alien math-en

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int i; // [rsp+8h] [rbp-8h]

  fjwpa_bpafbz();
  puts(&s);
  puts(&byte_5650D7613258);
  puts(&byte_5650D76132D0);
  putchar('\n');
  for ( i = 0; i <= 67; ++i )
  {
    if ( !(unsigned int)ajjgpfl(i) )
    {
      puts(&byte_5650D7613330);
      return 1;
    }
  }
  puts(&byte_5650D76133B0);
  puts(&byte_5650D7613418);
  puts(flag);
  return 0;
}
```

In the main, `fjwpa_bpafbz();` run `ajjgpfl(int foo)` 68 times and print out the flag when it passes safely.

Therefore, the above two functions should be carefully examined.

```
void fjwpa_bpafbz()
{
  unsigned int v0; // eax
  int i; // [rsp+Ch] [rbp-4h]

  v0 = time(0LL);
  srand(v0);
  for ( i = 0; i <= 62; ++i )
    rand();
}
```

`fjwpa_bpafbz();` → set `srand()` (seed of **`rand()`** function) and discards `rand()` function return value 62times.

```
__int64 __fastcall ajjgpfl(int a1)
{
  __int64 result; // rax
  int v2; // [rsp+14h] [rbp-13Ch]
  signed int v3; // [rsp+18h] [rbp-138h]
  unsigned int v4; // [rsp+1Ch] [rbp-134h]
  _BYTE *ptr; // [rsp+28h] [rbp-128h]
  _BYTE *v6; // [rsp+30h] [rbp-120h]
  _BYTE *v7; // [rsp+38h] [rbp-118h]
  char v8[264]; // [rsp+40h] [rbp-110h] BYREF
  unsigned __int64 v9; // [rsp+148h] [rbp-8h]

  v9 = __readfsqword(0x28u);
  v2 = bpafbz(1, 6);                            // 1~6 random
  ptr = rgisaz_jufrtme(a1 + 1);
  printf(aS, ptr);
  free(ptr);
  v3 = bpafbz(1, 511);                          // 1~511 random
  v4 = bpafbz(1, 511);                          // 1~511 random
  v6 = rgisaz_jufrtme(v3);
  v7 = rgisaz_jufrtme(v4);
  switch ( v2 )
  {
    case 1:
      printf(aS_0, v6, v7);
```

```
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( v3 == (unsigned int)emtrfuj_zasigr(v8) + v4 )
        goto LABEL_14;
      result = 0LL;
      break;
    case 2:
      printf(aS_1, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( v3 == (v4 ^ (unsigned int)emtrfuj_zasigr(v8)) )
        goto LABEL_14;
      result = 0LL;
      break;
    case 3:
      printf(aS_2, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( (unsigned int)emtrfuj_zasigr(v8) == (int)(3 * v4) / v3 )
        goto LABEL_14;
      result = 0LL;
      break;
    case 4:
      printf(aS_3, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( (unsigned int)emtrfuj_zasigr(v8) == 3 * v3 % (int)(3 * v4) )
        goto LABEL_14;
      result = 0LL;
      break;
    case 5:
      printf(aS_4, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( !(v3 * v3 - (unsigned int)emtrfuj_zasigr(v8) + v4) )
        goto LABEL_14;
      result = 0LL;
      break;
    case 6:
      printf(aS_5, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( v3 - v4 == (unsigned int)emtrfuj_zasigr(v8) - v3 + v4 )
        goto LABEL_14;
      result = 0LL;
      break;
    default:
LABEL_14:
      result = 1LL;
      break;
  }
  return result;
}
```

Return the random value between the two factors with `bpafbz(inta1, inta2)` -> `return (unsigned int) (rand() % (a2 - a1) + a1)` inside `ajjgpfl(inta1)`.

```
// a1 = 1 ~ 511 random
_BYTE *__fastcall rgisaz_jufrtme(unsigned int a1)
{
  double v1; // xmm0_8
  double v2; // xmm0_8
  double v3; // xmm1_8
  int v5; // [rsp+10h] [rbp-30h]
  int i; // [rsp+14h] [rbp-2Ch]
  int v7; // [rsp+18h] [rbp-28h]
  int v8; // [rsp+20h] [rbp-20h]
  int v9; // [rsp+24h] [rbp-1Ch]
```

```
  char *s; // [rsp+28h] [rbp-18h]
  _BYTE *v11; // [rsp+30h] [rbp-10h]
  char *v12; // [rsp+38h] [rbp-8h]

  v1 = log10((double)(int)(a1 + 1));
  v7 = (int)(ceil(v1) + 1.0);                   // ceil : ROUND UP
  v2 = log10((double)(int)(a1 + 1));
  v3 = (ceil(v2) + 1.0) * 8.0;
  s = (char *)malloc(v7 + 1);
  v11 = malloc((int)(3.0 * v3) + 1);
  sprintf(s, "%o", a1);
  trof_pripew(s);                               // reverse str
  v8 = strlen(s);
  v5 = 0;
  for ( i = 0; i < v8; ++i )
  {
    v12 = (char *)*(&lfwp + s[i] - '0');
    v9 = strlen(v12);
    memcpy(&v11[v5], v12, v9);
    v5 += v9;
  }
  v11[v5] = 0;
  free(s);
  return v11;
}
```

```
 sprintf(s, "%o", a1);
  trof_pripew(s);                               // reverse str
  v8 = strlen(s);
  v5 = 0;
  for ( i = 0; i < v8; ++i )
  {
    v12 = (char *)*(&lfwp + s[i] - '0');
    v9 = strlen(v12);
    memcpy(&v11[v5], v12, v9);
    v5 += v9;
  }
  v11[v5] = 0;
```

A place to pay attention to in this function is the code above. Let's look at the `trof_pripew(char *a1)` function.

```
// 1~511 random oct str
void __fastcall trof_pripew(char *a1)
{
  char *s; // [rsp+8h] [rbp-18h]
  const char *i; // [rsp+18h] [rbp-8h]

  s = a1;
  if ( a1 )
  {
    for ( i = &a1[strlen(a1) - 1]; s < i; --i ) // swap reverse
    {
      *s ^= *i;
      *i ^= *s;
      *s++ ^= *i;
    }
  }
}
```

`trof_pripew(char *a1)` → Assuming that the string a b c d is applied, in the first case,

`a^d b c d → a^d b c d^a^d → a^d b c a → a^d^a b c a → d b c a`

Then in the second case,

`d b^c c a → d b^c c^b^c a → d b^c b a → d b^c^b b a → d c b a`

That is, the function is a string reverse function.

```
sprintf(s, "%o", a1);
  trof_pripew(s);                               // reverse str
```

```
  v8 = strlen(s);
  v5 = 0;
  for ( i = 0; i < v8; ++i )
  {
    v12 = (char *)*(&lfwp + s[i] - '0');
    v9 = strlen(v12);
    memcpy(&v11[v5], v12, v9);
    v5 += v9;
  }
  v11[v5] = 0;
```

If you go back and look at the corresponding part, it's as follows.

First, the function's factor `a1` is replaced with an octal string and stored in `s`.

After that, the character string is reversed, and the character string is used as the actual index of `lfwp` and stored in `v11` assigned to the heap.

`lfwp` is initialized as follows.

```
.data:00005650D7615020 lfwp            dq offset unk_5650D7613008
.data:00005650D7615020                                         ; DATA XREF: rgisaz_jufrtme+103↑o
.data:00005650D7615020                                         ; emtrfuj_zasigr+92↑o ...
.data:00005650D7615028                 dq offset unk_5650D7613015
.data:00005650D7615030                 dq offset unk_5650D7613025
.data:00005650D7615038                 dq offset unk_5650D7613038
.data:00005650D7615040                 dq offset unk_5650D761304E
.data:00005650D7615048                 dq offset unk_5650D7613064
.data:00005650D7615050                 dq offset unk_5650D7613074
.data:00005650D7615058                 dq offset unk_5650D761308D
.data:00005650D7615060                 dq offset unk_5650D76130A3
```

```
.rodata:00005650D7613008 unk_5650D7613008 db 0E3h               ; DATA XREF: .data:lfwp↓o
.rodata:00005650D7613009                  db  83h
.rodata:00005650D761300A                  db 0BBh
.rodata:00005650D761300B                  db 0E2h
.rodata:00005650D761300C                  db  94h
.rodata:00005650D761300D                  db 0A4h
.rodata:00005650D761300E                  db 0E2h
.rodata:00005650D761300F                  db  94h
.rodata:00005650D7613010                  db 0A4h
.rodata:00005650D7613011                  db 0E2h
.rodata:00005650D7613012                  db  95h
.rodata:00005650D7613013                  db  99h
.rodata:00005650D7613014                  db    0
.rodata:00005650D7613015 unk_5650D7613015 db 0E3h               ; DATA XREF: .data:00005650D7615028↓o
.rodata:00005650D7613016                  db  83h
.rodata:00005650D7613017                  db 0BBh
.rodata:00005650D7613018                  db 0E2h
.rodata:00005650D7613019                  db  94h
.rodata:00005650D761301A                  db  9Ch
.rodata:00005650D761301B                  db 0E2h
.rodata:00005650D761301C                  db  94h
.rodata:00005650D761301D                  db 0B4h
.rodata:00005650D761301E                  db 0E2h
.rodata:00005650D761301F                  db  95h
.rodata:00005650D7613020                  db  97h
.rodata:00005650D7613021                  db 0E2h
.rodata:00005650D7613022                  db  95h
.rodata:00005650D7613023                  db 0ACh
.rodata:00005650D7613024                  db    0
.rodata:00005650D7613025 unk_5650D7613025 db 0E3h               ; DATA XREF: .data:00005650D7615030↓o
.rodata:00005650D7613026                  db  83h
.rodata:00005650D7613027                  db 0BBh
.rodata:00005650D7613028                  db 0E2h
.rodata:00005650D7613029                  db  95h
.rodata:00005650D761302A                  db  9Dh
.rodata:00005650D761302B                  db 0E2h
```

```
.rodata:00005650D761302C                db  94h
.rodata:00005650D761302D                db  94h
.rodata:00005650D761302E                db 0E2h
.rodata:00005650D761302F                db  94h
.rodata:00005650D7613030                db 0A4h
.rodata:00005650D7613031                db 0E2h
.rodata:00005650D7613032                db  94h
.rodata:00005650D7613033                db  90h
.rodata:00005650D7613034                db 0E2h
.rodata:00005650D7613035                db  94h
.rodata:00005650D7613036                db 0BCh
.rodata:00005650D7613037                db    0
.rodata:00005650D7613038 unk_5650D7613038 db 0E3h               ; DATA XREF: .data:00005650D7615038↓o
.rodata:00005650D7613039                db  83h
.rodata:00005650D761303A                db 0BBh
.rodata:00005650D761303B                db 0E2h
.rodata:00005650D761303C                db  94h
.rodata:00005650D761303D                db 0ACh
.rodata:00005650D761303E                db 0E2h
.rodata:00005650D761303F                db  95h
.rodata:00005650D7613040                db  92h
.rodata:00005650D7613041                db 0E2h
.rodata:00005650D7613042                db  94h
.rodata:00005650D7613043                db  98h
.rodata:00005650D7613044                db 0E2h
.rodata:00005650D7613045                db  94h
.rodata:00005650D7613046                db  80h
.rodata:00005650D7613047                db 0E2h
.rodata:00005650D7613048                db  94h
.rodata:00005650D7613049                db  94h
.rodata:00005650D761304A                db 0E2h
.rodata:00005650D761304B                db  94h
.rodata:00005650D761304C                db 0B4h
.rodata:00005650D761304D                db    0
.rodata:00005650D761304E unk_5650D761304E db 0E3h               ; DATA XREF: .data:00005650D7615040↓o
.rodata:00005650D761304F                db  83h
.rodata:00005650D7613050                db 0BBh
.rodata:00005650D7613051                db 0E2h
.rodata:00005650D7613052                db  94h
.rodata:00005650D7613053                db  9Ch
.rodata:00005650D7613054                db 0E2h
.rodata:00005650D7613055                db  95h
.rodata:00005650D7613056                db 0A1h
.rodata:00005650D7613057                db 0E2h
.rodata:00005650D7613058                db  94h
.rodata:00005650D7613059                db  8Ch
.rodata:00005650D761305A                db 0E2h
.rodata:00005650D761305B                db  94h
.rodata:00005650D761305C                db  94h
.rodata:00005650D761305D                db 0E2h
.rodata:00005650D761305E                db  94h
.rodata:00005650D761305F                db 0ACh
.rodata:00005650D7613060                db 0E2h
.rodata:00005650D7613061                db  95h
.rodata:00005650D7613062                db 0A5h
.rodata:00005650D7613063                db    0
.rodata:00005650D7613064 unk_5650D7613064 db 0E3h               ; DATA XREF: .data:00005650D7615048↓o
.rodata:00005650D7613065                db  83h
.rodata:00005650D7613066                db 0BBh
.rodata:00005650D7613067                db 0E2h
.rodata:00005650D7613068                db  94h
.rodata:00005650D7613069                db 0B4h
.rodata:00005650D761306A                db 0E2h
.rodata:00005650D761306B                db  94h
.rodata:00005650D761306C                db  90h
.rodata:00005650D761306D                db 0E2h
.rodata:00005650D761306E                db  94h
.rodata:00005650D761306F                db 0A4h
.rodata:00005650D7613070                db 0E2h
.rodata:00005650D7613071                db  94h
.rodata:00005650D7613072                db 0ACh
.rodata:00005650D7613073                db    0
.rodata:00005650D7613074 unk_5650D7613074 db 0E3h               ; DATA XREF: .data:00005650D7615050↓o
.rodata:00005650D7613075                db  83h
.rodata:00005650D7613076                db 0BBh
.rodata:00005650D7613077                db 0E2h
.rodata:00005650D7613078                db  94h
```

```
.rodata:00005650D7613079                db  94h
.rodata:00005650D761307A                db 0E2h
.rodata:00005650D761307B                db  94h
.rodata:00005650D761307C                db  80h
.rodata:00005650D761307D                db 0E2h
.rodata:00005650D761307E                db  94h
.rodata:00005650D761307F                db  90h
.rodata:00005650D7613080                db 0E2h
.rodata:00005650D7613081                db  94h
.rodata:00005650D7613082                db 0A4h
.rodata:00005650D7613083                db 0E2h
.rodata:00005650D7613084                db  94h
.rodata:00005650D7613085                db  80h
.rodata:00005650D7613086                db 0E2h
.rodata:00005650D7613087                db  94h
.rodata:00005650D7613088                db 0B4h
.rodata:00005650D7613089                db 0E2h
.rodata:00005650D761308A                db  94h
.rodata:00005650D761308B                db 0B4h
.rodata:00005650D761308C                db    0
.rodata:00005650D761308D unk_5650D761308D db 0E3h              ; DATA XREF: .data:00005650D7615058↓o
.rodata:00005650D761308E                db  83h
.rodata:00005650D761308F                db 0BBh
.rodata:00005650D7613090                db 0E2h
.rodata:00005650D7613091                db  94h
.rodata:00005650D7613092                db 0ACh
.rodata:00005650D7613093                db 0E2h
.rodata:00005650D7613094                db  95h
.rodata:00005650D7613095                db 0A7h
.rodata:00005650D7613096                db 0E2h
.rodata:00005650D7613097                db  94h
.rodata:00005650D7613098                db  80h
.rodata:00005650D7613099                db 0E2h
.rodata:00005650D761309A                db  94h
.rodata:00005650D761309B                db  98h
.rodata:00005650D761309C                db 0E2h
.rodata:00005650D761309D                db  95h
.rodata:00005650D761309E                db 0A3h
.rodata:00005650D761309F                db 0E2h
.rodata:00005650D76130A0                db  94h
.rodata:00005650D76130A1                db  90h
.rodata:00005650D76130A2                db    0
.rodata:00005650D76130A3 unk_5650D76130A3 db 0E3h              ; DATA XREF: .data:00005650D7615060↓o
.rodata:00005650D76130A4                db  83h
.rodata:00005650D76130A5                db 0BBh
.rodata:00005650D76130A6                db 0E2h
.rodata:00005650D76130A7                db  94h
.rodata:00005650D76130A8                db  80h
.rodata:00005650D76130A9                db    0
```

That is, the factor value of the function is substituted with the corresponding bytes arrays, and the substituted bytes array is returned.

```
switch ( v2 )
  {
    case 1:
      printf(aS_0, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( v3 == (unsigned int)emtrfuj_zasigr(v8) + v4 )
        goto LABEL_14;
      result = 0LL;
      break;
    case 2:
      printf(aS_1, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( v3 == (v4 ^ (unsigned int)emtrfuj_zasigr(v8)) )
        goto LABEL_14;
      result = 0LL;
      break;
    case 3:
```

```
      printf(aS_2, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( (unsigned int)emtrfuj_zasigr(v8) == (int)(3 * v4) / v3 )
        goto LABEL_14;
      result = 0LL;
      break;
    case 4:
      printf(aS_3, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( (unsigned int)emtrfuj_zasigr(v8) == 3 * v3 % (int)(3 * v4) )
        goto LABEL_14;
      result = 0LL;
      break;
    case 5:
      printf(aS_4, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( !(v3 * v3 - (unsigned int)emtrfuj_zasigr(v8) + v4) )
        goto LABEL_14;
      result = 0LL;
      break;
    case 6:
      printf(aS_5, v6, v7);
      free(v6);
      free(v7);
      __isoc99_scanf("%255s", v8);
      if ( v3 - v4 == (unsigned int)emtrfuj_zasigr(v8) - v3 + v4 )
        goto LABEL_14;
      result = 0LL;
      break;
    default:
LABEL_14:
      result = 1LL;
      break;
  }
```

In the `switch` syntax, an encrypted random value is output as a predetermined format, and it can be seen that the format contains a bytes array between `%s` at the time of confirmation. Therefore, it is possible to decode the random value used in the operation of each case from the output value, and to distinguish which operation is applied from the output value.

```
.rodata:00005650D7613008 unk_5650D7613008 db 0E3h               ; DATA XREF: .data:lfwp↓o
.rodata:00005650D7613009                  db  83h
.rodata:00005650D761300A                  db 0BBh
.rodata:00005650D761300B                  db 0E2h
.rodata:00005650D761300C                  db  94h
.rodata:00005650D761300D                  db 0A4h
.rodata:00005650D761300E                  db 0E2h
.rodata:00005650D761300F                  db  94h
.rodata:00005650D7613010                  db 0A4h
.rodata:00005650D7613011                  db 0E2h
.rodata:00005650D7613012                  db  95h
.rodata:00005650D7613013                  db  99h
.rodata:00005650D7613014                  db    0
.rodata:00005650D7613015 unk_5650D7613015 db 0E3h               ; DATA XREF: .data:00005650D7615028↓o
.rodata:00005650D7613016                  db  83h
.rodata:00005650D7613017                  db 0BBh
.rodata:00005650D7613018                  db 0E2h
.rodata:00005650D7613019                  db  94h
.rodata:00005650D761301A                  db  9Ch
.rodata:00005650D761301B                  db 0E2h
.rodata:00005650D761301C                  db  94h
.rodata:00005650D761301D                  db 0B4h
.rodata:00005650D761301E                  db 0E2h
.rodata:00005650D761301F                  db  95h
.rodata:00005650D7613020                  db  97h
.rodata:00005650D7613021                  db 0E2h
.rodata:00005650D7613022                  db  95h
```

```
.rodata:00005650D7613023                db 0ACh
.rodata:00005650D7613024                db    0
.rodata:00005650D7613025 unk_5650D7613025 db 0E3h               ; DATA XREF: .data:00005650D7615030↓o
.rodata:00005650D7613026                db  83h
.rodata:00005650D7613027                db 0BBh
.rodata:00005650D7613028                db 0E2h
.rodata:00005650D7613029                db  95h
.rodata:00005650D761302A                db  9Dh
.rodata:00005650D761302B                db 0E2h
.rodata:00005650D761302C                db  94h
.rodata:00005650D761302D                db  94h
.rodata:00005650D761302E                db 0E2h
.rodata:00005650D761302F                db  94h
.rodata:00005650D7613030                db 0A4h
.rodata:00005650D7613031                db 0E2h
.rodata:00005650D7613032                db  94h
.rodata:00005650D7613033                db  90h
.rodata:00005650D7613034                db 0E2h
.rodata:00005650D7613035                db  94h
.rodata:00005650D7613036                db 0BCh
.rodata:00005650D7613037                db    0
.rodata:00005650D7613038 unk_5650D7613038 db 0E3h               ; DATA XREF: .data:00005650D7615038↓o
.rodata:00005650D7613039                db  83h
.rodata:00005650D761303A                db 0BBh
.rodata:00005650D761303B                db 0E2h
.rodata:00005650D761303C                db  94h
.rodata:00005650D761303D                db 0ACh
.rodata:00005650D761303E                db 0E2h
.rodata:00005650D761303F                db  95h
.rodata:00005650D7613040                db  92h
.rodata:00005650D7613041                db 0E2h
.rodata:00005650D7613042                db  94h
.rodata:00005650D7613043                db  98h
.rodata:00005650D7613044                db 0E2h
.rodata:00005650D7613045                db  94h
.rodata:00005650D7613046                db  80h
.rodata:00005650D7613047                db 0E2h
.rodata:00005650D7613048                db  94h
.rodata:00005650D7613049                db  94h
.rodata:00005650D761304A                db 0E2h
.rodata:00005650D761304B                db  94h
.rodata:00005650D761304C                db 0B4h
.rodata:00005650D761304D                db    0
.rodata:00005650D761304E unk_5650D761304E db 0E3h               ; DATA XREF: .data:00005650D7615040↓o
.rodata:00005650D761304F                db  83h
.rodata:00005650D7613050                db 0BBh
.rodata:00005650D7613051                db 0E2h
.rodata:00005650D7613052                db  94h
.rodata:00005650D7613053                db  9Ch
.rodata:00005650D7613054                db 0E2h
.rodata:00005650D7613055                db  95h
.rodata:00005650D7613056                db 0A1h
.rodata:00005650D7613057                db 0E2h
.rodata:00005650D7613058                db  94h
.rodata:00005650D7613059                db  8Ch
.rodata:00005650D761305A                db 0E2h
.rodata:00005650D761305B                db  94h
.rodata:00005650D761305C                db  94h
.rodata:00005650D761305D                db 0E2h
.rodata:00005650D761305E                db  94h
.rodata:00005650D761305F                db 0ACh
.rodata:00005650D7613060                db 0E2h
.rodata:00005650D7613061                db  95h
.rodata:00005650D7613062                db 0A5h
.rodata:00005650D7613063                db    0
.rodata:00005650D7613064 unk_5650D7613064 db 0E3h               ; DATA XREF: .data:00005650D7615048↓o
.rodata:00005650D7613065                db  83h
.rodata:00005650D7613066                db 0BBh
.rodata:00005650D7613067                db 0E2h
.rodata:00005650D7613068                db  94h
.rodata:00005650D7613069                db 0B4h
.rodata:00005650D761306A                db 0E2h
.rodata:00005650D761306B                db  94h
.rodata:00005650D761306C                db  90h
.rodata:00005650D761306D                db 0E2h
.rodata:00005650D761306E                db  94h
.rodata:00005650D761306F                db 0A4h
```

```
.rodata:00005650D7613070                 db 0E2h
.rodata:00005650D7613071                 db  94h
.rodata:00005650D7613072                 db 0ACh
.rodata:00005650D7613073                 db    0
.rodata:00005650D7613074 unk_5650D7613074 db 0E3h               ; DATA XREF: .data:00005650D7615050↓o
.rodata:00005650D7613075                 db  83h
.rodata:00005650D7613076                 db 0BBh
.rodata:00005650D7613077                 db 0E2h
.rodata:00005650D7613078                 db  94h
.rodata:00005650D7613079                 db  94h
.rodata:00005650D761307A                 db 0E2h
.rodata:00005650D761307B                 db  94h
.rodata:00005650D761307C                 db  80h
.rodata:00005650D761307D                 db 0E2h
.rodata:00005650D761307E                 db  94h
.rodata:00005650D761307F                 db  90h
.rodata:00005650D7613080                 db 0E2h
.rodata:00005650D7613081                 db  94h
.rodata:00005650D7613082                 db 0A4h
.rodata:00005650D7613083                 db 0E2h
.rodata:00005650D7613084                 db  94h
.rodata:00005650D7613085                 db  80h
.rodata:00005650D7613086                 db 0E2h
.rodata:00005650D7613087                 db  94h
.rodata:00005650D7613088                 db 0B4h
.rodata:00005650D7613089                 db 0E2h
.rodata:00005650D761308A                 db  94h
.rodata:00005650D761308B                 db 0B4h
.rodata:00005650D761308C                 db    0
.rodata:00005650D761308D unk_5650D761308D db 0E3h               ; DATA XREF: .data:00005650D7615058↓o
.rodata:00005650D761308E                 db  83h
.rodata:00005650D761308F                 db 0BBh
.rodata:00005650D7613090                 db 0E2h
.rodata:00005650D7613091                 db  94h
.rodata:00005650D7613092                 db 0ACh
.rodata:00005650D7613093                 db 0E2h
.rodata:00005650D7613094                 db  95h
.rodata:00005650D7613095                 db 0A7h
.rodata:00005650D7613096                 db 0E2h
.rodata:00005650D7613097                 db  94h
.rodata:00005650D7613098                 db  80h
.rodata:00005650D7613099                 db 0E2h
.rodata:00005650D761309A                 db  94h
.rodata:00005650D761309B                 db  98h
.rodata:00005650D761309C                 db 0E2h
.rodata:00005650D761309D                 db  95h
.rodata:00005650D761309E                 db 0A3h
.rodata:00005650D761309F                 db 0E2h
.rodata:00005650D76130A0                 db  94h
.rodata:00005650D76130A1                 db  90h
.rodata:00005650D76130A2                 db    0
.rodata:00005650D76130A3 unk_5650D76130A3 db 0E3h               ; DATA XREF: .data:00005650D7615060↓o
.rodata:00005650D76130A4                 db  83h
.rodata:00005650D76130A5                 db 0BBh
.rodata:00005650D76130A6                 db 0E2h
.rodata:00005650D76130A7                 db  94h
.rodata:00005650D76130A8                 db  80h
.rodata:00005650D76130A9                 db    0
```

The input is then received from `__isoc99_scanf ("%255s", v8);` the input value is used in the if condition via `emtrfuj_zasigr (const char *a1).`

```
// a1 = input
__int64 __fastcall emtrfuj_zasigr(const char *a1)
{
  unsigned int v2; // [rsp+1Ch] [rbp-24h] BYREF
  int v3; // [rsp+20h] [rbp-20h]
  int v4; // [rsp+24h] [rbp-1Ch]
  int i; // [rsp+28h] [rbp-18h]
  int v6; // [rsp+2Ch] [rbp-14h]
  const char *v7; // [rsp+30h] [rbp-10h]
  unsigned __int64 v8; // [rsp+38h] [rbp-8h]

  v8 = __readfsqword(0x28u);
```

```
  v6 = strlen(a1);
  v7 = (const char *)malloc(v6 / 12 + 1);
  v3 = 0;
  v4 = 0;
LABEL_8:
  if ( v3 < v6 )
  {
    for ( i = 0; i <= 8; ++i )
    {
      if ( !ofwoa((const char *)*(&lfwp + i), &a1[v3]) )
      {
        v3 += strlen((const char *)*(&lfwp + i));
        v7[v4++] = pwfl[i];
        goto LABEL_8;
      }
    }
    return 1LL;
  }
  else
  {
    v7[v4] = 0;
    trof_pripew(v7);
    v2 = 0;
    __isoc99_sscanf(v7, "%o", &v2);
    return v2;
  }
}
```

```
// a2 = input[some]
int __fastcall ofwoa(const char *a1, const char *a2)
{
  size_t v2; // rax

  v2 = strlen(a1);
  return strncmp(a1, a2, v2);
}
```

```
.data:00005650D7615010 pwfl            db '01234567-',0
```

In this function, the input value is changed to a 0 to 7 string (eight digits), and the replaced octal string is reversed again and return after converting to `int`
That is, it is a decryption code.
Therefore, you can decode the output value to obtain the value used in the operation, find out which operation is applied, and then encrypt and transmit the result value of the operation.

Writing the code is as follows.

```
#cal.py

lfwp = [
    b'\xE3\x83\xBB\xE2\x94\xA4\xE2\x94\xA4\xE2\x95\x99'.decode(),
    b'\xE3\x83\xBB\xE2\x94\x9C\xE2\x94\xB4\xE2\x95\x97\xE2\x95\xAC'.decode(),
    b'\xE3\x83\xBB\xE2\x95\x9D\xE2\x94\x94\xE2\x94\xA4\xE2\x94\x90\xE2\x94\xBC'.decode(),
    b'\xE3\x83\xBB\xE2\x94\xAC\xE2\x95\x92\xE2\x94\x98\xE2\x94\x80\xE2\x94\x94\xE2\x94\xB4'.decode(),
    b'\xE3\x83\xBB\xE2\x94\x9C\xE2\x95\xA1\xE2\x94\x8C\xE2\x94\x94\xE2\x94\xAC\xE2\x95\xA5'.decode(),
    b'\xE3\x83\xBB\xE2\x94\xB4\xE2\x94\x90\xE2\x94\xA4\xE2\x94\xAC'.decode(),
    b'\xE3\x83\xBB\xE2\x94\x94\xE2\x94\x80\xE2\x94\x90\xE2\x94\xA4\xE2\x94\x80\xE2\x94\xB4\xE2\x94\xB4'.decode(),
    b'\xE3\x83\xBB\xE2\x94\xAC\xE2\x95\xA7\xE2\x94\x80\xE2\x94\x98\xE2\x95\xA3\xE2\x94\x90'.decode(),
    b'\xE3\x83\xBB\xE2\x94\x80'.decode(),
]

lfwp_dec = {
    b'\xE3\x83\xBB\xE2\x94\xA4\xE2\x94\xA4\xE2\x95\x99'.decode() : '0',
    b'\xE3\x83\xBB\xE2\x94\x9C\xE2\x94\xB4\xE2\x95\x97\xE2\x95\xAC'.decode(): '1',
    b'\xE3\x83\xBB\xE2\x95\x9D\xE2\x94\x94\xE2\x94\xA4\xE2\x94\x90\xE2\x94\xBC'.decode(): '2',
    b'\xE3\x83\xBB\xE2\x94\xAC\xE2\x95\x92\xE2\x94\x98\xE2\x94\x80\xE2\x94\x94\xE2\x94\xB4'.decode(): '3',
    b'\xE3\x83\xBB\xE2\x94\x9C\xE2\x95\xA1\xE2\x94\x8C\xE2\x94\x94\xE2\x94\xAC\xE2\x95\xA5'.decode(): '4',
```

```
    b'\xE3\x83\xBB\xE2\x94\xB4\xE2\x94\x90\xE2\x94\xA4\xE2\x94\xAC'.decode() : '5',
    b'\xE3\x83\xBB\xE2\x94\x94\xE2\x94\x80\xE2\x94\x90\xE2\x94\xA4\xE2\x94\x80\xE2\x94\xB4\xE2\x94\xB4'.decode() : '6',
    b'\xE3\x83\xBB\xE2\x94\xAC\xE2\x95\xA7\xE2\x94\x80\xE2\x94\x98\xE2\x95\xA3\xE2\x94\x90'.decode(): '7',
}


lfwp_enc = {
    '0': b'\xE3\x83\xBB\xE2\x94\xA4\xE2\x94\xA4\xE2\x95\x99'.decode(),
    '1': b'\xE3\x83\xBB\xE2\x94\x9C\xE2\x94\xB4\xE2\x95\x97\xE2\x95\xAC'.decode(),
    '2': b'\xE3\x83\xBB\xE2\x95\x9D\xE2\x94\x94\xE2\x94\xA4\xE2\x94\x90\xE2\x94\xBC'.decode(),
    '3': b'\xE3\x83\xBB\xE2\x94\xAC\xE2\x95\x92\xE2\x94\x98\xE2\x94\x80\xE2\x94\x94\xE2\x94\xB4'.decode(),
    '4': b'\xE3\x83\xBB\xE2\x94\x9C\xE2\x95\xA1\xE2\x94\x8C\xE2\x94\x94\xE2\x94\xAC\xE2\x95\xA5'.decode(),
    '5': b'\xE3\x83\xBB\xE2\x94\xB4\xE2\x94\x90\xE2\x94\xA4\xE2\x94\xAC'.decode(),
    '6': b'\xE3\x83\xBB\xE2\x94\x94\xE2\x94\x80\xE2\x94\x90\xE2\x94\xA4\xE2\x94\x80\xE2\x94\xB4\xE2\x94\xB4'.decode(),
    '7': b'\xE3\x83\xBB\xE2\x94\xAC\xE2\x95\xA7\xE2\x94\x80\xE2\x94\x98\xE2\x95\xA3\xE2\x94\x90'.decode(),
}

ptr_split = '。 ┴┘╟│ '
as0_split = '╞└╟╤├┼│ '
as1_split = '──╖┴╚┬╝═┴└╪┤┤'
as2_split = '┘└└┼┤┐╤┘┤┤╜ '
as3_split = '─╥╥┘└╙ '
as4_split = '┘┘│┬╥├ '
as5_split = '┼└╩╪╝╚│┴ '
as_split = [as0_split, as1_split, as2_split, as3_split, as4_split, as5_split]
s = '╜═─┬┼ │┘┘│╒┼ ┬┼─┤┴╣┼─ ┼┐┘┐└╨┴╓│ ┬├╪┘╚├。 '.encode()
bytes1 = '╨│┼╫╜┐┴┘╪ ┼╞═╣ ╖╪┴┼┴┬╝─ ┐├┬╫╢╡╨ ┤╞│║┐╘┴├'.encode()
bytes2 = '│┤╧│╬─ ╕╤┴┘ └┘┤┌╒┘ ┌┴╖┼═╪ ┘╒┴││├│？'.encode()
q_split = '？' # str


def dec(enc_str:str):
    res = 0
    tmp = enc_str
    for d in lfwp_dec.keys():
        tmp = tmp.replace(d, lfwp_dec[d])

    tmp = tmp[::-1]
    res = int(tmp, 8)
    return res

def enc(target:int):

    tmp = oct(target).replace('0o', '')[::-1]
    for d in lfwp_enc.keys():
        tmp = tmp.replace(d, lfwp_enc[d])
    return tmp
```

```
#exploit.py
import cal
from pwn import *

def solve():
    p.recvuntil(b"You can run the solver with:\n")
    a = p.recvline().strip()
    solution = subprocess.check_output(a, shell=True, executable="/bin/bash")
    print(solution)
    p.sendline(solution)

p = remote('alien.chal.irisc.tf', 10600)
#p = process('./alien_math') #in local

def tohex(val, nbits):
  return hex((val + (1 << nbits)) % (1 << nbits))

def calculation():

    datas = p.recv().decode().replace('\n', '')

    tmp = datas.split(cal.ptr_split)

    target = tmp[1].replace(cal.q_split, '')
    print('trial: ' + str(cal.dec(tmp[0])))

    cnt = 0
```

```
    for s in cal.as_split:
        if s in target:
            target = target.split(s)
            break
        cnt += 1
    res = 0
    a = cal.dec(target[0])
    b = cal.dec(target[1])
    if(cnt == 0):
        if(a > b):
            res = a-b
        else:
            res = int(tohex(a-b, 32), 16)
    elif(cnt == 1):
        res = b ^ a
    elif(cnt == 2):
        res = (b*3) // a
    elif(cnt == 3):
        res = (3 * a) %(3 * b)
    elif(cnt == 4):
        res = a**2 + b
    elif(cnt == 5):
        res = (2*a) - (2*b)

    return res

solve()

p.recvuntil(cal.bytes2)

for i in range(68):
    res = calculation()
    d = cal.enc(res)
    p.sendline(d.encode())
    sleep(0.15)

p.interactive()
```

```
trial: 63
trial: 64
trial: 65
trial: 66
trial: 67
trial: 68
[*] Switching to interactive mode
┘╤│╧ ┘╪├╨┤┬┘ ╣║┐ ╜╗┘┘┬┤╥┬ ┬┬┤┴╥╟┐┘
┘╫┼╤╘╤┤ ┴┼┌╨╟╜┌┘└ ╝╨┴╥─ ╗╬┐├┬┘├ ─╫┘╞╓╖ ╚╧╪╬┴─!
irisctf{w3_are_4_f1ng3r3d_cr34tur3s}
[*] Got EOF while reading in interactive
$
```

`flag: irisctf{w3_are_4_f1ng3r3d_cr34tur3s}`