Information Security Technologies COMP607
Tutorial
Session 2 Symmetric key cryptography

1. Alice and Bob shares a secret 8-bits key "`10110110`" for encryption using XOR. Alice wish to encrypt the following message to Bob: "NOMEETING"

   Refer to UTF-8 encoding to get the binary code for the letters, and then apply the key to each letter.

   Write the corresponding ciphert text in binary.

2. The following cipher text was sent from Bob to Alice using an XOR operation with an unknown 8-bit key. It is discovered that the 8-bit has been repeated reused (i.e. not a OTP) which is a fatal mistake. Given that the most common letter in the plaintext is "E", find the key and hence obtain the plaintext.

```
Cipher text: 11110000 11111001 11110000 11100011 11110000 11111011
```

3. Encryption using DES. Connect to the Linux server at 156.62.140.124 using your autlogin name and date of birth (e.g. 01apr) as password.
   (a). Create a test folder: `$ mkdir test`

   (b). Change into this directory: $ `cd test`

   (c). Open a text editor and create file containing some text, e.g.

   ```
   $ pico plain.txt
   ```

   Type in some text, e.g. This is a secret message
   Type Ctrl X to save and exit.

   (d). Encrypt the plain.txt to a new file called cipher.txt
   ```
   $ openssl enc -des -in plain.txt -out cipher1.enc
   ```
   Provide a suitable key

   (e). Display the encrypted file: $ `cat cipher1.enc`

   Display the encrypted file using a hexdump: `$ hexdump -C cipher1.enc`

   Display the encrypted file using a hex editor: $ hcurse cipher1.enc

   (f). Decrypt the file into plaintext:
   ```
   $ openssl enc -d -des -in cipher1.enc -out clear1.txt
   ```

   (g). View the decrypted file: $ `cat clear1.txt`

   You can also view the file using the text editor: $ pico clear1.txt

   (h) Encrypt the same plain.txt file again using the same key, call the output cipher2.enc
   ```
   $ openssl enc -des -in plain.txt -out cipher2.enc
   ```

Display and compare the contents of the two ciphertext files cipher1.enc and cipher2.enc

Are they the same? Why not?

(i) Encrypt the plaintext file again using “nosalt”

```
$ openssl enc -des -nosalt -in plain.txt -out cipherno1.enc
$ openssl enc -des -nosalt -in plain.txt -out cipherno2.enc
```

Display the encrypted files:

```
$ hexdump -C cipherno1.enc
$ hexdump -C cipherno2.enc
```

Are the same? Why?

4. Using the same procedure as above, encypt the file using AES-aes-256-cbc
e.g. `$ openssl enc -aes-256-cbc -in file.txt -out file.enc`

View the encrypted file: `$ cat file.enc`
(You can also use: `$ hexdump -C file.enc`)

Decrypt the file in the same way;

```
$ openssl enc -d -aes-256-cbc -in file.enc -out file.dec
```

view the decrypted file: `$ cat clear.dec`

5. The openssl commands are available in the Xenial Puppy linux OS /root/downloads folder. Use a browser and point to the folder. Experiment with other methods of encryption, key length, large files, etc.

6. *Assume we perform a known-plaintext attack against DES with one pair of plaintext and ciphertext. How many keys do we have to test in a worst-case scenario if we apply an exhaustive key search in a straightforward way? How many on average?

7. *We consider the long-term security of the Advanced Encryption Standard (AES) with a key length of 128-bit with respect to exhaustive key-search attacks. AES is perhaps the most widely used symmetric cipher at this time. 1. Assume that an attacker has a special purpose application specific integrated circuit (ASIC) which checks $5 \cdot 10^8$ keys per second, and she has a budget of $1 million. Assume one ASIC costs $50 including overhead for integrating the ASIC (manufacturing the printed circuit boards, power supply, cooling, etc.).

How many ASICs can we run in parallel with the given budget? How long does it to search for the brute force password attack. How long does it take for an average an average key search take? Relate these times to the age of the Universe, which is about 10^{10} years.