Misc-文件操作

2021-09-27

在Misc的题目中，经常可以遇到文件中隐写着其它文件的情况，那么这个时候就要灵活运用多种方法和工具来分离数据，以拿到最终的flag。任何要求分析一个静态数据文件从而获取隐藏文件或者信息的都可以被认为是涉及了文件操作（除非单纯地是密码学的知识）。

# Misc-文件操作

## 前置技能

* 了解常见的编码，能够对文件中出现的一些编码进行解码，并且对一些特殊的编码（Base64、十六进制、二进制等）有一定的敏感度，对其进行转换并得到最终的 flag。
* 能够利用脚本语言（Python 等）去操作二进制数据
* 熟知常见文件的文件格式，尤其是各类 [文件头](https://en.wikipedia.org/wiki/List_of_file_signatures)、协议、结构等
* 灵活运用常见的工具

## 常用工具

### 010 Editor

010 Editor 是一款专业的文本编辑器和十六进制编辑器,其设计旨在轻松简便地快速编辑您计算机上任何文件的内容。在CTF中，010 Editor常常以其优秀的模板功能来帮助选手来高效分析文件结构和类型。

### Winhex

Winhex是X-Ways公司出品的一款十六进制编辑、磁盘编辑软件。其优点是文件小，速度快，但是对于文件结构复杂的文件，建议使用010 Editor。

## 常见文件头，文件尾

### 文件头、文件尾

**善用010 Editor的模板功能**

| 文件类型 | 后缀 | 文件头 | 文件尾 |
| --- | --- | --- | --- |
| JPEG | .jpg | FF D8 FF | FF D9 |
| PNG | .png | 89 50 4E 47 | AE 42 60 82 |
| GIF | .gif | 47 49 46 38 | 00 3B |
| TIFF | .tif | 49 49 2A 00 |  |
| Windows Bitmap | .bmp | 42 4D C0 01 |  |
| ZIP Archive | .zip | 50 4B 03 04 | 50 4B |
| RAR Archive | .rar | 52 61 72 71 |  |
| MS Word/Excel | .xls .doc | D0 CF 11 E0 |  |
| Adobe Acrobat | .pdf | 25 50 44 46 2D 31 2E |  |
| Wave | .wav | 57 41 56 45 |  |
| AVI | .avi | 41 56 49 20 |  |
| pcap | .pcap | 4D 3C 2B 1A |  |

参考：[文件头文件尾总结](https://www.cnblogs.com/lwy-kitty/p/3928317.html)，[List of file signatures](https://en.wikipedia.org/wiki/List_of_file_signatures)

### 工具

#### file 指令（linux）

file 命令根据文件头（魔法字节）去识别一个文件的文件类型。

randark@randark-VirtualBox:~/shared$ file -h  
Usage: file [-bcCdEhikLlNnprsSvzZ0] [--apple] [--extension] [--mime-encoding]  
 [--mime-type] [-e <testname>] [-F <separator>] [-f <namefile>]  
 [-m <magicfiles>] [-P <parameter=value>] <file> ...  
 file -C [-m <magicfiles>]  
 file [--help]

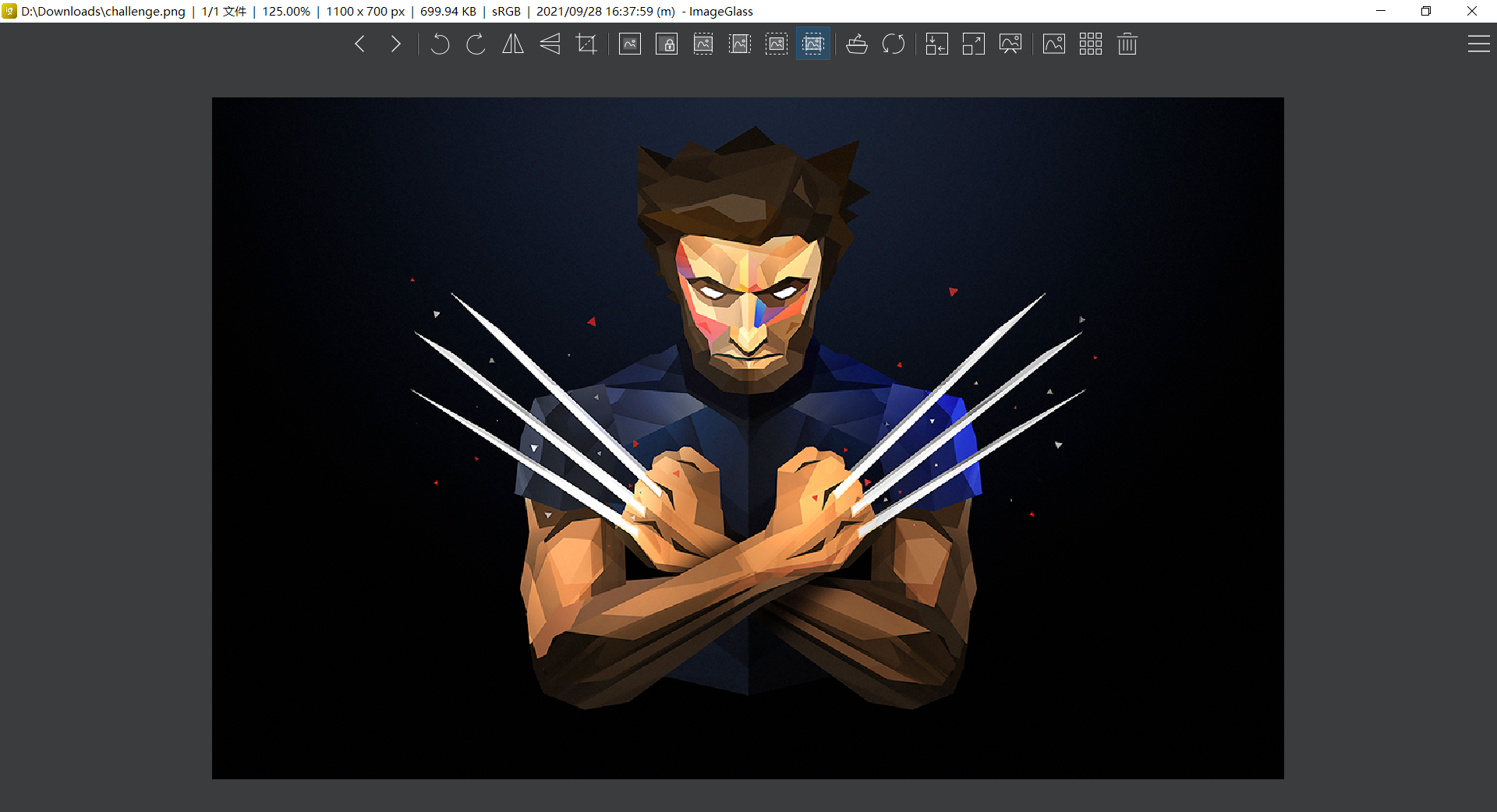
例如：

randark@randark-VirtualBox:~/shared$ file bronya.png  
bronya.png: PNG image data, 2457 x 2457, 8-bit/color RGB, non-interlaced

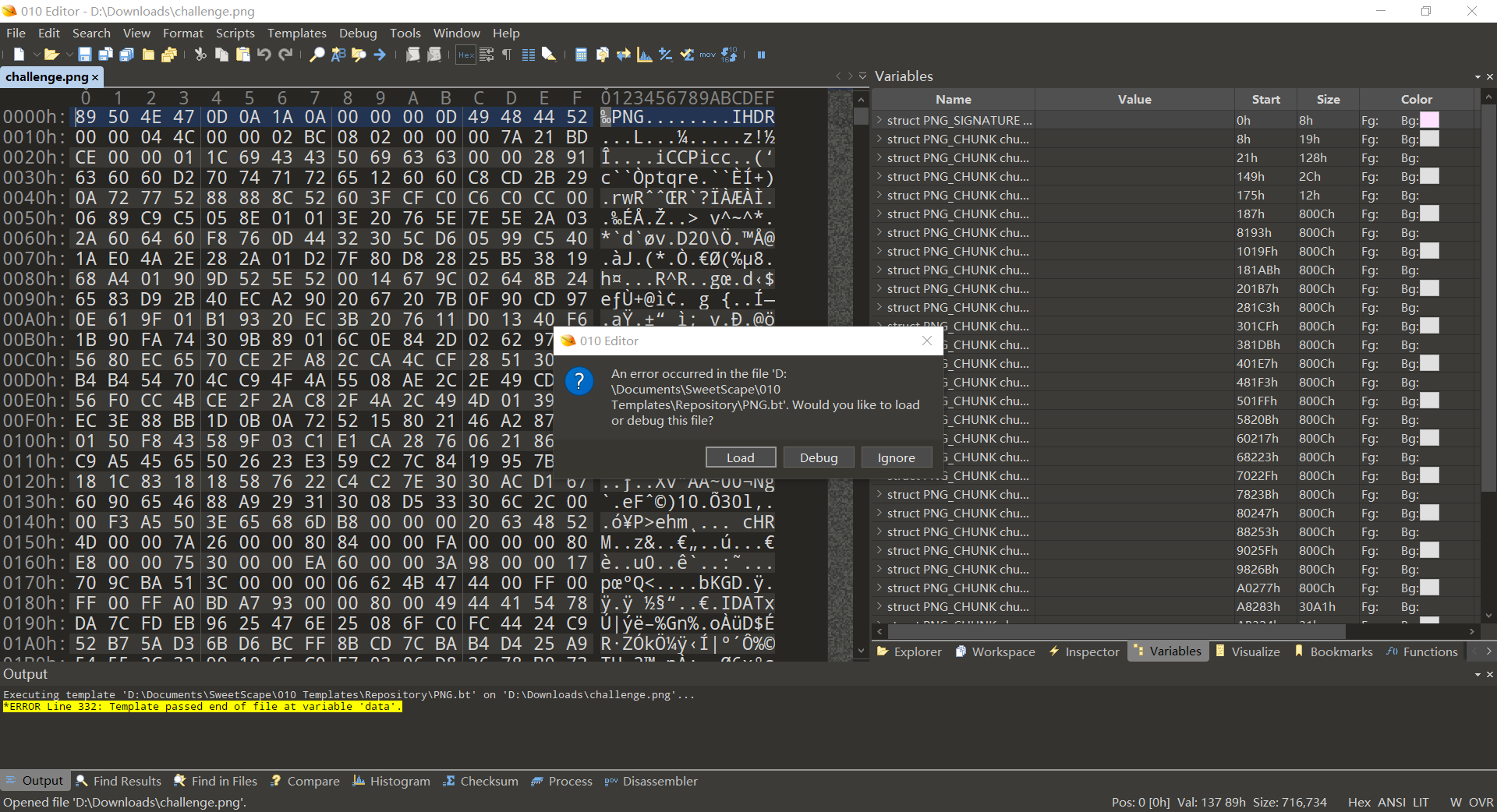
### 例题

[BUUCTF-Business Planning Group](https://buuoj.cn/challenges#Business%20Planning%20Group)

附件是一个png文件，可以正常打开来：



用010 Editor打开文件，就提示文件存在异常数据：

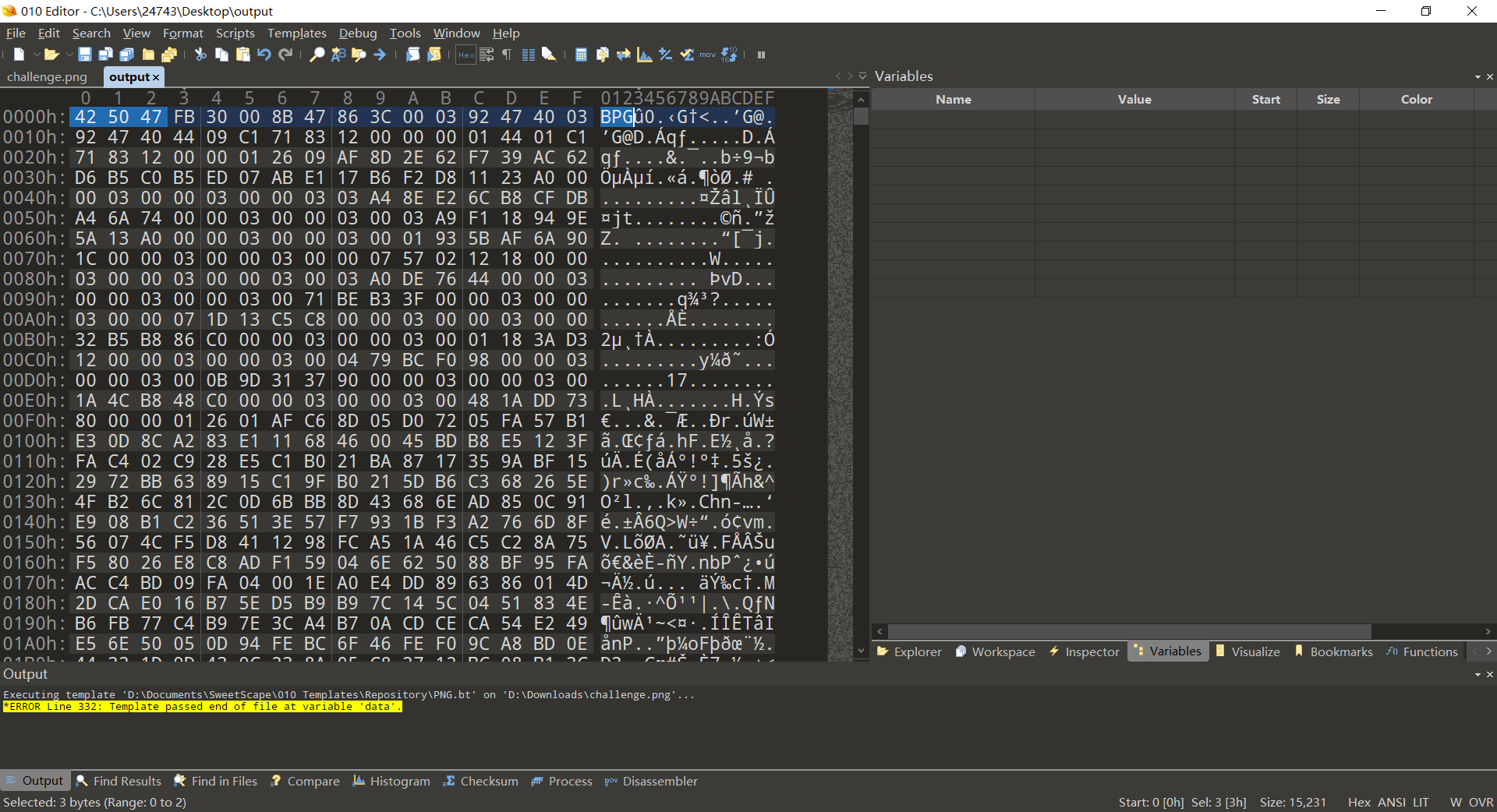


~~这就是模板功能的强大之处~~

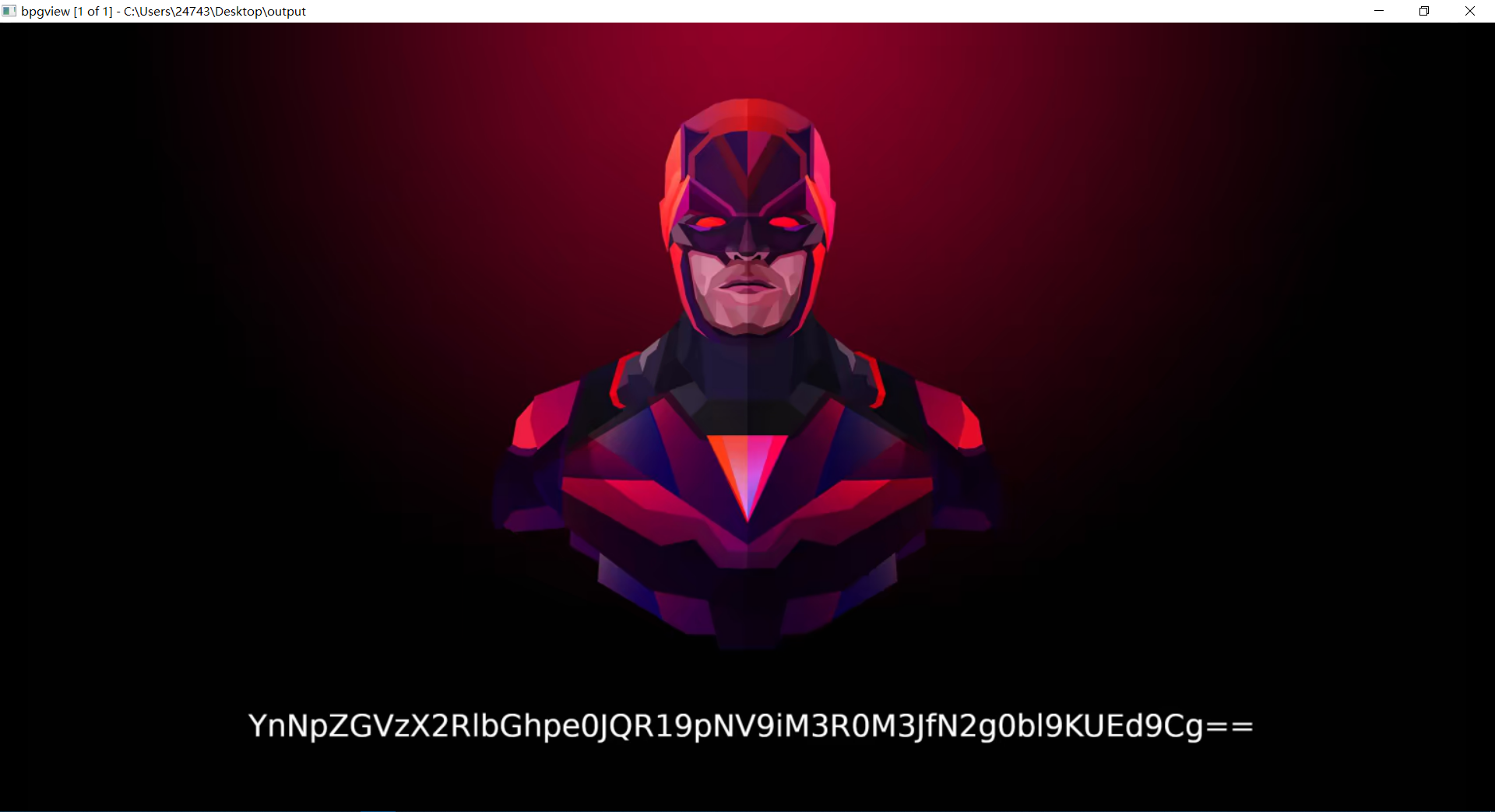
定位到异常数据的位置，发现有文件头的结构，但是BPG这种文件之前没有接触过，搜索引擎操作一下便知道了：

BPG（Better Portable Graphics，更好的可移植图形）是一种新的图像格式。它的目的是在质量或文件大小有问题时替换jpeg图像格式。

手动分离HEX数据（由于BPG文件比较少见，所以binwalk并不能分离出来）：



将文件保存，用bpgview打开，便可以看到叠加在原文件尾部的图像数据：



这题是一道比较特别的文件提取的题目，特别在于使用了一种较为少见的文件格式，从而避开了绝大多数自动分析工具的识别。在这里贴上一些常用工具的分析结果：

zsteg：（成功识别出）

randark@randark-VirtualBox:~/shared$ zsteg challenge.png  
[?] 15231 bytes of extra data after image end (IEND), offset = 0xab43f  
extradata:0 .. file: BPG (Better Portable Graphics)  
 00000000: 42 50 47 fb 30 00 8b 47 86 3c 00 03 92 47 40 03 |BPG.0..G.<...G@.|  
 00000010: 92 47 40 44 09 c1 71 83 12 00 00 00 01 44 01 c1 |.G@D..q......D..|  
 00000020: 71 83 12 00 00 01 26 09 af 8d 2e 62 f7 39 ac 62 |q.....&....b.9.b|  
 00000030: d6 b5 c0 b5 ed 07 ab e1 17 b6 f2 d8 11 23 a0 00 |.............#..|  
 00000040: 00 03 00 00 03 00 00 03 03 a4 8e e2 6c b8 cf db |............l...|  
 00000050: a4 6a 74 00 00 03 00 00 03 00 03 a9 f1 18 94 9e |.jt.............|  
 00000060: 5a 13 a0 00 00 03 00 00 03 00 01 93 5b af 6a 90 |Z...........[.j.|  
 00000070: 1c 00 00 03 00 00 03 00 00 07 57 02 12 18 00 00 |..........W.....|  
 00000080: 03 00 00 03 00 00 03 00 03 a0 de 76 44 00 00 03 |...........vD...|  
 00000090: 00 00 03 00 00 03 00 71 be b3 3f 00 00 03 00 00 |.......q..?.....|  
 000000a0: 03 00 00 07 1d 13 c5 c8 00 00 03 00 00 03 00 00 |................|  
 000000b0: 32 b5 b8 86 c0 00 00 03 00 00 03 00 01 18 3a d3 |2.............:.|  
 000000c0: 12 00 00 03 00 00 03 00 04 79 bc f0 98 00 00 03 |.........y......|  
 000000d0: 00 00 03 00 0b 9d 31 37 90 00 00 03 00 00 03 00 |......17........|  
 000000e0: 1a 4c b8 48 c0 00 00 03 00 00 03 00 48 1a dd 73 |.L.H........H..s|  
 000000f0: 80 00 00 01 26 01 af c6 8d 05 d0 72 05 fa 57 b1 |....&......r..W.|  
meta date:create .. text: "2019-09-18T17:16:33+06:00"  
meta date:modify .. text: "2017-05-08T12:47:24+05:00"  
meta icc:copyright .. text: "Copyright 1999 Adobe Systems Incorporated"  
meta icc:description.. text: "Apple RGB"  
meta icc:manufacturer.. [same as "meta icc:description"]  
meta icc:model .. [same as "meta icc:description"]  
imagedata .. text: "/3!D<&TE0LF-NA&@5!$&"  
b4,r,lsb,xy .. text: "!E#4B23U5C3D5TF4fs5UBUSDEGcUUfdvUfdvugvFgUuwXVfwug"  
b4,g,lsb,xy .. text: "2U$ESCDfFTDUFeWEftFfSfdUVXtffwu"  
b4,b,lsb,xy .. text: "S3\"T421%UTdU46R75CbEGWSEhEeuX"  
b4,rgb,lsb,xy .. text: "#Q$EuW\"CF4dWEr54b54cFV"  
b4,bgr,lsb,xy .. text: "S$!uGUB&Cd7TuE2d52d6C"

binwalk：

randark@randark-VirtualBox:~/shared$ binwalk challenge.png  
  
DECIMAL HEXADECIMAL DESCRIPTION  
--------------------------------------------------------------------------------  
0 0x0 PNG image, 1100 x 700, 8-bit/color RGB, non-interlaced  
399 0x18F Zlib compressed data, best compression

## 文件合并及分离

### 文件合并

#### copy

copy命令是cmd里面的一个命令，可以操作文件的复制操作：

C:\Users\24743>help copy  
将一份或多份文件复制到另一个位置。  
  
COPY [/D] [/V] [/N] [/Y | /-Y] [/Z] [/L] [/A | /B ] source [/A | /B]  
 [+ source [/A | /B] [+ ...]] [destination [/A | /B]]  
  
 source 指定要复制的文件。  
 /A 表示一个 ASCII 文本文件。  
 /B 表示一个二进位文件。  
 /D 允许解密要创建的目标文件  
 destination 为新文件指定目录和/或文件名。  
 /V 验证新文件写入是否正确。  
 /N 复制带有非 8dot3 名称的文件时，  
 尽可能使用短文件名。  
 /Y 不使用确认是否要覆盖现有目标文件  
 的提示。  
 /-Y 使用确认是否要覆盖现有目标文件  
 的提示。  
 /Z 用可重新启动模式复制已联网的文件。  
 /L 如果源是符号链接，请将链接复制  
 到目标而不是源链接指向的实际文件。  
  
命令行开关 /Y 可以在 COPYCMD 环境变量中预先设定。这可能会被命令行上的 /-Y 替代。除非 COPY命令是在一个批处理脚本中执行的，默认值应为在覆盖时进行提示。  
  
要附加文件，请为目标指定一个文件，为源指定数个文件(用通配符或 file1+file2+file3 格式)。

这里拿我当年自己写的ts文件合并bat举例：

copy /b E:\testts\temp\\* E:\testts\final\final.ts  
del E:\testts\temp\\*  
rename E:\testts\final\final.ts "%date:~0,4%-%date:~5,2%-%date:~8,2%\_%time:~0,2%H-%time:~3,2%M-%time:~6,2%S\_backup.ts"

其中第一行的copy命令的作用就是把E:\testts\temp\目录下的所有文件合并为final.ts单个文件。

#### cat

cat命令在日常中常常用来读取文件内容，比如刻在DNA里面的指令：

cat ./flag

Ubuntu中cat指令的帮助信息如下：

randark@randark-VirtualBox:~/shared$ cat --help  
用法：cat [选项]... [文件]...  
连接所有指定文件并将结果写到标准输出。  
  
如果没有指定文件，或者文件为"-"，则从标准输入读取。  
  
 -A, --show-all 等效于 -vET  
 -b, --number-nonblank 对非空输出行编号，同时取消 -n 选项效果  
 -e 等效于 -vE  
 -E, --show-ends 在每行结束处显示"$"  
 -n, --number 对输出的所有行编号  
 -s, --squeeze-blank 不输出多行空行  
 -t 与 -vT 等效  
 -T, --show-tabs 将跳格字符显示为^I  
 -u (被忽略)  
 -v, --show-nonprinting 使用^ 和M- 引用，除了LFD和 TAB 之外  
 --help 显示此帮助信息并退出  
 --version 显示版本信息并退出  
  
示例：  
 cat f - g 先输出f 的内容，然后输出标准输入的内容，最后输出g 的内容。  
 cat 将标准输入的内容复制到标准输出。  
  
GNU coreutils 在线帮助：<https://www.gnu.org/software/coreutils/>  
请向 <http://translationproject.org/team/zh\_CN.html> 报告 cat 的翻译错误  
完整文档请见：<https://www.gnu.org/software/coreutils/cat>  
或者在本地使用：info '(coreutils) cat invocation'

对于文件合并，也可以使用cat指令来实现，比如：

cat file1.txt file2.txt file3.txt > file\_total.txt

这个指令就实现了将file1.txt，file2.txt，file3.txt三个文件合并为file\_total.txt一个文件。

#### Pytyon脚本

##### struct 模块

有的时候需要用 Python 处理二进制数据，比如，存取文件，socket 操作时。这时候，可以使用 Python 的 struct 模块来完成。

struct 模块中最重要的三个函数是 pack()、unpack() 和 calcsize()

* pack(fmt, v1, v2, ...) 按照给定的格式（fmt），把数据封装成字符串（实际上是类似于 c 结构体的字节流）
* unpack(fmt, string) 按照给定的格式（fmt）解析字节流 string，返回解析出来的 tuple
* calcsize(fmt) 计算给定的格式（fmt）占用多少字节的内存

这里打包格式 fmt 确定了将变量按照什么方式打包成字节流，其包含了一系列的格式字符串。这里就不再给出不同格式字符串的含义了，详细细节可以参照 [Python Doc](https://docs.python.org/2/library/struct.html)

import struct  
struct.pack('>I',16)  
>>>'\x00\x00\x00\x10'

pack 的第一个参数是处理指令，'>I' 的意思是：> 表示字节顺序是 Big-Endian，也就是网络序，I 表示 4 字节无符号整数。

后面的参数个数要和处理指令一致。

读入一个 BMP 文件的前 30 字节，文件头的结构按顺序如下

* 两个字节：BM 表示 Windows 位图，BA 表示 OS/2 位图
* 一个 4 字节整数：表示位图大小
* 一个 4 字节整数：保留位，始终为 0
* 一个 4 字节整数：实际图像的偏移量
* 一个 4 字节整数：Header 的字节数
* 一个 4 字节整数：图像宽度
* 一个 4 字节整数：图像高度
* 一个 2 字节整数：始终为 1
* 一个 2 字节整数：颜色数

import struct  
bmp ='\x42\x4d\x38\x8c\x0a\x00\x00\x00\x00\x00\x36\x00\x00\x00\x28\x00\x00\x00\x80\x02\x00\x00\x68\x01\x00\x00\x01\x00\x18\x00'  
struct.unpack('<ccIIIIIIHH',bmp)  
>>>('B', 'M', 691256, 0, 54, 40, 640, 360, 1, 24)

##### bytearray 字节数组

将文件以二进制数组形式读取

data = bytearray(open('challenge.png', 'rb').read())

字节数组就是可变版本的字节

data[0] = '\x89'

这里拿一个合并目录下所有文件的脚本来举个例子：

import os  
import struct  
dir\_path = "d:/"  
out\_HEX = []  
  
file\_name\_all = os.listdir(dir\_path)  
for file\_name in file\_name\_all:  
 file\_path = dir\_path+"/"+file\_name  
 if os.path.isfile(file\_path) == True:  
 with open(file\_path, "rb") as file\_input:  
 while 1:  
 c = file\_input.read(1)  
 if not c:  
 break  
 out\_HEX.append(ord(c))  
file\_output\_path = dir\_path+"/out"  
with open(file\_output\_path, "wb") as outfile:  
 for bytes\_ in out\_HEX:  
 outfile.write(struct.pack("B", bytes\_))

## 文件分离

关于将文件数据拆分隐写在图像的RGB通道的这种情况，放在图片隐写里面讨论，这里不加以论述。

### binwalk

binwalk 本是一个固件的分析工具，比赛中常用来发现多个文件粘合再在一起的情况。根据文件头去识别一个文件中夹杂的其他文件，有时也会存在误报率（尤其是对 Pcap 流量包等文件时）。

一般只会binwalk <file>分析文件结构和binwalk -e <file>提取文件中黏合的其它文件。例如：

randark@randark-VirtualBox:~/CTF$ binwalk -e QR\_code.png  
  
DECIMAL HEXADECIMAL DESCRIPTION  
--------------------------------------------------------------------------------  
0 0x0 PNG image, 280 x 280, 1-bit colormap, non-interlaced  
  
WARNING: Extractor.execute failed to run external extractor 'jar xvf '%e'': [Errno 2] No such file or directory: 'jar', 'jar xvf '%e'' might not be installed correctly  
  
WARNING: Extractor.execute failed to run external extractor '7z x -y '%e' -p ''': [Errno 2] No such file or directory: '7z', '7z x -y '%e' -p ''' might not be installed correctly  
471 0x1D7 Zip archive data, encrypted at least v2.0 to extract, compressed size: 29, uncompressed size: 15, name: 4number.txt  
650 0x28A End of Zip archive, footer length: 22

### foremost

Foremost is a Linux program to recover files based on their headers and footers. Foremost can work on image files, such as those generated by dd, Safeback, Encase, etc, or directly on a drive. The headers and footers are specified by a configuration file, so you can pick and choose which headers you want to look for.

foremost常常用于binwalk无法正常工作的时候，比如用于流量包提取流量中的文件。

randark@randark-VirtualBox:~$ foremost -h  
foremost version 1.5.7 by Jesse Kornblum, Kris Kendall, and Nick Mikus.  
$ foremost [-v|-V|-h|-T|-Q|-q|-a|-w-d] [-t <type>] [-s <blocks>] [-k <size>]   
 [-b <size>] [-c <file>] [-o <dir>] [-i <file]   
  
-V - display copyright information and exit  
-t - specify file type. (-t jpeg,pdf ...)   
-d - turn on indirect block detection (for UNIX file-systems)   
-i - specify input file (default is stdin)   
-a - Write all headers, perform no error detection (corrupted files)   
-w - Only write the audit file, do not write any detected files to the disk   
-o - set output directory (defaults to output)  
-c - set configuration file to use (defaults to foremost.conf)  
-q - enables quick mode. Search are performed on 512 byte boundaries.  
-Q - enables quiet mode. Suppress output messages.   
-v - verbose mode. Logs all messages to screen

例如[BUUCTF-菜刀666](https://buuoj.cn/challenges#%E8%8F%9C%E5%88%80666)中，binwalk的分析结果如下：

randark@randark-VirtualBox:~/CTF$ binwalk 666666.pcapng  
  
DECIMAL HEXADECIMAL DESCRIPTION  
--------------------------------------------------------------------------------  
663085 0xA1E2D xz compressed data  
664045 0xA21ED xz compressed data  
812025 0xC63F9 xz compressed data  
814001 0xC6BB1 xz compressed data  
1238637 0x12E66D xz compressed data  
1240937 0x12EF69 xz compressed data  
1391563 0x153BCB xz compressed data  
1393067 0x1541AB xz compressed data  
1406647 0x1576B7 xz compressed data  
1412887 0x158F17 xz compressed data  
1422689 0x15B561 Zip archive data, encrypted at least v2.0 to extract, compressed size: 52, uncompressed size: 40, name: flag.txt

明显分析结果存在问题，用foremost就可以正常分析出文件：

randark@randark-VirtualBox:~/CTF$ foremost 666666.pcapng  
Processing: 666666.pcapng  
|foundat=flag.txtC����cS���J��Ea�v�  
 ��&e$K��2%�$��,�=�J��1p��p46PK?  
\*|

### 010 Editor手动分离

用010 Editor的模板套用在文件上，然后存在黏合文件的话，010 Editor会有提示，一般都在源文件的尾部，手动分离出HEX数据，粘贴到新文件中，就可以分离出来。

### dd

以下是dd的帮助信息：

randark@randark-VirtualBox:~/shared$ dd --help  
用法：dd [操作数] ...  
　或：dd 选项  
复制文件，依照指定操作数转换并格式化。  
  
 bs=字节数 一次读写的比特数（默认：512）；  
 会覆盖 ibs 和 obs 选项  
 cbs=字节数 一次转换的字节数  
 conv=CONVS 依照每个逗号分割的符号列表转换文件  
 count=块数 只将复制指定数量的输入块  
 ibs=字节数 一次读取的字节数（默认：512)  
 if=文件 从指定文件而非标准输入来进行读取  
 iflag=标志 按照以逗号分隔的符号列表指定的方式读取  
 obs=字节数 一次写入指定字节数（默认：512）  
 of=文件 写入到指定文件而非标准输出  
 oflag=标志 按照以逗号分隔的符号列表指定的方式写入  
 seek=块数 在输出开始处跳过指定的 obs 大小的块数  
 skip=块数 在输入开始处跳过指定的 ibs 大小的块数  
 status=等级 要输出到标准错误的信息等级；  
 'none' 将仅输出错误信息，  
 'noxfer' 将不输出最终传输统计信息，  
 'progress' 将显示周期性的传输统计信息  
  
N and BYTES may be followed by the following multiplicative suffixes:  
c =1, w =2, b =512, kB =1000, K =1024, MB =1000\*1000, M =1024\*1024, xM =M,  
GB =1000\*1000\*1000, G =1024\*1024\*1024, and so on for T, P, E, Z, Y.  
  
Each CONV symbol may be:  
  
 ascii from EBCDIC to ASCII  
 ebcdic from ASCII to EBCDIC  
 ibm from ASCII to alternate EBCDIC  
 block pad newline-terminated records with spaces to cbs-size  
 unblock replace trailing spaces in cbs-size records with newline  
 lcase change upper case to lower case  
 ucase change lower case to upper case  
 sparse try to seek rather than write the output for NUL input blocks  
 swab swap every pair of input bytes  
 sync pad every input block with NULs to ibs-size; when used  
 with block or unblock, pad with spaces rather than NULs  
 excl 如果输出文件已存在则认为操作失败  
 nocreat 不要创建输出文件  
 notrunc 不要截断输出文件  
 noerror 读取数据发生错误后仍然继续  
 fdatasync 结束前将输出文件数据物理上写入磁盘  
 fsync 与上者类似，但也将元数据一同写入  
  
FLAG 符号可以是：  
  
 append 追加模式(仅对输出有意义；隐含了conv=notrunc)  
 direct 使用直接I/O 存取模式  
 directory 除非是目录，否则操作失败  
 dsync 使用同步 I/O 存取模式  
 sync 与上者类似，但同时也对元数据生效  
 fullblock 为输入积累完整块(仅iflag)  
 nonblock 使用无阻塞I/O 存取模式  
 noatime 不更新访问时间  
 nocache 请求不使用缓存。参见 oflag=sync  
 noctty 不根据文件指派控制终端  
 nofollow 不跟随链接文件  
 count\_bytes 把 'count=N' 看作字节计数（仅 iflag）  
 skip\_bytes 把 'skip=N' 看作字节计数（仅 iflag）  
 seek\_bytes 把 'seek=N' 看作字节计数（仅 oflag）  
  
向正在运行的 'dd' 进程发送 USR1 信号可以令其向标准错误输出 I/O  
统计数据并继续进行复制。  
  
选项有：  
  
 --help 显示此帮助信息并退出  
 --version 显示版本信息并退出  
  
GNU coreutils 在线帮助：<https://www.gnu.org/software/coreutils/>  
请向 <http://translationproject.org/team/zh\_CN.html> 报告 dd 的翻译错误  
完整文档请见：<https://www.gnu.org/software/coreutils/dd>  
或者在本地使用：info '(coreutils) dd invocation'

假设一个文件big.bin是由3个文件合并成的，现在需要把他们都提取出来。其中part1大小1M，part2大小3m，part3大小8M

dd if=big.bin of=./part1 bs=1M count=1  
dd if=big.bin of=./part2 bs=1M skip=1 count=3  
dd if=big.bin of=./part3 bs=1M skip=4 count=8

if参数传入了输入文件，of参数定义了输出的文件。同时，一般obs和ibs的块大小都是512，但是由于指定了bs参数，导致obs和ibs两个参数都被改了，改为1M。如果要用bs参数的话，记得obs和ibs参数的默认大小随bs参数而改变。