the cipher, it can help to write the corresponding letter on the top right of each card - though of course, you don't want to be caught with such a marked-up deck if you're trying to hide your crypto habits from the Bad Guys. Where we refer to "the letter for the card", this means the letter from the table below, so the letter for the Jack of Hearts is "X".

We use this to represent the state of a pile of cards as a sequence of letters; the first letter in the sequence is the top card when the pile is face-up, so the sequence "CAR" represents a pile of three face-up cards, with a black 3 on top and a red 5 on the bottom. If the pile is face down, the "R" will be on the top, but we still represent the sequence as "CAR".

With this representation, you can't tell what suit the card is, only what colour; but the distinction between hearts and diamonds or between spades and clubs is never relevant for Mirdek encryption, except as a handy way to divide the deck into two functionally identical sub-piles.

Each letter also corresponds to a number from 1-26; thus, counting out five cards is the same as counting to "E".

	A	B	C	D	E	F	G	H	I	J	K	L	M
Black	Ace	2	3	4	5	6	7	8	9	10	Jack	Queen	King
	1	2	3	4	5	6	7	8	9	10	11	12	13
	14	15	16	17	18	19	20	21	22	23	24	25	26
Red	Ace	2	3	4	5	6	7	8	9	10	Jack	Queen	King
	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Basic operations

Mirdek divides the pack into two equal halves, one consisting entirely of spades and diamonds, the other of hearts and clubs. We step through each card in turn in the one half and use it to shuffle the other deterministically, then swap the two over and do it again. You will need only the standard 52 cards - the jokers can be discarded.

The easiest way to try out the basic operations described here is to shuffle the pack, then sort the cards onto two piles: hearts and clubs on the left, diamonds and spades on the right. We'll refer to the "left" and "right" piles throughout this description. Once you've got the hang of the basic operations, we'll start again and describe a complete encryption process. The example encryption contains lots of examples of both operations.

Hold the left pile face up in one hand, and place the right pile face down on the table in front of you. You'll need a table in front of you, since we're going to create three more piles: a "discard" pile for cards from the right half, and two "alternating" piles for cards from the left half.

Counted cut: take the top card from the face-down right pile and place it, face-up, on the discard pile. Now count cards one by one from the top of the left pile to the bottom, saying each successive letter of the alphabet as you do so; stop after you've said the letter for the card face-up on the discard pile. For example, if the discarded card was a 5 of Clubs, this corresponds to an

"E", so 5 cards should be moved; if it were a 3 of Diamonds corresponding to "P", then 16 cards move.

Example: supposing the piles start in this state:

- Discard: **ULX**
- Right: **IPDZOWKGSTVARMEQYBCFJNH**
- Left: **JHFDBZLMNOPQRSTUVWYCGIAKXE**

Since the right pile is face-down, the top card will be "H" (a black 8). We take this card and place it on top of the discard pile. We then count eight cards from the top of the left pile (counting "A, B, C, D, E, F, G, H!") and place each in turn on the bottom. The cards moved, in sequence, are "JHFDBZLM". The cards not moved are in the sequence "NOPQRSTUVWYCGIAKXE". After the count cut, the decks are left in this state:

- Discard: **HULX**
- Right: **IPDZOWKGSTVARMEQYBCFJN**
- Left: **NOPQRSTUVWYCGIAKXEJHFDBZLM**

This maneuver may sometimes leave the right pile empty. Immediately this happens, place the left pile face-down in place of the right pile, and pick up the discard pile, which becomes the new left pile. Then turn over one card from the new right pile onto the discard pile and use it for another counted cut.

Example: supposing the piles start in this state:

- Discard: **UFKZTGODRHCWQBJAPYLVXMISN**
- Right: **E**
- Left: **CUGWTBSJXZRHOLAQFNYMKPVIED**

So we move the "E" (black 5) to the top of the discard pile, and count five cards ("CUGWT") onto the bottom of the left pile.

- Discard: **EUFKZTGODRHCWQBJAPYLVXMISN**
- Right: *empty*
- Left: **BSJXZRHOLAQFNYMKPVIEDCUGWT**

But this leaves the right pile empty, and so the special rule comes into play: swap the piles

- Discard: *empty*
- Right: **BSJXZRHOLAQFNYMKPVIEDCUGWT**
- Left: **EUFKZTGODRHCWQBJAPYLVXMISN**

and cut again, this time moving the "T" (red 7) to the discard pile and moving 20 cards ("EUFKZTGODRHCWQBJAPYL") to the bottom of the new left pile.

- Discard: **T**
- Right: **BSJXZRHOLAQFNYMKPVIEDCUGW**

- Left: **VXMISNEUFKZTGODRHCWQBJAPYL**

Letter search: Choose a letter to search for; what determines this depends on how the search is used and is described later. Deal cards from the top of the left pile onto the two alternating piles, which (as the name suggests) each receive cards in turn just as with a normal deal, and stop immediately the card matching the searched-for letter is dealt. Now place the pile containing the searched-for card on top of the other pile (if there are any cards in it), and put them underneath the left pile in your hand. The right and discard piles are wholly unaffected.

Example: assume we're searching for the letter "S" and the left pile starts in this state:

- Left: **EJFDBZLMPNOQRSTUVWYAGICKXH**

So we start dealing the cards into two face-up piles: the "E" (black 5) onto one, the J (black 10) onto another, the "F" face-up on top of the "E" and the "D" face up onto the "J"... By the time we've dealt the "S" (red 6), we're left with two piles "ROPLBFE" and "SQNMZDJ" on the table, with "TUVWYAGICKXH" left in our hand. Gather together the two piles on the table in the other hand, making sure the searched-for "S" is on top ("SQNMZDJROPLBFE") and put that underneath the cards in your hand, to form

- Left: **TUVWYAGICKXHSQNMZDJROPLBFE**

Initialisation

These two basic operations are all that is needed to encrypt a message with Mirdek. Encryption is preceded by a setup phase in three parts: initialisation, keying, and mixing.

Sort the deck into suits, and sort each suit in order, with Aces face up on the top. Place the spades on top of the diamonds to make the left pile, and the clubs on top of the hearts to make the right pile. If the cards were marked with letters as suggested before, then each pile will read "A-Z" in order.

Now pick up the right pile (with the clubs and hearts) and shuffle them thoroughly. This introduces some randomness into the encryption process which allows the same passphrase to be used many times without harming security. Holding the deck face-up in your hand, take each card in turn from the top and write down the letter for the card, arranging them in groups of five letters; make a face-down pile of the cards you've recorded. Place the last card on the pile without writing down the letter; you should only write 25 letters. These form the first 25 letters of the ciphertext.

You now have the right pile face down on the table; pick up the left pile (diamonds and spades) and hold it face up in your hand, ready for the keying phase.

Keying

Keying consists of two alternating steps. For each letter in the keyphrase, take the following two steps:

- Make a counted cut.
- Then do a letter search for the next unused letter of the keyphrase.

If your keyphrase has fewer than 26 letters, then when you finish the number of letters in the keyphrase will be the number of cards in the discard pile.

Mixing

After the keying phase is complete, the next phase thoroughly mixes the state of the deck before encryption starts. Place the rest of the right hand pile **below** the discard pile (so the top card of the face-up discard pile is unchanged), put down the left pile face-down as the new right pile, and pick up the discard pile which becomes the new left pile. Then, for each card in the right deck in turn, starting with the face-down top card, convert the card to a letter and perform a letter search (on the left pile - letter search is always on the left pile). Once the right pile is empty, place the left pile face down and pick up the discard pile, so the piles are swapped.

You're now ready to begin encryption!

Encryption

As with Solitaire, all non-letters in the plaintext are discarded; numbers must be spelled out in full (unless you agree a code to represent them such as "XABCZDX = 12304"). It is then arranged into groups of five, and the last group padded with "X"s if it is short. Thus, in the example, we encrypt "plaintext" using "PLAIN TEXTX". The ciphertext should also be written in groups of five.

Encryption is very similar to keying with one important difference. For each letter in the plaintext, take the following two steps:

- Make a counted cut. Remember that every 25 letters this will exhaust the right deck, at which point you swap packs and cut again.
- Then do a letter search for the next unused letter of the plaintext. As you deal each card in the search, count using letters of the alphabet; the letter you reach when you deal the searched-for card is the next letter of the ciphertext. For example, if you deal five cards before the search stops, then the ciphertext letter is "E". If the searched-for card is on the top of the pack, the ciphertext will be "A"; if it's on the bottom, it's "Z".

Now that you've read this far, take a look at the example encryption and see if it agrees with the sense of the cipher you have so far.

Tidying up

Once encryption is complete, the state of the deck must be destroyed; otherwise, the cipher can be run backwards to recover the plaintext and possibly even the key. Shuffling works, but it's very hard to know when you've shuffled enough to hide not only your key and message but the fact that you've been using crypto at all. Better is to sort the deck entirely into order; an ordered deck should not attract suspicion since it's the result of an ordinary game of Patience, and much of the work of sorting it is already done; it will also make it easier next time you want to encrypt or decrypt a message. I've found the easiest way to do it is to sort each pile into reds and blacks first, then go through each pile in turn fanning out the 13 cards into your hand and taking out each card in turn, highest to lowest (ie look for the King, then the Queen...)

Decryption

Once you've got the hang of how encryption works, it should be clear how to decrypt a Mirdek-encrypted message as follows:

- Initialisation - the left deck is sorted as before, but the right deck must be arranged in the state indicated by the first 25 letters. Clearly, the 26th card will be whichever is left over once you've arranged the other 25.
- Keying and mixing - as before. If the key and initial states used are the same, your decks should now be in the same state as when your correspondent started their encryption process.
- Decryption - for each letter of the ciphertext:
    - Make a counted cut just as with encryption
    - Now deal cards as if in a letter search, but only deal as many cards as the ciphertext letter indicates (so "E" tells you to deal five cards). The last card dealt corresponds to the next plaintext letter. Pick up the piles as with a normal letter search.
- Tidying up - as before.

Timing notes

The counted cut, as described, is rather time consuming. You can speed this up by counting the cards into your hand before moving them to the bottom (so long as you're sure not to reverse the order), and by counting cards rather than letters: "Ace, 2, 3, 4, 5" in the case of the 5 of Clubs, and "Ace, 2, 3, ..., 9, 10, Jack, Queen, King, Ace, 2, 3" in the case of the 3 of Diamonds, since red cards come after black cards in the letter ordering. You can speed up handling the red cards even more by counting cards from the bottom: for an 8 of Diamonds, count "9, 10, Jack, Queen, King" from the bottom of the deck into your other hand and then place them on the top; this is easiest if you temporarily turn the pile upside down in your hand. And of course, a red King (corresponding to "Z") rotates the pile all the way around, leaving it unchanged.

You can also speed up the initialisation phase if you can get the knack of sweeping out the cards in the right pile along the table so that the number and suit of every card is visible, so you can just start writing without having to put down your pen to move cards. Remember that the face-up top card goes first, and the last card is not recorded at all.

After a little practice, it should be possible to encrypt a short message in under twenty minutes:

- Initialisation takes about three minutes, including a minute for shuffling the right half
- Keying takes twenty to thirty seconds per letter of the keyphrase
- Mixing takes about five minutes
- Encryption takes a little under thirty seconds per letter of the ciphertext
- Tidying up takes about two minutes.

Security notes

(obviously, more should go here)

- All operations are reversible - the "letter search" is reversible given either the plaintext or the ciphertext, because the searched-for card ends up as far from the back as it was from the front.
- Each possible card drawn from the right pile in a count cut leads to a different ciphertext for the next plaintext letter, and a different depth of mixing for the left pile.
- One likely side-channel attack poses great risk: a covert listening device allowing the listener to hear how many cards are moved by each step will give away the key and the ciphertext. If the key is longer than 25 characters, the IV can be picked up by this means as well, obviating the need to intercept the encrypted message in order to find the plaintext.

Example of the Mirdek card cipher at work

These examples use the representation from the cipher description pages; the table is reproduced below for convenience.

	A	B	C	D	E	F	G	H	I	J	K	L	M
Black	Ace	2	3	4	5	6	7	8	9	10	Jack	Queen	King
Red	Ace	2	3	4	5	6	7	8	9	10	Jack	Queen	King
	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Initialisation

Initialisation vector: IPDZO WKGST VARME QYBCF JNHUL
Key: KEYPHRASE
Plaintext:
PLAIN TEXTX

Initial state of piles:
Discard:
Right: IPDZOWKGSTVARMEQYBCFJNHULX
Left: ABCDEFGHIJKLMNOPQRSTUVWXYZ

Keying

Count cut:
Top card of discard pile: X

```
Putting together pile: YZ + ABCDEFGHIJKLMNOPQRSTUVWX
Discard: X
Right:   IPDZOWKGSTVARMEQYBCFJNHUL
Left:    YZABCDEFGHIJKLMNOPQRSTUVWX

Mixing in K from the passphrase:
Dealing cards: YZABCDEFGHIJK
Putting together pile: LMNOPQRSTUVWX + KIGECAY + JHFDBZ
Discard: X
Right:   IPDZOWKGSTVARMEQYBCFJNHUL
Left:    LMNOPQRSTUVWXKIGECAYJHFDBZ

Count cut:
Top card of discard pile: L
Putting together pile: XKIGECAYJHFDBZ + LMNOPQRSTUVW
Discard: LX
Right:   IPDZOWKGSTVARMEQYBCFJNHU
Left:    XKIGECAYJHFDBZLMNOPQRSTUVW

Mixing in E from the passphrase:
Dealing cards: XKIGE
Putting together pile: CAYJHFDBZLMNOPQRSTUVW + EIX + GK
Discard: LX
Right:   IPDZOWKGSTVARMEQYBCFJNHU
Left:    CAYJHFDBZLMNOPQRSTUVWEIXGK

Count cut:
Top card of discard pile: U
Putting together pile: EIXGK + CAYJHFDBZLMNOPQRSTUVW
Discard: ULX
Right:   IPDZOWKGSTVARMEQYBCFJNH
Left:    EIXGKCAYJHFDBZLMNOPQRSTUVW

Mixing in Y from the passphrase:
Dealing cards: EIXGKCAY
Putting together pile: JHFDBZLMNOPQRSTUVW + YCGI + AKXE
Discard: ULX
Right:   IPDZOWKGSTVARMEQYBCFJNH
Left:    JHFDBZLMNOPQRSTUVWYCGIAKXE

Count cut:
Top card of discard pile: H
Putting together pile: NOPQRSTUVWYCGIAKXE + JHFDBZLM
Discard: HULX
Right:   IPDZOWKGSTVARMEQYBCFJN
Left:    NOPQRSTUVWYCGIAKXEJHFDBZLM

Mixing in P from the passphrase:
Dealing cards: NOP
Putting together pile: QRSTUVWYCGIAKXEJHFDBZLM + PN + O
Discard: HULX
```

Right: IPDZOWKGSTVARMEQYBCFJN
Left: QRSTUVWYCGIAKXEJHFDBZLMPNO

Count cut:
Top card of discard pile: N
Putting together pile: EJHFDBZLMPNO + QRSTUVWYCGIAKX
Discard: NHULX
Right: IPDZOWKGSTVARMEQYBCFJ
Left: EJHFDBZLMPNOQRSTUVWYCGIAKX

Mixing in H from the passphrase:
Dealing cards: EJH
Putting together pile: FDBZLMPNOQRSTUVWYCGIAKX + HE + J
Discard: NHULX
Right: IPDZOWKGSTVARMEQYBCFJ
Left: FDBZLMPNOQRSTUVWYCGIAKXHEJ

Count cut:
Top card of discard pile: J
Putting together pile: RSTUVWYCGIAKXHEJ + FDBZLMPNOQ
Discard: JNHULX
Right: IPDZOWKGSTVARMEQYBCF
Left: RSTUVWYCGIAKXHEJFDBZLMPNOQ

Mixing in R from the passphrase:
Dealing cards: R
Putting together pile: STUVWYCGIAKXHEJFDBZLMPNOQ + R +
Discard: JNHULX
Right: IPDZOWKGSTVARMEQYBCF
Left: STUVWYCGIAKXHEJFDBZLMPNOQR

Count cut:
Top card of discard pile: F
Putting together pile: CGIAKXHEJFDBZLMPNOQR + STUVWY
Discard: FJNHULX
Right: IPDZOWKGSTVARMEQYBC
Left: CGIAKXHEJFDBZLMPNOQRSTUVWY

Mixing in A from the passphrase:
Dealing cards: CGIA
Putting together pile: KXHEJFDBZLMPNOQRSTUVWY + AG + IC
Discard: FJNHULX
Right: IPDZOWKGSTVARMEQYBC
Left: KXHEJFDBZLMPNOQRSTUVWYAGIC

Count cut:
Top card of discard pile: C
Putting together pile: EJFDBZLMPNOQRSTUVWYAGIC + KXH
Discard: CFJNHULX
Right: IPDZOWKGSTVARMEQYB
Left: EJFDBZLMPNOQRSTUVWYAGICKXH

Mixing in S from the passphrase:
Dealing cards: EJFDBZLMPNOQRS
Putting together pile: TUVWYAGICKXH + SQNMZDJ + ROPLBFE
Discard: CFJNHULX
Right: IPDZOWKGSTVARMEQYB
Left: TUVWYAGICKXHSQNMZDJROPLBFE

Count cut:
Top card of discard pile: B
Putting together pile: VWYAGICKXHSQNMZDJROPLBFE + TU
Discard: BCFJNHULX
Right: IPDZOWKGSTVARMEQY
Left: VWYAGICKXHSQNMZDJROPLBFETU

Mixing in E from the passphrase:
Dealing cards: VWYAGICKXHSQNMZDJROPLBFE
Putting together pile: TU + EBPRDMQHKIAW + FLOJZNSXCGYV
Discard: BCFJNHULX
Right: IPDZOWKGSTVARMEQY
Left: TUEBPRDMQHKIAWFLOJZNSXCGYV

Mixing
Discarding rest of right pile and swapping:
Discard:
Right: TUEBPRDMQHKIAWFLOJZNSXCGYV
Left: BCFJNHULXIPDZOWKGSTVARMEQY

Mixing left pile using right:
Top card of discard pile: V
Dealing cards: BCFJNHULXIPDZOWKGSTV
Putting together pile: ARMEQY + VSKODILHJC + TGWZPXUNFB
Discard: V
Right: TUEBPRDMQHKIAWFLOJZNSXCGY
Left: ARMEQYVSKODILHJCTGWZPXUNFB

Mixing left pile using right:
Top card of discard pile: Y
Dealing cards: ARMEQY
Putting together pile: VSKODILHJCTGWZPXUNFB + YER + QMA
Discard: YV
Right: TUEBPRDMQHKIAWFLOJZNSXCG
Left: VSKODILHJCTGWZPXUNFBYERQMA

Mixing left pile using right:
Top card of discard pile: G
Dealing cards: VSKODILHJCTG
Putting together pile: WZPXUNFBYERQMA + GCHIOS + TJLDKV
Discard: GYV
Right: TUEBPRDMQHKIAWFLOJZNSXC
Left: WZPXUNFBYERQMAGCHIOSTJLDKV

```
Mixing left pile using right:
Top card of discard pile: C
Dealing cards: WZPXUNFBYERQMAGC
Putting together pile: HIOSTJLDKV + CAQEBNXZ + GMRYFUPW
Discard: CGYV
Right:   TUEBPRDMQHKIAWFLOJZNSX
Left:    HIOSTJLDKVCAQEBNXZGMRYFUPW

Mixing left pile using right:
Top card of discard pile: X
Dealing cards: HIOSTJLDKVCAQEBNX
Putting together pile: ZGMRYFUPW + XBQCKLTOH + NEAVDJSI
Discard: XCGYV
Right:   TUEBPRDMQHKIAWFLOJZNS
Left:    ZGMRYFUPWXBQCKLTOHNEAVDJSI

Mixing left pile using right:
Top card of discard pile: S
Dealing cards: ZGMRYFUPWXBQCKLTOHNEAVDJS
Putting together pile: I + SDANOLCBWUYMZ + JVEHTKQXPFRG
Discard: SXCGYV
Right:   TUEBPRDMQHKIAWFLOJZN
Left:    ISDANOLCBWUYMZJVEHTKQXPFRG

Mixing left pile using right:
Top card of discard pile: N
Dealing cards: ISDAN
Putting together pile: OLCBWUYMZJVEHTKQXPFRG + NDI + AS
Discard: NSXCGYV
Right:   TUEBPRDMQHKIAWFLOJZ
Left:    OLCBWUYMZJVEHTKQXPFRGNDIAS

Mixing left pile using right:
Top card of discard pile: Z
Dealing cards: OLCBWUYMZ
Putting together pile: JVEHTKQXPFRGNDIAS + ZYWCO + MUBL
Discard: ZNSXCGYV
Right:   TUEBPRDMQHKIAWFLOJ
Left:    JVEHTKQXPFRGNDIASZYWCOMUBL

Mixing left pile using right:
Top card of discard pile: J
Dealing cards: J
Putting together pile: VEHTKQXPFRGNDIASZYWCOMUBL + J +
Discard: JZNSXCGYV
Right:   TUEBPRDMQHKIAWFLO
Left:    VEHTKQXPFRGNDIASZYWCOMUBLJ

Mixing left pile using right:
Top card of discard pile: O
```

```
Dealing cards: VEHTKQXPFRGNDIASZYWCO
Putting together pile: MUBLJ + OWZADGFXKHV + CYSINRPQTE
Discard: OJZNSXCGYV
Right:   TUEBPRDMQHKIAWFL
Left:    MUBLJOWZADGFXKHVCYSINRPQTE

Mixing left pile using right:
Top card of discard pile: L
Dealing cards: MUBL
Putting together pile: JOWZADGFXKHVCYSINRPQTE + LU + BM
Discard: LOJZNSXCGYV
Right:   TUEBPRDMQHKIAWF
Left:    JOWZADGFXKHVCYSINRPQTELUBM

Mixing left pile using right:
Top card of discard pile: F
Dealing cards: JOWZADGF
Putting together pile: XKHVCYSINRPQTELUBM + FDZO + GAWJ
Discard: FLOJZNSXCGYV
Right:   TUEBPRDMQHKIAW
Left:    XKHVCYSINRPQTELUBMFDZOGAWJ

Mixing left pile using right:
Top card of discard pile: W
Dealing cards: XKHVCYSINRPQTELUBMFDZOGAW
Putting together pile: J + WGZFBLTPNSCHX + AODMUEQRIYVK
Discard: WFLOJZNSXCGYV
Right:   TUEBPRDMQHKIA
Left:    JWGZFBLTPNSCHXAODMUEQRIYVK

Mixing left pile using right:
Top card of discard pile: A
Dealing cards: JWGZFBLTPNSCHXA
Putting together pile: ODMUEQRIYVK + AHSPLFGJ + XCNTBZW
Discard: AWFLOJZNSXCGYV
Right:   TUEBPRDMQHKI
Left:    ODMUEQRIYVKAHSPLFGJXCNTBZW

Mixing left pile using right:
Top card of discard pile: I
Dealing cards: ODMUEQRI
Putting together pile: YVKAHSPLFGJXCNTBZW + IQUD + REMO
Discard: IAWFLOJZNSXCGYV
Right:   TUEBPRDMQHK
Left:    YVKAHSPLFGJXCNTBZWIQUDREMO

Mixing left pile using right:
Top card of discard pile: K
Dealing cards: YVK
Putting together pile: AHSPLFGJXCNTBZWIQUDREMO + KY + V
Discard: KIAWFLOJZNSXCGYV
```

```
Right:   TUEBPRDMQH
Left:    AHSPLFGJXCNTBZWIQUDREMOKYV

Mixing left pile using right:
Top card of discard pile: H
Dealing cards: AH
Putting together pile: SPLFGJXCNTBZWIQUDREMOKYV + H + A
Discard: HKIAWFLOJZNSXCGYV
Right:   TUEBPRDMQ
Left:    SPLFGJXCNTBZWIQUDREMOKYVHA

Mixing left pile using right:
Top card of discard pile: Q
Dealing cards: SPLFGJXCNTBZWIQ
Putting together pile: UDREMOKYVHA + QWBNXGLS + IZTCJFP
Discard: QHKIAWFLOJZNSXCGYV
Right:   TUEBPRDM
Left:    UDREMOKYVHAQWBNXGLSIZTCJFP

Mixing left pile using right:
Top card of discard pile: M
Dealing cards: UDREM
Putting together pile: OKYVHAQWBNXGLSIZTCJFP + MRU + ED
Discard: MQHKIAWFLOJZNSXCGYV
Right:   TUEBPRD
Left:    OKYVHAQWBNXGLSIZTCJFPMRUED

Mixing left pile using right:
Top card of discard pile: D
Dealing cards: OKYVHAQWBNXGLSIZTCJFPMRUED
Putting together pile:  + DUMFCZSGNWAVK + ERPJTILXBQHYO
Discard: DMQHKIAWFLOJZNSXCGYV
Right:   TUEBPR
Left:    DUMFCZSGNWAVKERPJTILXBQHYO

Mixing left pile using right:
Top card of discard pile: R
Dealing cards: DUMFCZSGNWAVKER
Putting together pile: PJTILXBQHYO + RKANSCMD + EVWGZFU
Discard: RDMQHKIAWFLOJZNSXCGYV
Right:   TUEBP
Left:    PJTILXBQHYORKANSCMDEVWGZFU

Mixing left pile using right:
Top card of discard pile: P
Dealing cards: P
Putting together pile: JTILXBQHYORKANSCMDEVWGZFU + P +
Discard: PRDMQHKIAWFLOJZNSXCGYV
Right:   TUEB
Left:    JTILXBQHYORKANSCMDEVWGZFUP
```

Mixing left pile using right:
Top card of discard pile: B
Dealing cards: JTILXB
Putting together pile: QHYORKANSCMDEVWGZFUP + BLT + XIJ
Discard: BPRDMQHKIAWFLOJZNSXCGYV
Right: TUE
Left: QHYORKANSCMDEVWGZFUPBLTXIJ

Mixing left pile using right:
Top card of discard pile: E
Dealing cards: QHYORKANSCMDE
Putting together pile: VWGZFUPBLTXIJ + EMSARYQ + DCNKOH
Discard: EBPRDMQHKIAWFLOJZNSXCGYV
Right: TU
Left: VWGZFUPBLTXIJEMSARYQDCNKOH

Mixing left pile using right:
Top card of discard pile: U
Dealing cards: VWGZFU
Putting together pile: PBLTXIJEMSARYQDCNKOH + UZW + FGV
Discard: UEBPRDMQHKIAWFLOJZNSXCGYV
Right: T
Left: PBLTXIJEMSARYQDCNKOHUZWFGV

Mixing left pile using right:
Top card of discard pile: T
Dealing cards: PBLT
Putting together pile: XIJEMSARYQDCNKOHUZWFGV + TB + LP
Discard: TUEBPRDMQHKIAWFLOJZNSXCGYV
Right:
Left: XIJEMSARYQDCNKOHUZWFGVTBLP

Swapping piles:
Discard:
Right: XIJEMSARYQDCNKOHUZWFGVTBLP
Left: TUEBPRDMQHKIAWFLOJZNSXCGYV

Encryption

Count cut:
Top card of discard pile: P
Putting together pile: OJZNSXCGYV + TUEBPRDMQHKIAWFL
Discard: P
Right: XIJEMSARYQDCNKOHUZWFGVTBL
Left: OJZNSXCGYVTUEBPRDMQHKIAWFL

Encrypting letter: P
Dealing cards: OJZNSXCGYVTUEBP
Putting together pile: RDMQHKIAWFL + PETYCSZO + BUVGXNJ
Ciphertext letter: O
Discard: P
Right: XIJEMSARYQDCNKOHUZWFGVTBL

Left: RDMQHKIAWFLPETYCSZOBUVGXNJ

Count cut:
Top card of discard pile: L
Putting together pile: ETYCSZOBUVGXNJ + RDMQHKIAWFLP
Discard: LP
Right: XIJEMSARYQDCNKOHUZWFGVTB
Left: ETYCSZOBUVGXNJRDMQHKIAWFLP

Encrypting letter: L
Dealing cards: ETYCSZOBUVGXNJRDMQHKIAWFL
Putting together pile: P + LWIHMRNGUOSYE + FAKQDJXVBZCT
Ciphertext letter: Y
Discard: LP
Right: XIJEMSARYQDCNKOHUZWFGVTB
Left: PLWIHMRNGUOSYEFAKQDJXVBZCT

Count cut:
Top card of discard pile: B
Putting together pile: WIHMRNGUOSYEFAKQDJXVBZCT + PL
Discard: BLP
Right: XIJEMSARYQDCNKOHUZWFGVT
Left: WIHMRNGUOSYEFAKQDJXVBZCTPL

Encrypting letter: A
Dealing cards: WIHMRNGUOSYEFA
Putting together pile: KQDJXVBZCTPL + AESUNMI + FYOGRHW
Ciphertext letter: N
Discard: BLP
Right: XIJEMSARYQDCNKOHUZWFGVT
Left: KQDJXVBZCTPLAESUNMIFYOGRHW

Count cut:
Top card of discard pile: T
Putting together pile: YOGRHW + KQDJXVBZCTPLAESUNMIF
Discard: TBLP
Right: XIJEMSARYQDCNKOHUZWFGV
Left: YOGRHWKQDJXVBZCTPLAESUNMIF

Encrypting letter: I
Dealing cards: YOGRHWKQDJXVBZCTPLAESUNMI
Putting together pile: F + INSAPCBXDKHGY + MUELTZVJQWRO
Ciphertext letter: Y
Discard: TBLP
Right: XIJEMSARYQDCNKOHUZWFGV
Left: FINSAPCBXDKHGYMUELTZVJQWRO

Count cut:
Top card of discard pile: V
Putting together pile: QWRO + FINSAPCBXDKHGYMUELTZVJ
Discard: VTBLP

Right: XIJEMSARYQDCNKOHUZWFG
Left: QWROFINSAPCBXDKHGYMUELTZVJ

Encrypting letter: N
Dealing cards: QWROFIN
Putting together pile: SAPCBXDKHGYMUELTZVJ + NFRQ + IOW
Ciphertext letter: G
Discard: VTBLP
Right: XIJEMSARYQDCNKOHUZWFG
Left: SAPCBXDKHGYMUELTZVJNFRQIOW

Count cut:
Top card of discard pile: G
Putting together pile: KHGYMUELTZVJNFRQIOW + SAPCBXD
Discard: GVTBLP
Right: XIJEMSARYQDCNKOHUZWF
Left: KHGYMUELTZVJNFRQIOWSAPCBXD

Encrypting letter: T
Dealing cards: KHGYMUELT
Putting together pile: ZVJNFRQIOWSAPCBXD + TEMGK + LUYH
Ciphertext letter: I
Discard: GVTBLP
Right: XIJEMSARYQDCNKOHUZWF
Left: ZVJNFRQIOWSAPCBXDTEMGKLUYH

Count cut:
Top card of discard pile: F
Putting together pile: QIOWSAPCBXDTEMGKLUYH + ZVJNFR
Discard: FGVTBLP
Right: XIJEMSARYQDCNKOHUZW
Left: QIOWSAPCBXDTEMGKLUYHZVJNFR

Encrypting letter: E
Dealing cards: QIOWSAPCBXDTE
Putting together pile: MGKLUYHZVJNFR + EDBPSOQ + TXCAWI
Ciphertext letter: M
Discard: FGVTBLP
Right: XIJEMSARYQDCNKOHUZW
Left: MGKLUYHZVJNFREDBPSOQTXCAWI

Count cut:
Top card of discard pile: W
Putting together pile: AWI + MGKLUYHZVJNFREDBPSOQTXC
Discard: WFGVTBLP
Right: XIJEMSARYQDCNKOHUZ
Left: AWIMGKLUYHZVJNFREDBPSOQTXC

Encrypting letter: X
Dealing cards: AWIMGKLUYHZVJNFREDBPSOQTX
Putting together pile: C + XQSBEFJZYLGIA + TOPDRNVHUKMW

Ciphertext letter: Y
Discard: WFGVTBLP
Right: XIJEMSARYQDCNKOHUZ
Left: CXQSBEFJZYLGIATOPDRNVHUKMW

Count cut:
Top card of discard pile: Z
Putting together pile: + CXQSBEFJZYLGIATOPDRNVHUKMW
Discard: ZWFGVTBLP
Right: XIJEMSARYQDCNKOHU
Left: CXQSBEFJZYLGIATOPDRNVHUKMW

Encrypting letter: T
Dealing cards: CXQSBEFJZYLGIAT
Putting together pile: OPDRNVHUKMW + TILZFBQC + AGYJESX
Ciphertext letter: O
Discard: ZWFGVTBLP
Right: XIJEMSARYQDCNKOHU
Left: OPDRNVHUKMWTILZFBQCAGYJESX

Count cut:
Top card of discard pile: U
Putting together pile: YJESX + OPDRNVHUKMWTILZFBQCAG
Discard: UZWFGVTBLP
Right: XIJEMSARYQDCNKOH
Left: YJESXOPDRNVHUKMWTILZFBQCAG

Encrypting letter: X
Dealing cards: YJESX
Putting together pile: OPDRNVHUKMWTILZFBQCAG + XEY + SJ
Ciphertext letter: E
Discard: UZWFGVTBLP
Right: XIJEMSARYQDCNKOH
Left: OPDRNVHUKMWTILZFBQCAGXEYSJ

<u>IMPORTANT NOTE ON SECURITY</u>

I just finished Cryptonomicon last night, and as I played around with Solitaire, I also felt that the outputs were not totally random. Thanks for an interesting analysis. Another point I have noted, is that if the joker (53) output is not discarded, each successive round increases the distance between the jokers by exactly one card.

Interesting observation! However, it needs to be modified slightly. The "count cut" can come between the two jokers, meaning that the position of the two jokers is swapped; if it was x before, it will be 54-x afterwards. To get around this, we can define distance to mean the minimum distance going either in the middle of the pack or "around the outside"; in other words, we measure it as min(x, 54-x). This

measure of distance is unchanged by the count cut, and will change by
1 (either up or down) with very high probability each round.

This is the first time I've heard this fascinating observation. When
I get time, I'll add this to my Solitaire page and credit you. Thanks!
--
  __ Paul Crowley