Project 1

Admin

- Deadline: 2356h, Friday, 9 Oct 2015.
- Submit
  - A report in pdf format.
  - Source code.

  Likely I would not read the source code. If you want to highlight your algorithm, describe it in the report, not the source code.

- Upload to IVLE. Put everything into a single file. The file name should be *MatricNumber_Name.extension* for e.g,

  A000007A_AliceChang.zip

  The report (pdf) file name should be: *MatricNumber_Name*.pdf for e.g.

  A000007A_AliceChang.pdf

  If you prefer to hand in some handwritten notes, or hand-drawn figures, pass the report to me.

- Programming language: no restriction. Can be combined. For example, using C to perform some preprocessing, matlab to conduct some calculation, etc. The cracking of key can also be human + machine efforts. For example, for a particular data-set that can't be handled by the automated process, we can change some parameters or run a routine specially designed for that data-set.

- Machines: No restriction. Usage of grid, cloud, GPU, supercomputer, unless adopted in an "interesting" way, would not score extra marks on "method".

Data-sets

- Materials are stored in IVLE-Workbin-Project1:
  - 60 data-sets `A00.data, …, A59.data`

    (See forum on the data format. The zip of all these files is also available via www.comp.nus.edu.sg/~changec/cs4236.zip)
  - C++ program: `rc4.C`
  - A program that generate the script file: `gen.C`
  - Shell script that generates the data(with keys replaced): `sample.sh`
  - This file.

- The first 4 bytes in the file is the parameter L (i.e. the size of the array K). The next 4 bytes is the number of tuples.

- Each tuple contains 4 values,

  (v_1, v_2, v_3, x)

  where (v_1, v_2, v_3) is the 3 bytes IV and x is the first byte output by RC4. All the tuples in a file is generated using a common secret key.

- For convenient, the first byte x is

  S[(S[1]+S[S[1]]) mod N]

  Note that in actual implementation of RC4, there is a small chance that the first byte is not the above. We use the above version for simplicity.

- Each value v_1, v_2, v_3 and x is represented as a byte (i.e. unsigned char) in the file. It is very important that your program read in the correct values. As a reference, note that in A00.data, the first few values are

  ```
  8 5000000 81 137 43 151
  ```

  The data-sets are generated using the program `rc4.c`

Goal

- Each data-set uses different keys. Your goal is to find the key for each data-set.
- For the purpose of this project, the value "N" in the RC4 is 160, instead of 256.
- Each byte of the secret key is between (inclusive) 0 to 89.
- The size of L is also large, up to 18. When L=8, the size of the secret key is

  $5 \log_2 90 < 33$ bits
- As it is tedious to manually select the keys in `sample.sh` , I wrote a program `gen.C` to generate the script file.
- There is an intentional "implementation vulnerability". If you have discovered it, indicate it in the report. This vulnerability is inserted to add some fun. Exploiting this vulnerability can find all keys, but grading will be based on the keys that you can find when not exploiting this vulnerability. Successfully exploiting the vulnerability will gain extra marks under "methods".

Grading & Report

- In the report's first page, clearly state
  1. The number of found keys and the values.
  2. The configuration of machine. for e.g.

     A desktop. CPU: Core2 Quad CPU@3 GHz. Memory: 4GB

- Total mark is 25. This form 25% of overall grade. (Recall that 60% CA, 40% Exam).

- Grading is based on the number of key found + method employed + presentation.

  Tentatively,

# of keys found:	18 marks	(likely that a small number of found keys already give full marks)
methods:	7 marks	
presentation:	2 marks	(actual marks is min(25, total))

- Give your guess of the partial key if you are unable to find the full key. (If they happen to be correct, grade would be awarded).

Suggestions

- Exhaustive search for the last few bytes.
- Duplications of IVs.
- Korek's method.
- Searching strategy.