## Solution

First, you need to get the binary out so you can read what is at the end just as in the real cicada first challenge in 2012. You will find yourself with a deceptive wall. that says “you thougth it would be that easy this time? Try again or LEAVE if you are not ready scriptkido. 41 45 53 2D 43 54 52“ the numeric code 41 45 53 2D 43 54 52 is hexadecimal for “aes-ctr” and the word LEAVE is the password to decrypt the pixelated original image. The main problem here is to screenshot the proper pixels to decipher the picture but with a simple tool you can do it, the numbers on the phone are the exact pixels needed to cut the picture(I haven’t edited the phone to add this numbers)

## Technical solution

The method used to encrypt the image is based on the AES algorithm in **CTR mode** (Counter Mode). This mode offers several advantages relevant to this challenge:

* **Reversible XOR**: Both encryption and decryption use the XOR operation with a pseudo-random stream generated by AES-CTR.
* **Position Tolerance**: CTR mode does not rely on the order of data blocks, meaning that even if the image is shifted, cropped, or modified, it can still be decrypted correctly.
* **Visual Protection**: The result looks like random noise, preventing any visual inference about the content of the image.

**Solution Steps**

1. **Reading the Encrypted Image**  
   The new encrypted image is loaded using PIL and converted to RGB format to ensure we have access to the color pixel values.
2. **Key Generation**  
   The AES key is derived from the string "LEAVE" using SHA-256 hashing, then truncated to 16 bytes (128 bits), as this is the expected key size for AES in this case.

aes\_key = hashlib.sha256("LEAVE".encode()).digest()[:16]

1. **AES Configuration in CTR Mode**  
   We use the AES algorithm in CTR mode (Counter Mode). This mode generates a pseudo-random stream using a counter and the AES key:
   * The counter is initialized with a size of 128 bits.
   * The AES-CTR cipher is created with this key and the counter

counter = Counter.new(128)

cipher = AES.new(aes\_key, AES.MODE\_CTR, counter=counter)

1. **Decryption with XOR**  
   To decrypt the image, we perform an XOR operation between each byte of the encrypted image data and the pseudo-random stream generated by AES-CTR. The XOR operation is reversible: applying XOR a second time with the same stream restores the original data.

decrypted\_data = cipher.decrypt(bytes(flat\_encrypted\_data))

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement