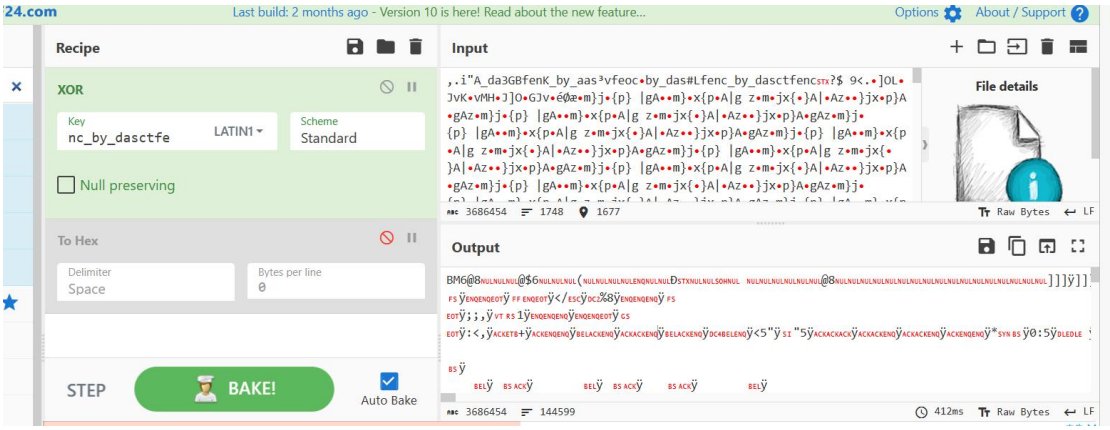
Cap：

将 cap.bin 文件和 nc_by_dasctfe 异或，得到一张图片

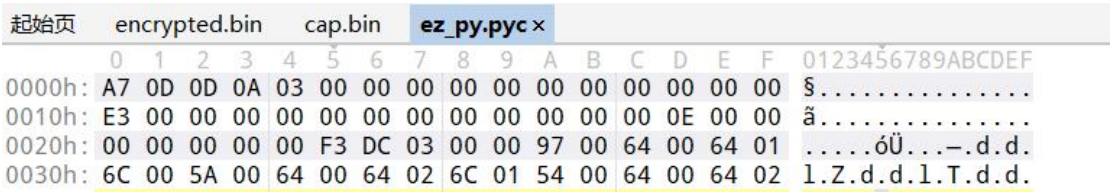


Ez_exe:

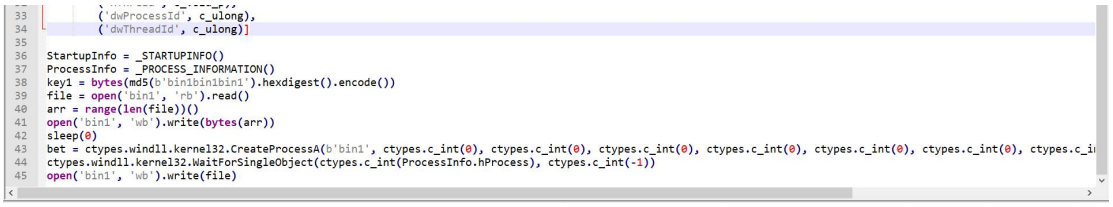
用 pyinstxtractor.py 解包

接着用 pycdc 将 ez_py.pyc 还原成 py 文件

一开始我的 pycdc 也转不了，后来按照 Gift1a 师傅的说法，改一下文件头



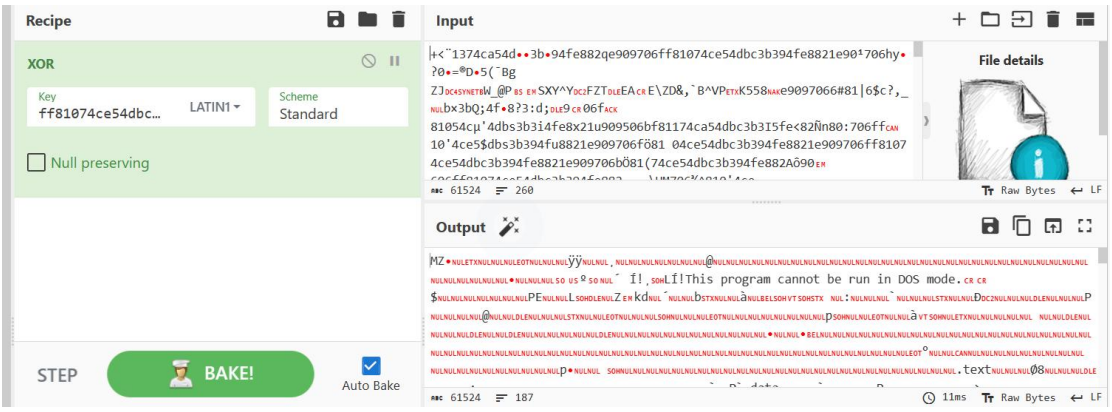
就可以还原了，得到如下代码



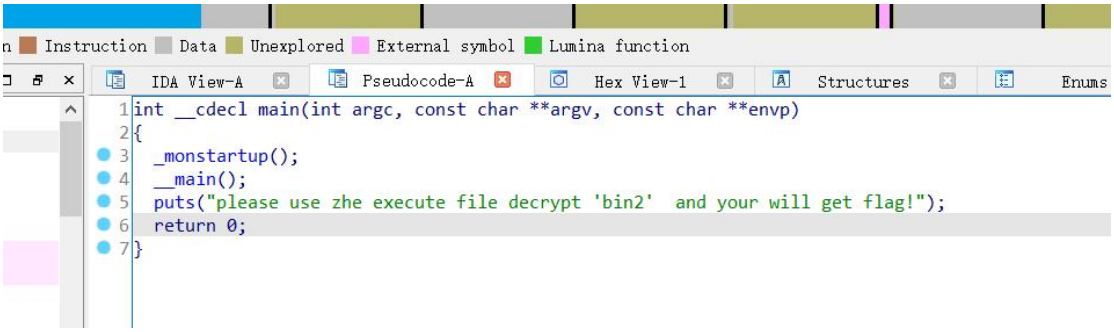
将 bin1bin1bin1 字符串 md5 加密得到 ff81074ce54dbc3b394fe8821e909706

字符串	bin1bin1bin1
16位 小写	e54dbc3b394fe882
16位 大写	E54DBC3B394FE882
32位 小写	ff81074ce54dbc3b394fe8821e909706
32位 大写	FF81074CE54DBC3B394FE8821E909706

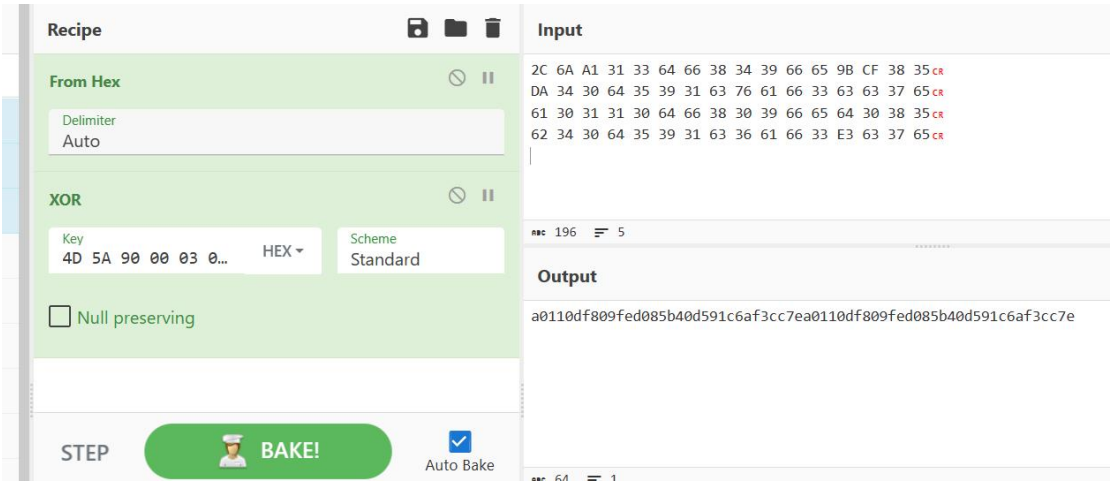
接着将 bin1 和 ff81074ce54dbc3b394fe8821e909706 异或一下
得到一个 exe 文件



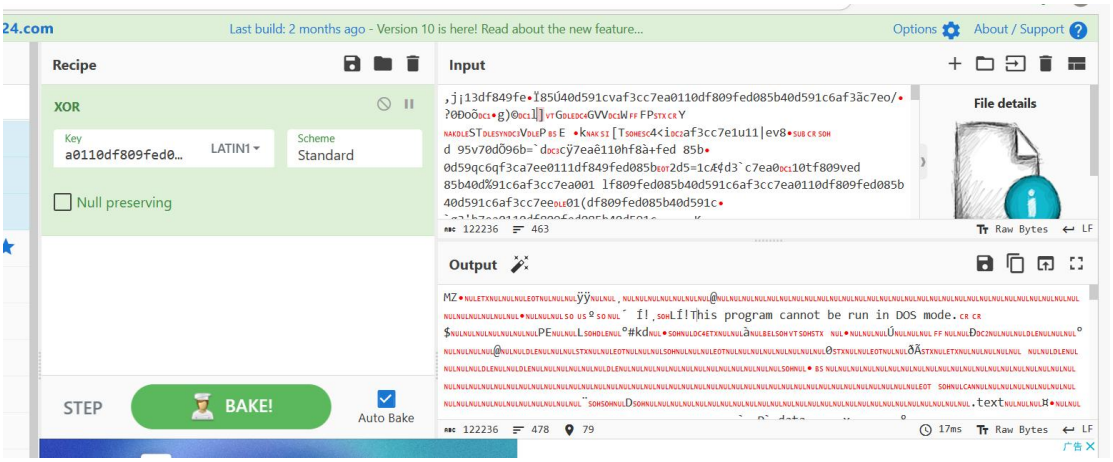
IDA 看一下，提示用得到的程序解密 bin2 文件



于是猜测 bin2 也是个 exe 文件，将 bin2 的文件头 64 个字节与 bin1 的文件头 64 个字节异或一下，得到 a0110df809fed085b40d591c6af3cc7ea0110df809fed085b40d591c6af3cc7e



最后用 bin2 文件和 a0110df809fed085b40d591c6af3cc7ea0110df809fed085b40d591c6af3cc7e 异或得到一个 exe 程序



IDA 看一下程序

```
    v12 += 4;
  }
  v6[0] = 19295;
  v6[1] = 57005;
  v6[2] = 4589;
  v6[3] = 46028;
  btea(v7, 11, v6);
  for ( j = 0; j <= 10; ++j )
    ;
  v5[0] = -867866211;
  v5[1] = 1748849490;
  v5[2] = -1195675232;
  v5[3] = 221779885;
  v5[4] = 1967646622;
  v5[5] = 1272810620;
  v5[6] = -1780327807;
  v5[7] = -1001238269;
  v5[8] = 1883878511;
  v5[9] = 1410856828;
  v5[10] = 1700163896;
  for ( k = 0; k <= 10; ++k )
  {
    if ( v5[k] != v7[k] )
    {
      MessageBoxA(0, "error!", &Caption, 0);
      exit(0);
    }
  }
  MessageBoxA(0, "right!", &Caption, 0);
  return 0;
}
```

跟进 btea

```
unsigned int v16; // [esp+1Ch] [ebp-Ch]
int v17; // [esp+34h] [ebp+Ch]

if ( a2 <= 1 )
{
  if ( a2 < -1 )
  {
    v17 = -a2;
    v10 = 52 / v17 + 6;
    v14 = 2033695134 * v10;
    v16 = *a1;
    do
    {
      v8 = (v14 >> 2) & 3;
      for ( i = v17 - 2; i; --i )
      {
        v6 = &a1[i];
        *v6 -= ((v16 ^ v14) + (a1[i - 1] ^ *(_DWORD *)(4 * (v8 ^ i & 3) + a3))) ^ (((4 * v16) ^ (a1[i - 1] >> 5))
                                                                                  + ((v16 >> 3) ^ (16 * a1[i - 1])));
        v16 = *v6;
      }
      *a1 -= (((4 * v16) ^ (a1[v17 - 1] >> 5)) + ((v16 >> 3) ^ (16 * a1[v17 - 1]))) ^ ((v16 ^ v14)
                                                                                      + (a1[v17 - 1] ^ *(_DWORD *)(4 * v8 + a3)));
      result = *a1;
      v16 = *a1;
      v14 -= 2033695134;
      --v10;
    }
    while ( v10 );
  }
}
else
00000839 btea:16 (401439)
```

发现是个 xxtea 算法，脚本直接解

```
#include <stdio.h>
#include <stdint.h>

#define DELTA 0x7937B99E
```

```
#define MX (((z>>5^y<<2) + (y>>3^z<<4)) ^ ((sum^y) + (key[(p&3)^e] ^ z)))

void btea(uint32_t *v, int n, uint32_t const key[4])
{
	uint32_t y, z, sum;
	unsigned p, rounds, e;
	if (n > 1)			/* Coding Part */
	{
		rounds = 52/n;
		sum = 0;
		z = v[n-1];
		do
		{
			sum += DELTA;
			e = (sum >> 2) & 3;
			for (p=0; p<n-1; p++)
			{
				y = v[p+1];
				z = v[p] += MX;
			}
			y = v[0];
			z = v[n-1] += MX;
		}
		while (--rounds);
		printf("\n%x\n",sum);
	}
	else if (n < -1)		/* Decoding Part */
	{
		n = -n;
		rounds = 52/n;
		sum = 0xe4dee678;
		y = v[0];
		do
		{
			e = (sum >> 2) & 3;
			for (p=n-1; p>0; p--)
			{
				z = v[p-1];
				y = v[p] -= MX;
			}
			z = v[n-1];
			y = v[0] -= MX;
			sum -= DELTA;
		}
```

```
        while (--rounds);
//          printf("\n%x\n",sum);
    }
}


int main()
{
    uint32_t v[11]= {0xCC45699D, 0x683D5352, 0xB8BB71A0, 0xD3817AD, 0x7547E79E, 0x4BDD8C7C, 0x95E25A81, 0xC4525103, 0x7049B46F, 0x5417F77C, 0x65567138};
    uint32_t const k[4]={0x4B5F,0xDEAD,0x11ED,0xB3CC};
    int i,n= 11; //n 的绝对值表示 v 的长度，取正表示加密，取负表示解密
    // v 为要加密的数据是两个 32 位无符号整数
    // k 为加密解密密钥，为 4 个 32 位无符号整数，即密钥长度为 128 位
    printf(" 解 密 前 原 始 数据：%x %x %x %x %x %x %x %x %x %x\n",v[0],v[1],v[2],v[3],v[4],v[5],v[6],v[7],v[8],v[9],v[10]);
    btea(v, -n, k);
    printf(" 解 密 后 的 数据：%x %x %x %x %x %x %x %x %x %x\n",v[0],v[1],v[2],v[3],v[4],v[5],v[6],v[7],v[8],v[9],v[10]);
    for(i=0;i<11;i++)
    {
        printf("%c",v[i]&0xff);
        printf("%c",(v[i]&0xff00)%0xff);
        printf("%c",(v[i]&0xff0000)%0xffff);

    printf("%c",(v[i]&0xff000000)%0xffffff);//DASCTF{7eb20cb2-deac-11ed-ae42-94085339ce84
}
    }
    btea(v, n, k);
    printf("\n 加 密 后 的 数据：%x %x %x %x %x %x %x %x %x %x\n",v[0],v[1],v[2],v[3],v[4],v[5],v[6],v[7],v[8],v[9],v[10]);
    system("pause");
    return 0;
}
```