Lecture slides of the course

Information hiding & secret sharing

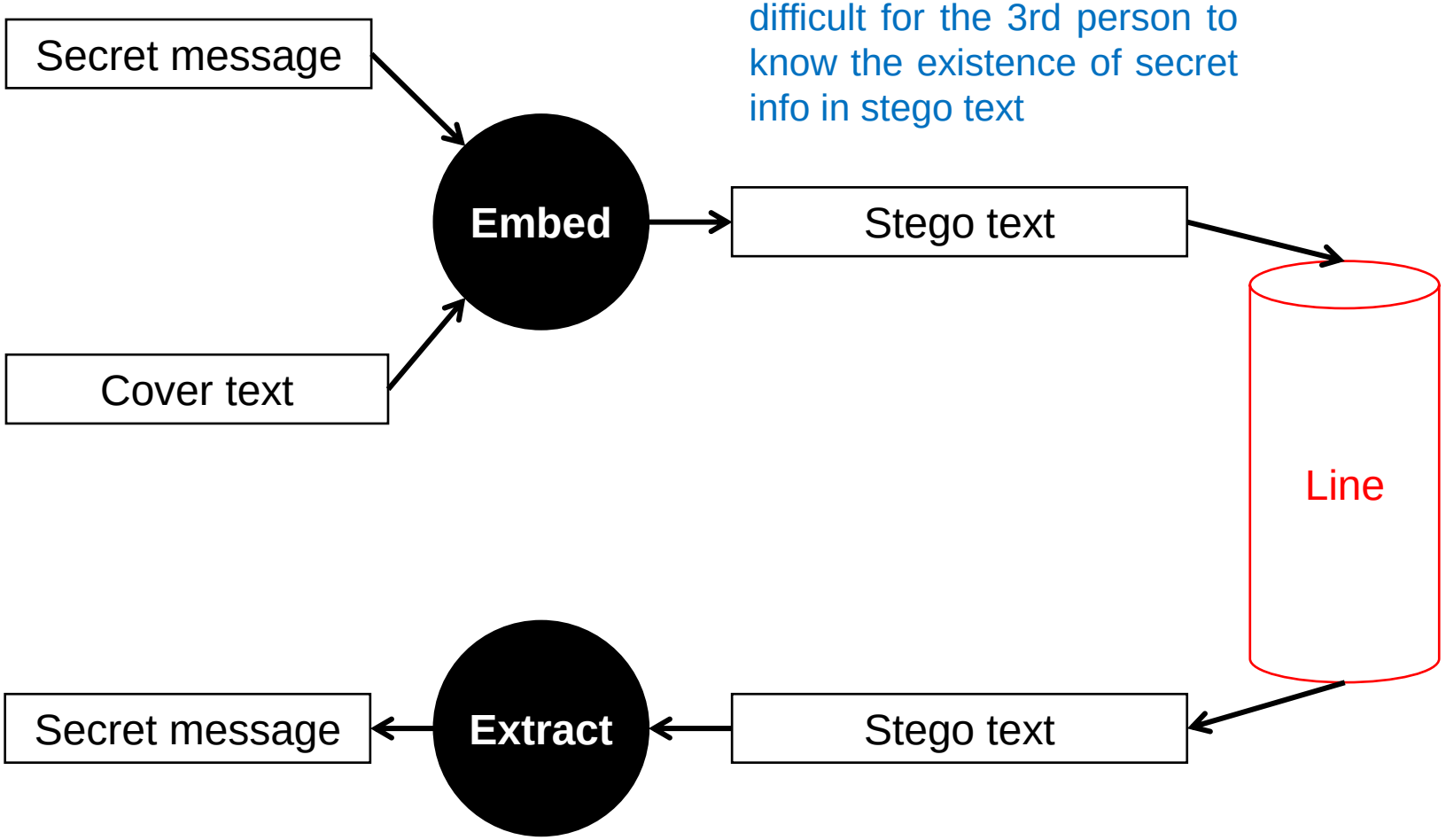
Text Steganography (P1)

Phạm Trọng Nghĩa

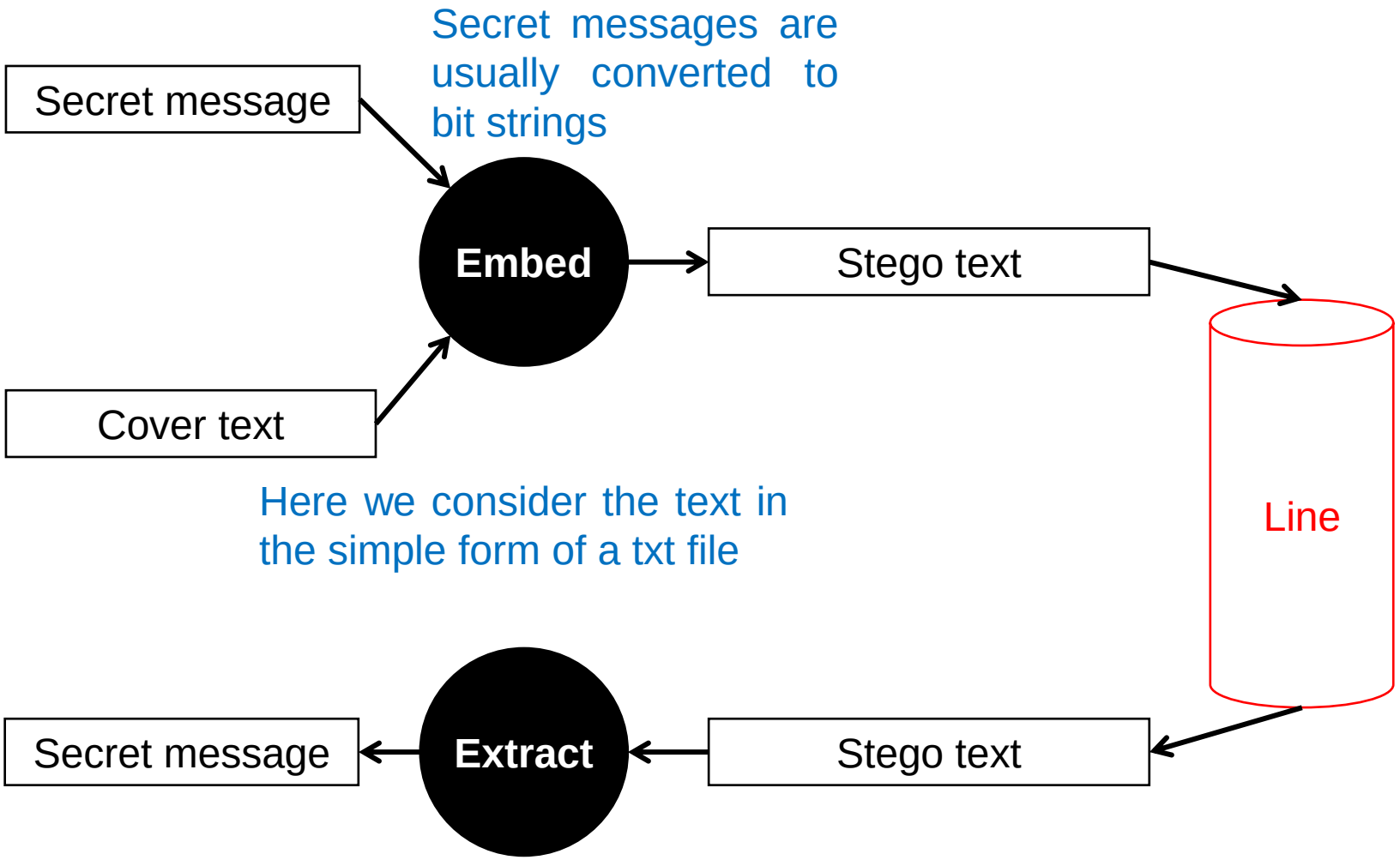
ptnghia@fit.hcmus.edu.vn

Text Steganography

Requirement: it must be difficult for the 3rd person to know the existence of secret info in stego text



Text Steganography



Text Steganography - Challenge

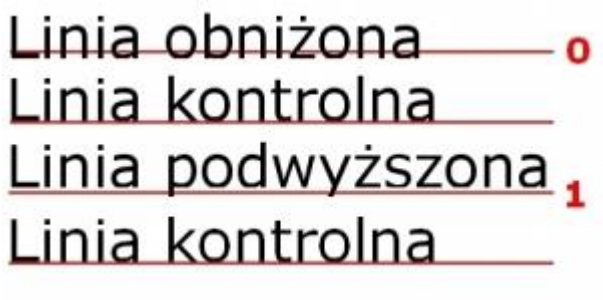
- Lacking of **redundant information** which is present in image, audio or a video file.
- The structure of text documents is **identical** with what we observe, while in other types of documents such as in picture, the structure of document is different from what we observe
- Storing text file require less memory and its faster as well as easier communication makes it preferable to other types of steganographic methods

Text Steganography

- Text Steganography is hiding information inside the text files.
  - Changing the format of existing text
  - Changing words within a text
  - Generating random character sequences
  - Using context-free grammars to generate readable texts.
- Various techniques used to hide the data in the text are:
  - Format Based Method
  - Random and Statistical Generation
  - Linguistic Method

Format Based Method

- Format based methods involve altering physically the format of text to conceal the information.
- The features are altered in such a manner that the human eye cannot sense them.



White space methods

- Example: bit 0 = 1 space, bit 1 = 2 spaces
- Why use whitespace
  - Not change the meaning of a phrase or sentence.
  - A casual reader is unlikely to take notice of slight modifications to white space

White space methods

- Example: bit 0 = 1 space, bit 1 = 2 spaces
- Where should white space be used to make the text look normal?
  - Inter-sentence spacing
  - End-of-line spaces
  - Inter-word spacing in justified text

Inter-sentence spacing

- Encodes a binary message into a text by placing either one or two spaces after each terminating character
- Demo!
- **Analyze**:
  - Invisiblility
  - Robustness
  - Capacity
  - Other problem?

End-of-line spaces

Idea

- At the end of each line, if you want to embed **bit 0** then **insert 1 space**, **bit 1 insert 2 spaces**, if you don't embed, **don't insert a space**
- Usually requires the length of the line after embedding ≤ a certain number (eg, 70) there are lines that will not be embedded

```
Hello Allice,  bit 1
 bit 0
Have you heard about steganography? I find it very interesting and  bit 1
want to share with you. Below is the introduction from Wiki. bit 0
  bit 1
Steganography is the practice of concealing a file, message, image, or
video within another file, message, image, or video. The word bit 0
steganography combines the Greek words steganos, meaning "covered,
concealed, or protected," and graphein meaning "writing".
```

No embed

No embed

No embed

End-of-line spaces

Analyze

- Invisiblility?
  - ☺
- Robustness?
  - Some word processors can automatically remove trailing whitespace ☹
- Capacity?
  - Can store **n bit** with **n = number of line** ☹
  - Some lines after embedded still have room to insert extra spaces → can take advantage of this to increase the capacity?
    - One way is to embed more than one bit in each line; e.g. embed 2 bits in each line: 00 = 1 space, 01 = 2 spaces, 10 = 3 spaces, 11 = 4 spaces
    - Is capacity increase guaranteed compared to embedding 1 bit in each line?

Inter-word spacing in justified text

Idea

- There are lots of spaces between words, why not embed bits (eg, **bit 0 = 1 space**, **bit 1 = 1 space insert 1 space**)?
  - Invisiblility ☹
- For text to be justified, it's normal to add some spaces between words take advantage of this to embed bits
- But can't simply do: **bit 0 = 1 space**, **bit 1 = 1 space insert 1 space**
  - Because of the constraint on the number of spaces to insert for each line
  - With the example below, line 1 must insert 0 spaces ➔ suppose we want to embed bit 1, you can't embed it, when extracting how do you know this line has no embedded bit?

```
Steganography is the practice of concealing a file, message, image, or<- Phải chèn 0 khoảng trắng
video within another file, message, image, or video. The word        <- Phải chèn 9 khoảng trắng
steganography combines the Greek words steganos, meaning "covered,   <- Phải chèn 4 khoảng trắng
concealed, or protected," and graphein meaning "writing".
```

Inter-word spacing in justified text

We only do it with alignment line (see the example in the image below).

Idea

With each line:

- Bit 0 = 10 = **1 space insert one space + 1 space**
- Bit 1 = 01 = **1 space + 1 space insert 1 space**
- No embedding = remaining cases

```
Hello Allice,

Have you heard about steganography? I find it very interesting and    <- Dòng căn lề
want to share with you. Below is the introduction from Wiki.

Steganography is the practice of concealing a file, message, image, or<- Dòng căn lề
video within another file, message, image, or video. The word         <- Dòng căn lề
steganography combines the Greek words steganos, meaning "covered,    <- Dòng căn lề
concealed, or protected," and graphein meaning "writing".
```

Inter-word spacing in justified text

Example 1: embed the bit string 101 into the cover text below

- Line 1: failed to embed. When extracting, encounter 1 space + 1 space ➔ this line is not embedded
- Line 2: has 9 spaces and must insert 9 more spaces → cannot be embedded, each space will be inserted one space. When extracting, encounter 2 spaces + 2 spaces immediately know that this line is not embedded
- Line 3: has 7 spaces and must insert 4 spaces → embeds 3 bits. When extracting
  - **1 space + 2 spaces → bit 1**
  - **2 spaces + 1 space → bit 0**;
  - Others, STOP

cover text

```
Steganography is the practice of concealing a file, message, image, or<- Phải chèn 0 khoảng trắng
video within another file, message, image, or video. The word         <- Phải chèn 9 khoảng trắng
steganography combines the Greek words steganos, meaning "covered,    <- Phải chèn 4 khoảng trắng
concealed, or protected," and graphein meaning "writing".
```

stego text

```
Steganography is the practice of concealing a file, message, image, or
video  within  another  file,  message,  image,  or  video.  The  word
steganography combines  the  Greek words steganos,  meaning  "covered,
concealed, or protected," and graphein meaning "writing".
```

bit 1 bit 0 bit 1

Inter-word spacing in justified text

Example 2: same with cover text in example 1, but embedded bit string is **10**

- Line 3 can embed 3 bits, but the embedded bit string has only 2 bits
- After embedding 2 bits, there are 3 spaces left and 2 spaces must be inserted → how to extract it?
  - One way is: 2 spaces + 2 spaces + 1 space
- But apart from this example, there are other cases; for example, in a line, after embedding all bits, there are 2 spaces left and 1 space must be inserted → how to extract it???
  - One way to make things simpler is to always have bits to embed: after all the bit strings are embedded, we embed: **one bit 1, the other bit 0**
  - When extract, do as usual. The extracted bit string will include embedded bit string + tail 100...; then cut the tail 100...

cover text

```
Steganography is the practice of concealing a file, message, image, or<- Phải chèn 0 khoảng trắng
video within another file, message, image, or video. The word        <- Phải chèn 9 khoảng trắng
steganography combines the Greek words steganos, meaning "covered,   <- Phải chèn 4 khoảng trắng
concealed, or protected," and graphein meaning "writing".
```

Inter-word spacing in justified text

Embedding algorithm

Input:

- msg_bits: bit string corresponding to secret message
- cover_text: text used to shield confidential information
- text_width: line length after alignment

Output:

- stego_text: là cover_text after embedded msg_bits

Inter-word spacing in justified text

Embedding algorithm

- b = 0 # Index of each bit in msg_bit
- l = 0 # Index of each line in cover_text
- Loop while l < number of line in cover_text
  - If l is aligned line # Can embedded bit
    - n = the number of inter-word in line l
    - m = the number of while space need to insert for line l be justified = text_width – len(line l)
    - If 0 < m < n # Can embedded min(m, n-m) bit
      - Traverse min(m, n-m) first whitespace pair in line l
        - If b < len(msg_bits) then embed msg_bits[b] into the whilespace pair: bit 0 = 1 space **imsert 1 more space**, bit 1 = 1 space + 1 space insert 1 more space; b += 1
        - Otherwises:
          - Insert bit 1 (only 1 time for whole embedding process)
          - Othertime: insert bit 0
      - For remaining while space of line l: if min(m, n-m) = n-m then insert 1 space for each inter-word
    - If m ≥ n: cannot embedded, but still need insert at least 1 more space for each interword for line l be justified
  - l += 1
- If still cannot insert bit 1 (in tail 100...): EMBEDDING FAIL!

Inter-word spacing in justified text

Extract algorithm

Input: stego_text; output: extracted_msg_bits

- extracted_msg_bits = empty
- l = 0 # Index of each line in cover_text
- Loop while l < #line in stego_text
  - If l is justified line
    - Traverse each inter-word of line l
      - If 2 space + 1 space: add bit 0 to extracted_msg_bits
      - Else if 1 space + 2 space: add bit 1 to extracted_msg_bits
      - Else: stop traverse
  - l += 1
- Cut tail 100.. out of extracted_msg_bits

Inter-word spacing in justified text

- Try pasting the content in the stego_text file into gmail to send it to Alice...

- **Why in gmail the content of stego_text is not side-aligned?**
- Because the default font allows different characters to have different widths

- **A font in gmail that all characters have the same width is Fixed Width**

Inter-word spacing in justified text

Analyze

- Invisiblility?
  - ☺
- Robustness?
  - Some word processors can automatically normalize spaces between words ☹
- Capacity?
  - Contains the range of **lines × min(m, n-m) bits** where m is the number of spaces that must be inserted for the line to be aligned and n is the number of spaces for the line

Conclusion about while space method

- **General observe**: the disadvantage of this group of methods is that it is unsustainable when reformatted (some word processors automatically normalize whitespace)
- The next two groups of methods will overcome this drawback and can also be used at the same time as the whitespace method group