**CyberBlitz 2024 Challenges (Anurag)**

**a. Challenge Name**

Da Vinci’s Hidden Palette

**b. Challenge Description**

“Leonardo da Vinci was not just a master painter but also a master of hidden messages. This challenge involves uncovering a secret hidden within his famous works. Find the hidden image and look within its colours.”

**c. Hints, if any**

Optional hints:

Bitwise operation

LSB in R.G.B

**d. Steps documenting the techniques, procedures, and thought processes to solve your challenge.**

Two images are given – blended\_art.png and mona\_lisa.png

A collage of a person looking at a painting

Description automatically generated

From blended\_art.png it is obvious that it is merged with another image.

1. Perform Bitwise operations (XOR) with the given images (using python script) to retrieve the other image (Vitruvian man)

A close-up of a drawing

Description automatically generated

1. Based on description – “Find image and look within its colors”, the focus is shifted towards RGB values of the image. Using a python script retrieve Least Significant Bit from R,G,B values to form the hidden text. (LSB steganography is a common technique)

A screenshot of a computer program

Description automatically generated

The flag can be identified as

CyberBlitz{R3al\_3y35\_R34l1z3\_R34L\_Li35}

Python Scripts

1. Script to XOR two images

**from** PIL **import** Image  
**import** numpy **as** np  
  
**def** xor\_images(image\_path1, image\_path2, output\_path):  
 # Open the images  
 img1 = Image.open(image\_path1).convert('RGB')  
 img2 = Image.open(image\_path2).convert('RGB')  
  
 # Ensure both images are the same size  
 **if** img1.size != img2.size:  
 **raise** ValueError("Images must be the same size")  
  
 # Convert images to numpy arrays  
 arr1 = np.array(img1)  
 arr2 = np.array(img2)  
  
 # Perform XOR operation  
 xor\_result = np.bitwise\_xor(arr1, arr2)  
  
 # Convert the result back to an image  
 result\_image = Image.fromarray(xor\_result)  
  
 # Save the result image  
 result\_image.save(output\_path)  
  
# Usage  
xor\_images('mona\_lisa.png', 'blended\_art.png', 'result.png')

1. Script to retrieve hidden text from LSB in RGB values

**from** PIL **import** Image  
  
  
**def** extract\_text(image\_path):  
 img = Image.open(image\_path)  
 binary\_text = ''  
  
 width, height = img.size  
 pixels = img.load()  
  
 **for** y **in** range(height):  
 **for** x **in** range(width):  
 r, g, b = pixels[x, y]  
 binary\_text += str(r & 1) # Extract least significant bit from R  
 binary\_text += str(g & 1) # Extract least significant bit from G  
 binary\_text += str(b & 1) # Extract least significant bit from B  
  
 binary\_chars = [binary\_text[i:i + 8] **for** i **in** range(0, len(binary\_text), 8)] # Split into 8-bit chunks  
 text = ''  
 **for** binary\_char **in** binary\_chars:  
 char = chr(int(binary\_char, 2)) # Convert binary to character  
 text += char  
 **if** char == '}':  
 **break** # Stop at '}'  
  
 **return** text # Return the extracted text  
  
  
# Usage  
print("extracting LSB bits from RBG to form: ")  
hidden\_text = extract\_text('result.png')  
print(hidden\_text)