链滴

一道另类而简单粗暴的题解

作者：someone48938

原文链接：https://ld246.com/article/1591734772761

来源网站：链滴

上周看到帖子《解密有奖活动》这个形式上类似 CTF 的题，有感而发，跟风也出了一道题。由于以前出过这类题，所以出的比较简单粗暴，本来可以更优雅的。但解决这道题仍然需要解题人有一定的耐和基础知识储备。

现在帖子沉下去了，看来没人打算解了，那这 250 积分就送给我自己了
oy

下面我将从一个小白的视角，分析并解决这个题目。

1. 密文在哪里?

有上一个解密帖子的经验，先把帖子正文复制到 vim 里面看下，看看有没有特殊字符（并没有）。看密文藏在别的地方。

正巧发现社区提供了查看帖子 Markdown 的功能，点开看看。

```
密文就藏在这个帖子里。答案是一个 11 字符长度的英文单词，全小写。

奖励：对于第一个包含正确题解的回帖，我将给予回帖人 250 积分作为奖励。:doge:

<details><summary><b>如果看到密文以后没有一点解题的头绪，请点开看提示</b></summary>
<p>

密文其实是一段可以执行的代码

</p>
</details>

<!--
```

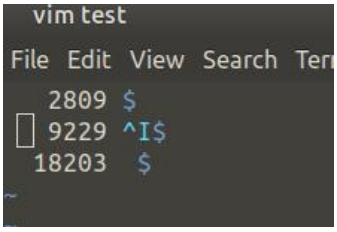
原文链接：一道另类而简单粗暴的题解

咦，发现了一段 HTML 注释（看上去全是空白字符，还挺长），看来这就是密文了，先保存到本地件中再说。

2. 密文该怎么解?

```
sed 's/\(.\)/\1\n/g' some_text | sort | uniq -c
```

先处理一下，看看都包含哪些字符。



看来还真的只有 \n、\t、 三种空白字符。到这里已经没有思路了，用三种字符做编码，可能的方式太多，难道要一个个去尝试？作为一个从来没打过 CTF 的小白，完全不知道接下来该怎么办。

诶，对了，帖子里不是有一个提示么？看一看，或许会有启发。

> 密文其实是一段可以执行的代码

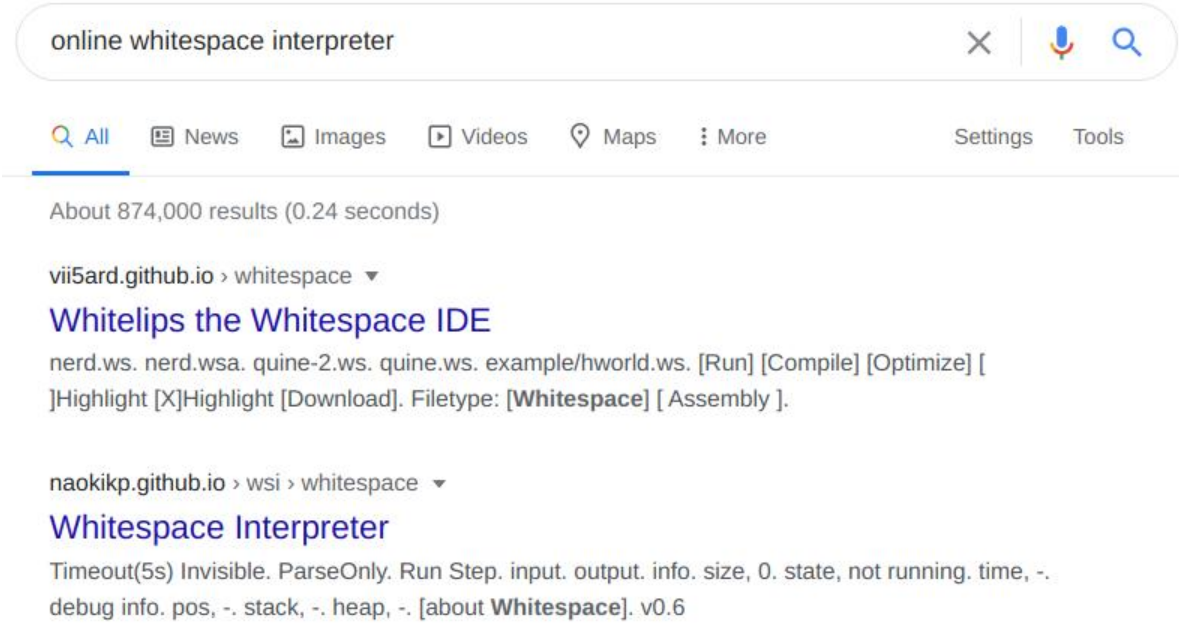
额，这不等于没说嘛，不还是得先解码再执行。。等一下，如果这段代码可以直接执行呢?

这时，Google 就派上用场了。搜索一下，有没有仅用空白字符编程的语言。

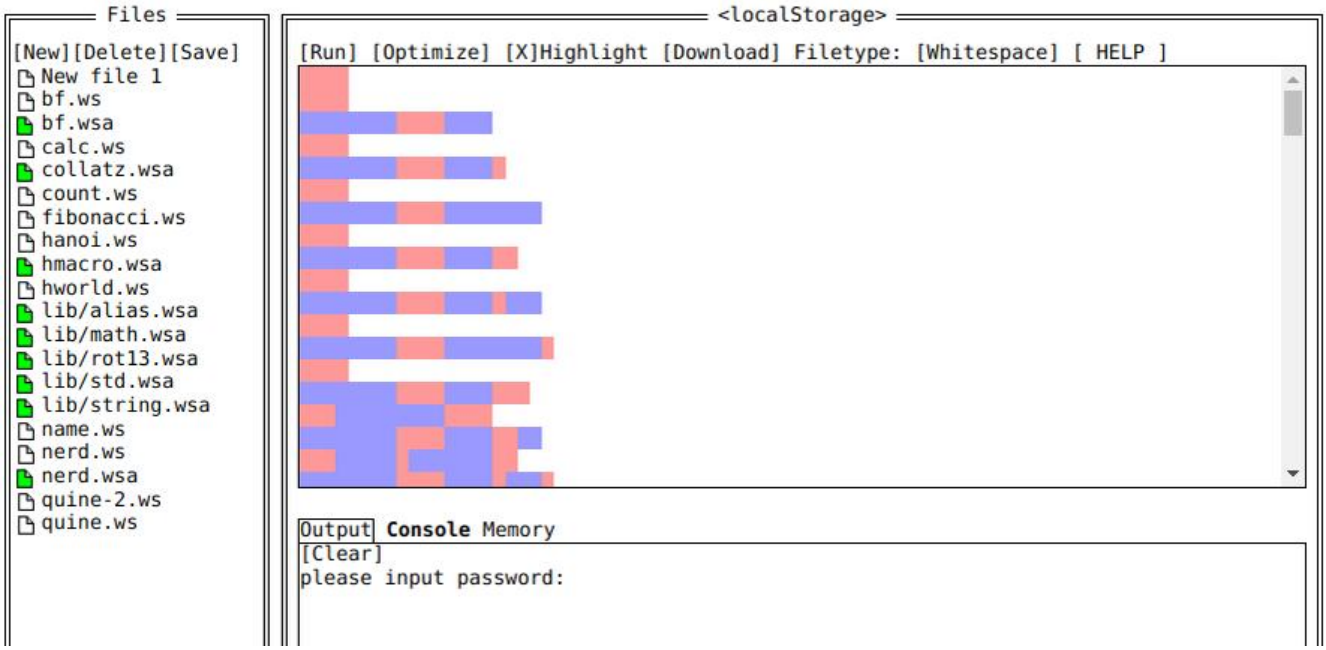
还真有。这个语言的名字就叫 Whitespace。点进 Wiki 简单看了下，这个语言大概就是用三种空白符组成一系列指令来操作栈和堆，还有一些数学计算和 I/O 相关的指令。但是用这种语言写的代码对类似乎并不友好，很难读。能不读还是先不读吧。执行一下看看结果再说。万一我的猜想不对，这不

一段 Whitespace 代码呢。

相信这个存在了已经快 20 个年头的语言一定有很多现成开源的解释器实现了。我希望找到一个可以线直接用的，省得下源码到本地自己编译了。



完美，一下子就找到了。那就试试第一个吧。



点新建文件，把那串空白字符复制进去，可以看到这个在线 IDE 做的有模有样，空格用淡粉色显示，Tb 用蓝紫色显示（但我并不觉得代码的可读性提高了多少）。

点执行，满怀期待地看输出结果。只见程序输出了“请输入密码”字样。看来我之前的猜想是正确的但距离正确答案还有一步之遥。先得知道密码是什么。

3. 密码是什么?

先随便输入一些东西，看看会输出什么。

```
Output Console Memory
[Clear]
please input password: helloworld
fail!
```

看来密码不是“helloworld”（能是这种密码就见鬼了），这也意味着我需要从程序中破解出密码了。

如果能读懂这段 Whitespace 代码，那应该就能知道密码是什么。然而，就算出题人没有刻意对代码过混淆，我也不想为了一道谜题，特意去学一门比汇编还难读的奇葩语言。

那我们就从另外一个方向下手。不管是什么语言写的程序，验证密码这一行为，无非是将用户输入的符串和一个现成的字符串做比较。对于这道题而言，那个现成的字符串，多半会事先放在内存的某个置，而且对于这种简单的程序，应该不会做 Hash 之类的处理。

重新运行程序，当程序暂停等待我们输入时，看一看 Whitespace VM 的内存使用情况。

```
Output Console Memory

Stack: [0]
Heap: { 0:0,    1:16777197,     2:0,    3:16777195,     4:16777199,     5:16777197,
6:0,    8:112,  9:108,  10:101, 11:97,  12:115, 13:101, 14:32,  15:105, 16:110,
17:112, 18:117, 19:116, 20:32,  21:112, 22:97,  23:115, 24:115, 25:119, 26:111,
27:114, 28:100, 29:58,  30:32,  31:0,   32:99,  33:111, 34:110, 35:118, 36:111,
37:108, 38:117, 39:116, 40:105, 41:111, 42:110, 43:0,   44:115, 45:117, 46:99,
47:99,  48:101, 49:115, 50:115, 51:33,  52:0,   53:102, 54:97,  55:105, 56:108,
57:33,  58:0,   59:52,  16777203:12,    16777204:0,     16777206:0,
16777207:16777215,      16777208:38,    16777209:16777203,      16777210:12,
16777223:0}
```

栈里面就一个 0，没有关注价值，主要看堆内存。

我发现，堆内存中有大量数值比较小的连续单元，用 0 隔开，而且都在可打印 ASCII 的范围，且 32 个值（对应 ASCII 的空格）多次出现。所以我大胆的猜测，这些多半是是明文存储的字符串。

接下来我打算把他们转换成对应的 ASCII 字符，然后进行观察。简单处理了一下，搞成一个 C 语言脚本：

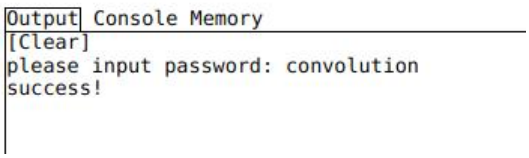
```c
#include <stdio.h>

int main() {
    char a[] = { 112, 108, 101, 97, 115, 101, 32, 105, 110, 112, 117, 116, 32, 112, 97, 115, 115, 119
 111, 114, 100, 58, 32, 0 };
    char b[] = { 99, 111, 110, 118, 111, 108, 117, 116, 105, 111, 110, 0};
    char c[] = { 115, 117, 99, 99, 101, 115, 115, 33, 0 };
    char d[] = { 102, 97, 105, 108, 33, 0 };
    printf("%s\n%s\n%s\n%s\n", a, b, c, d);
}
```

执行结果：

```
please input password:
convolution
success!
fail!
```

这个 "convolution" 在里面显得格格不入，想必它就是密码了，而且输入以后，程序应该会输出 "suc

ess!"。



经验证，我的猜想是正确的。由于题目里说，答案是一个长度为 11 字符的单词，那它显然应该是刚得到的密码 "convolution"。至此，题解完毕。