# Security in Computing & Information Technology (COSC2536/COSC2537)

## Lecture 8: Secret Data Hiding-Steganography



RMIT UNIVERSITY

# Lecture Overview

- During this lecture, we will learn
  - Basics of Secret Data Hiding(*i.e.* Steganography)
  - Why it is important
  - Secret Data Hiding Examples
    - Secret Data Hiding in Number
    - Secret Data Hiding in Signal
    - Secret Data Hiding in Image
    - Secret Data Hiding in Text

# What is Steganography

- Summarizing, steganography can be defined as hiding technique that is able to embed secret data inside another data host in such a way that prevents unauthorized persons accessing the secret message
- In recent years, data hiding models are gaining popularity in establishing more secured communication channels for delivering secret messages since there is huge demand for sending sensitive information over the net.

ancient Greek slaves/couriers shaving their head, tattooing a message on their scalp, and regrowing their hair.

Note: Historically, steganography used by military more often than cryptography

# What is Steganography

- An art of information hiding that hides a **secret message** within an object
- Existence of the hidden message in the object cannot be identified
- ***Steganography*** and ***cryptography*** share a common goal, however the way they are used differs significantly.
- ***Steganography*** hides the existence of secret message whereas ***cryptography*** provides security with respect to the content of message.
- Steganography is used to provide **privacy** and **authenticity** of sensitive data.

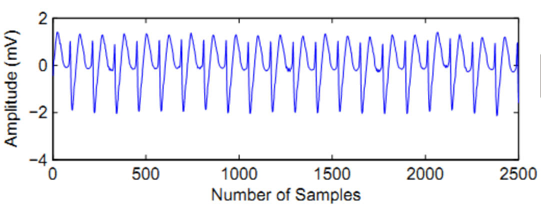
# General Concepts of Steganography

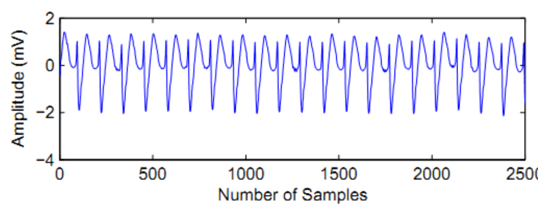
- The modern formulation of steganography is often given in terms of the **prisoner's problem** where ***Alice*** and ***Bob*** are two inmates who wish to communicate in order to hatch an escape plan.
- However, all communication between them is examined by the warden, ***Wendy***, who will put them in solitary confinement at the slightest suspicion of covert communication.
- Alice wishing to send a ***secret message (m)*** to Bob, "*embeds*" ***m*** into a ***cover-object (C)*** using a ***stego-key (K)***, and obtains a ***stego-object (S)***.
- The ***stego-object (S)*** is then sent through the public channel and ***stego-key (K)*** is sent to Bob using a private channel.

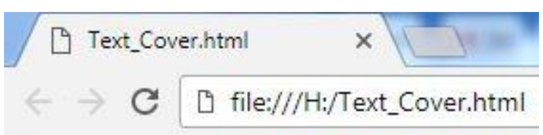
# General Concepts of Steganography

*cover-object*

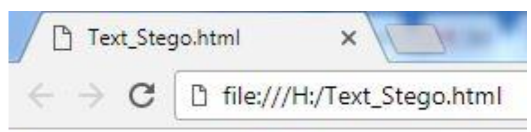
*stego-object*




"The time has come," the Walrus said,
"To talk of many things:
Of shoes and ships and sealing wax
Of cabbages and kings
And why the sea is boiling hot
And whether pigs have wings."

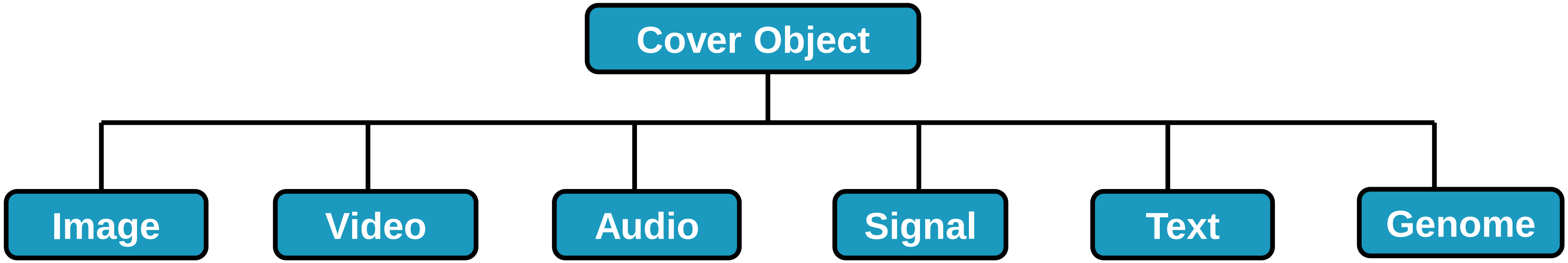



"The time has come," the Walrus said,
"To talk of many things:
Of shoes and ships and sealing wax
Of cabbages and kings
And why the sea is boiling hot
And whether pigs have wings."

- ***We can hide in images, ECG, texts, and many other objects***
- ***cover-object and stego-object appear to be same but stego-object contains hidden message***

# General Concepts of Steganography(1)

- ***Cover-object (C):*** refers to the object used as the carrier to embed messages into. Generally, less important in the communication.

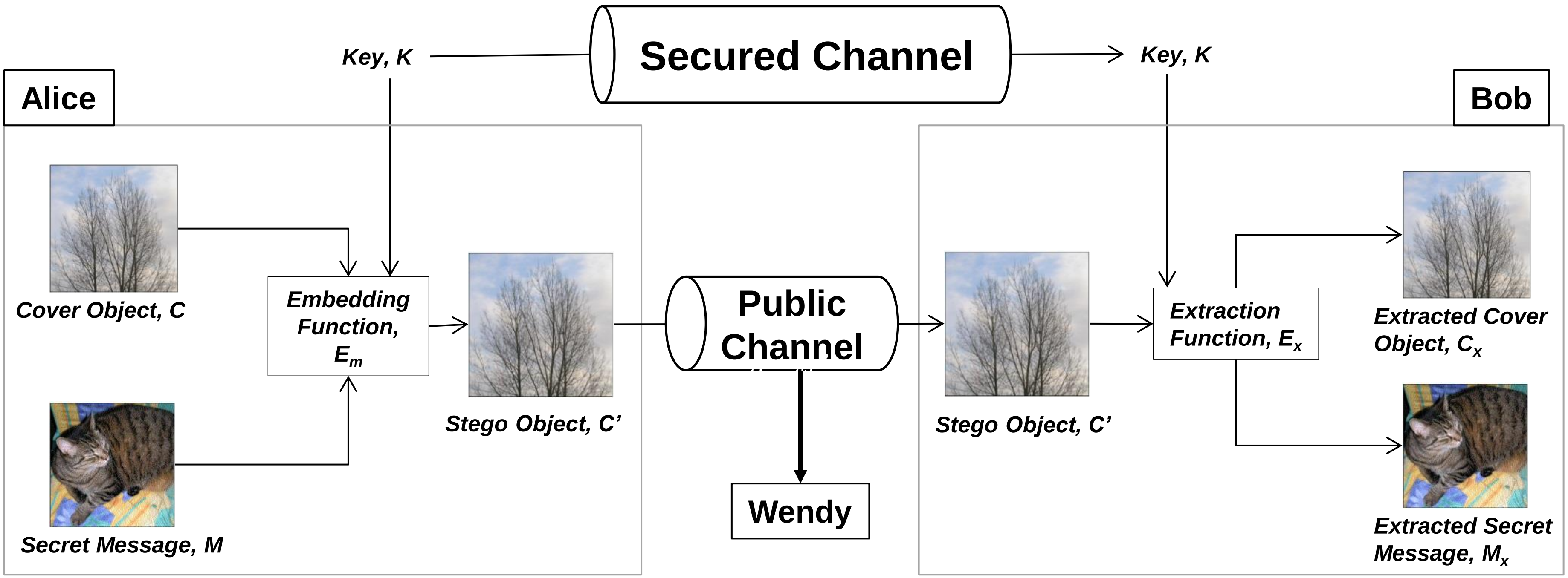



- ***Stego-object (S):*** refers to the object which is carrying a hidden **message**. So, given a **cover object** and a message. The goal of the steganographer is to produce a **stego object** which would carry the message.
- ***Stego-Key (K):*** refers to the secret information (i.e. secret key) that will be used to extract secret message. For example, location of hidden message in the stego-object.

# General Concepts of Steganography(2)

- ***Embedding Function ($E_M$):*** refers to the algorithm used by **Alice (sender)** to hide secret message into the cover-object and produce stego-object.
- ***Extraction Function ($E_X$):*** refers to the algorithm used by **Bob (receiver)** to extract secret message from the stego-object.
- Generally, the ***embedding function ($E_M$)*** and ***extraction function ($E_X$)*** are public, i.e. known by **Wendy (warden)**

# Modern Steganography: General Model




**Embedding Function,** $$E_m : C \oplus K \oplus M \rightarrow C'$$

**Extraction Function,** $$E_x(C', K) : C' \oplus K \rightarrow C_x, M_x$$

# Why Steganography is Important

- Steganography is used to provide **privacy** and **authenticity** of sensitive data.
- Sensitive data can be hidden in insensitive data before storing in *Cloud data centre (CDC)* **(provides privacy)**
  - For example, patient personal information can be hidden in patient medical record to provide privacy
- Data owners identification information can be hidden in data so that a data consumer can verify if the data is actually originated by the owner **(provides authenticity)**
  - For example, sensor ID can be hidden in signal (eg. ECG data) before sending it to *CDC* so that sensor data consumer (eg. doctor) can verify if the data is actually originated from that sensor or not.
  - Can also hide confidential patient information inside ECG which is usuallu huge in size.

# Steganography Example-1

## Secret Data Hiding in Numbers

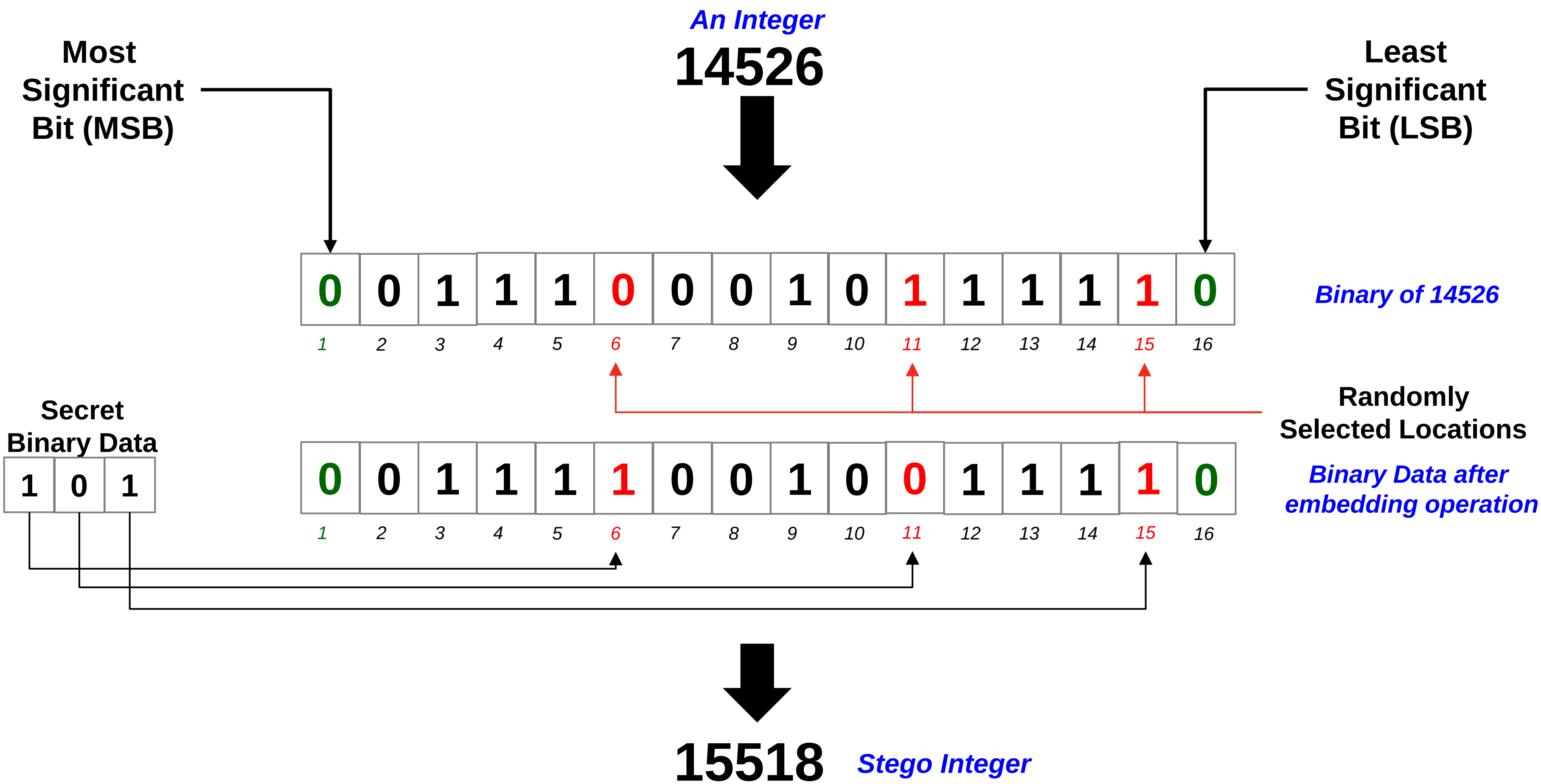
# Embedding Procedure of Bits in a Number(1)

- Assume that Alice wants to hide a ***secret binary message (M = 101)*** in **an integer** number **14526**
- Here, length of secret message is **3** *(i.e. length = 3) and cover data = {14256}*
- ***Embedding Procedure:***
  - The number in cover data is converted to ***16-bit binary string***: **0011100010111110**
  - Alice will hide each bit of the *secret binary message* in **randomly selected locations** of the binary strings.
  - Let, randomly selected locations are: 6th, 11th and 15th bits of **0011100010111110.** *(i.e. locations = {6,11,15})*
  - Therefore, **Stego-Key ($S_K$)** becomes:

$$S_K = \langle locations \rangle = \langle \{6,11,15\} \rangle$$

# Embedding Procedure of Bits in a Number (2)

- Secret bits are **embedded** as follows:



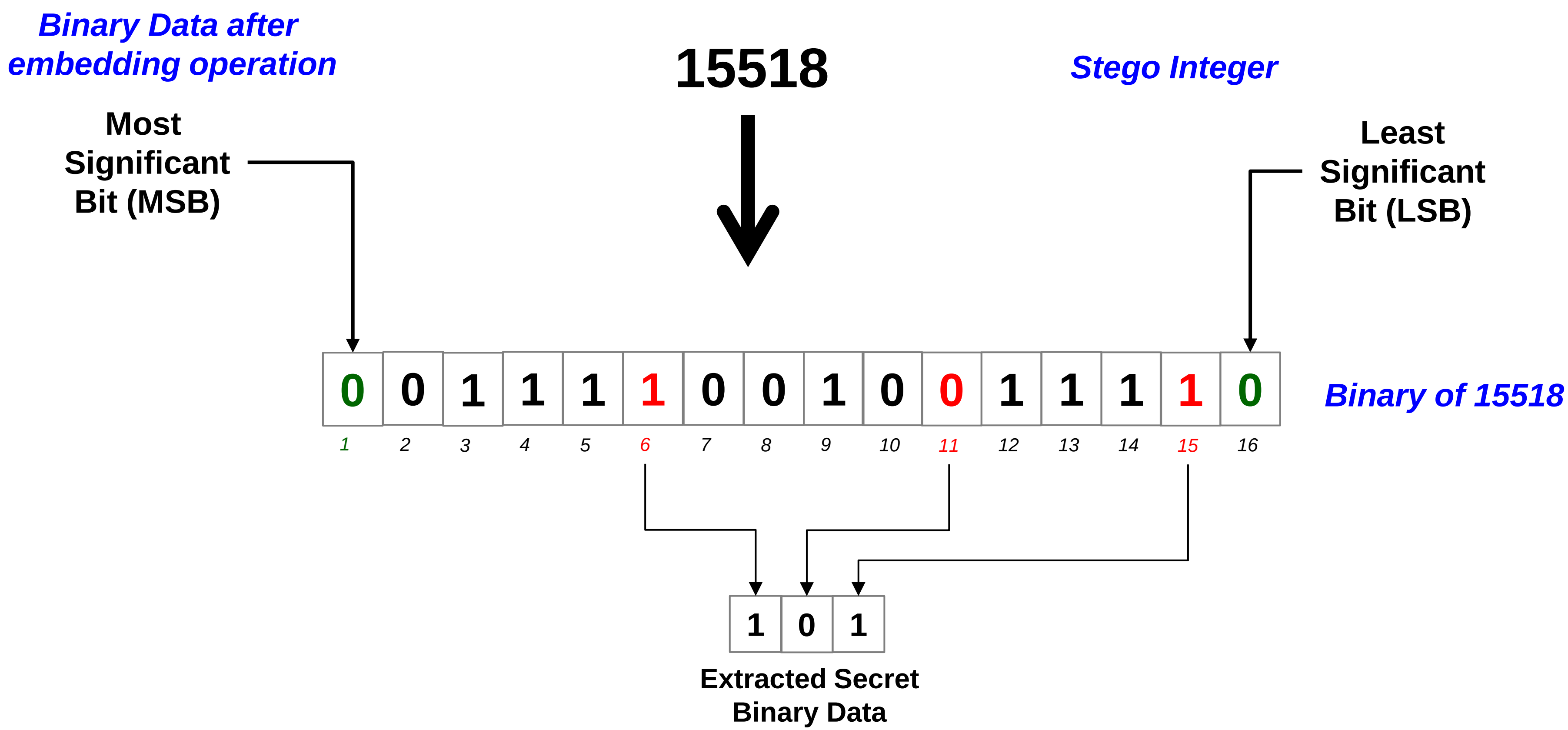

- Hence, ***stego data***, ***S = {15518}***
- $\mathbf{S_K}$ and **S** are sent to Bob.

# Extraction Procedure of Bits from A Number (3)

- Bob receives $\mathbf{S_K}$ and **S** from Alice and wants to extract secret message from stego data (S) .
- Bob knows from $S_K$ that the length of secret message is **3**
- ***Extraction Procedure:***
  - The number in stego data (S) (i.e. **15518**) is converted to ***16-bit binary string*** to obtain **binary string 0011110010011110.**
  - Bob will obtain **locations** of secret message bits in binary strings from $\mathbf{S_K}$
  - The locations are: 6th, 11th and 15th bits of **0011110010011110.** *(i.e. locations = {6,11,15})*

# Extraction Procedure of Bits in a Number (4)

- Secret bits are **extracted** as follows:




- Therefore, extracted secret message, $\mathbf{M_X} = \mathbf{\mathit{101}}$

# Embedding Procedure of Bits in Multiple Numbers (1)

- Assume that Alice wants to hide a ***secret binary message (M = 10010)*** in **five integer numbers: 19, 17, 11, 15, 21**
- Here, length of secret message is **5** *(i.e. length = 5) and cover data = {19, 17, 11, 15, 21}*
- ***Embedding Procedure:***
  - Each number in cover data is converted to ***8-bit binary string*** to obtain five **binary strings.**
  - Alice will hide each bit of the *secret binary message* in **randomly selected locations** of five binary strings.
  - Let, randomly selected locations are: 7th, 8th, 6th, 7th and 6th bits of **19, 17, 11, 15** and **21.** *(i.e. locations = {7,8,6,7,6})*
  - Therefore, **Stego-Key ($S_K$)** becomes:

    ***$S_K$ = <locations> = <{7,8,6,7,6}>***
- Secret bits are **embedded** as follows:

| ***Number*** | ***Binary String*** | ***Selected Locations*** | ***Secret Bit to Hide*** | ***Stego Binary*** | ***Stego Number*** |
|---|---|---|---|---|---|
| **19** | **00010011** | **00010011** | **1** | **00010011** | **19** |
| **17** | **00010001** | **00010001** | **0** | **00010000** | **16** |
| **11** | **00001011** | **00001011** | **0** | **00001011** | **11** |
| **15** | **00001111** | **00001111** | **1** | **00001111** | **15** |
| **21** | **00010101** | **00010101** | **0** | **00010001** | **17** |

- Hence, ***stego data***, ***S = {19,16,11,15,17}***
- **$S_K$** and **S** are sent to Bob.

# Extraction Procedure of Bits from Multiple Numbers (2)

- Bob receives $\mathbf{S_K}$ and **S** from Alice and wants to extract secret message from stego data (S) .
- Bob knows from $S_K$ that the length of secret message is **5**
- ***Extraction Procedure:***
  - Each number in stego data (S) is converted to ***8-bit binary string*** to obtain five **binary strings.**
  - Bob will obtain **locations** of secret message bits in five binary strings from $\mathbf{S_K}$
  - The locations are: 7th, 8th, 6th, 7th and 6th bits of **19, 16, 11, 16** and **17.** *(i.e. locations = {7,8,6,7,6})*
- Secret bits are **extracted** as follows:

| ***Stego Number*** | ***Binary String*** | ***Selected Locations*** | ***Extracted Bit*** |
|---|---|---|---|
| **19** | **00010011** | **00010011** | **1** |
| **16** | **00010000** | **00010000** | **0** |
| **11** | **00001011** | **00001011** | **0** |
| **15** | **00001111** | **00001111** | **1** |
| **17** | **00010001** | **00010001** | **0** |

- Therefore, extracted secret message, $\mathbf{M_X}$ **=** ***10010***

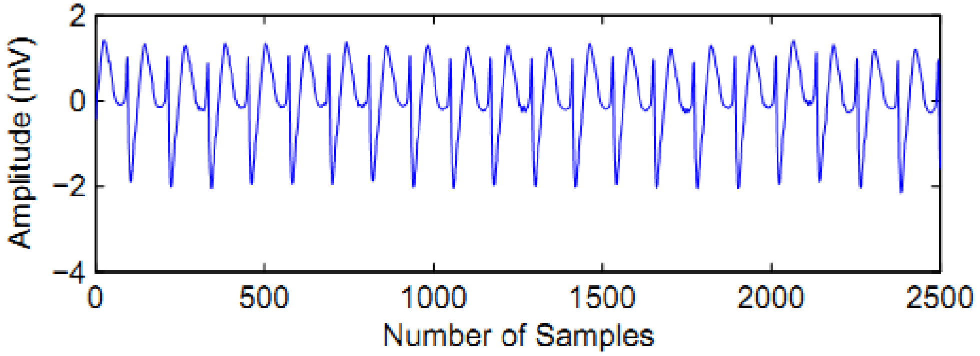
# Steganography Example-2

## Secret Data Hiding in Signal

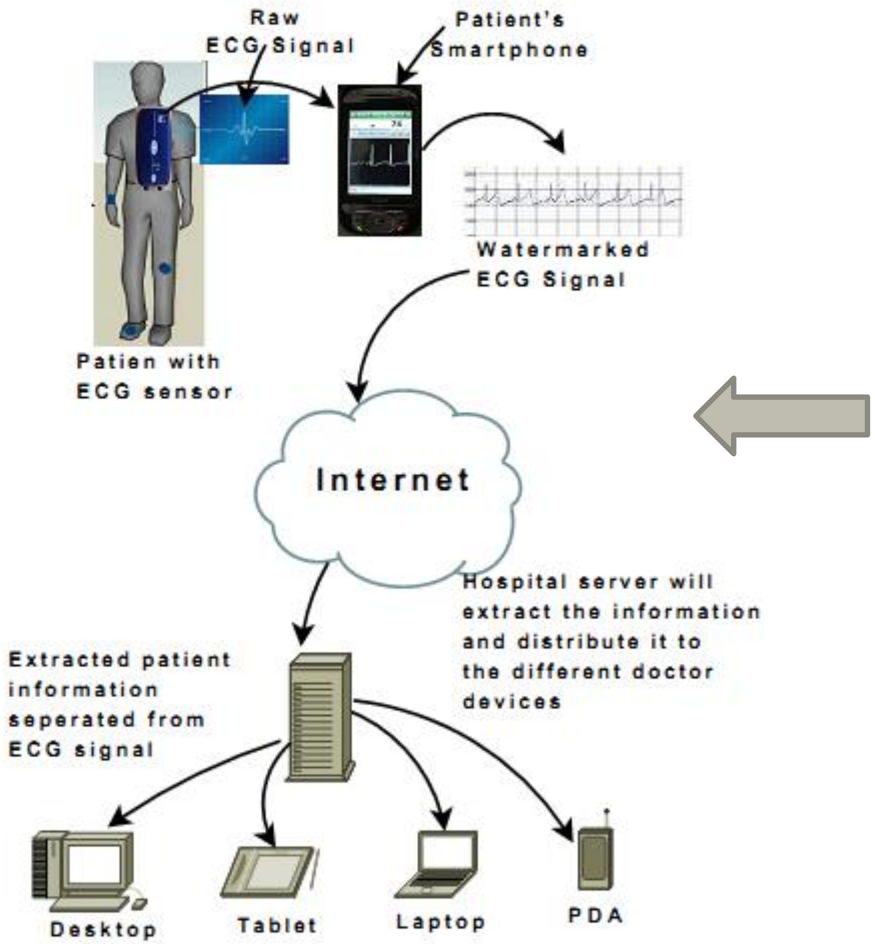
### (ECG Data Steganography)

# Steganography in Ehealth

- In a typical wireless telemonitoring scenario for remote cardiac patients wireless e-health systems send ECG signals of patients and their personal information separately to the hospital servers without any protection
- Health Insurance Portability and Accountability Act (HIPAA) of 1996 in US mandates that confidential and private information related to patients be protected and sent over the net and stored in a secured manner

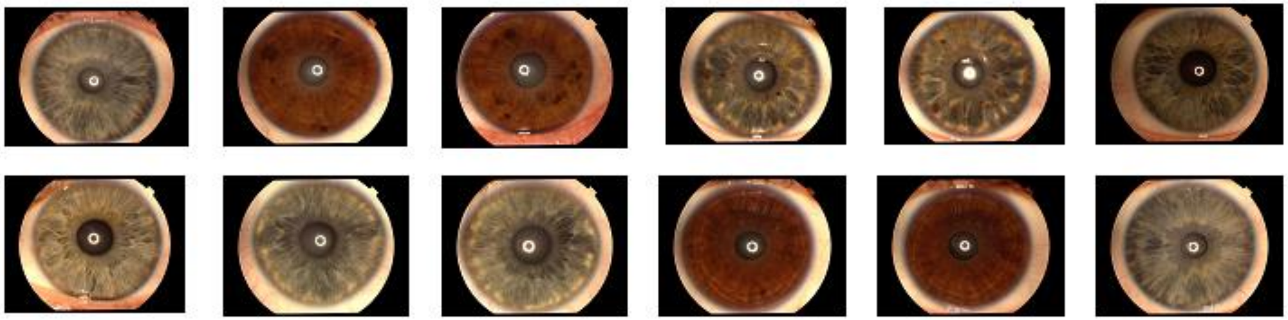


# Hiding in ECG: Wireless Health Scenario



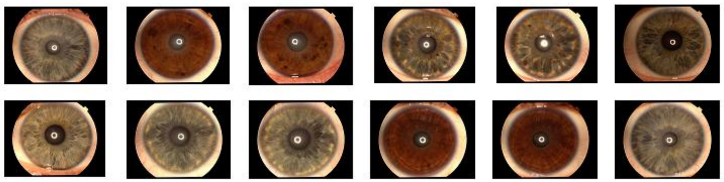

steganography technique as shown in the Figure allows ECG samples to be gathered on a wireless mobile device and sent with patient sensitive personal information such as name, age, personal ID, etc. At hospital servers, patients data can be extracted from the received ECG signal

# Steganography with ECG



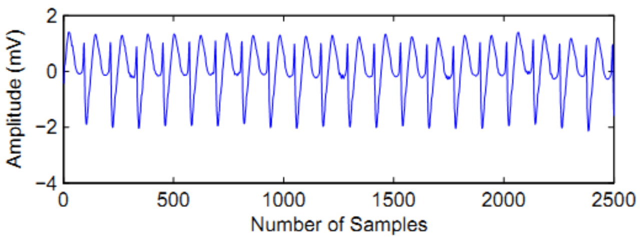
**Sensitive Biometric Data**

# Steganography with ECG



01100111011010011 1
10111110101000011 0
10001100011101100 0
10101011111000101 0

Iris and faces
Converted into secret
Binary data

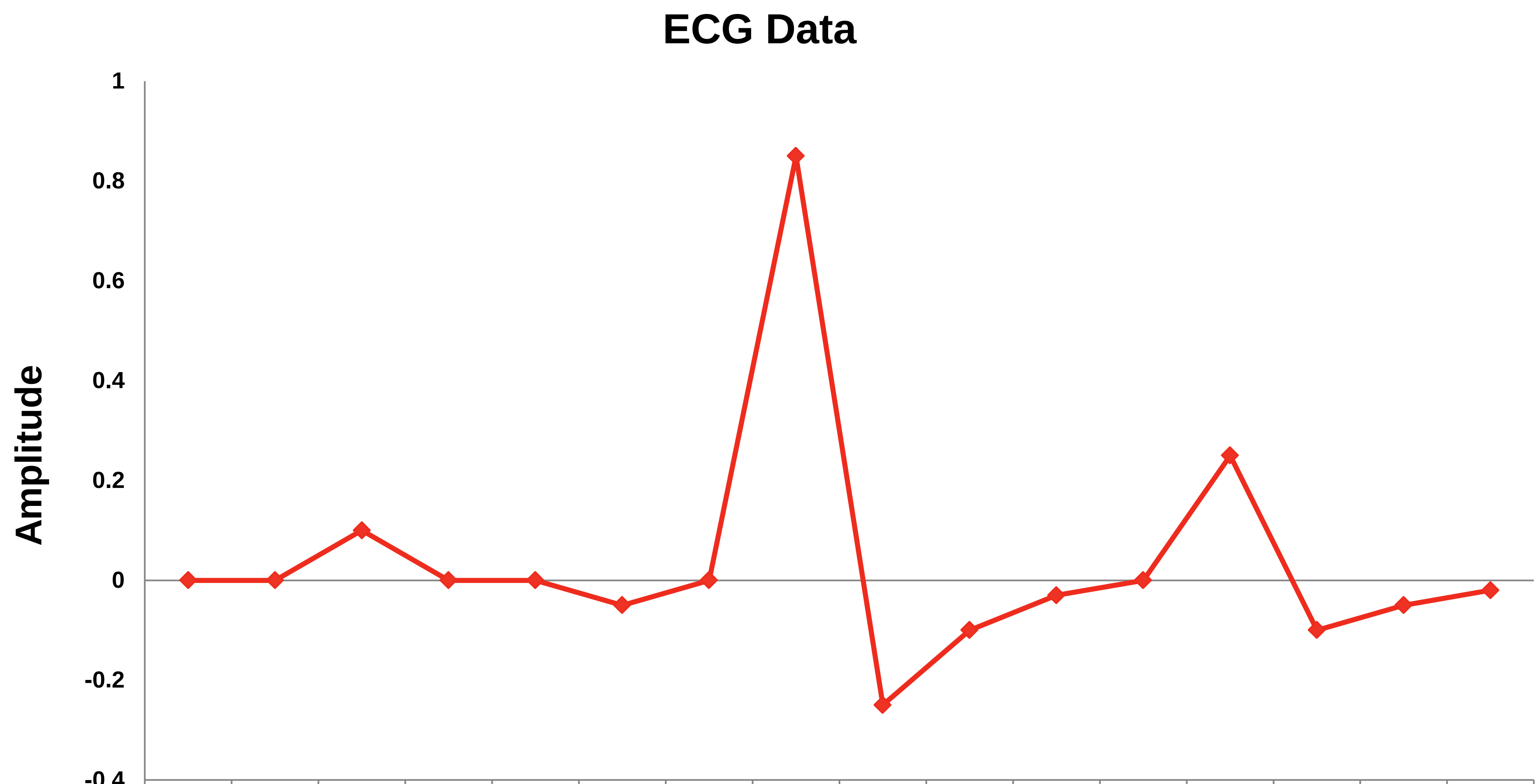



$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Hide secret
Binary data
Into ECG binary
data

# Embedding Procedure of Bits in ECG Signal (1)

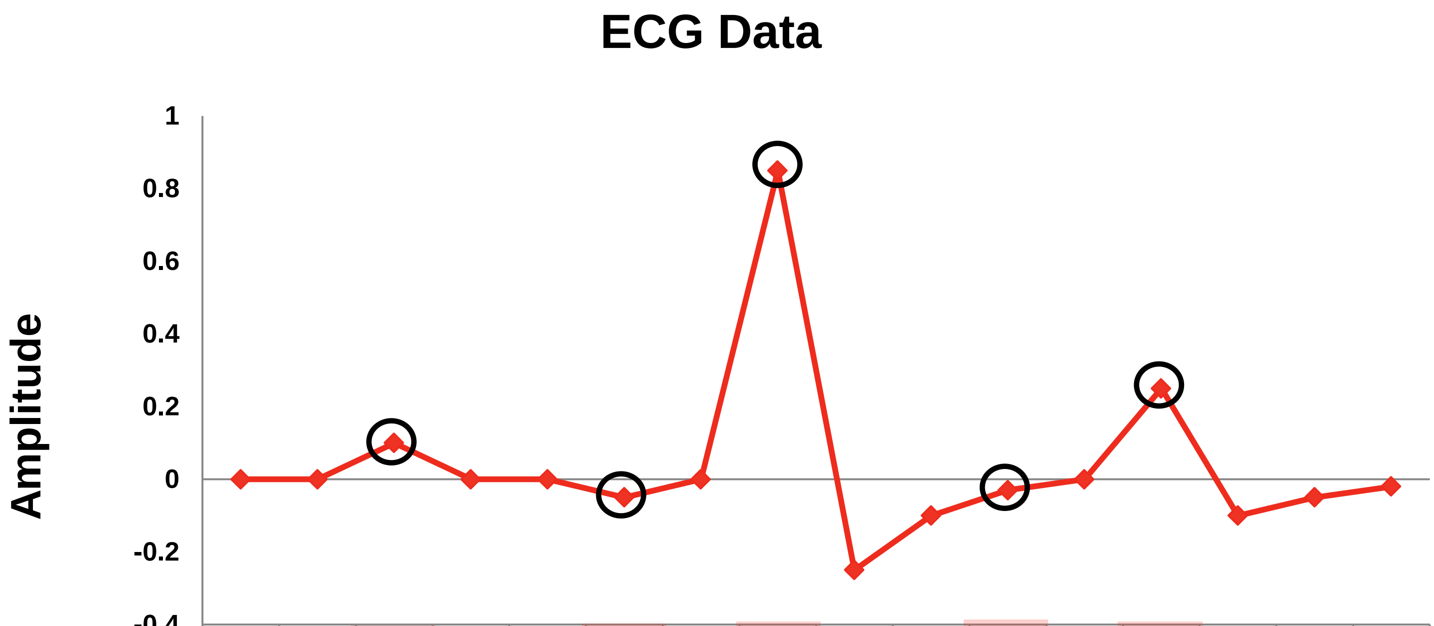
- Assume that Alice wants to hide a ***secret binary message (M = 00110)*** in **an ECG Signal** with **16** samples as shown below:




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Series1 | 0 | 0 | 0.1 | 0 | 0 | -0.05 | 0 | 0.85 | -0.25 | -0.1 | -0.03 | 0 | 0.25 | -0.1 | -0.05 | -0.02 |

# Embedding Procedure of Bits in ECG Signal (2)

- Here, length of secret message ***(M = 00110)*** is **5** *(i.e. length = 5) and cover data = {0, 0, 0.1, 0, 0, -0.05, 0, 0.85, -0.25, -0.1, -0.03, 0, 0.25, -0.1, -0.05, -0.02}*
- ***Embedding Procedure:***
  - Alice will **randomly** select **five** samples of cover ECG data in ascending order.
  - Let, randomly selected samples are: 3rd, 6th, 8th, 11th and 13th elements of ECG signal**.** *(i.e. locations = {3, 6, 8, 11, 13}). [See the following Figure.]*




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Series1 | 0 | 0 | 0.1 | 0 | 0 | -0.05 | 0 | 0.85 | -0.25 | -0.1 | -0.03 | 0 | 0.25 | -0.1 | -0.05 | -0.02 |

# Embedding Procedure of Bits in ECG Signal (3)

- ***Embedding Procedure (cont.):***
  - Each selected sample in cover ECG data is converted to ***32-bit binary string*** (using IEEE-754 Single Precision) to obtain **5 binary strings.**

    ***Use the following link for converting Floating Point Number into Binary String:***

    ***https://www.h-schmidt.net/FloatConverter/IEEE754.html***
  - Each secret bit is embedded in LSB of corresponding binary string and converted to floating point number as follows:

| ***Sample Number*** | ***Sample Value*** | ***Binary String*** | ***Secret Bit to Hide in LSB*** | ***Stego Binary*** | ***Stego Number*** |
|---|---|---|---|---|---|
| **3** | **0.1** | **00111101110011001100110011001101** | **0** | **00111101110011001100110011001100** | **0.099999994** |
| **6** | **-0.05** | **10111101010011001100110011001101** | **0** | **10111101010011001100110011001100** | **-0.049999997** |
| **8** | **0.85** | **00111111010110011001100110011010** | **1** | **00111111010110011001100110011011** | **0.8500001** |
| **11** | **-0.03** | **10111100111101011100001010001111** | **1** | **10111100111101011100001010001111** | **-0.03** |
| **13** | **0.25** | **00111110100000000000000000000000** | **0** | **00111110100000000000000000000000** | **0.25** |

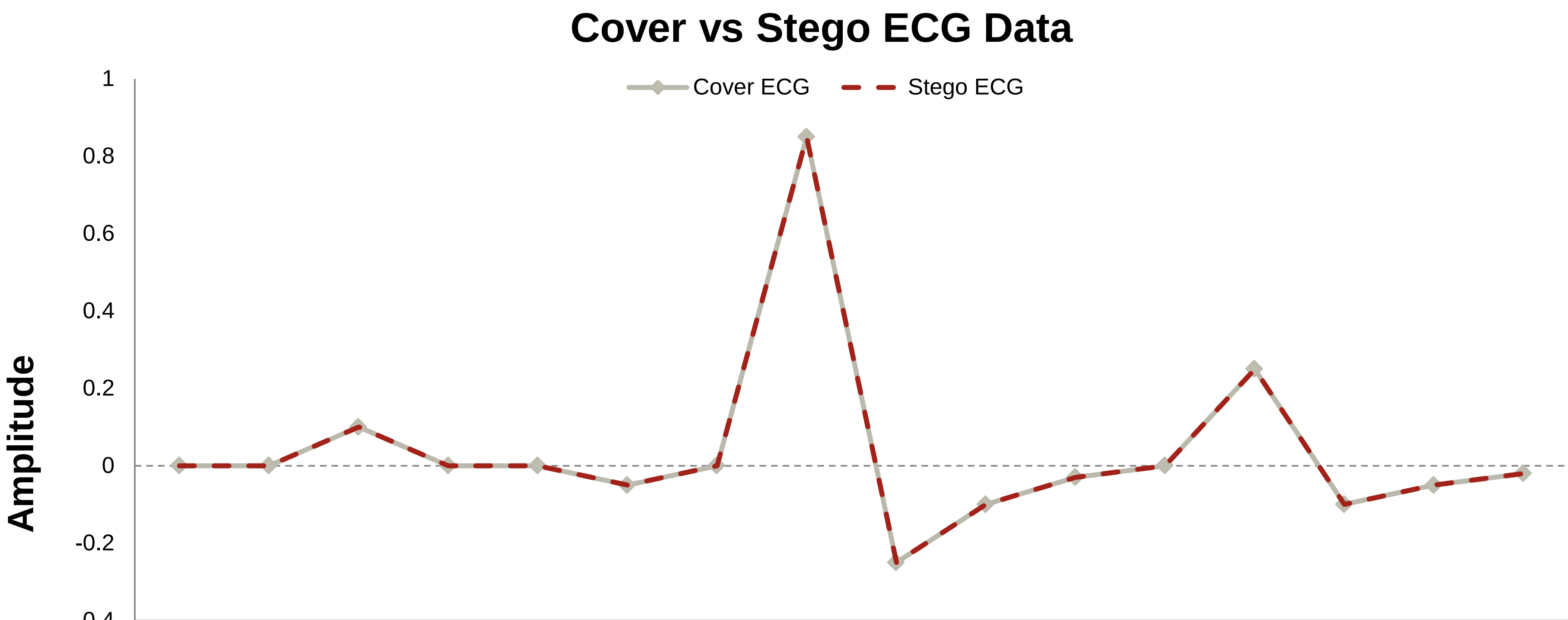
- Therefore, **Stego-Key ($S_K$)** becomes:

  $$S_K = <length, locations> = <5, \{3, 6, 8, 11, 13\}>$$
- Hence, ***stego data:***

***S = {****0, 0, 0.099999994, 0, 0, - 0.049999997, 0, 0.8500001, -0.25, -0.1, -0.03, 0, 0.25, -0.1, -0.05, -0.02****}***

- **$S_K$** and **S** are sent to Bob.

# Embedding Procedure of Bits in ECG Signal (4)




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cover ECG | 0 | 0 | 0.1 | 0 | 0 | -0.05 | 0 | 0.85 | -0.25 | -0.1 | -0.03 | 0 | 0.25 | -0.1 | -0.05 | -0.02 |
| Stego ECG | 0 | 0 | 0.09999 | 0 | 0 | -0.0499 | 0 | 0.85000 | -0.25 | -0.1 | -0.03 | 0 | 0.25 | -0.1 | -0.05 | -0.02 |

# Extraction Procedure of Bits from Stego ECG Data

- Bob receives $\mathbf{S_K}$ and **S** from Alice and wants to extract secret message from stego data (S) .
- Bob knows from $S_K$ that the length of secret message is **5**
- ***Extraction Procedure:***
  - Bob will obtain ECG samples that contain secret message bits from $\mathbf{S_K}$
  - The samples are: 3rd, 6th, 8th, 11th and 13th elements of ECG signal**.** *(i.e. locations = {3, 6, 8, 11, 13}).*
  - Each selected sample in stego data (S) is converted to ***32-bit binary string*** and LSB of corresponding binary string is extracted to construct secret message.
- Secret bits are **extracted** as follows:

| *Sample Number* | *Sample Value* | *Binary String* | *Secret Bit from LSB* |
|---|---|---|---|
| **3** | **0.099999994** | **00111101110011001100110011001100** | **0** |
| **6** | **-0.049999997** | **10111101010011001100110011001100** | **0** |
| **8** | **0.8500001** | **00111111010110011001100110011011** | **1** |
| **11** | **-0.03** | **10111100111101011100001010001111** | **1** |
| **13** | **0.25** | **00111110100000000000000000000000** | **0** |

- Therefore, extracted secret message, $\mathbf{M_X} = \boldsymbol{00110}$

# Steganography Example-3

## Secret Data Hiding in Images

# Image Steganography

Images use 24 bits for color: RGB

- 8 bits for red, 8 for green, 8 for blue

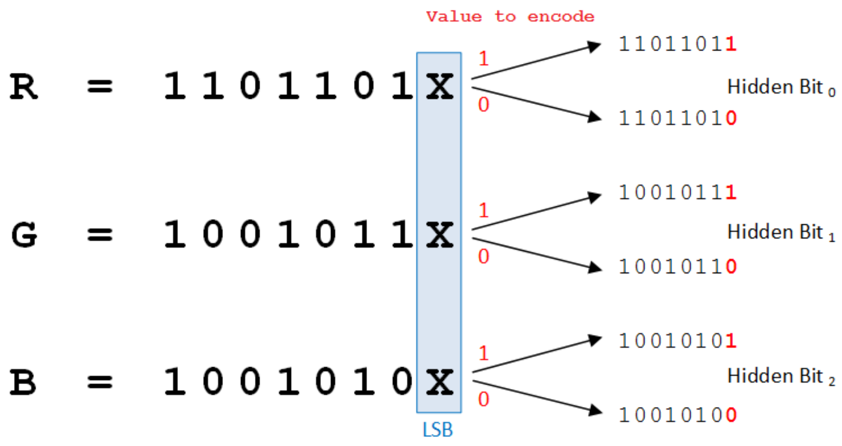
For example

- 0x7E 0x52 0x90 is this color
- 0xFE 0x52 0x90 is this color

While

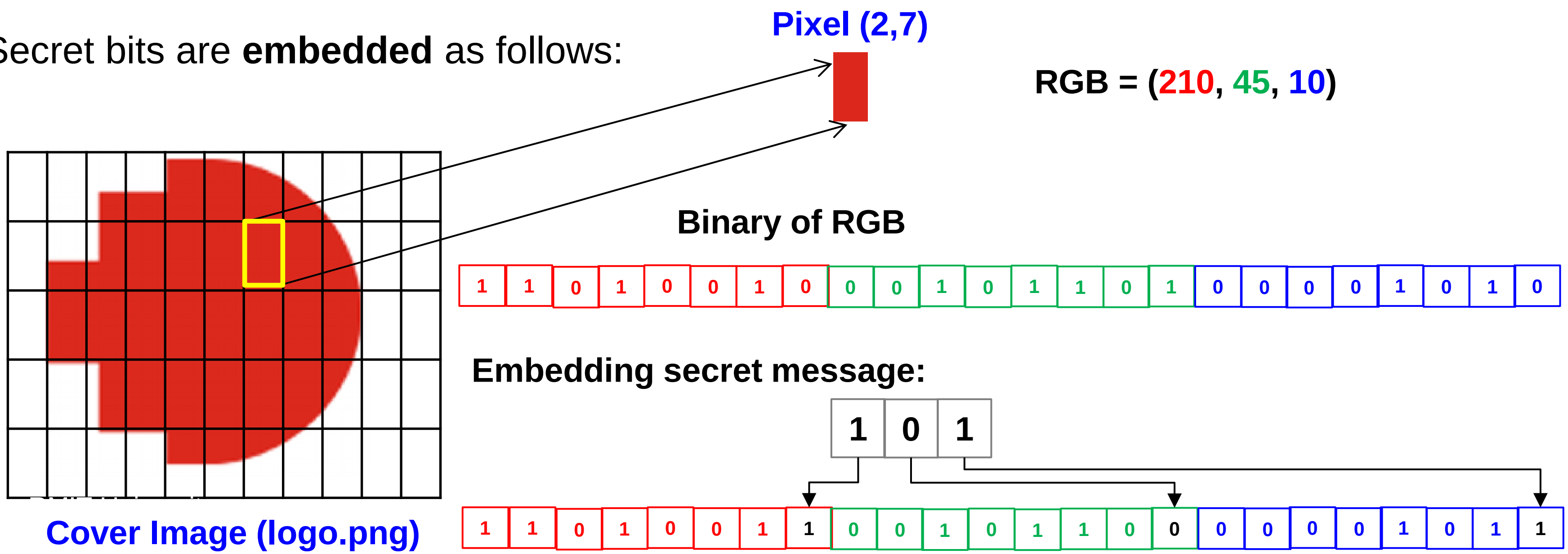
- 0xAB 0x33 0xF0 is this color
- 0xAB 0x33 0xF1 is this color

Low-order bits don't matter…

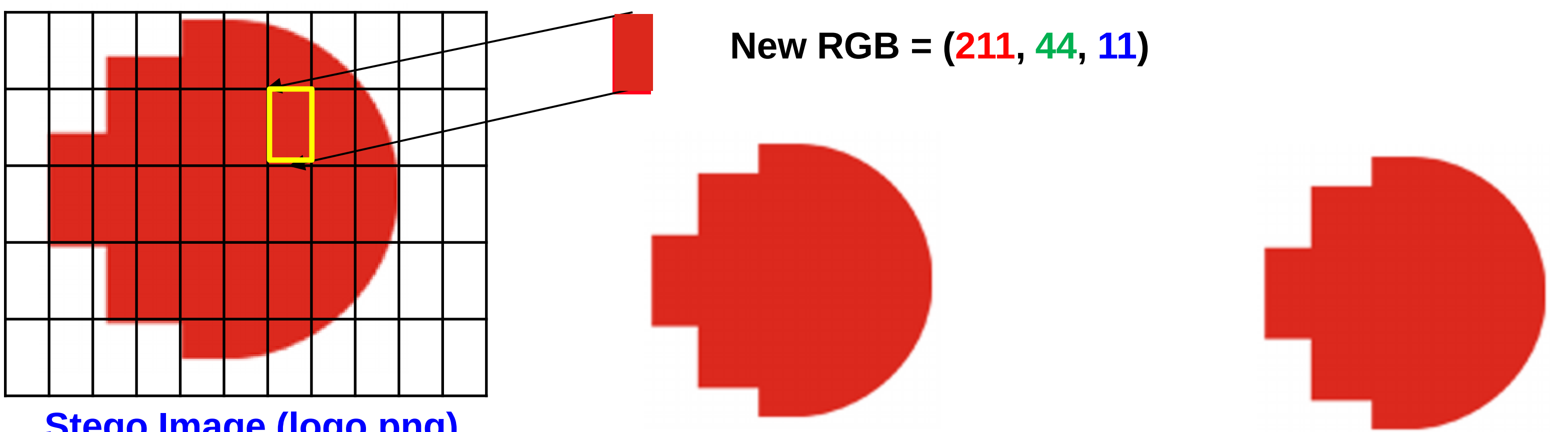


# Embedding Procedure of Bits in an Image

- Secret bits are **embedded** as follows:



Cover Image (logo.png)



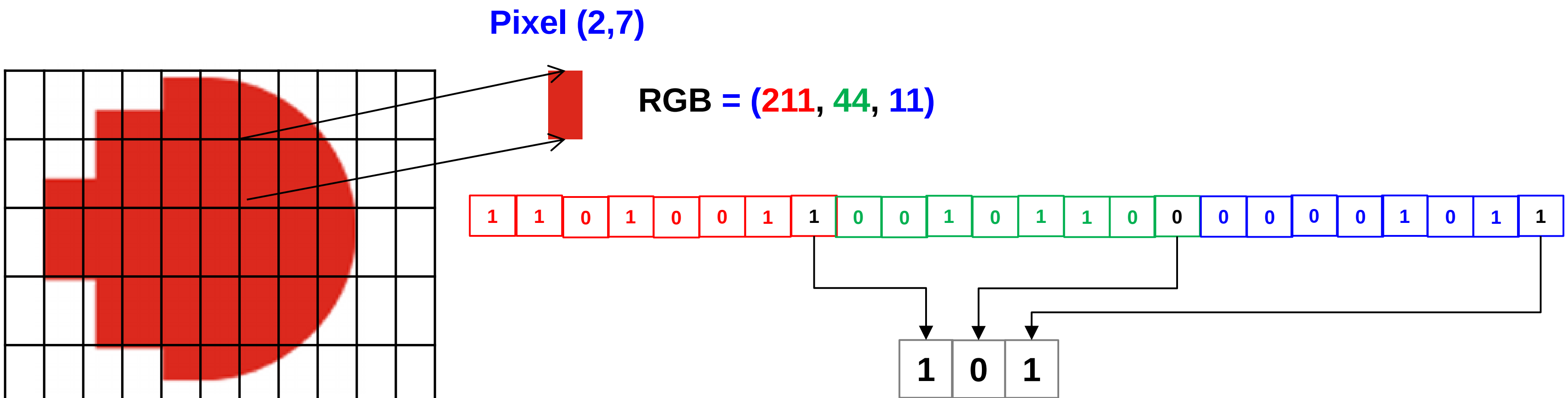
Stego Image (logo.png)

Cover Image

Stego Image

# Extraction Procedure of Bits from an Image

- Secret bits are **extracted** as follows:




**Stego Image (stego_logo.png)**

**Extracted secret message from LSBs**

# Embedding Procedure of Bits in an Image

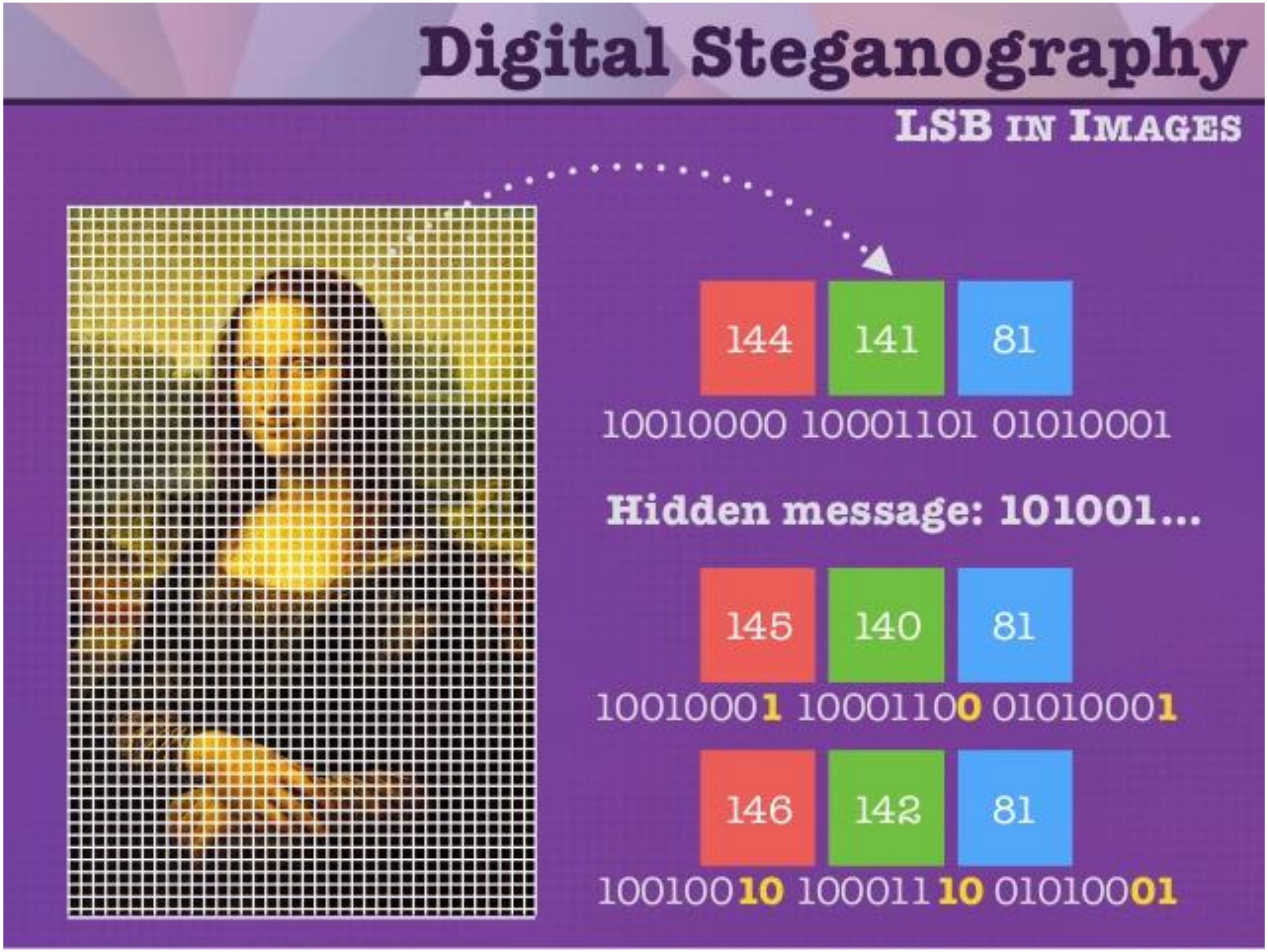
- Assume that Alice wants to hide a ***secret binary message (M = 101)*** in **an Image**
- Here, length of secret message is **3** *(i.e. length = 3) and cover data = {"logo.png"}*
- ***Embedding Procedure:***
  - Assume that Alice chooses a pixel in 2nd row and 7th column of the "logo.png", i.e. the coordinate of the pixel is (2,7). The RGB value of the pixel in (2,7) is (**210, 45, 10**).
  - Alice converts the pixel's RGB value into binary and gets: **11010010 00101101 00001010**, where there are 3 octets.
  - Alice hides the 101 as follows: 1st bit in LSB of 1st octet, 2nd bit in LSB of 2nd octet and 3rd bit in LSB of 3rd octet and produces stego image, S = {"stego_logo.png"}
  - Therefore, **Stego-Key ($S_K$)** becomes:

$$\boldsymbol{S_K = \langle length, locations \rangle = \langle 3,\{2,7\} \rangle}$$

# Extraction Procedure of Bits from an Image

- Bob receives $\mathbf{S_K}$ and **S** from Alice and wants to extract secret message from stego image (S) .
- Bob knows from $S_K$ that the length of secret message is **3**
- ***Extraction Procedure:***
  - Bob converts the pixel (2,7) RGB value (**211, 44, 11**) into binary and gets: **11010011 00101100 00001011**, where the 1st octet is RED, 2nd octet is GREEN and 3rd octet is BLUE.
  - Bob extracts the LSBs from the 3 octets as follows: LSB of 1st octet is **'1'**, LSB of 2nd octet is **'0'** and LSB of 3rd octet is **'1'**
  - Therefore, the secret message becomes **101**

# Image Steganography: Hiding in Monalisa



# Steganography Example-4

## Secret Data Hiding in Texts

## (Text Steganography)

# Text Stego Example: Simple Hiding in HTML

- Text_Cover.html in web browser

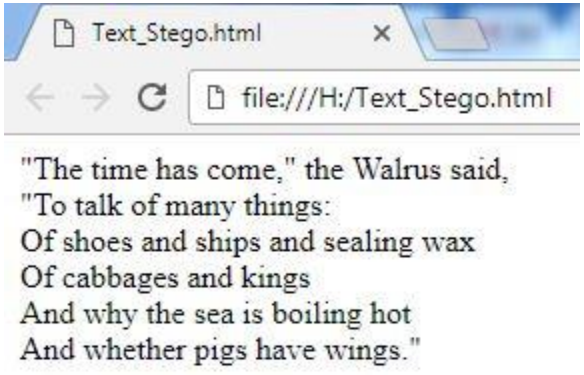

- “View source” reveals:

```
<font color=#000000>"The time has come," the Walrus said,</font><br>
<font color=#000000>"To talk of many things: </font><br>
<font color=#000000>Of shoes and ships and sealing wax </font><br>
<font color=#000000>Of cabbages and kings </font><br>
<font color=#000000>And why the sea is boiling hot </font><br>
<font color=#000000>And whether pigs have wings." </font><br>
```

# Embedding Procedure of Bits Hiding in HTML (1)

- Assume that Alice wants to hide a ***secret binary message (M)*** in **a HTML file** as ***font-color value***. Let, ***M = 010000010100001001000011***
- Here, length of secret message is **24** *(i.e. length = 24) and cover data = {"Text_cover.html"}*
- ***Embedding Procedure:***
  - Alice fragments 24 bits secret message into 4 segments of binary strings. Each segment has **6 bits** (as ***font-color*** takes **6 digit hexadecimal value**).
  - The segments are: ***010000, 010100, 001001, 000011***
  - Alice sets message segments as the font-color of 1st, 2nd , 5th and 6th lines as follows and the HTML file in the right (Text_stego.html) is obtained:

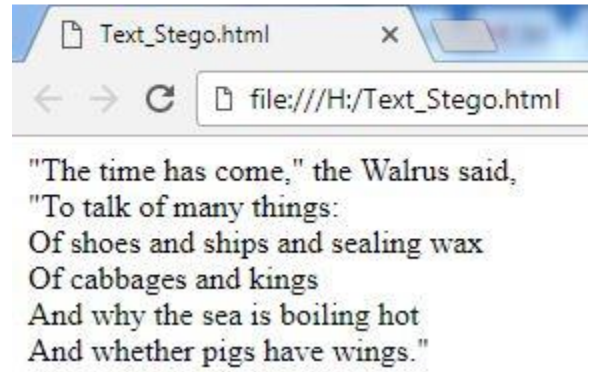
```
<font color=#010000>"The time has come," the Walrus said,</font><br>
<font color=#010100>"To talk of many things: </font><br>
<font color=#000000>Of shoes and ships and sealing wax </font><br>
<font color=#000000>Of cabbages and kings </font><br>
<font color=#001001>And why the sea is boiling hot </font><br>
<font color=#000011>And whether pigs have wings." </font><br>
```




  - Therefore, **Stego-Key ($S_K$)** becomes: ***$S_K$ = <lines> = <{1,2,5,6}>***

# Embedding Procedure of Bits Hiding in HTML (2)

- **Cover vs Stego Files:**


- **Pleas note that there is no visual difference between texts in Cover and Stego HTML files.**
- **If the line numbers are unknown then retrieving the message is challenging.**

# Extraction Procedure of Bits from HTML

- ***Extraction Procedure:***
  - Alice retrieves the colour codes from the lines of source files of Text_Stego.html as per given in $S_K$.
  - The retrieved segments are: ***010000, 010100, 001001, 000011***
  - The secret message is obtained as follows:

$$M_X = 010000010100001001000011$$

# Conclusion: Some Important Facts about Hiding Principles

- Some formats (e.g., image files) are more difficult than html for **humans** to read
  - But easy for computer programs to read…
- Easy to hide info in **unimportant bits**
- Easy to damage info in unimportant bits
- To be ***robust***, must use **important bits**
  - But stored info must not damage data
  - Collusion attacks are also a concern
- Robust steganography is tricky!