INSTALL – JOHN THE RIPPER

Download John the Ripper 1.8.0-jumbo-1 from http://www.openwall.com/john/
Unzip the file
Using Terminal, cd <path of the folder>/src
Then type ./configure && make
After successful compilation, cd ../run
Use ./john to make sure the program can run, you can also see all the command line options for john the ripper

CONFIGURATION

Check out **john.conf** file in the run directory. You can change any of the configurations from this file.

For example try finding these useful configurations:
--StatusShowCandidates (set it to Y)
--LogCrackedPasswords (set it to Y) => you can then check the cracked passwords in john.log
--CrackStatus (set it to N)

Exercise1:

Crack with different modes of john and get familiar with different options. More detailed explanation of all options can be found in doc/OPTIONS file. To learn more about cracking modes see doc/MODES file.

- Stdout
  This option, when used with a cracking mode, (except "single crack mode") makes John the Ripper output the candidate guesses to stdout instead of actually trying them against password hashes.

In this exercise we use this option to allow John the Ripper to print out the guesses to the terminal.

- Wordlist
  This is the simplest cracking mode supported by John. All you need to do is specify a wordlist (a text file containing one word per line)

  try:

  ```
  ./john –wordlist=../../Datasets/Dictionaries/simple_wordlist.txt –stdout
  ```

  see the guesses printed out and compare with the simple_wordlist.txt file

- Rules (enable word mangling rules)

  You can enable word-mangling rules (which are used to modify or "mangle" words producing other likely passwords). If enabled, all of the rules will be applied to every line in the wordlist file producing multiple candidate passwords from each source word.

--rules enables the word mangling rules. The word mangling rules are read from [List.Rules:Wordlist] section by default. Find this section in the <john the ripper folder>/run/john.conf file to see what word mangling rules will be applied to each word.

Look at the wordlist file Datasets/Dictionaries/small_wordlist.txt.
Then try this command to see the guesses:

```
./john -wordlist=../../Datasets/Dictionaries/small_wordlist.txt -rules -stdout
```

- Incremental
  See the explanation of incremental mode in doc/MODES. This mode can try all possible character combinations as passwords. You can define the mode's parameters, including password length limits and the charset to use. These parameters are defined in the configuration file sections called [Incremental:MODE]

  ```
  ./john –incremental –stdout
  ```

- Markov
  You can find more details about this option in doc/MARKOV.
  This option uses the Markov model to generate guesses.
  Try: `./john –markov –stdout`

Exercise2:
Crack myspace hash file

```
./john <cracking_mode> -format=Raw-MD5 ../../Datasets/myspace_hashes.txt
```

Exercise3: Crack with john and use checkPass to see the result
In this exercise the goal is to crack the file yahoo_test.txt. Use any of the options for generating guesses with John the Ripper. Then use the –stdout option to only generate the guesses and then pipe the guesses into the program checkPass.
The checkPass program needs three options:

-c: the number of files you want to crack which in our case is 1. So we use –c 1
-f: path and name of the file you want to crack
-l: the maximum number of candidate guesses we want to generate

First go to checkPass folder and type **make** to compile the program.

Example command line:
From john-1.8.0-jumbo-1/run directory:

```
./john –incremental –stdout | ../../checkPass/checkPass -c 1 -f
../../Datasets/yahoo_test.txt -l 1000000000
```

The final result will be stored in result1.txt file in the run directory.