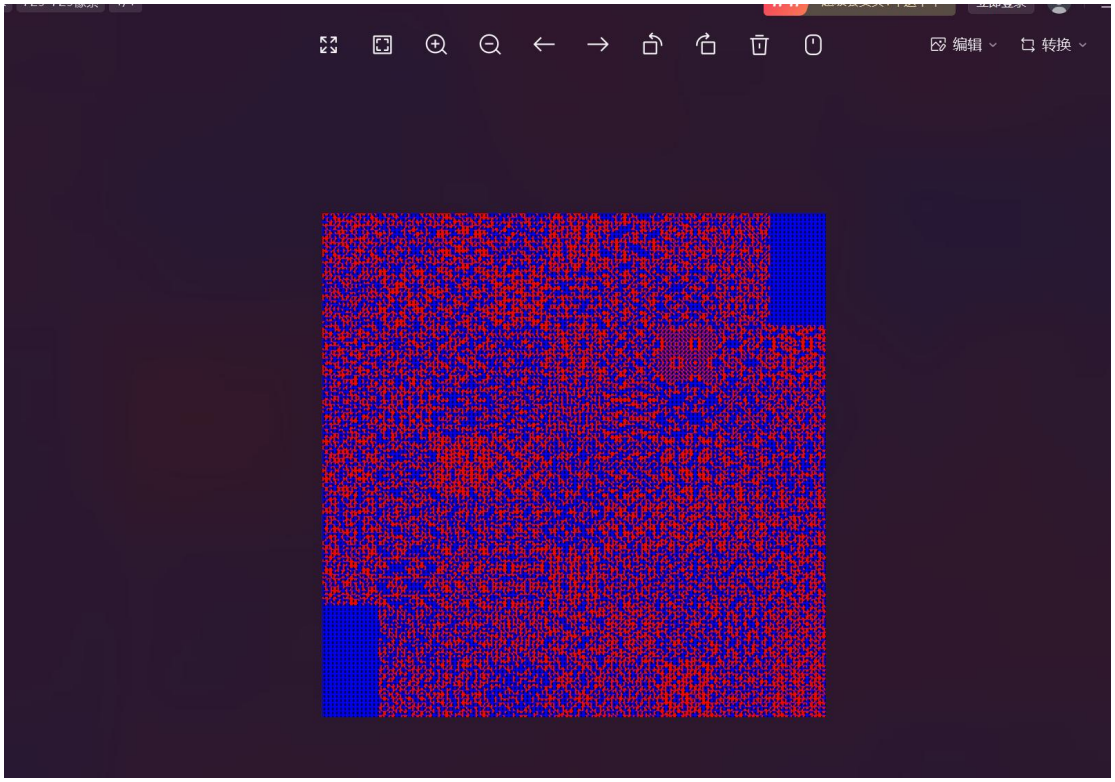
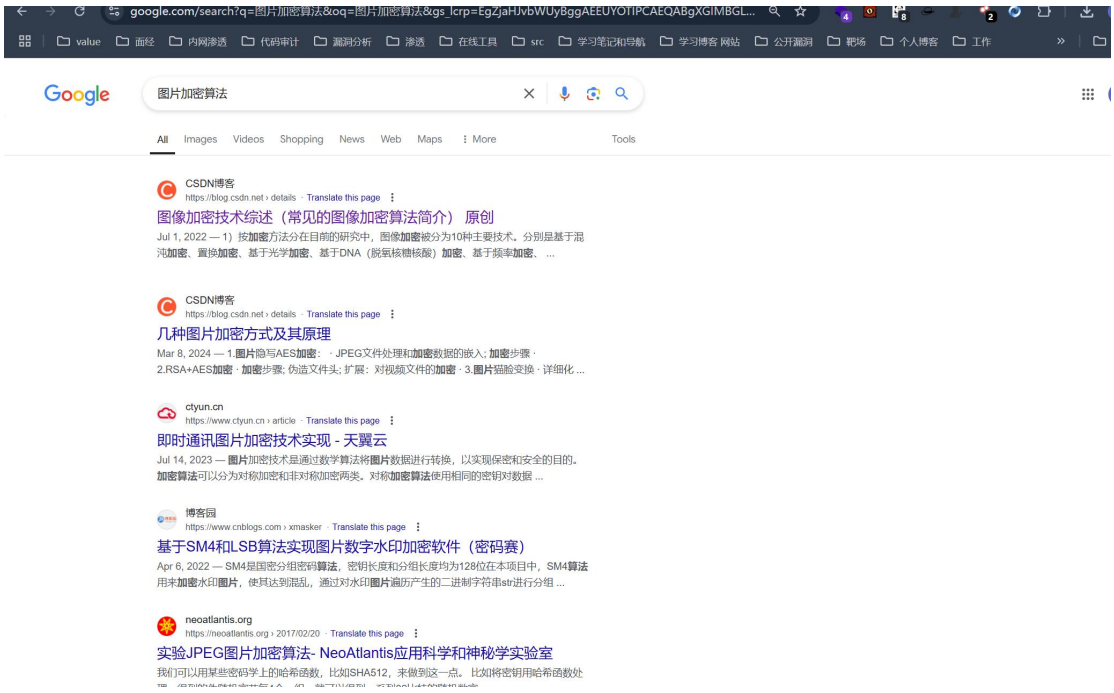
赛题名称：MISC04

解题步骤（WriteUp）

第一步：下载附件

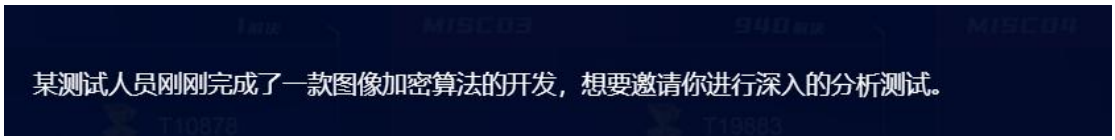


第二步： 图片加密算法，我第一想法就去搜索图片加密算法有哪些

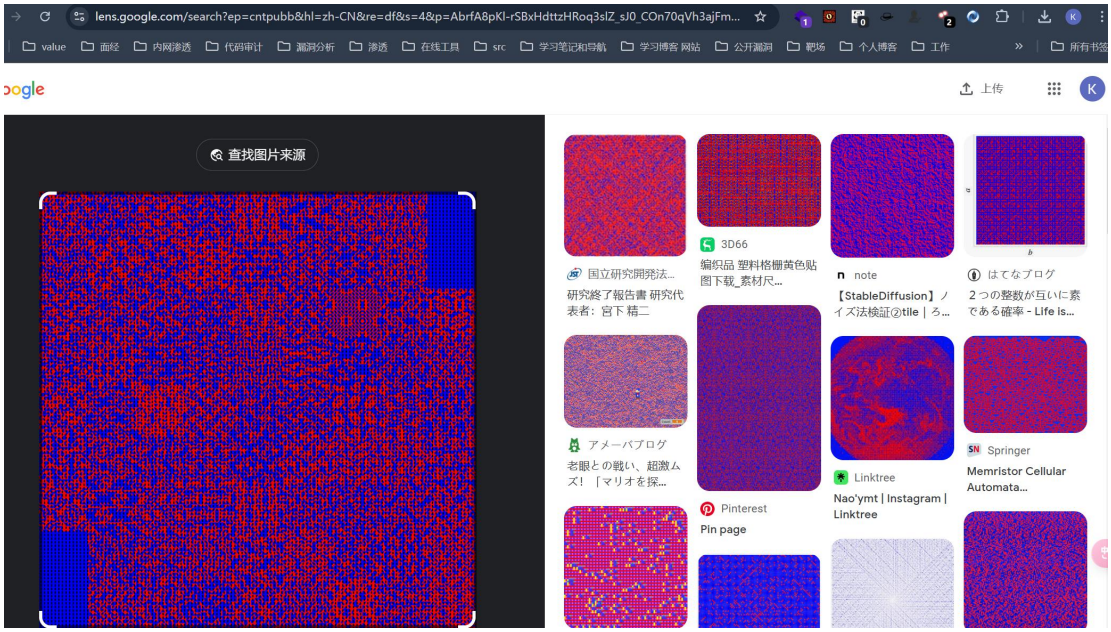


但是找不到对应的一些图片，不能发现这是什么加密算法。主要是感觉没有什么图片能够跟他对应起来。

第三步：按照常规的思路，我先是使用了 zsteg Binwalk foremost 分析和分离文件发现都没有什么用。但是观察 010 和一些隐写爆破都没有思路。查看属性发现也没有提示，之后又想到这是个



图像加密算法，感觉肯定跟我们平常做的肯定是不一样。只能去识图看看有没有类似的图像我们可以去查查

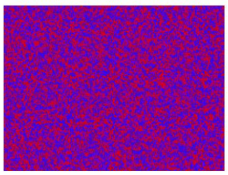


value 面经 内网渗透 代码审计 漏洞分析 渗透 在线工具 src 学习笔记和导航 学习博客 网站 公开漏洞 靶场 个人博客 工作

O-R-G 奥格

Breaking Like Surf on a Shore Until

就像海岸上的冲浪一样，直到



macOS v10.12+ Screensaver

macOS v10.12+ 屏幕保护程序

Designed by David Reinfurt

由大卫·赖因福特设计

While living with William Burroughs at the Beat hotel on Gît-le-Coeur, Paris, and with the aid of Cambridge mathematician Ian Somerville, poet Brion Gysin designed a simple paper cylinder with periodic apertures surrounding a bare light bulb placed on a rotating turntable at 78 rpm.

与威廉·巴勒斯 (William Burroughs) 住在巴黎吉特勒科尔 (Gît-le-Coeur) 的比特酒店 (Beat hotel) 时，在剑桥数学家伊恩·萨默维尔 (Ian Somerville) 的帮助下，诗人布里昂·吉辛 (Brion Gysin) 设计了一个简单的纸圆筒，圆筒上有周期性的孔，围绕着一个放置在旋转转盘上的裸露灯泡。 78 转/分。

The resulting flickering light repeats at a constant frequency between 8 and 13 Hz matching the brain's alpha-waves present in deep relaxation, such as drifting into sleep. When a viewer closes their eyes, sits close to the machine, and the turntable is started, the flickering light induces waves of color and repeating geometric patterns that form and re-form in the mind's eye. Ian Somerville described the experience in a letter to Gysin:

由此产生的闪烁光以 8 至 13 Hz 之间的恒定频率重复，与深度放松（例如进入睡眠）时大脑的 α 波相匹配。当观看者闭上眼睛，坐在靠近机器的地方，转盘启动时，闪烁的光线会引发色彩波动和重复的几何图案，这些图案在脑海中形成并重新形成。伊恩·萨默维尔在给吉辛的一封信中描述了这段经历：

> Visions start with a kaleidoscope of colors on a plane in front of the eyes and gradually become more complex and beautiful, breaking like surf on a shore until whole patterns of color are bounding to get in. After awhile the visions were permanently behind my eyelids and I was in the middle of the whole scene with limitless patterns being generated around me.
>
> 幻象从眼前平面上的万花筒色彩开始，逐渐变得更加复杂和美丽，就像海岸上的海浪一样破碎，直到整个颜色图案都跃入其中。过了一会儿，幻象就永久地出现在我的眼睑后面，我处于整个场景的中央，周围产生了无限的图案。

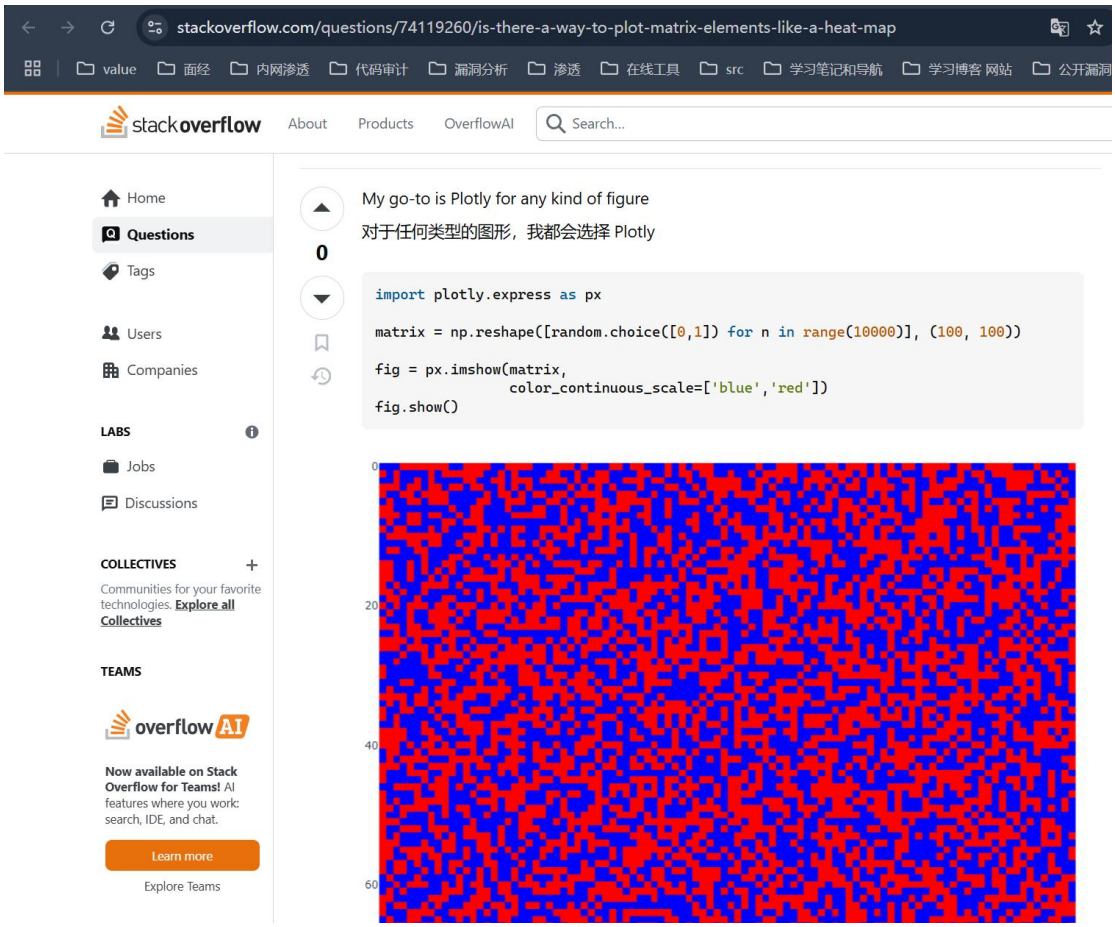
Unlike a pharmacological trip, th experience ends when the user opens their eyelids. Gysin called vision machine for closed eyes tl Dreamachine.

与药理之旅不同的是，当用户睁开时，体验就结束了。吉辛将他的闭觉机器称为“梦想机器”。

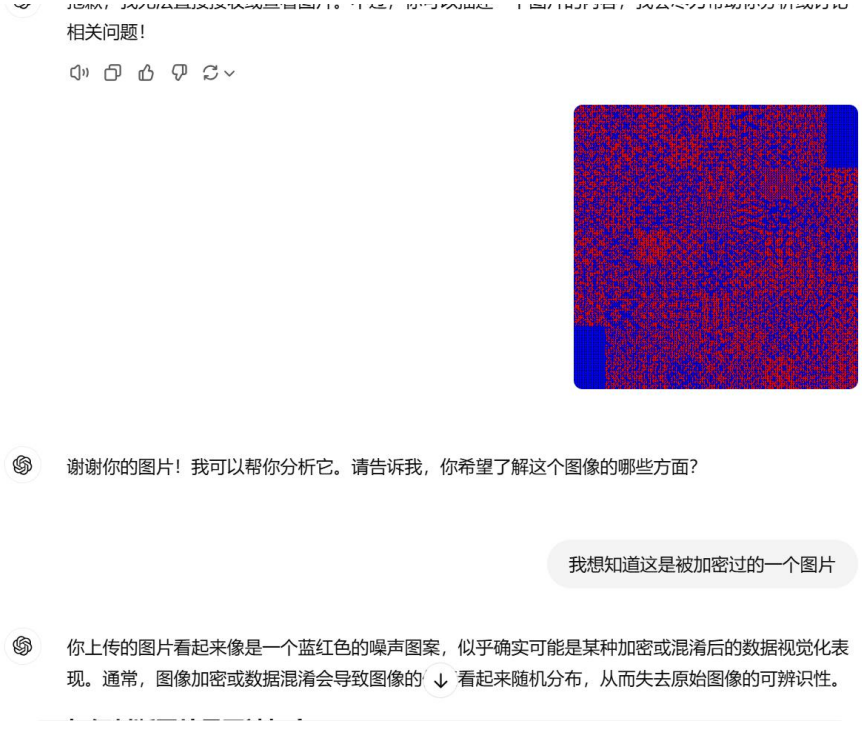
More . . . 更多的。。。

*

$25

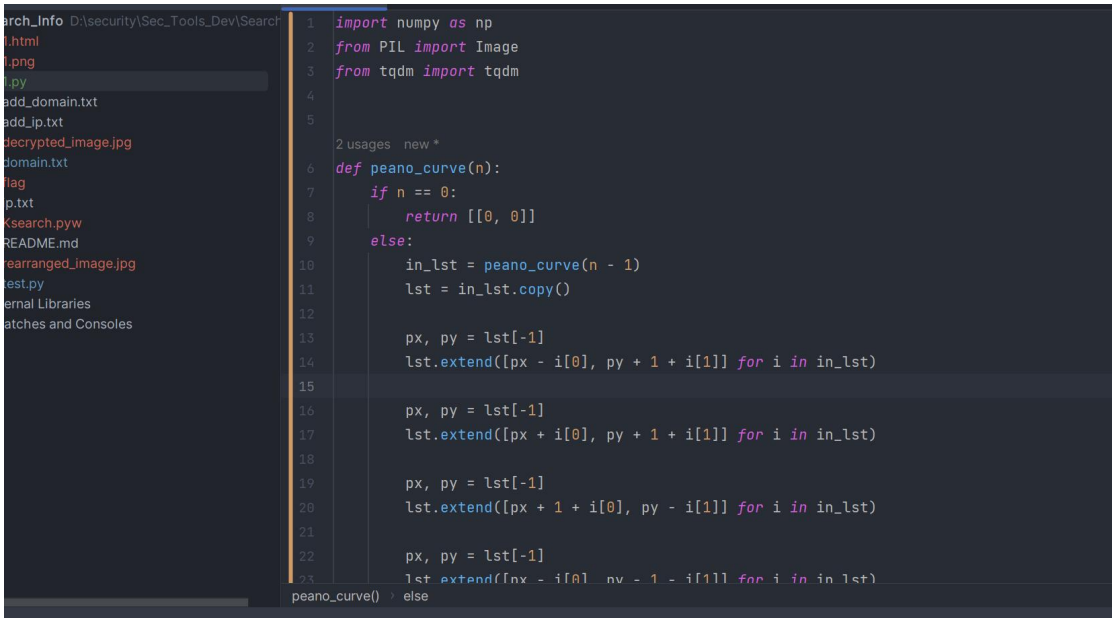


通过这些矩阵画出了一个红蓝红蓝的操作……但是没用.....

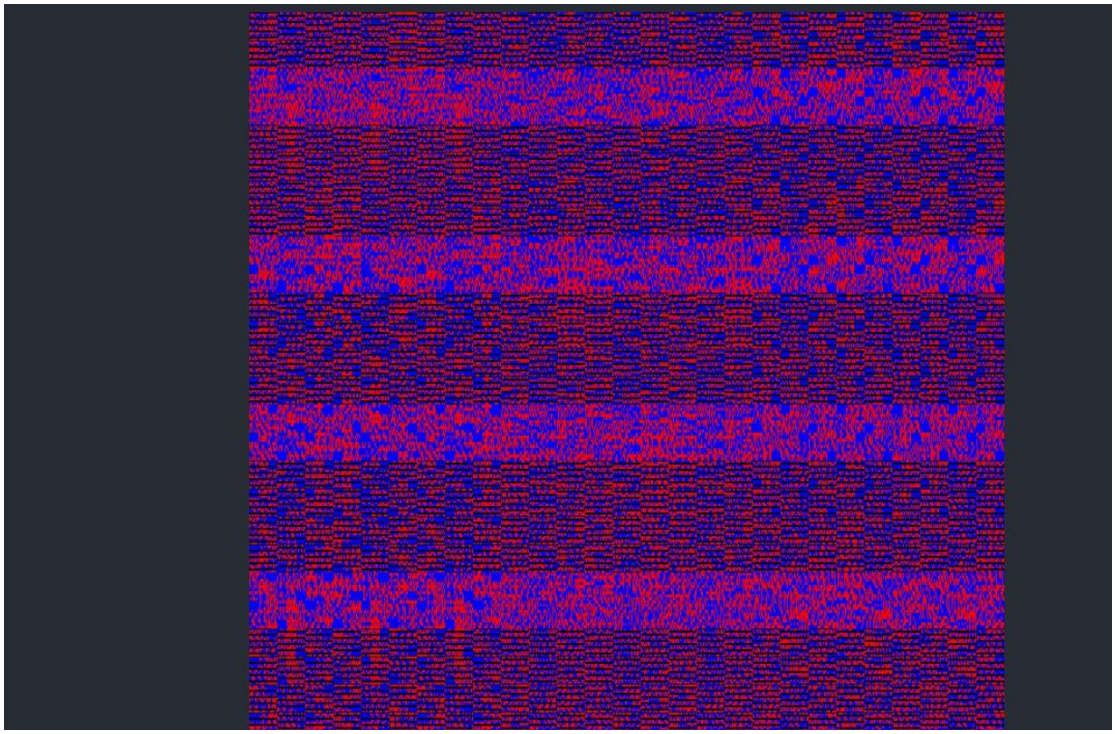


发现都是红蓝并且 跟二维码很对应，因此就可以猜测加密肯定就是像素被打乱顺序，我们

尝试一些算法打乱顺序，找到了很多算法尝试。发现可以使用佩亚诺曲线,之前还以为是这个像素被加密好了，需要逆佩亚诺 。



第 N 步：但是一直都是乱码



但是得不到，之后使用正向。Gpt 生成

```
from PIL import Image
from tqdm import tqdm

# 定义生成佩亚诺曲线的函数
```

```
def generate_peano_curve(n):
    if n == 0:
        return [(0, 0)]
    else:
        prev_curve = generate_peano_curve(n - 1)
        curve = prev_curve.copy()

        px, py = curve[-1]
        curve.extend([(px - x, py + 1 + y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px + x, py + 1 + y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px + 1 + x, py - y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px - x, py - 1 - y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px + x, py - 1 - y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px + 1 + x, py + y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px - x, py + 1 + y) for x, y in prev_curve])

        px, py = curve[-1]
        curve.extend([(px + x, py + 1 + y) for x, y in prev_curve])

        return curve

# 解密图片函数
def decrypt_image_using_peano_curve(input_image_path, output_image_path, order):
    # 打开图像并获取其尺寸
    img = Image.open(input_image_path)
    width, height = img.size

    # 生成指定阶数的佩亚诺曲线
    peano_curve = generate_peano_curve(order)

    # 验证曲线点数量是否匹配图像像素数量
    if len(peano_curve) != width * height:
```

```
        raise ValueError("佩亚诺曲线的点数与图像尺寸不匹配，请调整阶数。")

    # 创建空白图像来存储解密结果
    decrypted_image = Image.new("RGB", (width, height))

    # 使用佩亚诺曲线顺序重新排列像素
    for i, (x, y) in tqdm(enumerate(peano_curve), total=len(peano_curve), desc="解密中"):
        new_x, new_y = i % width, i // width
        pixel = img.getpixel((x, height - 1 - y))
        decrypted_image.putpixel((new_x, new_y), pixel)

    # 保存解密后的图像
    decrypted_image.save(output_image_path)
    print(f"解密后的图像已保存为 {output_image_path}")

# 使用示例
decrypt_image_using_peano_curve("1.png", "decrypted_image.jpg", order=6)
```