Breaking the XoR Cipher

Information Security – Lecture 04

Aadil Zia Khan

Cryptanalysis

- Cryptanalysis - the process of attempting to discover the plaintext or key

- An encryption scheme is **computationally secure** if the ciphertext generated by the scheme meets one or both of the following criteria
  - The cost of breaking the cipher exceeds the value of the encrypted information
  - The time required to break the cipher exceeds the useful lifetime of the information

Attacking Encrypted Messages

Type of Attack	Cryptanalyst's Knowledge (Encryption algorithm and Ciphertext known)
Ciphertext only	• No additional information - *most difficult*
Known plaintext	• One or more plaintext–ciphertext pairs formed with the secret key • In **Probable plaintext** attack, attacker can guess parts of the plaintext
Chosen plaintext	• Plaintext chosen by cryptanalyst, together with its corresponding ciphertext • Analyst may deliberately pick patterns that can reveal structure of the key
Chosen ciphertext	• Ciphertext chosen by cryptanalyst, together with its corresponding plaintext

Known Plaintext???

Is it possible to get your hands on a plaintext-ciphertext pairs??? => **Yes**

- The source code files for a program developed by a corporation might include a copyright statement in some standardized position
- An encrypted image from some TV channel will always contain the channel logo
- A file encoded in the Postscript format always begins with the same pattern
- In an accounting file, the adversary may know the placement of certain words in the header of the file


```
/*
 * Copyright IBM Corporation, 2010
 * Author Aneesh Kumar K.V <aneesh.kumar@linux.vnet.ibm.com>
 *
 * This program is free software; you can redistribute it and/or modify it
 * under the terms of version 2.1 of the GNU Lesser General Public License
 * as published by the Free Software Foundation.
 *
 * This program is distributed in the hope that it would be useful, but
 * WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
 *
 */
```

Known Plaintext???

Is there any other way to get your hands on a plaintext–ciphertext pairs??? => **Yes**

- We can also infer the plaintext from some ciphertext
- E.g., If there is a single letter word, chances are that it would either be "I" or "a" – double letter word could be "an", "us", "we", "to", "if", etc.
- E.g., if a letter occurs most frequently, chances are that it is the encrypted form of space character or "e" because they are most common characters in English text

Lets Break the XoR Cipher

Rules:

- Plaintext xor Key = Cipher
- Cipher xor Key = Plaintext
- Plaintext xor Cipher = Key

- Cipher1 xor Cipher2 = Plaintext1 xor Plaintext2

Example

P1	= 1001	P2	= 0011
K	= 1100	K	= 1100
C1	= 0101	C2	= 1111
P1 xor K	= 0101	P2 xor K	= 1111
C1 xor K	= 1001	C2 xor K	= 0011
P1 xor C1	= 1100	P2 xor C2	= 1100
C1 xor C2	=1010	P1 xor P2	= 1010

Break the XOR Cipher - Known Plaintext Attack

```
/*
This program is free software; you can redistribute it
and/or modify it under the terms of version 2.1 of the
GNU Lesser General Public License as published by
the Free Software Foundation.

This program is distributed in the hope that it would
be useful, but WITHOUT ANY WARRANTY; without
even the implied warranty of MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE.
*/
#include <linux/module.h>
#include <linux/fs.h>
```

- Suppose you have an encrypted source file
  - You know that many source files have the license statement as shown in the excerpt on the left
- XoR the starting text of the encrypted file with the license text => you will get the key

Break the Single-byte Key XOR Cipher - Ciphertext only

- Brute force – since the key (8bits) is small, we can try all possible key values (2^8=256)
- If for any key value, you are able to find English words in the resulting decrypted plaintext => you have found the key

Break the Single-byte Key XOR Cipher - Ciphertext only Example

- Plaintext (not *known*)
  - 01101000 01100101 01101100 01101100 01101111 00100000 01101000 01101111 01110111 00100000 01100001 01110010 01100101 00100000 01111001 01101111 01110101 (*hello how are you*)
- Ciphertext (*known*)
  - 00001010 00000111 00001110 00001110 00001101 01000010 00001010 00001101 00010101 01000010 00000011 00010000 00000111 01000010 00011011 00001101 00010111
- Key (need to find out)
  - Lets try key=a (01100001) => 01000010 xor 01100001 = 00100011 (#) => doesn't make sense
  - Lets try key=b (01100010) => 01000010 xor 01100010 = 00100000 (space) => could be
    - Lets use b to decrypt the entire text => "hello how are you"

Break the Multiple-byte Repeating Key XOR Cipher - Ciphertext only

- Note, the key of length L encrypts a block of plaintext of length L and then repeats with the next block - take two such encrypted blocks
- XOR them with each other - you'll get the XOR of the two original unencrypted messages since the identical keys cancel each other out
- Now what?
  - Take a guess of a common phrase that may appear in one of the plaintexts (e.g., the 5 letter " the ")
  - XoR that against the XoR of the two original messages at different locations
  - If one of the plaintexts had the text (" the "), the result of the XoR will be what the other plaintext had in that position; otherwise the result will be garbage

Assumptions in our Attack

- Key length is known
  - Not always true, but guessing the key length is not too difficult

- The plaintext is in regular English => not numbers, not compressed text
  - It would become difficult to determine if the encryption has been broken
  - We wont be able to make use of the properties of the English language
    - Like frequency of different letters, words of length less than three, etc.

XoR Encryption is Weak - Now What???

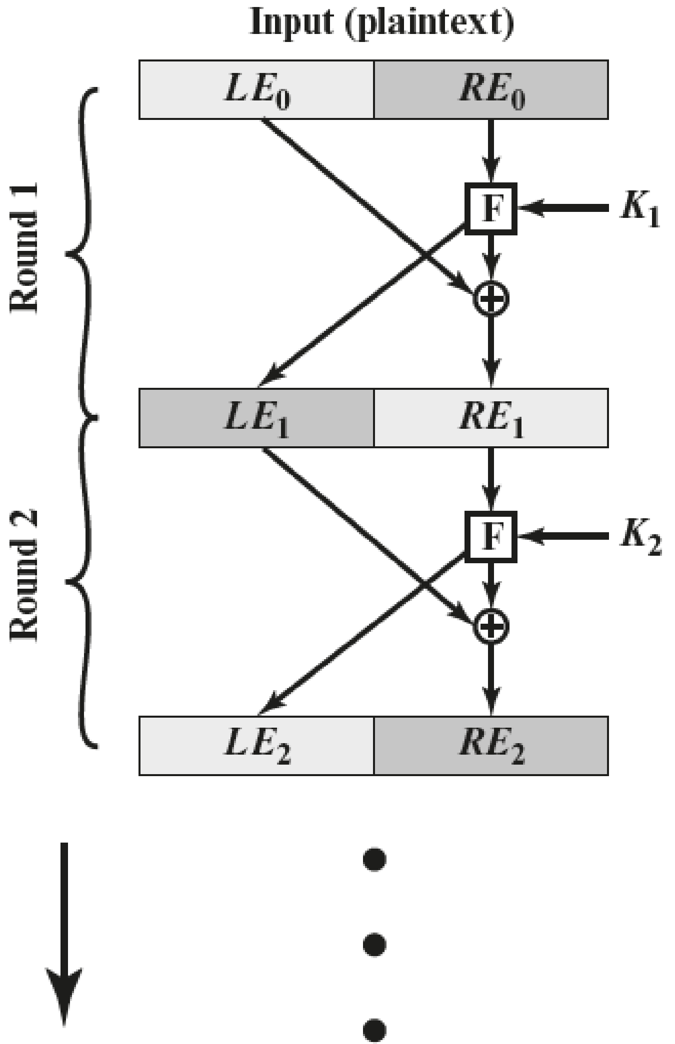
- We have to design a system which is (ideally) immune to such cryptanalysis

Diffusion and Confusion – The Way Forward

- Assume the attacker has knowledge of the statistical properties of the plaintext
- The cipher needs to completely obscure statistical properties of original message
- Shannon introduced substitution-permutation (S-P) networks with the following goal
  - Diffusion – dissipate statistical structure of plaintext over bulk of ciphertext
  - Confusion – make the relationship between ciphertext and key as complex as possible

Feistel Cipher Structure

- Key is used to generate multiple subkeys
- Plaintext is split into two halves
  - Right half is fed to a Round function together with the subkey of that round – it remains unchanged and becomes the left half in the next round
  - Left half is XoR-ed with the output of the Round function – output becomes the right half in the next round
- Swapping of the two halves is permutation and operation on the left half is substitution
- There are multiple rounds like this
- Decryption is the reverse of this procedure - use the ciphertext as input to the algorithm, but use the subkeys in reverse order



Feistel Cipher Parameters

- Block size: Larger block sizes mean greater security
- Key size: Larger key size means greater security but may decrease encryption/decryption speed
- Number of rounds: Multiple rounds offer increasing security – (typically 10 to 16 rounds)
- Subkey Generation and Round Functions: Complex functions make cryptanalysis difficult