



PhotoEncryptor

By Marcos Barbieri

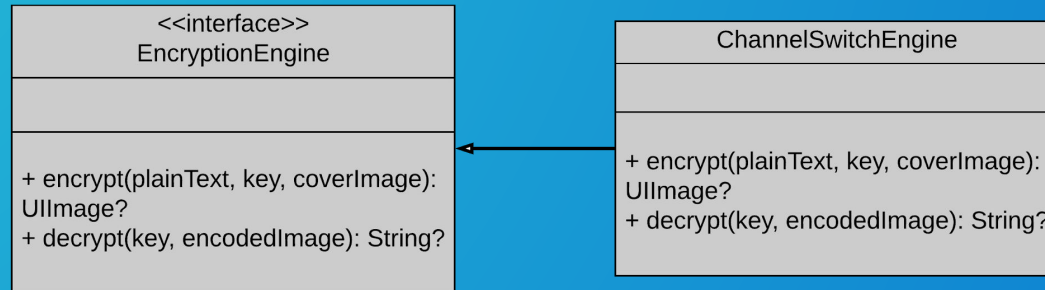
Project Background

- Platform: **iOS**
- Language: **Swift**
- Implementation:
 - ~Image Steganography
 - ~Least Significant Bit (LSB)

Channel Switching Encryption

- Red Green Blue Alpha (RGBA)
 - #00FF06 FF
- Switching channels creates diffusion
- Which channel?
 - $\{ B_i \mid i = c_j \% k \}$
 - B represents the set of channels (4)
 - c represents the character at position j
 - k represents the total number of channels

Implementation



Implementation (continued)

- Memory Access High Level Languages
 - Swift doesn't want you to access the individual pixels

UnsafeMutableRawPointer
?

```
guard let context = CGContext(data: nil,
                               width: width,
                               height: height,
                               bitsPerComponent: bitsPerComponent,
                               bytesPerRow: bytesPerRow,
                               space: colorSpace,
                               bitmapInfo: bitmapInfo) else {
    print("[ERROR] Unable to create context")
    return nil
}
context.draw(inputCGImage, in: CGRect(x: 0, y: 0, width: width, height: height))
guard let buffer = context.data else {
    print("[ERROR] Unable to extract buffer.")
    return nil
}
let pixelBuffer = buffer.bindMemory(to: RGBA32.self, capacity: width * height)
```

Things to Consider

- High level languages sometimes abstract too much
- Be mindful of the type of image that you are encrypting
- Be aware of what encoding you are using

The background is a solid blue color with a pattern of small, white, line-art icons scattered across it. These icons represent various office and technology items, including papers, folders, pens, pencils, erasers, rulers, protractors, scissors, staplers, paper clips, mobile phones, tablets, and computer mice. The icons are distributed evenly across the entire background.

DEMO