ECE 364
Software Engineering Tools Laboratory

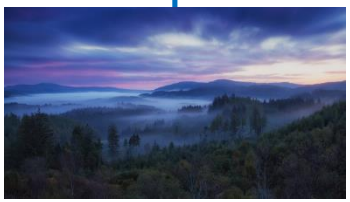
Project Description

Steganography

Motivation

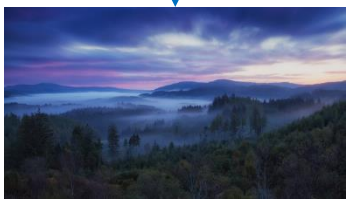
Embedding

Extraction



Payload

Carrier



Extracted Payload

For the required section of the project, we will only work with color images

Setup Requirements

You need to update your account with 3 modules:

- `numpy (1.12.1)` `<= Most Important`
- `scipy (0.19.0)`
- `Pillow (4.0.0)`

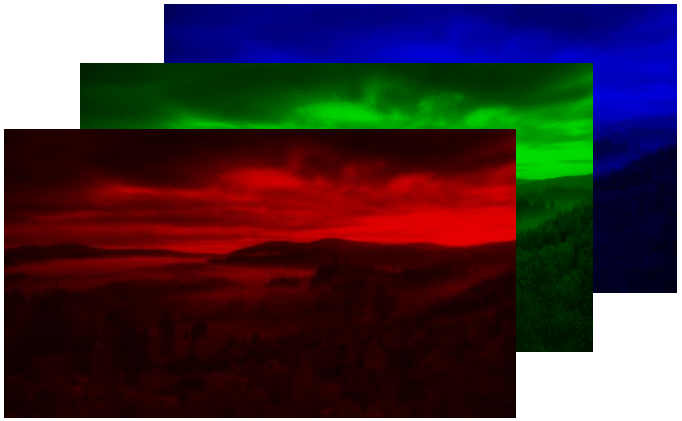
- You must update these libraries.
  - Your code must run with these versions, or it will NOT be graded!

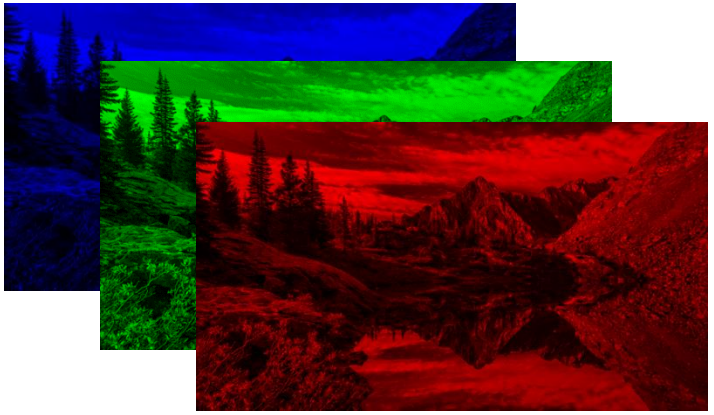
- You cannot use any other library.

- You will primarily need **numpy**
  - **There are numerous examples on the web, e.g.:**

http://www.kdnuggets.com/2017/03/working-numpy-matrices.html

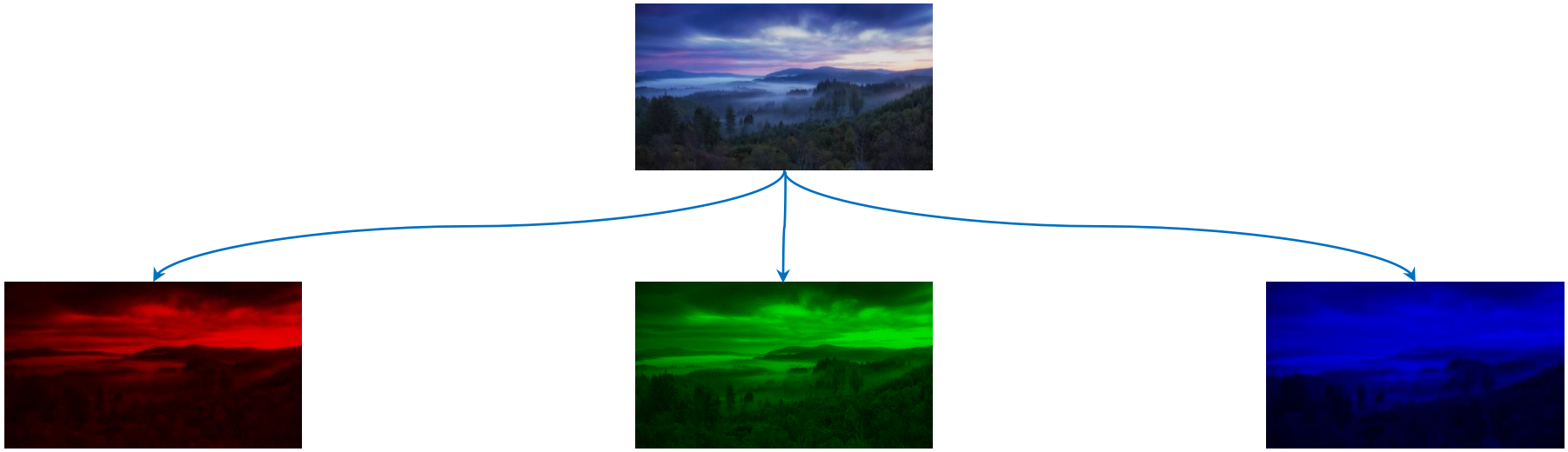
About Images





- Color images consist of 3 Bands/Channels: Red, Green & Blue values for every Pixel.
- Each channel value is one byte, i.e. each pixel is comprised of 3 independent bytes.

Preparing the Payload



$R_1\ R_2$
$R_3\ R_n$

$G_1\ G_2$
$G_3\ G_n$

$B_1\ B_2$
$B_3\ B_n$

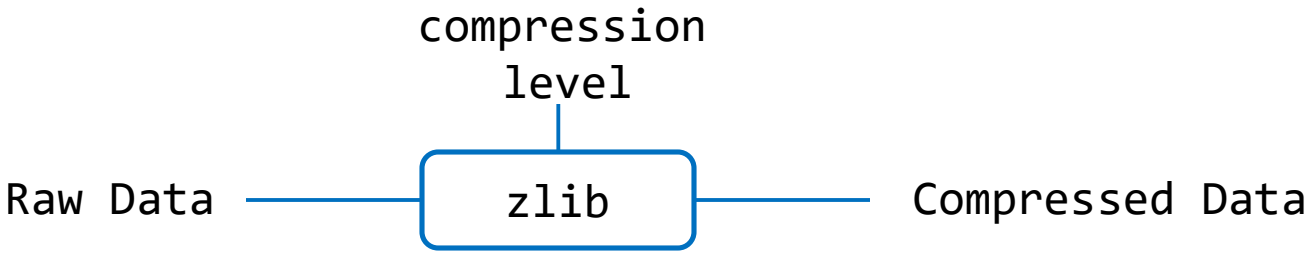
Step 1: Raster Scan & Concatenate:

$$\text{Raw Data} = [\ R_1\ R_2\ R_3\ R_n\ \ G_1\ G_2\ G_3\ G_n\ \ B_1\ B_2\ B_3\ B_n\ ]$$

Note: Each value in this array is one byte, i.e. in [0 – 255]

Preparing the Payload (2)

Step 2: Compress



Compressed Data = [Z_1 Z_2 Z_3 … Z_n]

Step 3: Pack as XML:

type = "Color" or "Gray"
size="rows,columns"
compressed="True" or "False"

Example:

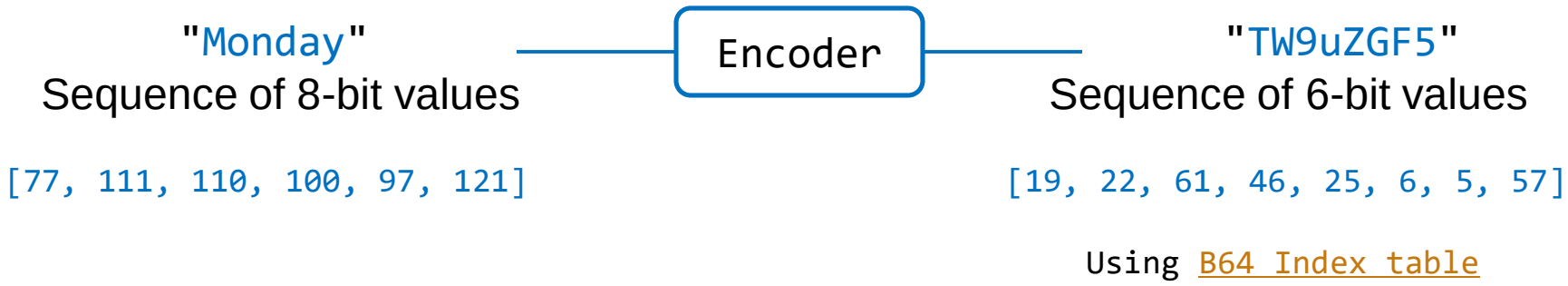
```
<?xml version="1.0" encoding="UTF-8"?>
<payload type="Color" size="360,640" compressed="True">
218,77,155,9,212,77,229,26,199,77,155,9,212,77,229,26,...
</payload>
```

Note: The XML string must NOT contain the "\n" character:

```
<?xml version="1.0" encoding="UTF-8"?><payload type="Color" size="360,640" compressed="True">218,77,155,...</payload>
```

Preparing the Payload (3)

Step 4: Convert XML String to Base64:



Note:

- Python has the **`base64`** Module.
- Use **`UTF-8`** for encoding/decoding.
- You will need to handle the padding.
- You cannot use any external Python Modules. You can, however, write your own implementation.

Embedding in the Carrier

Base64 Value to Embed:
30
01 11 10

B = 47

0 0 1 0 1 1 1 1

0 0 1 0 1 1 **0 1**

B = **45**

G = 112

0 1 1 1 0 0 0 0

0 1 1 1 0 0 **1 1**

G = **115**

R = 85

0 1 0 1 0 1 0 1

0 1 0 1 0 1 **1 0**

R = **86**

Project Block Diagram

Scanning

Compression

XML Serialization

Base64 Encoding

Embedding

Extraction

Base64 Decoding

XML De-Serialization

Decompression

Generation

Administrative Details

- This Phase is due on: **Sunday, Apr 9th @ 11:59 PM**.
  - You will be given Unit Tests and data files.
  - These tests will help identify any overlooked requirements. They also help identify possible bugs.
  - We will use these tests for grading, but against different data.
- Starting next week, lab attendance is optional.
  - Lab sessions will be open office hours for everyone.
  - Current Office Hours schedule will change.
- Phase II: (Heads up)
  - Due on: Sunday, Apr 23rd @ 11:59 PM (just before dead week.)
  - Attendance of your lab in dead week is **Mandatory**.
  - You will need to demo your work.

Extra Credit

- Grayscale Images:
  - Payload: This is straightforward.
  - Carrier: Each three consecutive pixels will hold one value.
  - XML: Set the type to “Gray”.

- Performance:
  - Vectorization: Your implementation must contain no loops (for-loop, while-loop ... etc.) whatsoever!
  - Execution Time: needs to be comparable to provided values.

- Variable Starting Point and Encoding Step:
  - You must first complete the “Performance” Requirement.
  - Embedded payload can be anywhere.
  - Encoding is done every $\mathcal{N}$ pixel, where $\mathcal{N}$ is unknown.
  - Execution time must be comparable to provided values.

Important: *Help will NOT be provided for this section.*