

Re: Paper & pencil password algorithm

Source: <http://www.derkeiler.com/Newsgroups/sci.crypt/2009-02/msg00169.html>

- *From:* Maaartin <grajcar1@xxxxxxxxx>
 - *Date:* Tue, 10 Feb 2009 12:37:03 -0800 (PST)
-

That just moves the problem to "how do I remember these different usernames?"

I don't think so. Either you choose your usenames with purpose and than you can remember them easily.
Or you can generate them about the same way as the password.

So now my new mixing rounds:

*mulOrAddRight: Given the two pieces of rows

A B C

a b c

with a and b already computed you compute c using a, b and C as follows:

Do a multiplication or an addition of a and C, where the decision is based on b.

More exactly written as programm expression: $c = (b==0 \parallel b==3 \parallel b==6 \parallel b==9) ? \text{mul}(a, C) : \text{add}(a, C)$

where mul and add are as defined above. Why this way?

- A ternary operation is/may be stronger than a binary one.
- Computing something like $\text{mul}(\text{add}(a, b), C)$ would be twice as much work.
- Switching the operations is slow for computers (conditional branch misprediction penalty) but easy for humans.
- Switching based on the last value computed (i.e., b) is for me easier than the other way round.
- Moreover, this way you get two strains where each following digit depends bijectively on the previous one.
- Using some other test would work as well, but I prefer this one for the following reasons:
 - Testing the LSB (i.e., even/odd) is not very good since in addition the LSB doesn't get influenced by the higher bits.
 - Testing $b < 5$ seems fine, but I wanted the row consisting of all zeros or all ones to be no fixed point.
 - Making the more demanding operation slightly less often then the other is easier.
 - Testing if a digit is multiple of three is easy to remember.

A small example starting with

Re: Paper & pencil password algorithm

5 1 4 6 1

5 1 4 6 1

6 _ _ _ assume zeros to the left, zero is a multiple of three, so
compute $\text{mul}(0, 5) = 10 * 5 \% 11 = 50 - 44 = 6$

5 1 4 6 1

6 0 _ _ six is a multiple of three, so compute $\text{mul}(0, 1) = 0$

5 1 4 6 1

6 0 2 _ _ zero is a multiple of three, so compute $\text{mul}(6, 4) = 6 * 4 \% 11 = 24 - 22 = 2$

5 1 4 6 1

6 0 2 6 _ two is no multiple of three, so compute $\text{add}(0, 6) = 6 \% 10 = 6$

5 1 4 6 1

6 0 2 6 2 six is a multiple of three, so compute $\text{mul}(2, 1) = 2$

There may be some errors above, I did it directly on computer.

*confuseRight: Look at the first empty slot from the left, call it's position (column) p0.

Let the digit just to the left of it (i.e., on position p0-1) be N.

Go over the empty slots from the right counting from 0 to N, wrap around as needed; call the resulting position p1.

Copy the digit from the old row at position p0 in the new row at position p1 and vice versa.

In case of p0=p1 this reduces in copying a single digit.

A small example starting with

6 0 2 6 2

6 0 2 6 2

2 _ _ _ 6 assume zero to the left, there's p0=0, counting from the right you get the rightmost slot (p1=4);
actually in the first step you always swap the leftmost and the rightmost digits

6 0 2 6 2

2 0 _ _ 6 there's p1=1, counting from 0 to 2 from the right you get to p1=2, because of p0=p1 you simply copy the next digit

6 0 2 6 2

2 0 6 2 6 there's p1=2, counting from 0 to 0 from the right you get to p1=3, so you swap the 2 and the 6

Re: Paper & pencil password algorithm

This example was too small in order to see how nicely the row gets permuted, I'll make a larger one after having implemented it.

*addKey: Add the key digitwise to the last row. The length of the key must match the row length.

The whole algorithm should go as follows:

1. Start with a row of digits computed using the url and the nonce (not the key).
2. addKey
3. mulOrAddRight
4. confuseRight
5. addKey
6. mulOrAddRight
7. confuseRight
8. addKey
9. Convert the digits to characters.

Actually this maps the starting row bijectively, so it's a cipher not a keyed hash.

But this shouldn't matter for you, right?

In case it does, than add the first row instead of the key in step 8.

I can't tell nothing about the security (except that I couldn't break it, but this means nothing).

Maybe I'll evaluate how long it takes to avalanche and similar simple things.

.