第四届“强网”拟态防御国际精英挑战赛_wp（上）

原创

合天网安实验室 于 2021-10-28 16:20:00 发布 196 收藏 1
文章标签： aof processing gwt rust asynctask

本文链接：https://blog.csdn.net/qq_38154820/article/details/121026870

Re

1、fastjs

提取字节码

```
/* File generated automatically by the QuickJS compiler. */


#include "quickjs-libc.h"


const uint32_t qjsc_hello_size = 2269;


const uint8_t qjsc_hello[2269] = {
   0x02, 0x3A, 0x10, 0x6C, 0x6F, 0x6E, 0x67, 0x32, 0x73, 0x74, 0x72, 0x10, 0x73, 0x74, 0x72, 0x32,
   0x6C, 0x6F, 0x6E, 0x67, 0x10, 0x73, 0x64, 0x66, 0x73, 0x66, 0x73, 0x64, 0x66, 0x0E, 0x73, 0x74,
   0x72, 0x32, 0x48, 0x65, 0x78, 0x0E, 0x68, 0x65, 0x78, 0x32, 0x73, 0x74, 0x72, 0x0C, 0x78, 0x78,
   0x78, 0x66, 0x73, 0x73, 0x08, 0x6D, 0x61, 0x69, 0x6E, 0x08, 0x61, 0x72, 0x67, 0x73, 0x82, 0x01,
   0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E, 0x4F, 0x50,
   0x51, 0x52, 0x53, 0x54, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5A, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66,
   0x67, 0x68, 0x69, 0x6A, 0x6B, 0x6C, 0x6D, 0x6E, 0x6F, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76,
   0x77, 0x78, 0x79, 0x7A, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x2B, 0x2F,
   0x3D, 0x0E, 0x5F, 0x6B, 0x65, 0x79, 0x53, 0x74, 0x72, 0x0A, 0x64, 0x66, 0x73, 0x66, 0x73, 0x14,
   0x73, 0x63, 0x72, 0x69, 0x70, 0x74, 0x41, 0x72, 0x67, 0x73, 0x0A, 0x73, 0x68, 0x69, 0x66, 0x74,
   0x0A, 0x65, 0x72, 0x72, 0x6F, 0x72, 0x20, 0x2E, 0x2F, 0x74, 0x65, 0x73, 0x74, 0x5F, 0x65, 0x6E,
   0x63, 0x6F, 0x64, 0x65, 0x2E, 0x6A, 0x73, 0x02, 0x76, 0x02, 0x77, 0x04, 0x76, 0x6C, 0x04, 0x73,
   0x6C, 0x02, 0x69, 0x18, 0x66, 0x72, 0x6F, 0x6D, 0x43, 0x68, 0x61, 0x72, 0x43, 0x6F, 0x64, 0x65,
   0x12, 0x73, 0x75, 0x62, 0x73, 0x74, 0x72, 0x69, 0x6E, 0x67, 0x02, 0x73, 0x06, 0x6C, 0x65, 0x6E,
   0x14, 0x63, 0x68, 0x61, 0x72, 0x43, 0x6F, 0x64, 0x65, 0x41, 0x74, 0x06, 0x73, 0x74, 0x72, 0x06,
   0x6B, 0x65, 0x79, 0x02, 0x6B, 0x02, 0x6E, 0x02, 0x7A, 0x02, 0x79, 0x0A, 0x64, 0x65, 0x6C, 0x74,
   0x61, 0x04, 0x6D, 0x78, 0x02, 0x65, 0x02, 0x71, 0x06, 0x73, 0x75, 0x6D, 0x02, 0x70, 0x0A, 0x66,
   0x6C, 0x6F, 0x6F, 0x72, 0x0C, 0x6F, 0x75, 0x74, 0x70, 0x75, 0x74, 0x08, 0x63, 0x68, 0x72, 0x31,
   0x10, 0x70, 0x61, 0x72, 0x73, 0x65, 0x49, 0x6E, 0x74, 0x0C, 0x73, 0x75, 0x62, 0x73, 0x74, 0x72,
   0x08, 0x63, 0x68, 0x72, 0x32, 0x08, 0x63, 0x68, 0x72, 0x33, 0x08, 0x65, 0x6E, 0x63, 0x31, 0x08,
   0x65, 0x6E, 0x63, 0x32, 0x08, 0x65, 0x6E, 0x63, 0x33, 0x08, 0x65, 0x6E, 0x63, 0x34, 0x0A, 0x69,
   0x73, 0x4E, 0x61, 0x4E, 0x0C, 0x63, 0x68, 0x61, 0x72, 0x41, 0x74, 0x12, 0x64, 0x66, 0x73, 0x66,
   0x64, 0x73, 0x66, 0x73, 0x64, 0x0C, 0x66, 0x77, 0x64, 0x65, 0x72, 0x66, 0x0A, 0x70, 0x72, 0x69,
   0x6E, 0x74, 0x18, 0x79, 0x6F, 0x75, 0x72, 0x20, 0x69, 0x6E, 0x70, 0x75, 0x74, 0x3A, 0x20, 0x20,
   0x6E, 0x6F, 0x5F, 0x74, 0x68, 0x69, 0x6E, 0x67, 0x5F, 0x69, 0x73, 0x5F, 0x74, 0x72, 0x75, 0x65,
   0x0E, 0x64, 0x66, 0x64, 0x66, 0x77, 0x66, 0x33, 0xE0, 0x01, 0x30, 0x35, 0x61, 0x65, 0x64, 0x30,
   0x63, 0x65, 0x34, 0x34, 0x31, 0x66, 0x38, 0x30, 0x62, 0x35, 0x62, 0x63, 0x33, 0x36, 0x61, 0x66,
   0x34, 0x63, 0x36, 0x39, 0x38, 0x35, 0x30, 0x39, 0x66, 0x63, 0x36, 0x63, 0x63, 0x33, 0x63, 0x39,
   0x37, 0x31, 0x34, 0x36, 0x33, 0x35, 0x33, 0x64, 0x65, 0x35, 0x61, 0x39, 0x35, 0x63, 0x36, 0x61,
   0x62, 0x65, 0x61, 0x30, 0x37, 0x66, 0x64, 0x34, 0x61, 0x37, 0x30, 0x37, 0x30, 0x39, 0x33, 0x32,
   0x64, 0x38, 0x36, 0x61, 0x63, 0x33, 0x32, 0x64, 0x36, 0x32, 0x38, 0x36, 0x37, 0x32, 0x61, 0x35,
```

```
0x39, 0x31, 0x32, 0x33, 0x65, 0x35, 0x39, 0x37, 0x32, 0x33, 0x33, 0x31, 0x64, 0x62, 0x35, 0x64,
0x66, 0x66, 0x65, 0x37, 0x30, 0x35, 0x37, 0x33, 0x36, 0x32, 0x06, 0x79, 0x65, 0x73, 0x0E, 0x00,
0x06, 0x00, 0xA0, 0x01, 0x00, 0x01, 0x00, 0x03, 0x00, 0x08, 0xE4, 0x01, 0x01, 0xA2, 0x01, 0x00,
0x00, 0x00, 0x3F, 0xE1, 0x00, 0x00, 0x00, 0x40, 0x3F, 0xE2, 0x00, 0x00, 0x00, 0x40, 0x3F, 0xE3,
0x00, 0x00, 0x00, 0x40, 0x3F, 0xE4, 0x00, 0x00, 0x00, 0x40, 0x3F, 0xE5, 0x00, 0x00, 0x00, 0x40,
0x3F, 0xE6, 0x00, 0x00, 0x00, 0x40, 0x3F, 0xE7, 0x00, 0x00, 0x00, 0x40, 0x3F, 0xE8, 0x00, 0x00,
0x00, 0x00, 0xC0, 0x00, 0x40, 0xE1, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x01, 0x40, 0xE2, 0x00, 0x00,
0x00, 0x00, 0xC0, 0x02, 0x40, 0xE3, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x03, 0x40, 0xE4, 0x00, 0x00,
0x00, 0x00, 0xC0, 0x04, 0x40, 0xE5, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x05, 0x40, 0xE6, 0x00, 0x00,
0x00, 0x00, 0xC0, 0x07, 0x40, 0xE7, 0x00, 0x00, 0x00, 0x00, 0x3E, 0xE8, 0x00, 0x00, 0x00, 0x00,
0x38, 0xE6, 0x00, 0x00, 0x00, 0x04, 0xE9, 0x00, 0x00, 0x00, 0x15, 0x43, 0xEA, 0x00, 0x00, 0x00,
0xC9, 0x38, 0xE6, 0x00, 0x00, 0x00, 0xC0, 0x06, 0x15, 0x43, 0xEB, 0x00, 0x00, 0x00, 0xC9, 0x06,
0xC9, 0x37, 0xEC, 0x00, 0x00, 0x00, 0xF4, 0xEB, 0x1D, 0x38, 0xEC, 0x00, 0x00, 0x00, 0x11, 0x39,
0xE8, 0x00, 0x00, 0x00, 0xC9, 0x38, 0xE8, 0x00, 0x00, 0x00, 0x42, 0xED, 0x00, 0x00, 0x00, 0x24,
0x00, 0x00, 0xC9, 0xEC, 0x25, 0x06, 0xC9, 0x37, 0x4D, 0x00, 0x00, 0x00, 0xF4, 0xEB, 0x0F, 0x38,
0x4D, 0x00, 0x00, 0x00, 0x11, 0x39, 0xE8, 0x00, 0x00, 0x00, 0xC9, 0xEC, 0x0D, 0x04, 0xEE, 0x00,
0x00, 0x00, 0x11, 0x39, 0xE8, 0x00, 0x00, 0x00, 0xC9, 0x38, 0xE7, 0x00, 0x00, 0x00, 0x38, 0xE8,
0x00, 0x00, 0x00, 0xEF, 0xCD, 0x28, 0xDE, 0x03, 0x01, 0x13, 0xF1, 0x00, 0x3E, 0xB0, 0x01, 0x58,
0x00, 0x05, 0x36, 0x00, 0x09, 0x40, 0x35, 0x3F, 0x49, 0x3F, 0x3F, 0x0D, 0x41, 0x0E, 0x43, 0x06,
0x00, 0xC2, 0x03, 0x02, 0x03, 0x02, 0x09, 0x00, 0x01, 0x6E, 0x05, 0xE0, 0x03, 0x00, 0x01, 0x00,
0xE2, 0x03, 0x00, 0x01, 0x00, 0xE4, 0x03, 0x00, 0x00, 0x00, 0xE6, 0x03, 0x00, 0x01, 0x00, 0xE8,
0x03, 0x00, 0x02, 0x00, 0xD1, 0xE9, 0xC9, 0xD1, 0xC5, 0xB6, 0x9E, 0x47, 0xBF, 0x00, 0xAD, 0xCA,
0xB5, 0xCB, 0xC7, 0xC5, 0xA3, 0xEA, 0x3B, 0xD1, 0xC7, 0x71, 0x38, 0x99, 0x00, 0x00, 0x00, 0x42,
0xF5, 0x00, 0x00, 0x00, 0xD1, 0xC7, 0x47, 0xBE, 0xFF, 0x00, 0xAD, 0xD1, 0xC7, 0x47, 0xBD, 0x08,
0xA2, 0xBE, 0xFF, 0x00, 0xAD, 0xD1, 0xC7, 0x47, 0xBD, 0x10, 0xA2, 0xBE, 0xFF, 0x00, 0xAD, 0xD1,
0xC7, 0x47, 0xBD, 0x18, 0xA2, 0xBE, 0xFF, 0x00, 0xAD, 0x24, 0x04, 0x00, 0x49, 0x93, 0x02, 0xEC,
0xC2, 0xD2, 0xEA, 0x15, 0xD1, 0x42, 0x5B, 0x00, 0x00, 0x00, 0xC1, 0x24, 0x01, 0x00, 0x42, 0xF6,
0x00, 0x00, 0x00, 0xB5, 0xC6, 0x25, 0x02, 0x00, 0xD1, 0x42, 0x5B, 0x00, 0x00, 0x00, 0xC1, 0x25,
0x01, 0x00, 0xDE, 0x03, 0x01, 0x0B, 0x03, 0x12, 0x30, 0x27, 0x67, 0x35, 0x35, 0x49, 0x17, 0x12,
0x69, 0x06, 0x00, 0x00, 0xE0, 0xFF, 0xFF, 0xFF, 0xEF, 0x41, 0x0E, 0x43, 0x06, 0x00, 0xC4, 0x03,
0x02, 0x03, 0x02, 0x07, 0x00, 0x00, 0x5D, 0x05, 0xEE, 0x03, 0x00, 0x01, 0x00, 0xE2, 0x03, 0x00,
0x01, 0x00, 0xF0, 0x03, 0x00, 0x00, 0x00, 0xE0, 0x03, 0x00, 0x01, 0x00, 0xE8, 0x03, 0x00, 0x02,
0x00, 0xD1, 0xE9, 0xC9, 0x26, 0x00, 0x00, 0xCA, 0xB5, 0xCB, 0xC7, 0xC5, 0xA3, 0xEA, 0x46, 0xC6,
0xC7, 0xB7, 0xA1, 0x71, 0xD1, 0x42, 0xF9, 0x00, 0x00, 0x00, 0xC7, 0x24, 0x01, 0x00, 0xD1, 0x42,
0xF9, 0x00, 0x00, 0x00, 0xC7, 0xB6, 0x9D, 0x24, 0x01, 0x00, 0xBD, 0x08, 0xA0, 0xAF, 0xD1, 0x42,
0xF9, 0x00, 0x00, 0x00, 0xC7, 0xB7, 0x9D, 0x24, 0x01, 0x00, 0xBD, 0x10, 0xA0, 0xAF, 0xD1, 0x42,
0xF9, 0x00, 0x00, 0x00, 0xC7, 0xB8, 0x9D, 0x24, 0x01, 0x00, 0xBD, 0x18, 0xA0, 0xAF, 0x49, 0xB9,
0x94, 0x02, 0xEC, 0xB7, 0xD2, 0xEA, 0x06, 0xC6, 0xC6, 0xE9, 0xC5, 0x49, 0xC6, 0x28, 0xDE, 0x03,
0x13, 0x0B, 0x03, 0x12, 0x17, 0x27, 0x4E, 0x53, 0x53, 0x58, 0x1C, 0x12, 0x1D, 0x0E, 0x43, 0x06,
0x00, 0xC6, 0x03, 0x02, 0x0B, 0x02, 0x06, 0x00, 0x04, 0xF8, 0x01, 0x0D, 0xF4, 0x03, 0x00, 0x01,
0x00, 0xF6, 0x03, 0x00, 0x01, 0x00, 0xE0, 0x03, 0x00, 0x00, 0x00, 0xF8, 0x03, 0x00, 0x01, 0x00,
0xFA, 0x03, 0x00, 0x02, 0x00, 0xFC, 0x03, 0x00, 0x03, 0x00, 0xFE, 0x03, 0x00, 0x04, 0x00, 0x80,
0x04, 0x00, 0x05, 0x00, 0x82, 0x04, 0x00, 0x06, 0x00, 0x84, 0x04, 0x00, 0x07, 0x00, 0x86, 0x04,
0x00, 0x08, 0x00, 0x88, 0x04, 0x00, 0x09, 0x00, 0x8A, 0x04, 0x00, 0x0A, 0x00, 0xD1, 0xC1, 0xA9,
0xEA, 0x03, 0xC1, 0x28, 0x38, 0xE2, 0x00, 0x00, 0x00, 0xD1, 0x0A, 0xF0, 0xC9, 0x38, 0xE2, 0x00,
0x00, 0x00, 0xD2, 0x09, 0xF0, 0xCA, 0xC5, 0xE9, 0xB6, 0x9E, 0xCB, 0xC5, 0xC7, 0x47, 0xCC, 0xC5,
0xB5, 0x47, 0xC3, 0x04, 0xBF, 0x00, 0xC3, 0x05, 0x38, 0x9D, 0x00, 0x00, 0x00, 0x42, 0x06, 0x01,
0x00, 0x00, 0xBB, 0xBD, 0x34, 0xC7, 0xB6, 0x9D, 0x9B, 0x9D, 0x24, 0x01, 0x00, 0xC3, 0x08, 0xB5,
0xC3, 0x09, 0xC2, 0x08, 0x90, 0xC3, 0x08, 0xB5, 0xA5, 0x69, 0x9B, 0x00, 0x00, 0x00, 0xC2, 0x09,
0xC2, 0x05, 0x9D, 0xBF, 0x01, 0xAD, 0xC4, 0x09, 0xB7, 0xA2, 0xB8, 0xAD, 0xC3, 0x07, 0xB5, 0xC3,
0x0A, 0xC2, 0x0A, 0xC7, 0xA3, 0xEA, 0x43, 0xC5, 0xC2, 0x0A, 0xB6, 0x9D, 0x47, 0xC3, 0x04, 0xC8,
0xBA, 0xA2, 0xC2, 0x04, 0xB7, 0xA0, 0xAE, 0xC2, 0x04, 0xB8, 0xA2, 0xC8, 0xB9, 0xA0, 0xAE, 0x9D,
0xC2, 0x09, 0xC2, 0x04, 0xAE, 0xC6, 0xC2, 0x0A, 0xB8, 0xAD, 0xC2, 0x07, 0xAE, 0x47, 0xC8, 0xAE,
0x9D, 0xAE, 0xC3, 0x06, 0xC5, 0xC2, 0x0A, 0x71, 0xC5, 0xC2, 0x0A, 0x47, 0xC2, 0x06, 0x9D, 0xBF,
0x02, 0xAD, 0x16, 0x49, 0xCC, 0x93, 0x0A, 0xEC, 0xB9, 0xC5, 0xB5, 0x47, 0xC3, 0x04, 0xC8, 0xBA,
0xA2, 0xC2, 0x04, 0xB7, 0xA0, 0xAE, 0xC2, 0x04, 0xB8, 0xA2, 0xC8, 0xB9, 0xA0, 0xAE, 0x9D, 0xC2,
0x09, 0xC2, 0x04, 0xAE, 0xC6, 0xC2, 0x0A, 0xB8, 0xAD, 0xC2, 0x07, 0xAE, 0x47, 0xC8, 0xAE, 0x9D,
0xAE, 0xC3, 0x06, 0xC5, 0xC7, 0x71, 0xC5, 0xC7, 0x47, 0xC2, 0x06, 0x9D, 0xBF, 0x03, 0xAD, 0x16,
0x49, 0xCC, 0xED, 0x5F, 0xFF, 0x38, 0xE4, 0x00, 0x00, 0x00, 0x38, 0xE1, 0x00, 0x00, 0x00, 0xC5,
0x09, 0xF0, 0x23, 0x01, 0x00, 0xDE, 0x03, 0x23, 0x15, 0x03, 0x1C, 0x08, 0x08, 0x30, 0x30, 0x1D,
```

```
   0x44, 0x85, 0x3F, 0x35, 0x21, 0x30, 0x2B, 0xBC, 0x58, 0x17, 0x1C, 0xBC, 0x4E, 0x13, 0x06, 0x00,
   0x00, 0x20, 0x37, 0xEF, 0xC6, 0xE3, 0x41, 0x06, 0x00, 0x00, 0xE0, 0xFF, 0xFF, 0xFF, 0xEF, 0x41,
   0x06, 0x00, 0x00, 0xE0, 0xFF, 0xFF, 0xFF, 0xEF, 0x41, 0x06, 0x00, 0x00, 0xE0, 0xFF, 0xFF, 0xFF,
   0xEF, 0x41, 0x0E, 0x43, 0x06, 0x00, 0xC8, 0x03, 0x01, 0x03, 0x01, 0x04, 0x00, 0x01, 0x32, 0x04,
   0xB0, 0x01, 0x00, 0x01, 0x00, 0x8E, 0x04, 0x00, 0x00, 0x00, 0x90, 0x04, 0x00, 0x01, 0x00, 0xE8,
   0x03, 0x00, 0x02, 0x00, 0xC1, 0xC9, 0xC1, 0xCA, 0xB5, 0xCB, 0xD1, 0x42, 0xF9, 0x00, 0x00, 0x00,
   0xC7, 0x91, 0xCB, 0x24, 0x01, 0x00, 0x42, 0x37, 0x00, 0x00, 0x00, 0xBD, 0x10, 0x24, 0x01, 0x00,
   0xCE, 0xE9, 0xB6, 0xA9, 0xEA, 0x06, 0xBF, 0x00, 0xC6, 0x9D, 0xCA, 0xC6, 0x94, 0x00, 0xC7, 0xD1,
   0xE9, 0xA3, 0xEB, 0xD7, 0xC5, 0x28, 0xDE, 0x03, 0x3E, 0x08, 0x03, 0x0D, 0x0D, 0x0E, 0x76, 0x35,
   0x12, 0x21, 0x07, 0x02, 0x30, 0x0E, 0x43, 0x06, 0x00, 0xCA, 0x03, 0x01, 0x03, 0x01, 0x06, 0x00,
   0x00, 0x52, 0x04, 0xB0, 0x01, 0x00, 0x01, 0x00, 0x8E, 0x04, 0x00, 0x00, 0x00, 0xE8, 0x03, 0x00,
   0x01, 0x00, 0xF8, 0x03, 0x00, 0x02, 0x00, 0xC1, 0xC9, 0xB5, 0xCA, 0xC6, 0xD1, 0xE9, 0xA3, 0xEA,
   0x47, 0x38, 0x09, 0x01, 0x00, 0x00, 0xD1, 0x42, 0x0A, 0x01, 0x00, 0x00, 0xC6, 0xB6, 0x24, 0x02,
   0x00, 0xBD, 0x10, 0xF0, 0xB9, 0xA0, 0x38, 0x09, 0x01, 0x00, 0x00, 0xD1, 0x42, 0x0A, 0x01, 0x00,
   0x00, 0xC6, 0x8F, 0xCE, 0xB6, 0x24, 0x02, 0x00, 0xBD, 0x10, 0xF0, 0xAF, 0xCF, 0xBE, 0xFF, 0x00,
   0xAD, 0xCB, 0xC5, 0x38, 0x99, 0x00, 0x00, 0x00, 0x42, 0xF5, 0x00, 0x00, 0x00, 0xC7, 0x24, 0x01,
   0x00, 0x9D, 0xC9, 0x93, 0x01, 0xEC, 0xB5, 0xC5, 0x28, 0xDE, 0x03, 0x4A, 0x09, 0x03, 0x0D, 0x0D,
   0x21, 0xDA, 0x21, 0x58, 0x0D, 0x0D, 0x0E, 0x43, 0x06, 0x00, 0xCC, 0x03, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x01, 0x00, 0x29, 0xDE, 0x03, 0x56, 0x01, 0x03, 0x0E, 0x43, 0x06, 0x00, 0x00, 0x01,
   0x09, 0x01, 0x04, 0x00, 0x00, 0xE6, 0x01, 0x0A, 0xB0, 0x01, 0x00, 0x01, 0x00, 0x8E, 0x04, 0x00,
   0x00, 0x00, 0x90, 0x04, 0x00, 0x01, 0x00, 0x96, 0x04, 0x00, 0x02, 0x00, 0x98, 0x04, 0x00, 0x03,
   0x00, 0x9A, 0x04, 0x00, 0x04, 0x00, 0x9C, 0x04, 0x00, 0x05, 0x00, 0x9E, 0x04, 0x00, 0x06, 0x00,
   0xA0, 0x04, 0x00, 0x07, 0x00, 0xE8, 0x03, 0x00, 0x08, 0x00, 0xC1, 0xC9, 0xC1, 0xCC, 0xC1, 0xC3,
   0x07, 0xB5, 0xC3, 0x08, 0xD1, 0x42, 0xF9, 0x00, 0x00, 0x00, 0xC2, 0x08, 0x91, 0xC3, 0x08, 0x24,
   0x01, 0x00, 0xCA, 0xD1, 0x42, 0xF9, 0x00, 0x00, 0x00, 0xC2, 0x08, 0x91, 0xC3, 0x08, 0x24, 0x01,
   0x00, 0xCB, 0xD1, 0x42, 0xF9, 0x00, 0x00, 0x00, 0xC2, 0x08, 0x91, 0xC3, 0x08, 0x24, 0x01, 0x00,
   0xCC, 0xC6, 0xB7, 0xA1, 0xC3, 0x04, 0xC6, 0xB8, 0xAD, 0xB9, 0xA0, 0xC7, 0xB9, 0xA1, 0xAF, 0xC3,
   0x05, 0xC7, 0xBD, 0x0F, 0xAD, 0xB7, 0xA0, 0xC8, 0xBB, 0xA1, 0xAF, 0xC3, 0x06, 0xC8, 0xBD, 0x3F,
   0xAD, 0xC3, 0x07, 0x38, 0x11, 0x01, 0x00, 0x00, 0xC7, 0xEF, 0xEA, 0x09, 0xBD, 0x40, 0xC4, 0x07,
   0xC3, 0x06, 0xEC, 0x0E, 0x38, 0x11, 0x01, 0x00, 0x00, 0xC8, 0xEF, 0xEA, 0x05, 0xBD, 0x40, 0xC3,
   0x07, 0xC5, 0x38, 0xE6, 0x00, 0x00, 0x00, 0x41, 0xEA, 0x00, 0x00, 0x00, 0x42, 0x12, 0x01, 0x00,
   0x00, 0xC2, 0x04, 0x24, 0x01, 0x00, 0x9D, 0x38, 0xE6, 0x00, 0x00, 0x00, 0x41, 0xEA, 0x00, 0x00,
   0x00, 0x42, 0x12, 0x01, 0x00, 0x00, 0xC2, 0x05, 0x24, 0x01, 0x00, 0x9D, 0x38, 0xE6, 0x00, 0x00,
   0x00, 0x41, 0xEA, 0x00, 0x00, 0x00, 0x42, 0x12, 0x01, 0x00, 0x00, 0xC2, 0x06, 0x24, 0x01, 0x00,
   0x9D, 0x38, 0xE6, 0x00, 0x00, 0x00, 0x41, 0xEA, 0x00, 0x00, 0x00, 0x42, 0x12, 0x01, 0x00, 0x00,
   0xC2, 0x07, 0x24, 0x01, 0x00, 0x9D, 0xC9, 0xC1, 0xD0, 0xCF, 0xCA, 0xC1, 0xC4, 0x07, 0xC4, 0x06,
   0xC4, 0x05, 0xC3, 0x04, 0xC2, 0x08, 0xD1, 0xE9, 0xA3, 0x6A, 0x2A, 0xFF, 0xFF, 0xFF, 0xC5, 0x28,
   0xDE, 0x03, 0x59, 0x18, 0x03, 0x0D, 0x0D, 0x12, 0x13, 0x4E, 0x4E, 0x4E, 0x1C, 0x3A, 0x3F, 0x21,
   0x30, 0x21, 0x3A, 0x18, 0x08, 0x6C, 0x6C, 0x6C, 0x71, 0x17, 0x30, 0x35, 0x0E, 0x43, 0x06, 0x00,
   0xCE, 0x03, 0x01, 0x04, 0x01, 0x04, 0x00, 0x00, 0x84, 0x01, 0x05, 0xD0, 0x03, 0x00, 0x01, 0x00,
   0xB0, 0x01, 0x00, 0x00, 0x00, 0xA6, 0x04, 0x00, 0x01, 0x00, 0xA8, 0x04, 0x00, 0x02, 0x00, 0xAC,
   0x01, 0x00, 0x03, 0x00, 0xD1, 0xE9, 0xB6, 0xA3, 0xEA, 0x0D, 0x38, 0x15, 0x01, 0x00, 0x00, 0x04,
   0xEE, 0x00, 0x00, 0x00, 0xEF, 0x29, 0x38, 0x15, 0x01, 0x00, 0x00, 0x04, 0x16, 0x01, 0x00, 0x00,
   0xD1, 0xB5, 0x47, 0x9D, 0xEF, 0x0E, 0xD1, 0xB5, 0x47, 0xC9, 0x38, 0xE6, 0x00, 0x00, 0x00, 0x42,
   0xEB, 0x00, 0x00, 0x00, 0xC5, 0x24, 0x01, 0x00, 0xCA, 0x04, 0x17, 0x01, 0x00, 0x00, 0x11, 0x39,
   0x18, 0x01, 0x00, 0x00, 0x0E, 0x38, 0xE3, 0x00, 0x00, 0x00, 0xC6, 0x38, 0x18, 0x01, 0x00, 0x00,
   0xF0, 0xCF, 0xE9, 0xBD, 0x70, 0xAA, 0xEA, 0x0D, 0x38, 0x15, 0x01, 0x00, 0x00, 0x04, 0xEE, 0x00,
   0x00, 0x00, 0xEF, 0x29, 0x04, 0x19, 0x01, 0x00, 0x00, 0xCC, 0xC7, 0xC8, 0xA9, 0xEA, 0x0E, 0x38,
   0x15, 0x01, 0x00, 0x00, 0x04, 0x1A, 0x01, 0x00, 0x00, 0xEF, 0x0E, 0x29, 0x38, 0x15, 0x01, 0x00,
   0x00, 0x04, 0xEE, 0x00, 0x00, 0x00, 0xEF, 0x29, 0xDE, 0x03, 0x76, 0x11, 0x03, 0x21, 0x3B, 0x08,
   0x54, 0x17, 0x4F, 0x3F, 0x40, 0x26, 0x3B, 0x09, 0x21, 0x1D, 0x3F, 0x09, 0x3C
};


static JSContext *JS_NewCustomContext(JSRuntime *rt)
{
 JSContext *ctx = JS_NewContextRaw(rt);
 if (!ctx)
   return NULL;
```

```
 JS_AddIntrinsicBaseObjects(ctx);
 return ctx;
}


int main(int argc, char **argv)
{
 JSRuntime *rt;
 JSContext *ctx;
 rt = JS_NewRuntime();
 js_std_set_worker_new_context_func(JS_NewCustomContext);
 js_std_init_handlers(rt);
 ctx = JS_NewCustomContext(rt);
 js_std_add_helpers(ctx, argc, argv);
 js_std_eval_binary(ctx, qjsc_hello, qjsc_hello_size, 0);
 js_std_loop(ctx);
 JS_FreeContext(ctx);
 JS_FreeRuntime(rt);
 return 0;
}
```

编译dump出bytecode

```
0000:  02 3a                    58 atom indexes {
1"long2str"
1"str2long"
1"sdfsfsdf"
1"str2Hex"
1"hex2str"
1"xxxfss"
1"main"
1"args"
1"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/="
1"_keyStr"
1"dfsfs"
1"scriptArgs"
1"shift"
1"error"
1"./test_encode.js"
1"v"
1"w"
1"vl"
1"sl"
1"i"
1"fromCharCode"
1"substring"
1"s"
1"len"
1"charCodeAt"
1"str"
1"key"
1"k"
1"n"
1"z"
1"y"
1"delta"
1"mx"
1"e"
```

```
1"q"
1"sum"
1"p"
1"floor"
1"output"
1"chr1"
1"parseInt"
1"substr"
1"chr2"
1"chr3"
1"enc1"
1"enc2"
1"enc3"
1"enc4"
1"isNaN"
1"charAt"
1"dfsfdsfsd"
1"fwderf"
1"print"
1"your input: "
1"no_thing_is_true"
1"dfdfwf3"
1"05aed0ce441f80b5bc36af4c698509fc6cc3c97146353de5a95c6abea07fd4a7070932d86ac32d628672a59123e5972331db5dffe
1"yes"
0002:  10 6c 6f 6e 67 32 73 74
      72 10 73 74 72 32 6c 6f
      6e 67 10 73 64 66 73 66
      73 64 66 0e 73 74 72 32
      48 65 78 0e 68 65 78 32
      73 74 72 0c 78 78 78 66
      73 73 08 6d 61 69 6e 08
      61 72 67 73 82 01 41 42
      43 44 45 46 47 48 49 4a
      4b 4c 4d 4e 4f 50 51 52
      53 54 55 56 57 58 59 5a
      61 62 63 64 65 66 67 68
      69 6a 6b 6c 6d 6e 6f 70
      71 72 73 74 75 76 77 78
      79 7a 30 31 32 33 34 35
      36 37 38 39 2b 2f 3d 0e
      5f 6b 65 79 53 74 72 0a
      64 66 73 66 73 14 73 63
      72 69 70 74 41 72 67 73
      0a 73 68 69 66 74 0a 65
      72 72 6f 72 20 2e 2f 74
      65 73 74 5f 65 6e 63 6f
      64 65 2e 6a 73 02 76 02
      77 04 76 6c 04 73 6c 02
      69 18 66 72 6f 6d 43 68
      61 72 43 6f 64 65 12 73
      75 62 73 74 72 69 6e 67
      02 73 06 6c 65 6e 14 63
      68 61 72 43 6f 64 65 41
      74 06 73 74 72 06 6b 65
      79 02 6b 02 6e 02 7a 02
      79 0a 64 65 6c 74 61 04
      6d 78 02 65 02 71 06 73
      75 6d 02 70 0a 66 6c 6f
      6f 72 0c 6f 75 74 70 75
      74 08 63 68 72 31 10 70
```

```
      61 72 73 65 49 6e 74 0c
      73 75 62 73 74 72 08 63
      68 72 32 08 63 68 72 33
      08 65 6e 63 31 08 65 6e
      63 32 08 65 6e 63 33 08
      65 6e 63 34 0a 69 73 4e
      61 4e 0c 63 68 61 72 41
      74 12 64 66 73 66 64 73
      66 73 64 0c 66 77 64 65
      72 66 0a 70 72 69 6e 74
      18 79 6f 75 72 20 69 6e
      70 75 74 3a 20 20 6e 6f
      5f 74 68 69 6e 67 5f 69
      73 5f 74 72 75 65 0e 64
      66 64 66 77 66 33 e0 01
      30 35 61 65 64 30 63 65
      34 34 31 66 38 30 62 35
      62 63 33 36 61 66 34 63
      36 39 38 35 30 39 66 63
      36 63 63 33 63 39 37 31
      34 36 33 35 33 64 65 35
      61 39 35 63 36 61 62 65
      61 30 37 66 64 34 61 37
      30 37 30 39 33 32 64 38
      36 61 63 33 32 64 36 32
      38 36 37 32 61 35 39 31
      32 33 65 35 39 37 32 33
      33 31 64 62 35 64 66 66
      65 37 30 35 37 33 36 32
      06 79 65 73             }
020e:  0e                       function {
020f:  00 06 00 a0 01 00 01 00
      03 00 08 e4 01 01          name: "<eval>"
                                 args=0 vars=1 defargs=0 closures=0 cpool=8
                                 stack=3 bclen=228 locals=1
                                 vars {
021d:  a2 01 00 00 00               name: "<ret>"
                                 }
                                 bytecode {
0222:  3f e1 00 00 00 40 3f e2
      00 00 00 40 3f e3 00 00
      00 40 3f e4 00 00 00 40
      3f e5 00 00 00 40 3f e6
      00 00 00 40 3f e7 00 00
      00 40 3f e8 00 00 00 00
      c0 00 40 e1 00 00 00 00
      c0 01 40 e2 00 00 00 00
      c0 02 40 e3 00 00 00 00
      c0 03 40 e4 00 00 00 00
      c0 04 40 e5 00 00 00 00
      c0 05 40 e6 00 00 00 00
      c0 07 40 e7 00 00 00 00
      3e e8 00 00 00 00 38 e6
      00 00 00 04 e9 00 00 00
      15 43 ea 00 00 00 c9 38
      e6 00 00 00 c0 06 15 43
      eb 00 00 00 c9 06 c9 37
      ec 00 00 00 f4 eb 1d 38
      ec 00 00 00 11 39 e8 00
```

```
      00 00 c9 38 e8 00 00 00
      42 ed 00 00 00 24 00 00
      c9 ec 25 06 c9 37 4d 00
      00 00 f4 eb 0f 38 4d 00
      00 00 11 39 e8 00 00 00
      c9 ec 0d 04 ee 00 00 00
      11 39 e8 00 00 00 c9 38
      e7 00 00 00 38 e8 00 00
      00 ef cd 28                  at 1, fixup atom: long2str
                                   at 7, fixup atom: str2long
                                   at 13, fixup atom: sdfsfsdf
                                   at 19, fixup atom: str2Hex
                                   at 25, fixup atom: hex2str
                                   at 31, fixup atom: xxxfss
                                   at 37, fixup atom: main
                                   at 43, fixup atom: args
                                   at 51, fixup atom: long2str
                                   at 59, fixup atom: str2long
                                   at 67, fixup atom: sdfsfsdf
                                   at 75, fixup atom: str2Hex
                                   at 83, fixup atom: hex2str
                                   at 91, fixup atom: xxxfss
                                   at 99, fixup atom: main
                                   at 105, fixup atom: args
                                   at 111, fixup atom: xxxfss
                                   at 116, fixup atom: "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxy
                                   at 122, fixup atom: _keyStr
                                   at 128, fixup atom: xxxfss
                                   at 136, fixup atom: dfsfs
                                   at 144, fixup atom: scriptArgs
                                   at 152, fixup atom: scriptArgs
                                   at 158, fixup atom: args
                                   at 164, fixup atom: args
                                   at 169, fixup atom: shift
                                   at 182, fixup atom: arguments
                                   at 190, fixup atom: arguments
                                   at 196, fixup atom: args
                                   at 204, fixup atom: error
                                   at 210, fixup atom: args
                                   at 216, fixup atom: main
                                   at 221, fixup atom: args
                                }
                                 debug {
0306:  de 03 01 13 f1 00 3e b0
      01 58 00 05 36 00 09 40
      35 3f 49 3f 3f 0d 41         filename: "./test_encode.js"
                                }
                                 cpool {
031d:  0e                           function {
031e:  43 06 00 c2 03 02 03 02
      09 00 01 6e 05                  name: long2str
                                      args=2 vars=3 defargs=2 closures=0 cpool=1
                                      stack=9 bclen=110 locals=5
                                      vars {
032b:  e0 03 00 01 00                    name: v
0330:  e2 03 00 01 00                    name: w
0335:  e4 03 00 00 00                    name: vl
033a:  e6 03 00 01 00                    name: sl
033f:  e8 03 00 02 00                    name: i
                                      }
```

```
                                    }
                                     bytecode {
0344:  d1 e9 c9 d1 c5 b6 9e 47
       bf 00 ad ca b5 cb c7 c5
       a3 ea 3b d1 c7 71 38 99
       00 00 00 42 f5 00 00 00
       d1 c7 47 be ff 00 ad d1
       c7 47 bd 08 a2 be ff 00
       ad d1 c7 47 bd 10 a2 be
       ff 00 ad d1 c7 47 bd 18
       a2 be ff 00 ad 24 04 00
       49 93 02 ec c2 d2 ea 15
       d1 42 5b 00 00 00 c1 24
       01 00 42 f6 00 00 00 b5
       c6 25 02 00 d1 42 5b 00
       00 00 c1 25 01 00               at 23, fixup atom: String
                                       at 28, fixup atom: fromCharCode
                                       at 82, fixup atom: join
                                       at 91, fixup atom: substring
                                       at 102, fixup atom: join
                                    }
                                     debug {
03b2:  de 03 01 0b 03 12 30 27
       67 35 35 49 17 12 69            filename: "./test_encode.js"
                                    }
                                     cpool {
03c1:  06                               float64 {
03c2:  00 00 e0 ff ff ff ef 41            4.29497e+09
                                      }
                                    }
./test_encode.js:1: function: long2str
 args: v w
 locals:
   0: var vl
   1: var sl
   2: var i
 stack_size: 9
 opcodes:
       get_arg0 0: v
       get_length
       put_loc0 0: vl
       get_arg0 0: v
       get_loc0 0: vl
       push_1 1
       sub
       get_array_el
       push_const8 0: 4294967295
       and
       put_loc1 1: sl
       push_0 0
       put_loc2 2: i
  14:  get_loc2 2: i
       get_loc0 0: vl
       lt
       if_false8 77
       get_arg0 0: v
       get_loc2 2: i
       to_propkey2
       get_var String
       get_field2 fromCharCode
```

```
       get_arg0 0: v
       get_loc2 2: i
       get_array_el
       push_i16 255
       and
       get_arg0 0: v
       get_loc2 2: i
       get_array_el
       push_i8 8
       shr
       push_i16 255
       and
       get_arg0 0: v
       get_loc2 2: i
       get_array_el
       push_i8 16
       shr
       push_i16 255
       and
       get_arg0 0: v
       get_loc2 2: i
       get_array_el
       push_i8 24
       shr
       push_i16 255
       and
       call_method 4
       put_array_el
       inc_loc 2: i
       goto8 14
  77:  get_arg1 1: w
       if_false8 100
       get_arg0 0: v
       get_field2 join
       push_empty_string
       call_method 1
       get_field2 substring
       push_0 0
       get_loc1 1: sl
       tail_call_method 2
 100:  get_arg0 0: v
       get_field2 join
       push_empty_string
       tail_call_method 1



                                    }
03ca:  0e                             function {
03cb:  43 06 00 c4 03 02 03 02
       07 00 00 5d 05                  name: str2long
                                       args=2 vars=3 defargs=2 closures=0 cpool=0
                                       stack=7 bclen=93 locals=5
                                       vars {
03d8:  ee 03 00 01 00                     name: s
03dd:  e2 03 00 01 00                     name: w
03e2:  f0 03 00 00 00                     name: len
03e7:  e0 03 00 01 00                     name: v
03ec:  e8 03 00 02 00                     name: i
                                       }
                                       bytecode {
```

```
                                     bytecode {
03f1:  d1 e9 c9 26 00 00 ca b5
       cb c7 c5 a3 ea 46 c6 c7
       b7 a1 71 d1 42 f9 00 00
       00 c7 24 01 00 d1 42 f9
       00 00 00 c7 b6 9d 24 01
       00 bd 08 a0 af d1 42 f9
       00 00 00 c7 b7 9d 24 01
       00 bd 10 a0 af d1 42 f9
       00 00 00 c7 b8 9d 24 01
       00 bd 18 a0 af 49 b9 94
       02 ec b7 d2 ea 06 c6 c6
       e9 c5 49 c6 28                  at 21, fixup atom: charCodeAt
                                       at 31, fixup atom: charCodeAt
                                       at 47, fixup atom: charCodeAt
                                       at 63, fixup atom: charCodeAt
                                     }
                                      debug {
044e:  de 03 13 0b 03 12 17 27
       4e 53 53 58 1c 12 1d            filename: "./test_encode.js"
                                     }
./test_encode.js:19: function: str2long
 args: s w
 locals:
   0: var len
   1: var v
   2: var i
 stack_size: 7
 opcodes:
       get_arg0 0: s
       get_length
       put_loc0 0: len
       array_from 0
       put_loc1 1: v
       push_0 0
       put_loc2 2: i
   9:  get_loc2 2: i
       get_loc0 0: len
       lt
       if_false8 83
       get_loc1 1: v
       get_loc2 2: i
       push_2 2
       sar
       to_propkey2
       get_arg0 0: s
       get_field2 charCodeAt
       get_loc2 2: i
       call_method 1
       get_arg0 0: s
       get_field2 charCodeAt
       get_loc2 2: i
       push_1 1
       add
       call_method 1
       push_i8 8
       shl
       or
       get_arg0 0: s
       get_field2 charCodeAt
```

```
       get_loc2 2: i
       push_2 2
       add
       call_method 1
       push_i8 16
       shl
       or
       get_arg0 0: s
       get_field2 charCodeAt
       get_loc2 2: i
       push_3 3
       add
       call_method 1
       push_i8 24
       shl
       or
       put_array_el
       push_4 4
       add_loc 2: i
       goto8 9
  83:  get_arg1 1: w
       if_false8 91
       get_loc1 1: v
       get_loc1 1: v
       get_length
       get_loc0 0: len
       put_array_el
  91:  get_loc1 1: v
       return



                                  }
045d:  0e                           function {
045e:  43 06 00 c6 03 02 0b 02
       06 00 04 f8 01 0d             name: sdfsfsdf
                                     args=2 vars=11 defargs=2 closures=0 cpool=4
                                     stack=6 bclen=248 locals=13
                                     vars {
046c:  f4 03 00 01 00                   name: str
0471:  f6 03 00 01 00                   name: key
0476:  e0 03 00 00 00                   name: v
047b:  f8 03 00 01 00                   name: k
0480:  fa 03 00 02 00                   name: n
0485:  fc 03 00 03 00                   name: z
048a:  fe 03 00 04 00                   name: y
048f:  80 04 00 05 00                   name: delta
0494:  82 04 00 06 00                   name: mx
0499:  84 04 00 07 00                   name: e
049e:  86 04 00 08 00                   name: q
04a3:  88 04 00 09 00                   name: sum
04a8:  8a 04 00 0a 00                   name: p
                                     }
                                     bytecode {
04ad:  d1 c1 a9 ea 03 c1 28 38
       e2 00 00 00 d1 0a f0 c9
       38 e2 00 00 00 d2 09 f0
       ca c5 e9 b6 9e cb c5 c7
       47 cc c5 b5 47 c3 04 bf
       00 c3 05 38 9d 00 00 00
       42 06 01 00 00 bb bd 34
```

```
      42 06 01 00 00 bb bd 34
      c7 b6 9d 9b 9d 24 01 00
      c3 08 b5 c3 09 c2 08 90
      c3 08 b5 a5 69 9b 00 00
      00 c2 09 c2 05 9d bf 01
      ad c4 09 b7 a2 b8 ad c3
      07 b5 c3 0a c2 0a c7 a3
      ea 43 c5 c2 0a b6 9d 47
      c3 04 c8 ba a2 c2 04 b7
      a0 ae c2 04 b8 a2 c8 b9
      a0 ae 9d c2 09 c2 04 ae
      c6 c2 0a b8 ad c2 07 ae
      47 c8 ae 9d ae c3 06 c5
      c2 0a 71 c5 c2 0a 47 c2
      06 9d bf 02 ad 16 49 cc
      93 0a ec b9 c5 b5 47 c3
      04 c8 ba a2 c2 04 b7 a0
      ae c2 04 b8 a2 c8 b9 a0
      ae 9d c2 09 c2 04 ae c6
      c2 0a b8 ad c2 07 ae 47
      c8 ae 9d ae c3 06 c5 c7
      71 c5 c7 47 c2 06 9d bf
      03 ad 16 49 cc ed 5f ff
      38 e4 00 00 00 38 e1 00
      00 00 c5 09 f0 23 01 00          at 8, fixup atom: str2long
                                       at 17, fixup atom: str2long
                                       at 44, fixup atom: Math
                                       at 49, fixup atom: floor
                                       at 233, fixup atom: str2Hex
                                       at 238, fixup atom: long2str
                                     }
                                      debug {
05a5:  de 03 23 15 03 1c 08 08
      30 30 1d 44 85 3f 35 21
      30 2b bc 58 17 1c bc 4e
      13                               filename: "./test_encode.js"
                                     }
                                      cpool {
05be:  06                                float64 {
05bf:  00 00 20 37 ef c6 e3 41             2.65444e+09
                                       }
05c7:  06                                float64 {
05c8:  00 00 e0 ff ff ff ef 41             4.29497e+09
                                       }
05d0:  06                                float64 {
05d1:  00 00 e0 ff ff ff ef 41             4.29497e+09
                                       }
05d9:  06                                float64 {
05da:  00 00 e0 ff ff ff ef 41             4.29497e+09
                                       }
                                     }
./test_encode.js:35: function: sdfsfsdf
 args: str key
 locals:
   0: var v
   1: var k
   2: var n
   3: var z
   4: var y
   5: var delta
```

```
  6: var mx
  7: var e
  8: var q
  9: var sum
 10: var p
stack_size: 6
opcodes:
      get_arg0 0: str
      push_empty_string
      eq
      if_false8 7
      push_empty_string
      return
  7:  get_var str2long
      get_arg0 0: str
      push_true
      call2 2
      put_loc0 0: v
      get_var str2long
      get_arg1 1: key
      push_false
      call2 2
      put_loc1 1: k
      get_loc0 0: v
      get_length
      push_1 1
      sub
      put_loc2 2: n
      get_loc0 0: v
      get_loc2 2: n
      get_array_el
      put_loc3 3: z
      get_loc0 0: v
      push_0 0
      get_array_el
      put_loc8 4: y
      push_const8 0: 2654435769
      put_loc8 5: delta
      get_var Math
      get_field2 floor
      push_6 6
      push_i8 52
      get_loc2 2: n
      push_1 1
      add
      div
      add
      call_method 1
      put_loc8 8: q
      push_0 0
      put_loc8 9: sum
 69:  get_loc8 8: q
      post_dec
      put_loc8 8: q
      push_0 0
      gt
      if_false 232
      get_loc8 9: sum
      get_loc8 5: delta
      add
```

```
       add
       push_const8 1: 4294967295
       and
       set_loc8 9: sum
       push_2 2
       shr
       push_3 3
       and
       put_loc8 7: e
       push_0 0
       put_loc8 10: p
100:   get_loc8 10: p
       get_loc2 2: n
       lt
       if_false8 172
       get_loc0 0: v
       get_loc8 10: p
       push_1 1
       add
       get_array_el
       put_loc8 4: y
       get_loc3 3: z
       push_5 5
       shr
       get_loc8 4: y
       push_2 2
       shl
       xor
       get_loc8 4: y
       push_3 3
       shr
       get_loc3 3: z
       push_4 4
       shl
       xor
       add
       get_loc8 9: sum
       get_loc8 4: y
       xor
       get_loc1 1: k
       get_loc8 10: p
       push_3 3
       and
       get_loc8 7: e
       xor
       get_array_el
       get_loc3 3: z
       xor
       add
       xor
       put_loc8 6: mx
       get_loc0 0: v
       get_loc8 10: p
       to_propkey2
       get_loc0 0: v
       get_loc8 10: p
       get_array_el
       get_loc8 6: mx
       add
       push_const8 2: 4294967295
```

```
      and
      insert3
      put_array_el
      put_loc3 3: z
      inc_loc 10: p
      goto8 100
172:  get_loc0 0: v
      push_0 0
      get_array_el
      put_loc8 4: y
      get_loc3 3: z
      push_5 5
      shr
      get_loc8 4: y
      push_2 2
      shl
      xor
      get_loc8 4: y
      push_3 3
      shr
      get_loc3 3: z
      push_4 4
      shl
      xor
      add
      get_loc8 9: sum
      get_loc8 4: y
      xor
      get_loc1 1: k
      get_loc8 10: p
      push_3 3
      and
      get_loc8 7: e
      xor
      get_array_el
      get_loc3 3: z
      xor
      add
      xor
      put_loc8 6: mx
      get_loc0 0: v
      get_loc2 2: n
      to_propkey2
      get_loc0 0: v
      get_loc2 2: n
      get_array_el
      get_loc8 6: mx
      add
      push_const8 3: 4294967295
      and
      insert3
      put_array_el
      put_loc3 3: z
      goto16 69
232:  get_var str2Hex
      get_var long2str
      get_loc0 0: v
      push_false
      call2 2
```

```
       tail_call 1


                                    }
05e2:  0e                             function {
05e3:  43 06 00 c8 03 01 03 01
       04 00 01 32 04                  name: str2Hex
                                       args=1 vars=3 defargs=1 closures=0 cpool=1
                                       stack=4 bclen=50 locals=4
                                       vars {
05f0:  b0 01 00 01 00                     name: input
05f5:  8e 04 00 00 00                     name: output
05fa:  90 04 00 01 00                     name: chr1
05ff:  e8 03 00 02 00                     name: i
                                    }
                                       bytecode {
0604:  c1 c9 c1 ca b5 cb d1 42
       f9 00 00 00 c7 91 cb 24
       01 00 42 37 00 00 00 bd
       10 24 01 00 ce e9 b6 a9
       ea 06 bf 00 c6 9d ca c6
       94 00 c7 d1 e9 a3 eb d7
       c5 28                          at 8, fixup atom: charCodeAt
                                      at 19, fixup atom: toString
                                    }
                                       debug {
0636:  de 03 3e 08 03 0d 0d 0e
       76 35 12 21                    filename: "./test_encode.js"
                                    }
                                       cpool {
0642:  07                                string {
1"0"
0643:  02 30                           }
                                    }
./test_encode.js:62: function: str2Hex
 args: input
 locals:
   0: var output
   1: var chr1
   2: var i
 stack_size: 4
 opcodes:
       push_empty_string
       put_loc0 0: output
       push_empty_string
       put_loc1 1: chr1
       push_0 0
       put_loc2 2: i
   6:  get_arg0 0: input
       get_field2 charCodeAt
       get_loc2 2: i
       post_inc
       put_loc2 2: i
       call_method 1
       get_field2 toString
       push_i8 16
       call_method 1
       set_loc1 1: chr1
       get_length
       push 1 1
```

```
       eq
       if_false8 39
       push_const8 0: 1"0"
       get_loc1 1: chr1
       add
       put_loc1 1: chr1
  39:  get_loc1 1: chr1
       add_loc 0: output
       get_loc2 2: i
       get_arg0 0: input
       get_length
       lt
       if_true8 6
       get_loc0 0: output
       return


                                  }
0645:  0e                           function {
0646:  43 06 00 ca 03 01 03 01
       06 00 00 52 04                name: hex2str
                                     args=1 vars=3 defargs=1 closures=0 cpool=0
                                     stack=6 bclen=82 locals=4
                                     vars {
0653:  b0 01 00 01 00                   name: input
0658:  8e 04 00 00 00                   name: output
065d:  e8 03 00 01 00                   name: i
0662:  f8 03 00 02 00                   name: k
                                     }
                                     bytecode {
0667:  c1 c9 b5 ca c6 d1 e9 a3
       ea 47 38 09 01 00 00 d1
       42 0a 01 00 00 c6 b6 24
       02 00 bd 10 f0 b9 a0 38
       09 01 00 00 d1 42 0a 01
       00 00 c6 8f ce b6 24 02
       00 bd 10 f0 af cf be ff
       00 ad cb c5 38 99 00 00
       00 42 f5 00 00 00 c7 24
       01 00 9d c9 93 01 ec b5
       c5 28                            at 11, fixup atom: parseInt
                                        at 17, fixup atom: substr
                                        at 32, fixup atom: parseInt
                                        at 38, fixup atom: substr
                                        at 61, fixup atom: String
                                        at 66, fixup atom: fromCharCode
                                     }
                                     debug {
06b9:  de 03 4a 09 03 0d 0d 21
       da 21 58 0d 0d                   filename: "./test_encode.js"
                                     }
./test_encode.js:74: function: hex2str
 args: input
 locals:
   0: var output
   1: var i
   2: var k
 stack_size: 6
 opcodes:
```

```
       push_empty_string
       put_loc0 0: output
       push_0 0
       put_loc1 1: i
   4:  get_loc1 1: i
       get_arg0 0: input
       get_length
       lt
       if_false8 80
       get_var parseInt
       get_arg0 0: input
       get_field2 substr
       get_loc1 1: i
       push_1 1
       call_method 2
       push_i8 16
       call2 2
       push_4 4
       shl
       get_var parseInt
       get_arg0 0: input
       get_field2 substr
       get_loc1 1: i
       inc
       set_loc1 1: i
       push_1 1
       call_method 2
       push_i8 16
       call2 2
       or
       set_loc2 2: k
       push_i16 255
       and
       put_loc2 2: k
       get_loc0 0: output
       get_var String
       get_field2 fromCharCode
       get_loc2 2: k
       call_method 1
       add
       put_loc0 0: output
       inc_loc 1: i
       goto8 4
  80:  get_loc0 0: output
       return



                                  }
06c6:  0e                           function {
06c7:  43 06 00 cc 03 00 00 00
       00 00 00 01 00                name: xxxfss
                                     args=0 vars=0 defargs=0 closures=0 cpool=0
                                     stack=0 bclen=1 locals=0
                                     bytecode {
06d4:  29                             }
                                     debug {
06d5:  de 03 56 01 03                   filename: "./test_encode.js"
                                     }
./test_encode.js:86: function: xxxfss
 stack size: 0
```

```
 opcodes:
       return_undef



                                    }
06da:  0e                            function {
06db:  43 06 00 00 01 09 01 04
       00 00 e6 01 0a                 name: "<null>"
                                      args=1 vars=9 defargs=1 closures=0 cpool=0
                                      stack=4 bclen=230 locals=10
                                      vars {
06e8:  b0 01 00 01 00                    name: input
06ed:  8e 04 00 00 00                    name: output
06f2:  90 04 00 01 00                    name: chr1
06f7:  96 04 00 02 00                    name: chr2
06fc:  98 04 00 03 00                    name: chr3
0701:  9a 04 00 04 00                    name: enc1
0706:  9c 04 00 05 00                    name: enc2
070b:  9e 04 00 06 00                    name: enc3
0710:  a0 04 00 07 00                    name: enc4
0715:  e8 03 00 08 00                    name: i
                                      }
                                      bytecode {
071a:  c1 c9 c1 cc c1 c3 07 b5
       c3 08 d1 42 f9 00 00 00
       c2 08 91 c3 08 24 01 00
       ca d1 42 f9 00 00 00 c2
       08 91 c3 08 24 01 00 cb
       d1 42 f9 00 00 00 c2 08
       91 c3 08 24 01 00 cc c6
       b7 a1 c3 04 c6 b8 ad b9
       a0 c7 b9 a1 af c3 05 c7
       bd 0f ad b7 a0 c8 bb a1
       af c3 06 c8 bd 3f ad c3
       07 38 11 01 00 00 c7 ef
       ea 09 bd 40 c4 07 c3 06
       ec 0e 38 11 01 00 00 c8
       ef ea 05 bd 40 c3 07 c5
       38 e6 00 00 00 41 ea 00
       00 00 42 12 01 00 00 c2
       04 24 01 00 9d 38 e6 00
       00 00 41 ea 00 00 00 42
       12 01 00 00 c2 05 24 01
       00 9d 38 e6 00 00 00 41
       ea 00 00 00 42 12 01 00
       00 c2 06 24 01 00 9d 38
       e6 00 00 00 41 ea 00 00
       00 42 12 01 00 00 c2 07
       24 01 00 9d c9 c1 d0 cf
       ca c1 c4 07 c4 06 c4 05
       c3 04 c2 08 d1 e9 a3 6a
       2a ff ff ff c5 28                at 12, fixup atom: charCodeAt
                                        at 27, fixup atom: charCodeAt
                                        at 42, fixup atom: charCodeAt
                                        at 90, fixup atom: isNaN
                                        at 107, fixup atom: isNaN
                                        at 121, fixup atom: xxxfss
                                        at 126, fixup atom: _keyStr
                                        at 131, fixup atom: charAt
```

```
                                      at 142, fixup atom: xxxfss
                                      at 147, fixup atom: _keyStr
                                      at 152, fixup atom: charAt
                                      at 163, fixup atom: xxxfss
                                      at 168, fixup atom: _keyStr
                                      at 173, fixup atom: charAt
                                      at 184, fixup atom: xxxfss
                                      at 189, fixup atom: _keyStr
                                      at 194, fixup atom: charAt
                                    }
                                     debug {
0800:  de 03 59 18 03 0d 0d 12
      13 4e 4e 4e 1c 3a 3f 21
      30 21 3a 18 08 6c 6c 6c
      71 17 30 35                      filename: "./test_encode.js"
                                    }
./test_encode.js:89: function: <null>
 args: input
 locals:
   0: var output
   1: var chr1
   2: var chr2
   3: var chr3
   4: var enc1
   5: var enc2
   6: var enc3
   7: var enc4
   8: var i
 stack_size: 4
 opcodes:
       push_empty_string
       put_loc0 0: output
       push_empty_string
       put_loc3 3: chr3
       push_empty_string
       put_loc8 7: enc4
       push_0 0
       put_loc8 8: i
  10:  get_arg0 0: input
       get_field2 charCodeAt
       get_loc8 8: i
       post_inc
       put_loc8 8: i
       call_method 1
       put_loc1 1: chr1
       get_arg0 0: input
       get_field2 charCodeAt
       get_loc8 8: i
       post_inc
       put_loc8 8: i
       call_method 1
       put_loc2 2: chr2
       get_arg0 0: input
       get_field2 charCodeAt
       get_loc8 8: i
       post_inc
       put_loc8 8: i
       call_method 1
       put_loc3 3: chr3
       get_loc1 1: chr1
```

```
      get_loc1 1: chr1
      push_2 2
      sar
      put_loc8 4: enc1
      get_loc1 1: chr1
      push_3 3
      and
      push_4 4
      shl
      get_loc2 2: chr2
      push_4 4
      sar
      or
      put_loc8 5: enc2
      get_loc2 2: chr2
      push_i8 15
      and
      push_2 2
      shl
      get_loc3 3: chr3
      push_6 6
      sar
      or
      put_loc8 6: enc3
      get_loc3 3: chr3
      push_i8 63
      and
      put_loc8 7: enc4
      get_var isNaN
      get_loc2 2: chr2
      call1 1
      if_false8 106
      push_i8 64
      set_loc8 7: enc4
      put_loc8 6: enc3
      goto8 119
106:  get_var isNaN
      get_loc3 3: chr3
      call1 1
      if_false8 119
      push_i8 64
      put_loc8 7: enc4
119:  get_loc0 0: output
      get_var xxxfss
      get_field _keyStr
      get_field2 charAt
      get_loc8 4: enc1
      call_method 1
      add
      get_var xxxfss
      get_field _keyStr
      get_field2 charAt
      get_loc8 5: enc2
      call_method 1
      add
      get_var xxxfss
      get_field _keyStr
      get_field2 charAt
      get_loc8 6: enc3
      call_method 1
```

```
       add
       get_var xxxfss
       get_field _keyStr
       get_field2 charAt
       get_loc8 7: enc4
       call_method 1
       add
       put_loc0 0: output
       push_empty_string
       set_loc3 3: chr3
       set_loc2 2: chr2
       put_loc1 1: chr1
       push_empty_string
       set_loc8 7: enc4
       set_loc8 6: enc3
       set_loc8 5: enc2
       put_loc8 4: enc1
       get_loc8 8: i
       get_arg0 0: input
       get_length
       lt
       if_true 10
       get_loc0 0: output
       return


                                  }
081c:  0e                           function {
081d:  43 06 00 ce 03 01 04 01
      04 00 00 84 01 05              name: main
                                     args=1 vars=4 defargs=1 closures=0 cpool=0
                                     stack=4 bclen=132 locals=5
                                     vars {
082b:  d0 03 00 01 00                   name: args
0830:  b0 01 00 00 00                   name: input
0835:  a6 04 00 01 00                   name: dfsfdsfsd
083a:  a8 04 00 02 00                   name: fwderf
083f:  ac 01 00 03 00                   name: target
                                     }
                                     bytecode {
0844:  d1 e9 b6 a3 ea 0d 38 15
      01 00 00 04 ee 00 00 00
      ef 29 38 15 01 00 00 04
      16 01 00 00 d1 b5 47 9d
      ef 0e d1 b5 47 c9 38 e6
      00 00 00 42 eb 00 00 00
      c5 24 01 00 ca 04 17 01
      00 00 11 39 18 01 00 00
      0e 38 e3 00 00 00 c6 38
      18 01 00 00 f0 cf e9 bd
      70 aa ea 0d 38 15 01 00
      00 04 ee 00 00 00 ef 29
      04 19 01 00 00 cc c7 c8
      a9 ea 0e 38 15 01 00 00
      04 1a 01 00 00 ef 0e 29
      38 15 01 00 00 04 ee 00
      00 00 ef 29                      at 7, fixup atom: print
                                       at 12, fixup atom: error
                                       at 19, fixup atom: print
                                       at 24, fixup atom: "your input: "
```

```
                                      at 24, fixup atom: "your input: "
                                      at 39, fixup atom: xxxfss
                                      at 44, fixup atom: dfsfs
                                      at 54, fixup atom: no_thing_is_true
                                      at 60, fixup atom: dfdfwf3
                                      at 66, fixup atom: sdfsfsdf
                                      at 72, fixup atom: dfdfwf3
                                      at 85, fixup atom: print
                                      at 90, fixup atom: error
                                      at 97, fixup atom: "05aed0ce441f80b5bc36af4c698509fc6cc3c97146353de5
                                      at 108, fixup atom: print
                                      at 113, fixup atom: yes
                                      at 121, fixup atom: print
                                      at 126, fixup atom: error
                                    }
                                     debug {
08c8:  de 03 76 11 03 21 3b 08
       54 17 4f 3f 40 26 3b 09
       21 1d 3f 09 3c                 filename: "./test_encode.js"
                                    }
./test_encode.js:118: function: main
 args: args
 locals:
   0: var input
   1: var dfsfdsfsd
   2: var fwderf
   3: var target
 stack_size: 4
 opcodes:
       get_arg0 0: args
       get_length
       push_1 1
       lt
       if_false8 18
       get_var print
       push_atom_value error
       call1 1
       return_undef
  18:  get_var print
       push_atom_value "your input: "
       get_arg0 0: args
       push_0 0
       get_array_el
       add
       call1 1
       drop
       get_arg0 0: args
       push_0 0
       get_array_el
       put_loc0 0: input
       get_var xxxfss
       get_field2 dfsfs
       get_loc0 0: input
       call_method 1
       put_loc1 1: dfsfdsfsd
       push_atom_value no_thing_is_true
       dup
       put_var dfdfwf3
       drop
       get_var sdfsfsdf
```

```
       get_loc1 1: dfsfdsfsd
       get_var dfdfwf3
       call2 2
       set_loc2 2: fwderf
       get_length
       push_i8 112
       neq
       if_false8 96
       get_var print
       push_atom_value error
       call1 1
       return_undef
  96:  push_atom_value "05aed0ce441f80b5bc36af4c698509fc6cc3c97146353de5a95c6abea07fd4a7070932d86ac32d62867
       put_loc3 3: target
       get_loc2 2: fwderf
       get_loc3 3: target
       eq
       if_false8 120
       get_var print
       push_atom_value yes
       call1 1
       drop
       return_undef
 120:  get_var print
       push_atom_value error
       call1 1
       return_undef



                                  }
                                 }
./test_encode.js:1: function: <eval>
 locals:
   0: var <ret>
 stack_size: 3
 opcodes:
       check_define_var long2str,64
       check_define_var str2long,64
       check_define_var sdfsfsdf,64
       check_define_var str2Hex,64
       check_define_var hex2str,64
       check_define_var xxxfss,64
       check_define_var main,64
       check_define_var args,0
       fclosure8 0: [bytecode long2str]
       define_func long2str,0
       fclosure8 1: [bytecode str2long]
       define_func str2long,0
       fclosure8 2: [bytecode sdfsfsdf]
       define_func sdfsfsdf,0
       fclosure8 3: [bytecode str2Hex]
       define_func str2Hex,0
       fclosure8 4: [bytecode hex2str]
       define_func hex2str,0
       fclosure8 5: [bytecode xxxfss]
       define_func xxxfss,0
       fclosure8 7: [bytecode main]
       define_func main,0
       define_var args,0
       get_var xxxfss
```

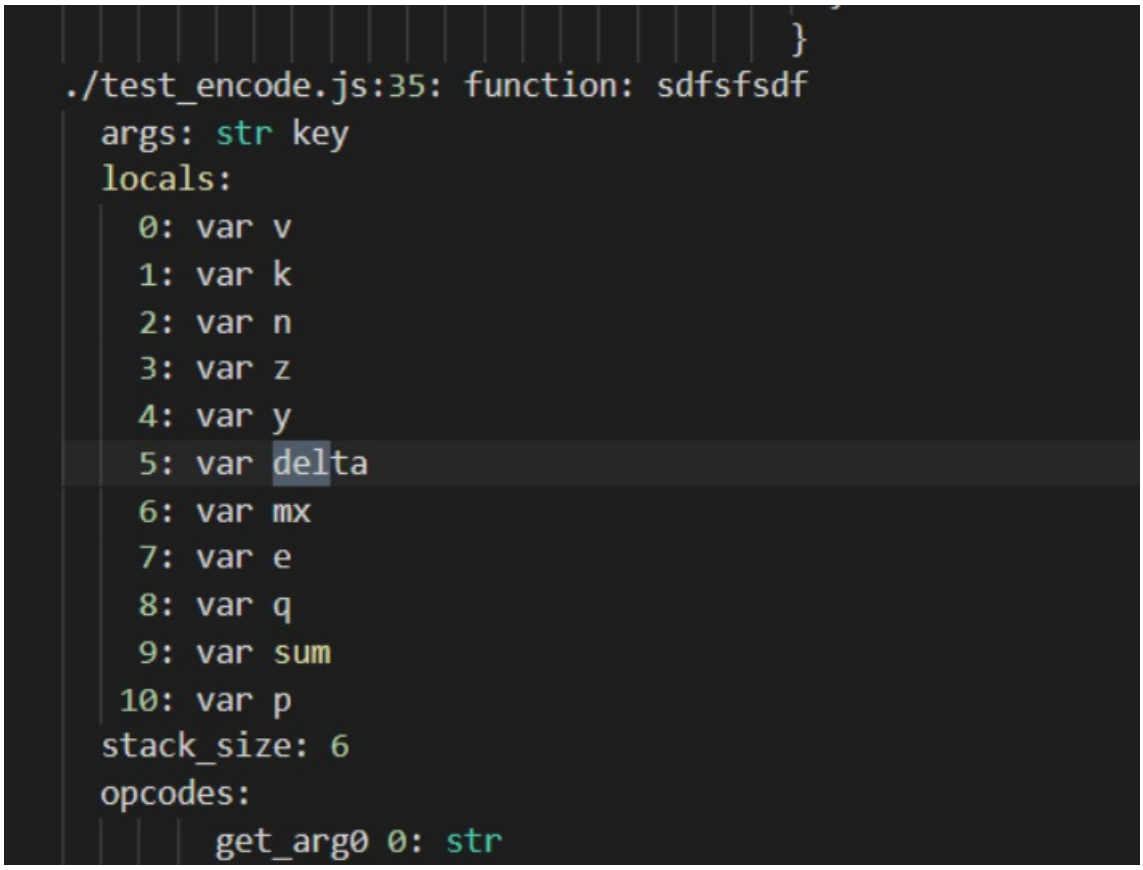
```
       get_var xxxfss
       push_atom_value "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/="
       insert2
       put_field _keyStr
       put_loc0 0: "<ret>"
       get_var xxxfss
       fclosure8 6: [bytecode <null>]
       insert2
       put_field dfsfs
       put_loc0 0: "<ret>"
       undefined
       put_loc0 0: "<ret>"
       get_var_undef scriptArgs
       typeof_is_undefined
       if_true8 179
       get_var scriptArgs
       dup
       put_var args
       put_loc0 0: "<ret>"
       get_var args
       get_field2 shift
       call_method 0
       put_loc0 0: "<ret>"
       goto8 215
 179:  undefined
       put_loc0 0: "<ret>"
       get_var_undef arguments
       typeof_is_undefined
       if_true8 203
       get_var arguments
       dup
       put_var args
       put_loc0 0: "<ret>"
       goto8 215
 203:  push_atom_value error
       dup
       put_var args
       put_loc0 0: "<ret>"
 215:  get_var main
       get_var args
       call1 1
       set_loc0 0: "<ret>"
       return



                              }
error
```

一共有这些函数

eval、long2str、str2long、sdfsfsdf、str2Hex、hex2str、xxxfss、<null>、main

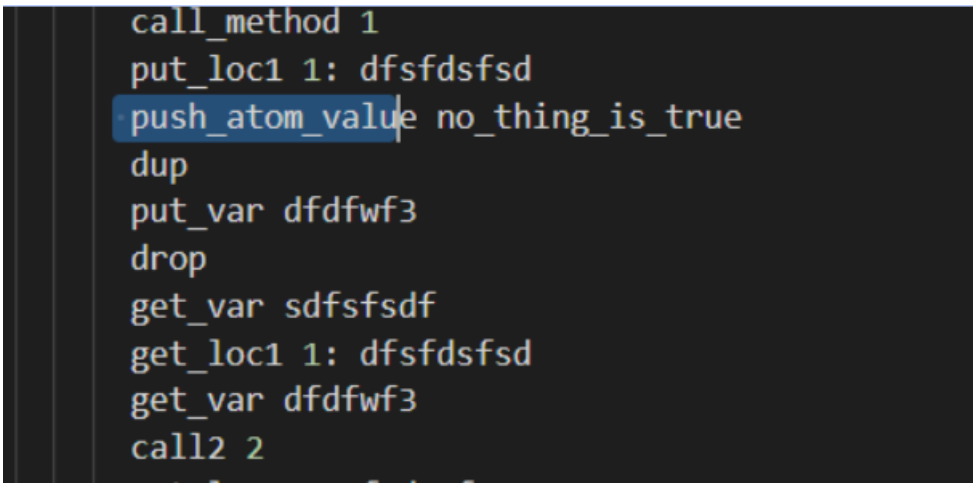
接下来分析各个函数的功能

sdfsfsdf:



应该为tea系列

然后key为 no_thing_is_true

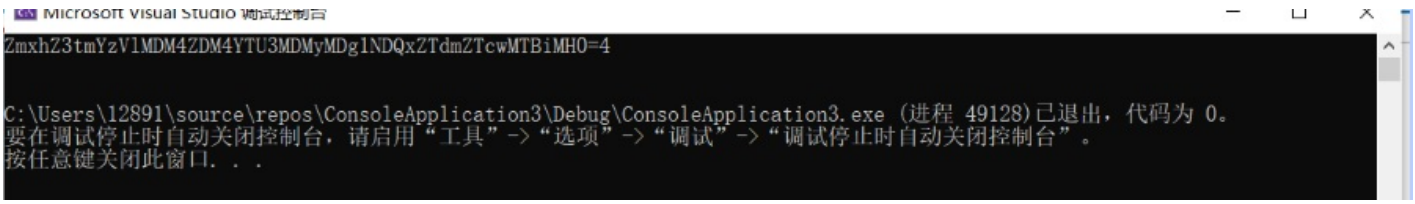


直接上脚本

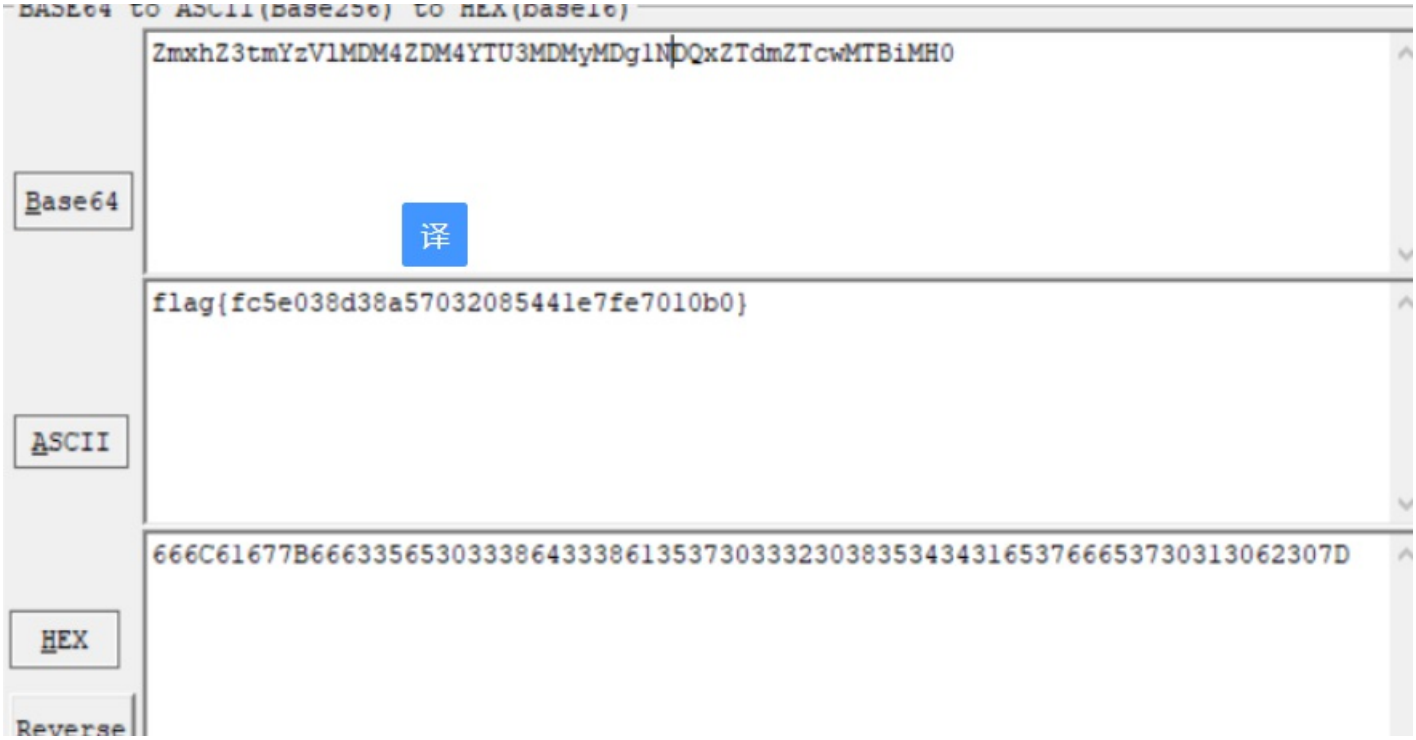
```
#include <stdio.h>
#include <stdlib.h>
#define DELTA 0x9e3779b9//0x61C88647
int main()
{
   char v1[] =
  {
       0x05,0xae,0xd0,0xce,
       0x44,0x1f,0x80,0xb5,
       0xbc,0x36,0xaf,0x4c,
       0x69,0x85,0x09,0xfc,
       0x6c,0xc3,0xc9,0x71,
       0x46,0x35,0x3d,0xe5,
```

```
        0xa9,0x5c,0x6a,0xbe,
        0xa0,0x7f,0xd4,0xa7,
        0x07,0x09,0x32,0xd8,
        0x6a,0xc3,0x2d,0x62,
        0x86,0x72,0xa5,0x91,
        0x23,0xe5,0x97,0x23,
        0x31,0xdb,0x5d,0xff,
        0xe7,0x05,0x73,0x62
    };


        char key1[] = "no_thing_is_true";//0x6e6f5f74,0x68696e67,0x5f69735f,0x74727565
    unsigned int sum = 0;
    unsigned int y, z, p, rounds, e;
    int n = 14;  //


    unsigned int* v = (unsigned int*)v1;
    unsigned int* key = (unsigned int*)key1;
    int i = 0;
    rounds = 6 + 52 / n;
    y = v[0];
    sum = (rounds * DELTA) & 0xffffffff;
    do                  //0x9E3779B9*(52/35)-0x4AB325AA，测试来要循环7次
    {
        e = sum >> 2 & 3;
        for (p = n - 1; p > 0; p--)    //34次循环
        {
            z = v[p - 1];
            v[p] = (v[p] - ((((z >> 5) ^ (y << 2)) + ((y >> 3) ^ (z << 4))) ^ ((key[(p ^ e) & 3] ^ z) + (y ^
            y = v[p];
        }
        z = v[n - 1];
        v[0] = (v[0] - (((key[(p ^ e) & 3] ^ z) + (y ^ sum)) ^ (((y << 2) ^ (z >> 5)) + ((z << 4) ^ (y >> 3)
        y = v[0];
        sum = (sum - DELTA) & 0xffffffff;
    } while (--rounds);
    for (i = 0; i < 56; i++)
    {
        printf("%c", v1[i]);
    }
    printf("\n\n");


    return 0;
}
```

解密得到



之后base直接转得到flag



2、marmagic

前面的vm都没用

```
import itertools
import string
from zio import *
s = [0x2FA4B4E6, 0x3AA763C7, 0x67057097, 0x155D753, 0x423159C0, 0x9F54FA00, 0x8AAAFCD, 0xE468BB71, 0x4CCCFB
s += [0xB8F2D37F, 0x19794181, 0x993737EE, 0x749C7985, 0x5A9734C3, 0xEA59355A, 0xC73E3987, 0x22EB465B, 0x8B5
s += [0x3A661CBD, 0x5578F063, 0x76D83FC5, 0x699EA4A7, 0xCA324955, 0xC6A8998D, 0xFA2A09D2, 0x158673D9, 0x66C
s += [0xD22BC207, 0x4000F0AC, 0x24C4BA66, 0x8523EA8A, 0x66D95F75, 0x988ADF8A, 0x76F57D90, 0x42BA7FCF, 0x22F
s += [0xC5919DF, 0x10271DCA, 0xBE0AABA8, 0xF3001CAD, 0x3D9BFB3D, 0xBCEC5E4D, 0x10D780AC, 0x9537CA60, 0x9F02
s += [0x472ECBDF, 0x20DB1A28, 0xA8FD9C14, 0x2A5800F0, 0xE1B8C757, 0x5AA73EE9, 0xEF262FE2, 0x2341E61E, 0x4F8
cmps = [0x99C57520, 0x177E3A5A, 0x717D831F, 0x69A5977F, 0x8CAF8A2D, 0x23B374E2, 0x8062C165, 0xEFE98B3F, 0x6
       0xAAB6B3F, 0x45057390, 0x2EA6430]


mydict = {}
for p in itertools.combinations(s, 5):
   sum = (p[0] + p[1] + p[2] + p[3] + p[4])&0xffffffff
   if sum in cmps:
       print ('p=%s, sum=%x' %(p, sum))
       mydict[sum] = p


k = 0
s1 = list(mydict[cmps[2*k]])
s2 = list(mydict[cmps[2*k+1]])


v = list(set(s1).intersection(set(s2)))[0]
s1.remove(v)
s2.remove(v)
if k == 1:
   s3 = [0x79C1E7FF, 0x3092311F, 0x379526BD, 0xA776EEFC, 0xFCA2C92C, 0x20000000, 0x2FD60000, 0xE2B81F5A, 0x
elif k == 2:
   s3 = [0x58D1FFFF, 0x5B7F0D80, 0xEC3A938F, 0xCCBAB640, 0x3EA60422, 0x3F80589B, 0x436C57F8, 0xA2A000C0, 0x
```

```
elif k == 0:
   s3 = [0x7E1F777C, 0x81550ED6, 0x61D70225, 0x10602644, 0x05CD4A40, 0xC0120000, 0xF8214B14, 0x77000000, 0x
elif k == 3:
   s3 = [0x02A20833, 0x37FFFEED, 0x83220421, 0x2B8A0884, 0xF1719016, 0x552BC509, 0xCB104A8D, 0xBD4B0696, 0x
elif k == 4:
   s3 = [0x32C600E7, 0x588001F3, 0x24224983, 0x3C804776, 0xA07004E0, 0x0C079035, 0xCCA4D614, 0x4780C084, 0x
elif k == 5:
   s3 = [0x81BFE181, 0x544F470A, 0x728880C0, 0xD613137B, 0x86970C9F, 0xE1000000, 0x9CA8076B, 0xDA10007D, 0x


for p1 in itertools.permutations(s1):
   for p2 in itertools.permutations(s2):
       d = [p1[0], p2[0], p1[1], p2[1], v, p2[2], p1[2], p2[3], p1[3]]
       sum = 0
       for i in range(9):
           sum += d[i]*s3[i]
           sum = sum&0xffffffff
       c = l32(sum)
       is_printable = True
       for j in range(4):
           if c[j] not in string.printable:
               is_printable = False


       if is_printable:
           print ('d=%s, c=%s' %(d, c))
```

3、showyourflag

本题有幸拿到唯一一血 □

```
from zio import *
with open('./yourflag', 'rb') as f:
   d = f.read()


def handle_one(d, d2, idx):
   common_len = 0
   while True:
       if idx >= (len(d)):
           break
       size = l8(d[idx])
       print ('idx=%x, size=%x, idx2=%x' %(idx, size, len(d2)))
       if size == 0x1f:
           d2 += d[idx+1:idx+0x21]
           idx += 0x21
       elif size < 0x1f:
           d2 += d[idx+1:idx+2+size]
           idx += size + 2
       elif (size >= 0x20)&(size < 0xe0):
           common_len += (size>>5)+2
           distance = ((size&0x1f)<<8) + l8(d[idx+1]) + 1
           s = d2[-distance:]
           while len(s) < common_len:
               s += d2[-distance:]
           s = s[:common_len]
           print ('common_len=%x, distance=%x, data=%s' %(common_len, distance, s.encode('hex')))
           d2 += s
```

```
            common_len = 0
            idx += 2
        elif (size >= 0xe0):
            size2 = l8(d[idx+1])
            if size2 == 0xfd:
                common_len += 262
                idx += 3
            else:
                common_len += size2 + 2 + 7
                distance = (((size+32)&0xff) << 8) + l8(d[idx + 2]) + 1
                s = d2[-distance:]
                while len(s) < common_len:
                    s += d2[-distance:]
                s = s[:common_len]
                print ('common_len=%x, distance=%x, data=%s' % (common_len, distance, s.encode('hex')))
                d2 += s
                common_len = 0
                idx += 3
    return idx, d2


idx = 0
d2 = ''
idx, d2 = handle_one(d, d2, idx)


d3 = ''
for i in range(len(d2)):
    c = (0xff-ord(d2[i]))
    c = ((c>>1)+(c<<7))&0xff
    d3 += chr(c)


with open('flag3.png', 'wb') as f:
    f.write(d3)
```

Mobile

1、HaHaHaHa

先计算出各个hash加密的顺序：

```
from hashlib import *
import hashlib
op = [0xAF, 0xA1, 0xA4, 170, 0xA5, 0xAE, 0xA0, 0xA3]
for i in range(len(op)):
    op[i] ^= 0xAB
for i in range(len(op)):
    if (op[i] >> 3) & 1:
        print('SHA-512')
    elif (op[i]&7) == 0:
        print('MD2')
    elif (op[i]&7) == 1:
        print('MD5')
    elif (op[i]&7) == 2:
        print('SHA-1')
    elif (op[i]&7) == 3:
        print('SHA-224')
    elif (op[i]&7) == 4:
        print('SHA-256')
    elif (op[i]&7) == 5:
        print('SHA-384')
```

得到加密顺序为

SHA-256

HmacSha512

HmacSha512

MD5

HmacSha512

SHA-384

HmacSha512

HmacSha512

之后逐个爆破

SHA-256：

```
from hashlib import *
import hmac
key = 0
for i in range(0x20,0x7f):
   for j in range(0x20,0x7f):
       for k in range(0x20,0x7f):
           for l in range(0x20,0x7f):
               tmp = bytes([i, j, k, l])
               if sha256(tmp).hexdigest()[:16] == 'fc7466e55fbf37b1':
                   print(tmp)
```

```
C:\Users\12891\Desktop\拟态\HaHaHaHa>python burp_256.py
b'_8@P'
```

Sha512：

```
from hashlib import *
import hmac


a = [4,10,15,1,14,5,11,8]
b = ["WIgD1ZNZ0ilJqFpw", "4811tjOZjoiXpjdq", "ALFjcgztxnUaC89v", "ZgHzTu79Zwhoi0PB", "UYBfajKYrDFE1zJs", "y
for i in range(len(b)):
   b[i] = md5(b[i].encode()).digest()
final = ["fc7466e55fbf37b1", "78b0be39e63b6837", "c2f9c805d0442203", "c11a61bb60d79dab", "869e650ee55bd9f6"


def burp_512(order,key):
   for i in range(0x20,0x7f):
       for j in range(0x20,0x7f):
           for k in range(0x20,0x7f):
               for l in range(0x20,0x7f):
                   tmp = bytes([i, j, k, l])
                   if hmac.new(key, tmp, sha512).hexdigest()[:16] == final[order]:
                       print(tmp)
                       break
for i in range(len(a)):
   if (a[i] >> 3) & 1:
       burp_512(i,b[a[i]&7])
```

```
C:\Users\12891\Desktop\拟态\HaHaHaHa>python burp2.py
b'51_H'
b'}P11'
b'C_55'
b'3H7_'
b'GALF'
```

SHA-384：

```
from hashlib import *
import hmac
key = 0
for i in range(0x20,0x7f):
   for j in range(0x20,0x7f):
       for k in range(0x20,0x7f):
           for l in range(0x20,0x7f):
               tmp = bytes([i, j, k, l])
               if sha384(tmp).hexdigest()[:16] == 'f2dda5fc021fe2bf':
                   print(tmp)
```

```
C:\Users\12891\Desktop\拟态\HaHaHaHa>python burp_384.py
b'P1N3'
```

MD5：

```
from hashlib import *
import hashlib


for i in range(0x20,0x7f):
   for j in range(0x20,0x7f):
       for k in range(0x20,0x7f):
           for l in range(0x20,0x7f):
               tmp = bytes([i, j, k, l])
               if hashlib.md5(tmp).hexdigest()[:16] == 'c11a61bb60d79dab':
                   print(tmp)
```

```
C:\Users\12891\Desktop\拟态\HaHaHaHa>python burp_md5.py
b'{H@5'
```

拼接起来得到flag为 FLAG{H@5H_15_7H3_8@PP1N355_C11P}

Misc

1、WeirdPhoto

crc32爆破得到真正的宽和高

```
00000000  89 50 4E 47 0D 0A 1A 0A  00 00 00 0D 49 48 44 52
00000010  00 00 03 E8 00 00 01 F4  08 06 00 00 00 9E 91 69
00000020  64 00 00 00 01 73 52 47  42 00 AE CE 1C E9 00 00
```

```
#Python3爆 png长和高
import binascii
import struct



for i in range(20000):#一般 20000就够
   wide = struct.pack('>i',i)
   for j in range(20000):
       high = struct.pack('>i',j)
       data = b'\x49\x48\x44\x52' + wide+ high+b'\x08\x06\x00\x00\x00'
       #因为是 Py3，byte和str型不能直接进行运算，要把 str写 b'...'。不然把 wide和 high写成 str(...)




       crc32 = binascii.crc32(data) & 0xffffffff
       if crc32 == 0x9E916964: # 0x9E916964是这个 png文件头的 CRC校验码，在 21~25byte处
           print('\n\n',i,j,crc32)
           print(type(data))
           exit(0)
   print(i, end=' ')
```

运行后爆破出真实的宽为1420，高为500，在winhex中修改后保存

00000000	89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52
00000010	00	00	05	8C	00	00	01	F4	08	06	00	00	00	9E	91	69
00000020	64	00	00	00	01	73	52	47	42	00	AE	CE	1C	E9	00	00

打开图片，已被还原

猜了半天，猜到是栅栏密码，而且key为4

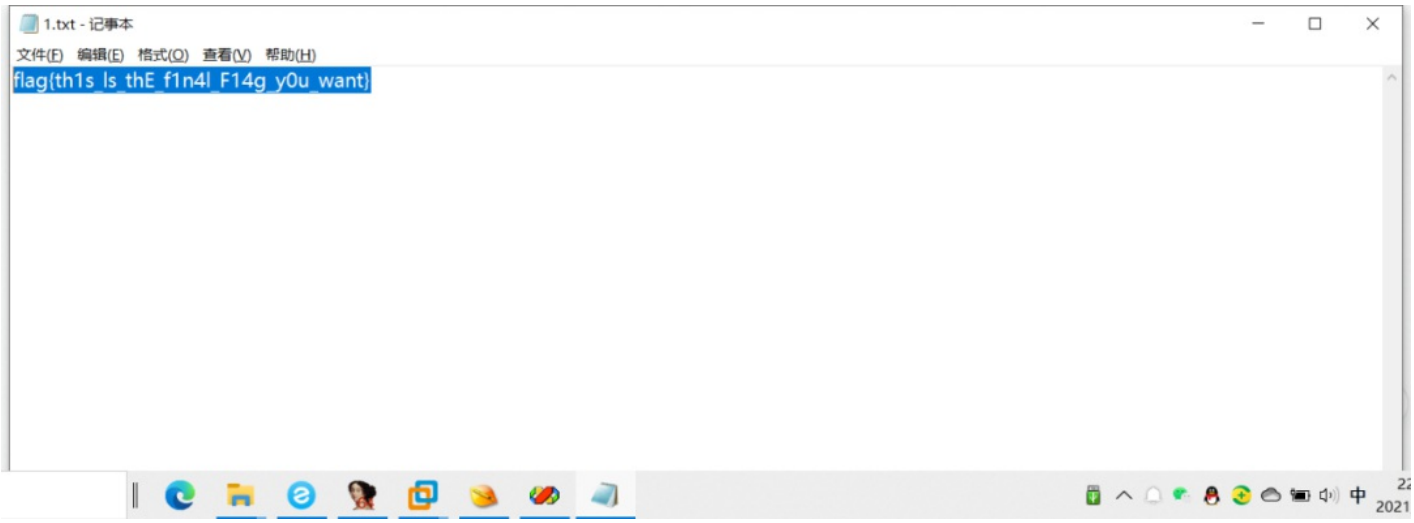
解开之后里面有一份out文件

搜索头文件里面的/BitsPerComponent

发现将原pdf文件4字节改了，应该是在pdf文件中隐藏了flag

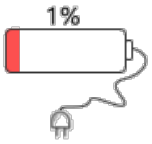
恢复pdf文件

用wbStego4恢复，得到flag



flag{th1s_ls_thE_f1n4l_F14g_y0u_want}

实操推荐：https://www.hetianlab.com/pages/CTFLaboratory.jsp



戳“阅读原文”体验免费靶场!