Solitaire cryptography

Encrypting messages with a deck of cards

In this issue...

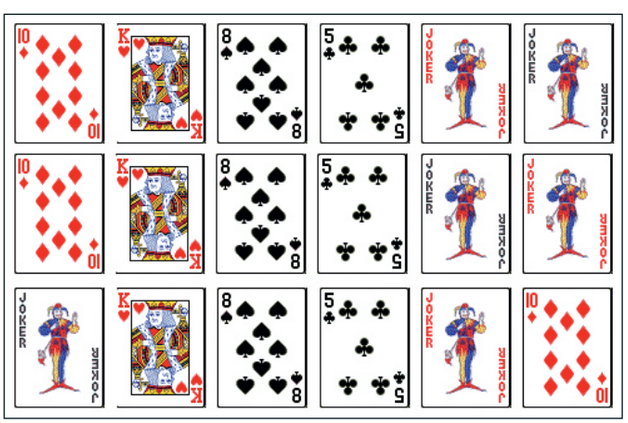
WHAT'S COVERED

You're an agent behind enemy lines. Having a computer with some encryption program on it or any weird specialised hardware is going to get you noticed and dragged into a little room by the secret police. Yet you must communicate with the outside world – and those messages must use strong and unbreakable encryption. Luckily, you have an ordinary deck of cards and the Solitaire Encryption Algorithm.

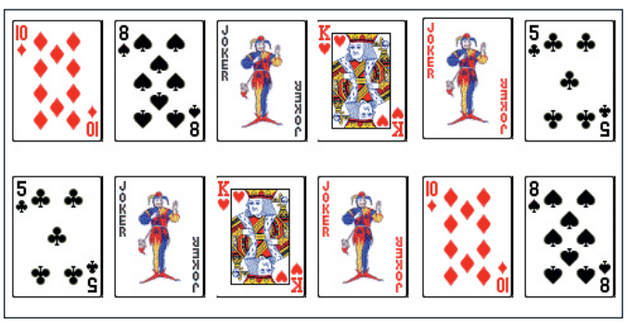
Neal Stephenson's critically regarded novel *Cryptonomicon* is a sprawling book about code makers and code breakers, and fast-forwards from World War II to the present day and back again. While writing the book, Stephenson asked the well-known cryptographer Bruce Schneier to create an unbreakable encryption algorithm that his characters could use. Crucially, it had to be usable without a computer or other specialised computing device. Schneier invented the Solitaire Encryption Algorithm, which used an ordinary deck of cards as the encryption device.

In the novel, the algorithm is called 'Pontifex' in an attempt to throw the bad guys off the scent – calling it Solitaire would make it obvious that a deck of cards is being used. Schneier describes how the algorithm works in an appendix to *Cryptonomicon*.

The algorithm may be low tech, but it's certainly high strength. It gets its security from the inherent randomness of a deck of cards. The deck is manipulated from a known initial state (called the 'key') to produce a stream of random letters, which is called a keystream. The convenient thing about using a pack of cards is that there are 52 cards in the deck, meaning that they easily map twice over onto the alphabet.



▲ **Figure one: Moving the two jokers through the deck.**



▲ **Figure two: The deck before and after a triple cut.**

Learning the maths

To encrypt words and messages with Solitaire, you start off with a known initial ordering of the cards, and then manipulate the deck to generate as many keystream letters as there are plaintext letters (ignoring punctuation and spaces). You then add the corresponding letters together, keystream plus plaintext, modulo 26 to create the cyphertext.

To decrypt, you start off with the same initial ordering, manipulate the deck in the same way to generate as many keystream letters as you have cyphertext letters, and then subtract the keystream from the cyphertext, modulo 26.

Let's see how the encryption and decryption work in practice. Imagine our plaintext message is PCPLUSCODE. Split it up into five-letter chunks, filling the last chunk with 'X's if need be. So ours will become PCPLU SCODE. Now, using the deck of cards generate 10 keystream letters. We'll see how to do this in a moment. Imagine that in this case, the keystream letters are TFCYG THMRG.

You now add these to the plaintext, modulo 16. The easiest way to do this is to replace each letter with its numerical equivalent, so A=one, B=two and so on. Perform the addition, and then if the result is greater than 26, subtract 26.

For our example, the first plaintext character is P, which is 16, and the first keystream letter is T, which is 20. The sum is 36, which is 10 modulo 26. And 10 is equivalent to J. So the first cyphertext letter is J. The remaining text is encrypted the same way and should give JISKB MKBVL.

To decrypt this, we generate the same keystream and then subtract it letter by letter from the cyphertext, using the same trick of converting to a number from one to 26, performing the subtraction and then converting back to a letter. If the number you're subtracting is greater than the number you are subtracting from, add 26 before subtracting.

So, to subtract T from J, we convert to numbers, 20 and 10.

Bridge keys

Another way to get a keyed deck is to use a published bridge game. Many newspapers print bridge puzzles showing a complete deck split between the four players. Use one of these puzzles to key your deck. The problem with this particular keying method is presumably that if the bad guys guess you are using Solitaire as an encryption method, then they would have access to the same bridge puzzles. Even though there are many such puzzles that could be used, a simple computer program could be written to quickly decrypt your message using each of them as the basis for the keyed deck. ■

Since 10 is less than 20, we add 26 to it first to give 36, and then subtract the 20, to give 16. This gets converted back to a P.

All that was easy enough, and given some practice you would quickly learn the numeric equivalents of each letter and become able to do the arithmetic in your head. What still remains, though, is defining the initial state and the manipulation of the card deck. It is here that the strength of the algorithm lies.

Shuffling the deck

The starting order of the cards is possibly the most problematic. Schneier describes a few tactics, but the simplest is to use a passphrase that both you and your counterpart have memorised. For good security, the passphrase must be at least 80 letters long (ignoring spaces and punctuation of course), but for the strongest security you should have one that's 120 letters or more.

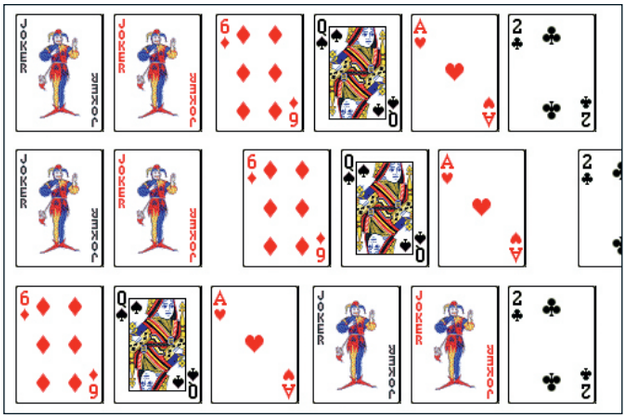
Each passphrase (that is, key) must only be used for one message and then discarded, Using the same passphrase for more than one message would make it easier for any intercepter to crack the code. Obviously, writing down a lot of passphrases is a red flag, so remember to have a book or books handy and some agreed-on methodology for selecting passphrases from the text.

To use a passphrase, start off with the cards in order (lowest cards to highest cards, with the suits in a standard bridge order: clubs, hearts, diamonds, spades) and then 'key' the deck using the Solitaire algorithm and your most recent passphrase.

The Solitaire algorithm requires the use of the two jokers. These jokers must be different – call them A and B. Most decks should have slightly different jokers (the ones I've looked at had a black and a red joker, so the black joker would be B and the red one A). If not, you can mark one of them in a special way to differentiate them. Add the jokers into the deck at some agreed on position, such as both at the beginning or end, or one at the end of the first suit with the second at the start of the last suit. The Solitaire Shuffling algorithm goes like this:

1) Find the A joker in the keyed deck. Move it one card downwards to the end of the deck. If it is the bottom card of the deck, assume the deck 'wraps' and move it to the second place in the deck, underneath the top card. (Figure one, top line, shows the original deck; the second line shows the result of applying this rule.)
2) Now find the B joker in the deck. Move it two cards downwards. Again, if it is at the end of the deck (or the last but one), assume the deck wraps bottom to top, and move it. (Figure one, third line shows the result of this shuffle.)
3) Perform a triple cut. This means take the cards above the first joker in the deck (this could be either the A or the B joker), and swap them with the cards below the second joker in the deck. Neither the jokers nor the cards between them move. (Figure two shows the deck before and after a triple cut.)
4) Perform a count cut. This is a little more complicated. Remove the bottom card. Convert it to a number between one and 53 using bridge ordering (clubs are counted one to 13, hearts 14 to 26, diamonds 27 to 39, spades 40 to 52, and either joker counts as 53). Count down that number of cards from the top of the deck. Cut after that card and swap over the two parts. Finally, add the card you removed to the bottom of the deck again. (Figure three shows the original deck, the three parts, and the result of the count cut.)
5) It's now time to get the next keystream letter. Look at the top card. Convert it to a number from 1 to 53 as in step 4. Count down that number of cards (the top card itself is number one). Write down the card after the one you get to.
6) Convert the card to a number from one to 26 by using standard bridge ordering. Clubs and diamonds will give you one through 13, hearts and spades, 14 to 26. Make a note of that number (and hence letter), and go back to step one to find the next keystream letter.

If in step five you hit a joker, don't do step six. Instead, jump back to step one again.



▲ **Figure three: Perform a count cut to reveal the next keystream letter.**

Using the passphrase

If you are keying the deck using your passphrase, then instead of step five you do another count cut (step four). You still remove the bottom card, but you don't use its value to calculate the cut point: instead, you use the next character of your passphrase, converted to a number between one and 26. Perform the cut and then add the card you removed but didn't use back to the bottom of the deck. Then you jump back to step one again. After you've used up all the letters in your passphrase the deck will be nicely shuffled, but in a very particular order. You can now use this keyed deck to perform the Solitaire encryption and decryption.

Although the algorithm seems long-winded, with a bit of practice you can become adept at manipulating the cards and doing addition and subtraction modulo 26. Before you know it, you'll be sending off your CV to M. ■

Spotlight on… Bruce Schneier

Bruce Schneier is one of the best-known and foremost cryptographers in the world today. He has written many books, both for a technical and a layman's audience, such as *Advanced Cryptography* and *Beyond Fear*. He has invented, either by himself or with various co-designers, such cryptographic algorithms as Blowfish and Twofish. Twofish was one of the finalists under consideration for the replacement algorithm for DES (Data Encryption Standard). Schneier is CTO for BT Counterpane, a company he started to monitor and help develop other companies' security. In recent years, he's recognised that security is more than the application of mathematical cryptography and he now writes about security from a man-in-the-street perspective, especially with regard to personal security (with particular attention to security when travelling by air) and identity security. ■

Julian M Bucknall has worked for companies ranging from TurboPower to Microsoft and is now CTO for Developer Express.
feedback@pcplus.co.uk