COMPSCI 755: Assignment 1
Gunnar Yonker

1.

(1) One active attack is Masquerade where the attacker would pretend to be a different entity, this could be accomplished through IP spoofing where the victim machine believes that they are communicating with the right machine, but it is the attacker's machine spoofing their IP address pretending to be the intended machine. This could compromise the confidentiality, integrity, and authenticity of the communication instance due to the data being transmitted to an attacker not the intended machine. Another attack is the Replay attack in which the attacker would passively capture a data unit and then later they send it to the receiver which would then produce an unauthorized effect. This could compromise the integrity and authenticity of the communication because the user would no longer be communicating with an authorized machine. A third active attack is modification of messages, which is when the messages are altered, delayed, or reordered. This could compromise the integrity of the messages if they were altered, delayed, or reordered.

(2)

a. Since a mono-alphabetic cipher is a fixed substitution when changing the plaintext to the ciphertext, it follows the patterns of what letters are used more often. A cryptanalytic attack with a mono-alphabetic cipher is when the mathematical pattern of the most used letters and least used letters is used to start decrypting the text. This would then reveal enough information that would allow one to look at the ciphertext and assign letters based on what is most commonly used, from there the key can be solved for. A brute-force attack is when every key option is tested and tried until the correct key is found. With a mono-alphabetic cipher there are 26 letters in the alphabet so there would be 26! different keys that would need to be tried. If this process is used on a supercomputer it would take 6.4x10e6 years to try all possible key combinations.

b. A polyalphabetic cipher is stronger than a mono-alphabetic cipher because instead of having a fixed substitution rule, there are multiple substitution rules when encrypting the plaintext. This would make a cryptanalytic attack more difficult because the pattern of frequency of letters would no longer be as useful. A polyalphabetic cipher also has a larger amount of keys that could be used, which would ultimately make a brute-force attack take longer to break the ciphertext than a mono-alphabetic cipher. The substitution rule changes for each letter of the plaintext file with a polyalphabetic cipher, which makes both the cryptanalytic attack and brute-force attack more difficult to break than a mono-alphabetic cipher.

(3) I don't think that it would be feasible to use pure substitution operations to implement the block cipher for a block size of 128 bits because the amount of keys available would be limited and the pure substitution would require a unique mapping. The plaintext to ciphertext would require 2^128 and the cipher text would require the same amount for a block size of 128. A pure substitution would then require a key space of 2^256, which would not be feasible for a key space that is limited to 2^128 so there would not be enough space. This is the reason that encryption methods such as AES and DES are used for block ciphers because they become more complex using multiple encryption methods.

COMPSCI 755: Assignment 1
Gunnar Yonker

2.

(1) I wrote a java program that would check each shift and print out the key shift used along with the plaintext it would result in. A key of 22 was used and resulted in “GOWARHAWKS”.

```
Key: 0
Plaintext: CKSWNDWSGO
Key: 1
Plaintext: BJRVMCVRFN
Key: 2
Plaintext: AIQULBUQEM
Key: 3
Plaintext: ZHPTKATPDL
Key: 4
Plaintext: YGOSJZSOCK
Key: 5
Plaintext: XFNRIYRNBJ
Key: 6
Plaintext: WEMQHXQMAI
Key: 7
Plaintext: VDLPGWPLZH
Key: 8
Plaintext: UCKOFVOKYG
Key: 9
Plaintext: TBJNEUNJXF
Key: 10
Plaintext: SAIMDTMIWE
Key: 11
Plaintext: RZHLCSLHVD
Key: 12
Plaintext: QYGKBRKGUC
Key: 13
Plaintext: PXFJAQJFTB
Key: 14
Plaintext: OWEIZPIESA
Key: 15
Plaintext: NVDHYOHDRZ
Key: 16
Plaintext: MUCGXNGCQY
Key: 17
Plaintext: LTBFWMFBPX
Key: 18
Plaintext: KSAEVLEAOW
Key: 19
Plaintext: JRZDUKDZNV
Key: 20
Plaintext: IQYCTJCYMU
Key: 21
Plaintext: HPXBSIBXLT
Key: 22
Plaintext: GOWARHAWKS
Key: 23
Plaintext: FNVZQGZVJR
Key: 24
Plaintext: EMUYPFYUIQ
Key: 25
Plaintext: DLTXOEXTHP
```

COMPSCI 755: Assignment 1
Gunnar Yonker

(2) Plaintext: “spring is coming”

Key: “song”

Ciphertext: “kdeofu vy uczofu”

3. LD2 = RE14 and RD2 = LE14

Can use the proof of LDn = RE16-n; RDn = LE16-n

to prove the above

n = 1 proof

Encryption:

LE16 = RE15

RE16 = LE15 XOR F(RE15, K16)

Decryption:

LD1 = RD0 = LE16 = RE15

RD1 = LD0 XOR F(RD0, K16)

= RE16, XOR F(RE15, K16)

= (LE15 XOR (RE15, 16)) XOR F(RE15, K16)

= LE15 XOR (F(RE15, K16) XOR F(RE16, K16))

= LE15 XOR 0

= LE15

Therefor:

LD1 = RE15

RD1 = LE15

Now, n=2 proof

Encryption:

LE15 = RE14

RE16 = LE15 XOR F(RE15, K16)

Decryption:

LD2 = RD0 = LE15 = RE14

RD2 = LD0 XOR F(RD0, K16)

= RE16, XOR F(RE15, K16)

= (LE15 XOR (RE15, 16)) XOR F(RE15, K16)

= LE15 XOR (F(RE15, K16) XOR F(RE16, K16))

= LE15 XOR 0

= LE15

Therefor:

LD2 = RE14

RD2 = LE14

4.

(1) 8-byte message: meetat12

Hex: 6d65657461743132

(2) 8-Byte key: hiddenin

Hex: 68696464656e696e

Binary: 0110 1000 0110 1001 0110 0100 0110 0100 0110 0101 0110 1110 0110 1001 0110 1110

(3) Encrypted: e3fa3875e0e7c544

a.

Round 1: 3fe82f57 | dc5a67ea

0011 1111 1110 1000 0010 1111 0101 0111 1101 1100 0101 1010 0110 0111 1110 1010

Round 2: e3b248bd|6fd3ccae

1110 0011 1011 0010 0100 1000 1011 1101 0110 1111 1101 0011 1100 1100 1010 1110

Round 3: 6f2ccd2e | 038705e7

0110 1111 0010 1100 1100 1101 0010 1110 0000 0011 1000 0111 0000 0101 1110 0111

Round 4: e0354d5a | 9be500f0

1110 0000 0011 0101 0100 1101 0101 1010 1001 1011 1110 0101 0000 0000 1111 0000

Round 5: f4c9cdde | 0a79034b

1111 0100 1100 1001 1100 1101 1101 1110 0000 1010 0111 1001 0000 0011 0100 1011

Round 6: ea4c4e11 | 3dbae1d1

COMPSCI 755: Assignment 1
Gunnar Yonker

1110 1010 0100 1100 0100 1110 0001 0001 0011 1101 1011 1010 1110 0001 1101 0001

b.

Round 1:

Li-1 = 3fe82f57

Ri-1 = 3fe82f57

F(Ri-1,Ki1) = dc5a67ea

Li = Ri-1 = 3fe82f57

Ri = Li-1 XOR F(Ri-1,Ki) = 3fe82f57 XOR dc5167ea = e3b248bd

Round 2:

Li-1 = e3b248bd

Ri-1 = e3b248bd

F(Ri-1,Ki1) = 6fd3ccae

Li = Ri-1 = e3b248bd

Ri = Li-1 XOR F(Ri-1,Ki) = e3b248bd XOR 6fd3ccae = 6f2ccd2e

Round 3:

Li-1 = 6f2ccd2e

Ri-1 = 6f2ccd2e

F(Ri-1,Ki1) = 038705e7

Li = Ri-1 = 6f2ccd2e

Ri = Li-1 XOR F(Ri-1,Ki) = 6f2ccd2e XOR 038705e7 = e0354d5a

(4)

Key in 4-bit binary: 0110 1000 0110 1001 0110 0100 0110 0100 0110 0101 0110 1110 0110 1001 0110 1110

Bit 38 changed: 0110 1000 0110 1001 0110 0100 0110 0100 0110 0001 0110 1110 0110 1001 0110 1110

Round 1: 3fe82f57 | de4a63eb

0011 1111 1110 1000 0010 1111 0101 0111 1101 1110 0100 1010 0110 0011 1110 1011

Round 2: e1a24cbc | fb90a7b0

1110 0001 1010 0010 0100 1100 1011 1100 1111 1011 1001 0000 1010 0111 1011 0000

COMPSCI 755: Assignment 1
Gunnar Yonker

Round 3: fb6fa630 | 0ad26c7c

1111 1011 0110 1111 1010 0110 0011 0000 0000 1010 1101 0010 0110 1100 0111 1100

Round 4: eb7020c0 | 38d6c7b5

1110 1011 0111 0000 0010 0000 1100 0000 0011 1000 1101 0110 1100 0111 1011 0101

Round 5: c3b96185 | 8826b35c

1100 0011 1011 1001 0110 0001 1000 0101 1000 1000 0010 0110 1011 0011 0101 1100

Round 6: 6356939c | 7a441e79

0110 0011 0101 0110 1001 0011 1001 1100 0111 1010 0100 0100 0001 1110 0111 1001

(5)

Comparison for Avalanche effect:

O = Original

M = Modified 38-bit key

Round 1:

O: 0011 1111 1110 1000 0010 1111 0101 0111 1101 1100 0101 1010 0110 0111 1110 1010

M: 0011 1111 1110 1000 0010 1111 0101 0111 1101 1110 0100 1010 0110 0011 1110 1011

Round 2:

O: 1110 0011 1011 0010 0100 1000 1011 1101 0110 1111 1101 0011 1100 1100 1010 1110

M: 1110 0001 1010 0010 0100 1100 1011 1100 1111 1011 1001 0000 1010 0111 1011 0000

Round 3:

O: 0110 1111 0010 1100 1100 1101 0010 1110 0000 0011 1000 0111 0000 0101 1110 0111

M: 1111 1011 0110 1111 1010 0110 0011 0000 0000 1010 1101 0010 0110 1100 0111 1100

Round 4:
O: 1110 0000 0011 0101 0100 1101 0101 1010 1001 1011 1110 0101 0000 0000 1111 0000

M: 1110 1011 0111 0000 0010 0000 1100 0000 0011 1000 1101 0110 1100 0111 1011 0101

Round 5:

O: 1111 0100 1100 1001 1100 1101 1101 1110 0000 1010 0111 1001 0000 0011 0100 1011

M: 1100 0011 1011 1001 0110 0001 1000 0101 1000 1000 0010 0110 1011 0011 0101 1100

Round 6:
O: 1110 1010 0100 1100 0100 1110 0001 0001 0011 1101 1011 1010 1110 0001 1101 0001

COMPSCI 755: Assignment 1
Gunnar Yonker

M: 0110 0011 0101 0110 1001 0011 1001 1100 0111 1010 0100 0100 0001 1110 0111 1001

Above, the differences between the original(O) and modified(M) rounds are shown with the differences highlighted in the M line. The avalanche effect is when a small change in the input data, in this case bit 38 in the key, produces a significant change in the output data. The avalanche effect is observed here because the small change which was the change of bit 38 produces a significant change in the output of the data. In the first round there are only 4 discrepancies noted which is still different after the change, but as we move to round 2 there are already significantly more changes. This trend is seen continuing through the rounds as that initial small change resulted in many of the bits being different from the original to the modified version in each round. The data collected from the rounds above show that the avalanche effect was observed when bit 38 was changed in the original key to the modified key during the encryption. This is an important part of the DES encryption algorithm because with one small change resulted in a significant change of the output data, it makes it more difficult for an attacker to break the cipher.