

IY2760/CS3760: Coursework 1

Solutions

1. Compile a glossary of 25 commonly used security terms. A good starting point is the glossary of IT security terminology at the ISO/IEC JTC1 SC27 web site – see:

<http://www.jtc1sc27.din.de/en>

[5 marks]

Any reasonable set of definitions will do.

2. What is the main purpose of an encryption algorithm?

[2 marks]

The main purpose of an encryption algorithm is to provide confidentiality (secrecy) for data. An encryption algorithm transforms plaintext into ciphertext, from which no information about the plaintext can be recovered without the appropriate key.

3. What is an exhaustive key search? What property is required of an encryption function to ensure that such a search is infeasible?

[3 marks]

An exhaustive key search using ciphertext-plaintext pairs involves systematically decrypting the ciphertext with all possible key values until the correct plaintext is found. If ciphertext but no plaintext is available, then candidate keys can be tested by decrypting the ciphertext until the output “makes sense” (or in some other way is recognisably valid plaintext).

Exhaustive key searches become infeasible if the length of the key is so large that it is infeasible to check all the keys in a reasonable amount of time. Use of a 64-bit key means that the number of possible keys is 2^{64} , which is relatively small, although it would still take a long time to work through such a set of keys, even at upwards of 10^6 keys per second. Currently, a key length of at least 80 bits is considered desirable.

4. Find two four-letter words that yield the same ciphertext using the Caesar cipher (under two different keys). E.g. *bee* and *loo* both encipher to *mpp* – in the first case using a key which shifts every letter forward through the alphabet by 11 places, and in the second case using a key which shifts every letter by one place).

[6 marks]

There are many possible pairs of 4-letter words, one of which encrypts to the other using the Caesar Cipher. In the list below, the number after the pair of words indicates the shift required. My thanks to past students from this course who contributed these suggestions.

<i>Word 1</i>	<i>Word 2</i>	<i>Shift</i>
beef	loop	10
bits	talk	18
boob	naan	12
boon	reed	16
clap	rape	15
curl	wolf	20
Dahl	help	4
damp	road	14
deed ¹	noon	10
dune	ribs	14
envy	rail	13
feed	tool	14
goon	weed	16
hack	Lego	4
lawn	pear	4
meet	wood	10
pelt	show	3

There are also some 5-letter pairs, as follows.

<i>Word 1</i>	<i>Word 2</i>	<i>Shift</i>
chain	ingot	6
dolls	wheel	19

5. Describe how a simple substitution cipher works. How would you set about writing a computer program to automatically cryptanalyse ciphertext produced using a simple substitution cipher? You need only provide a *top level description* of the program.

[8 marks]

A simple substitution cipher involves defining a permutation of the set of letters. Each letter in the plaintext is replaced by its image under the permutation to obtain the ciphertext.

It would be no good writing a program to perform an exhaustive key search attack, as there is such a large number of keys, i.e. $26! \approx 4 \times 10^{26}$. As a result, performing such a search would be infeasible. One way of

¹ If you allow 'Abba' as a word, then Abba, deed and noon are all shifts of each other!

attempting to break a simple substitution cipher is by using the frequencies of letters occurring in the ciphertext.

A program could process the ciphertext to compute the letter frequency distribution in the ciphertext. The program could then compare these results with the typical letter frequency distribution for the English language. The program could then compare these results with the statistics from the ciphertext and match each ciphertext letter with the most likely plaintext letter. Of course, this will not give a completely correct decryption.

One way of completing the decryption process would be to get the user to provide input. The program might, for example, ask the user whether the plaintext makes sense. If some of the frequency statistics are very close, then the computer could find all the possible plaintext letters to which a particular ciphertext letter might correspond, and then present the options to the user.

The following table gives typical letter frequencies for English text:

A=8.167	F=2.228	K=0.772	P=1.929	U=2.406	Z=0.074
B=1.492	G=2.015	L=2.782	Q=0.095	V=0.978	
C=2.758	H=6.094	M=2.406	R=5.987	W=2.36	
D=4.253	I=6.966	N=6.749	S=6.327	X=0.15	
E=12.7	J=0.153	O=7.507	T=9.056	Y=1.974	

A pseudo-code description of such a program follows.

```

float english_freq[27] = {freq. % of each letter in English lang.}
char english_alphabet[27] = {"zqxjkbpygfwmucldrhsnioate"}
float cipher_freq[27] = {0}
char cipher_alphabet[27] = {"abcdefghijklmnopqrstuvwxy"}
char ciphertext[] = {ciphertext}
cipher_length = length of ciphertext
//Calculate letter occurrences in ciphertext
FOR (loop = 1 TO cipher_length)
    letter = ciphertext[loop]
    cipher_freq[ letter ] += 1
//Work out % of cipher letter frequencies
FOR (loop = 1 TO 26)
    cipher_freq[loop] = (cipher_freq[ loop ]/cipher_length)*100
//Bubblesort frequency of ciphertext letters into numerical order
//Compare cipher_freq and english_freq and choose the most
appropriate substitutions.
//Display the computer's results

```

The ciphertext will need to be reasonably long (i.e. at least several hundred characters), or the above process will not even get close to yielding the correct plaintext.

The program could move closer to an automatic decryption process by using statistics of bigrams and trigrams (i.e. pairs and triples of

consecutive letters). An initial matching of plaintext against ciphertext letters could be derived using single letter frequencies (as above), and then a combinatorial optimisation approach (e.g. hill climbing) could be used to try to obtain a better match of bigram statistics.

6. Explain in your own words the properties needed for a stream cipher key stream to be secure. How do these properties relate to the key stream used in a one-time pad?

[3 marks]

Three necessary properties for a key stream are:

- *pseudo-randomness, for example the numbers of zeros and ones should be close to equal, and pairs 00, 01, 10, and 11 should all occur roughly equally often;*
- *long period, e.g. 2^{64} bits or more before the pattern is repeated;*
- *large linear equivalence; i.e. the shortest linear recurrence formula that generates the sequence should involve a large number of terms.*

The one-time pad is totally secure because:

- *the key stream is truly random;*
- *the period could be considered to be infinite since there is no repetition;*
- *no recurrence formula exists.*

7. The Vigenère cipher can be considered as a type of stream cipher. State what the keystream generator is in this case, and describe whether or not it meets the properties given in your answer to the previous question.

[7 marks]

In this case the keystream is simply the keyword repeated over and over again, i.e. the keystream generator outputs the keyword repeatedly. Addition modulo 26 replaces use of the exclusive-or operation. Clearly the keystream has very short period (the keyword length), breaking the second property. The other two properties also fail because of the very short period.

8. Discuss the security of triple DES (as opposed to DES) with reference to key length / key space and implementation issues.

[3 marks]

Triple DES uses two 56-bit keys so has an effective key length of 112 bits. The implementation involves encrypt-decrypt-encrypt, with one key used for the encryptions and the other for the decryption. This protects against the meet-in-the-middle attack that would be possible with the more straightforward use of encrypt-encrypt with two different keys. A key space of size 2^{112} is likely to be sufficiently secure for the foreseeable future.

The encrypt-decrypt-encrypt pattern used with two equal keys means the process is the same as single DES. This property is useful for backwards compatibility with systems using single DES.

9. What is a dictionary attack on a block cipher, and how can we ensure that such an attack is infeasible?

[2 marks]

A cryptanalyst equipped with pairs of matching plaintext and ciphertext blocks may be able to compile a 'dictionary' listing which plaintext block maps to which ciphertext block under a certain key. Any further ciphertext generated using the same secret key can then be decrypted by looking up the block in the 'dictionary'.

Choosing a large block size ensures such attacks are infeasible (the dictionary becomes too large).