

WRITEUP EKOPARTY PRECTF 2016 by @fz

BACKDOOR

- Con 7z descomprimi el archivo que resulto ser un binario de vsftpd
- Haciendo `strings` sobre el archivo me entere que era la version 3.0.3 por lo que busque el codigo fuente, lo compile e hice un diff entre los strings de los dos binarios; una cosa que salto inmediatamente es que el backdoor tenia `execl`.
- Usando radare2 busque las referencias de donde se usaba excel y comparando con el binario original me di cuenta que estaba reemplazando la funcion `str_contains_space` que se ejecuta en el chequeo del comando `USER` informacion que resulto ser util para debuggear con gdb.
- Leyendo el ASM de esta nueva funcion deduje que el string se esperaba que sea de 18 chars (cmp eax, 0x12) y que el string era validado con la regla:

```
#13 + #14 = 0xcb
#13 + #12 = 0xcf
#10 + #11 = 0xc3
#8 + #9 = 0xd4
#12 + #11 = 0xc9
#0 = 'E'
#15 + #14 = 0xd7
#5 + #6 = 0xd9
#10 + #9 = 0xc3
#4 + #5 = 0xe9
#3 + #4 = 0xf1
#16 + #17 = 0xf2
#1 + #2 = 0x9a
#3 + #2 = 0xca
#1 = 'K'
#7 + #6 = 0xda
#7 + #8 = 0xe4
#16 + #15 = 0xeb
```

Luego corri el script que adjunto (backdoor.py) y obtuve la bandera

```
$ 7z x backdoor
$ radare2 vsftpd
$ strings vsftpd-3.0.3/vsftpd > 1
$ strings vsftpd > 2
$ diff -Nru 1 2
```

```
(...)
@@ -11,6 +11,7 @@
chroot
socket
fchmod
+execl
stand
(...)
```

```
$ radare2 vftpd
[0x...] V
```

p
 p (para ver asm)
 _ (para buscar el symbol)
 execl

```
> execl
- 0x00221fb0 reloc.execl_176
  0x00000000 sym.imp.execl
  0x00003fd8 sub.execl_176_fd8
```

Despues segui a ver quien llamaba a la función que tiene un xref a execl y era:

```
[0x00015ce4 62% 171 vsftpd]> pd $r @ fcn.00015c20+196 # 0x15ce4
||| 0x00015ce4 e867e0feff call sub.dup2_24_d50
||| 0x00015ce9 89df mov edi, ebx
||| 0x00015ceb be01000000 mov esi, 1
||| 0x00015cf0 e85be0feff call sub.dup2_24_d50
||| 0x00015cf5 89df mov edi, ebx
||| 0x00015cf7 be02000000 mov esi, 2
||| 0x00015cfc e84fe0feff call sub.dup2_24_d50
||| 0x00015d01 31d2 xor edx, edx
||| 0x00015d03 4c89ee mov rsi, r13
||| 0x00015d06 4c89e7 mov rdi, r12
||| 0x00015d09 31c0 xor eax, eax
||| 0x00015d0b c644241573 mov byte [rsp + local_15h]
||| 0x00015d10 c64424136e mov byte [rsp + local_13h]
||| 0x00015d15 c644241162 mov byte [rsp + local_11h]
||| 0x00015d1a c644241700 mov byte [rsp + local_17h]
||| 0x00015d1f c644241668 mov byte [rsp + local_16h]
||| 0x00015d24 c64424142f mov byte [rsp + local_14h]
||| 0x00015d29 c644241269 mov byte [rsp + local_12h]
||| 0x00015d2e c64424102f mov byte [rsp + local_10h]
||| 0x00015d33 c6042473 mov byte [rsp], 0x73
||| 0x00015d37 c644240168 mov byte [rsp + local_1h],
||| 0x00015d3c c644240200 mov byte [rsp + local_2h],
||| 0x00015d41 e892e2feff call sub.execl_176_fd8
||| 0x00015d46 e96dffffff jmp 0x15cb8
||| 0x00015d4b bf01000000 mov edi, 1
||| 0x00015d50 e863e2feff call sub.exit_136_fb8
||| 0x00015d55 90 nop
||| 0x00015d56 662e0f1f8400. nop word cs:[rax + rax]
/ (fcn) fcn.00015d60 15
||| ; CALL XREF from 0x000045ec (main)
||| ; CALL XREF from 0x0000f57b (fcn.0000f530)
||| 0x00015d60 4889fe mov rsi, rdi
||| 0x00015d63 488d3d66dc20. lea rdi, [rip + 0x20dc66]
```

Que tiene un xref a 0xcccd con la logica mencionada anteriormente:

```
||||| 0x0000cd19 0f1f80000000. nop dword [rax]
||||| 0x0000cd20 488b07 mov rax, qword [rdi]
||||| 0x0000cd23 0f8b480d movsx ecx, byte [rax + 0xd]
||||| 0x0000cd27 0f8b500e movsx edx, byte [rax + 0xe]
||||| 0x0000cd2b 8d3411 lea esi, [rcx + rdx]
||||| 0x0000cd2e 81fecb000000 cmp esi, 0xcb
|||< 0x0000cd34 75a3 jne 0xcccd9
||||| 0x0000cd36 0f8b700a movsx esi, byte [rax + 0xa]
||||| 0x0000cd3a 0f8b780b movsx edi, byte [rax + 0xb]
||||| 0x0000cd3e 448d043e lea r8d, [rsi + rdi]
||||| 0x0000cd42 4181f8c30000. cmp r8d, 0xc3
|||< 0x0000cd49 758e jne 0xcccd9
||||| 0x0000cd4b 440f8b480c movsx r9d, byte [rax + 0xc]
||||| 0x0000cd50 4401c9 add ecx, r9d
||||| 0x0000cd53 81f9cf000000 cmp ecx, 0xcf
|||< 0x0000cd59 0f857affffff jne 0xcccd9
||||| 0x0000cd5f 440f8b4008 movsx r8d, byte [rax + 8]
||||| 0x0000cd64 440f8b5009 movsx r10d, byte [rax + 9]
||||| 0x0000cd69 438d0c10 lea ecx, [r8 + r10]
||||| 0x0000cd6d 81f9d4000000 cmp ecx, 0xd4
|||< 0x0000cd73 0f8560ffffff jne 0xcccd9
||||| 0x0000cd79 4401cf add edi, r9d
||| 0x0000cd7c 81ffc9000000 cmp edi, 0xc9
|||< 0x0000cd82 0f8551ffffff jne 0xcccd9
||| 0x0000cd88 803845 cmp byte [rax], 0x45
||< 0x0000cd8b 0f8548ffffff jne 0xcccd9
||| 0x0000cd91 0f8b480f movsx ecx, byte [rax + 0xf]
||||| 0x0000cd95 01ca add edx, ecx
||||| 0x0000cd97 81fad7000000 cmp edx, 0xd7
|||< 0x0000cd9d 0f8536ffffff jne 0xcccd9
```

Backdoor.py

```
known = {
    0: 'E',
    1: 'K',
    2: 'O',
    3: '{',
    17: '}',
}

equation = [
    (13, 14, 0xcb),
    (13, 12, 0xcf),
    (10, 11, 0xc3),
    (8, 9, 0xd4),
    (12, 11, 0xc9),
    (15, 14, 0xd7),
    (5, 6, 0xd9),
    (10, 9, 0xc3),
    (4, 5, 0xe9),
    (3, 4, 0xf1),
    (16, 17, 0xf2),
    (1, 2, 0x9a),
    (3, 2, 0xca),
    (7, 6, 0xda),
    (7, 8, 0xe4),
    (16, 15, 0xeb),
]

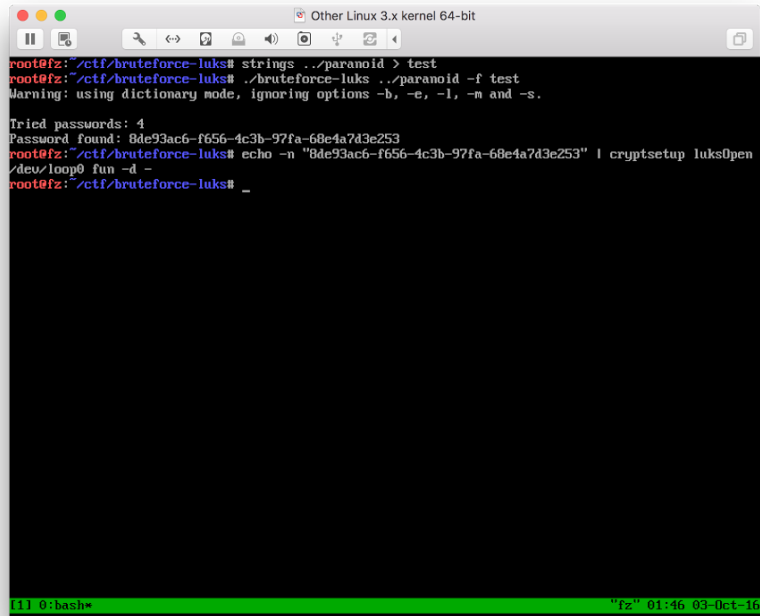
while len(known) < 17:
    for a, b, total in equation:
        if a in known and b in known:
            continue

        if a in known:
            known[b] = chr(total - ord(known[a]))
        elif b in known:
            known[a] = chr(total - ord(known[b]))

print "".join(c for i, c in sorted(known.items()))
```

PARANOID

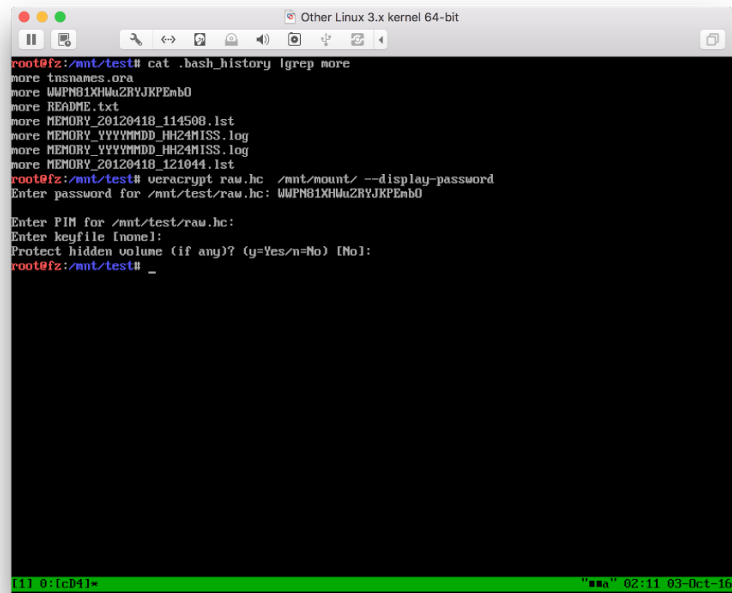
- Primero hice un brute force en el archivo luks usando como diccionario el output del comando `strings paranoid`



```
root@fz:~/ctf/bruteforce-luks# strings ../paranoid > test
root@fz:~/ctf/bruteforce-luks# ./bruteforce-luks ../paranoid -f test
Warning: using dictionary mode, ignoring options -b, -e, -l, -n and -s.

Tried passwords: 4
Password found: 8de93ac6-f656-4c3b-97fa-68e4a7d3e253
root@fz:~/ctf/bruteforce-luks# echo -n "8de93ac6-f656-4c3b-97fa-68e4a7d3e253" | cryptsetup luksOpen
/dev/loop0 fun -d -
root@fz:~/ctf/bruteforce-luks# _
```

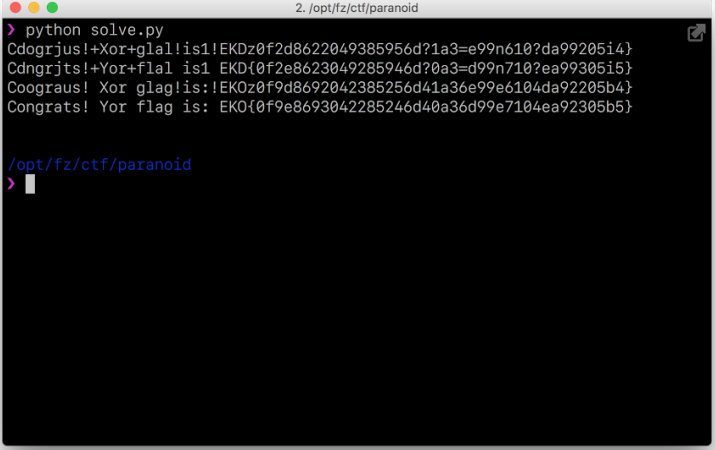
- En el archivo montado encontré un `.bash_history` con un more medio raro; probando me di cuenta que era la clave del veracrypt (por la extension del archivo me di cuenta que era un veracrypt)



```
root@fz:~/mnt/test# cat .bash_history |grep more
more tnsnames.ora
more WJPNB1XHMuZRYJKPEmb0
more README.txt
more MEMORY_20120418_114508.lst
more MEMORY_VVVVMMDD_HH24MISS.log
more MEMORY_VVVVMMDD_HH24MISS.log
more MEMORY_20120418_121044.lst
root@fz:~/mnt/test# veracrypt raw.hc /mnt/mount/ --display-password
Enter password for /mnt/test/raw.hc: WJPNB1XHMuZRYJKPEmb0

Enter PIN for /mnt/test/raw.hc:
Enter keyfile [none]:
Protect hidden volume (if any)? (y=Yes/n=No) [No]:
root@fz:~/mnt/test# _
```

- Ahi se encontraba un flag.cipher que depuse de _MUCHO_ probar me di cuenta que era un XOR (en realidad me costo un huevo porque asumi siempre que empezaba con EKO{)... Use el script que adjunto para brute forcearlo:

A terminal window with a black background and white text. The title bar at the top shows three colored circles (red, yellow, green) and the text '2. /opt/fz/ctf/paranoid'. The terminal content shows a command prompt followed by 'python solve.py'. The output consists of four lines of text, each containing a ciphered string followed by a closing curly brace. The last line indicates the flag has been found. The prompt then changes to '/opt/fz/ctf/paranoid' and a new command prompt is shown.

```
> python solve.py
Cdogrjus!+Xor+glal!is1!EKDz0f2d8622049385956d?1a3=e99n610?da99205i4}
Cdngrjts!+Yor+flal is1 EKD{0f2e8623049285946d?0a3=d99n710?ea99305i5}
Coograus! Xor glag!is:!EK0z0f9d8692042385256d41a36e99e6104da92205b4}
Congrats! Yor flag is: EKO{0f9e8693042285246d40a36d99e7104ea92305b5}

/opt/fz/ctf/paranoid
> 
```

Paranoid.py

```
import itertools
import string

from collections import defaultdict

cipher = open("flag.cipher").read()

def xor(key, cipher):
    content = ""
    key_iter = itertools.cycle(key)
    for letter in cipher:
        k = key_iter.next()
        content += chr(ord(k) ^ ord(letter))

    return content

def valid(decrypted):
    alphabet = string.ascii_letters + string.digits + "=+/\n\r\t\"'{}!?:"
    return all(x in alphabet for x in decrypted)

def check(position, key, cipher):
    decrypted = xor(key, cipher)
    parts = [decrypted[i:i + len(key)]
              for i in xrange(0, len(decrypted), len(key))]

    return all(valid(p[position:position + 1]) for p in parts)

key = ["\x00", "\x00", "\x00", "\x00"]
possible = defaultdict(list)
for i in xrange(len(key)):
    for c in map(chr, xrange(255)):
        key[i] = c
        if check(i, key, cipher):
            possible[i].append(c)

for f in possible[0]:
    for s in possible[1]:
        for t in possible[2]:
            for c in possible[3]:
                print xor(f + s + t + c, cipher)
```

CODEOP

- Primero traduje los opcodes de python que están en el archivo a un script mas o menos legible (ver constructed.py).

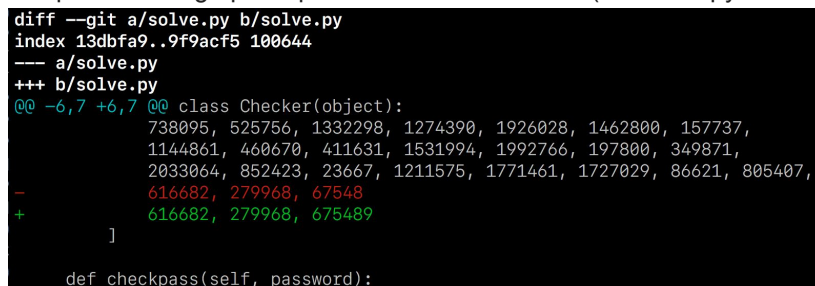


```
1. /opt/fz/ctf/opc
> python constructed.py > my_opc

/opt/fz/ctf/opc master*
> diff -Nrbu my_opc opc
--- my_opc      2016-10-02 12:09:56.000000000 -0700
+++ opc 2016-10-02 11:44:14.000000000 -0700
@@ -121,4 +121,3 @@
                182 LOAD_FAST          4 (00000000000000000000)
                185 COMPARE_OP        2 (==)
                188 RETURN_VALUE

/opt/fz/ctf/opc master*
>
```

- Segundo limpie un poco el código para que se entienda mas fácil (ver solve.py - método checkpass).

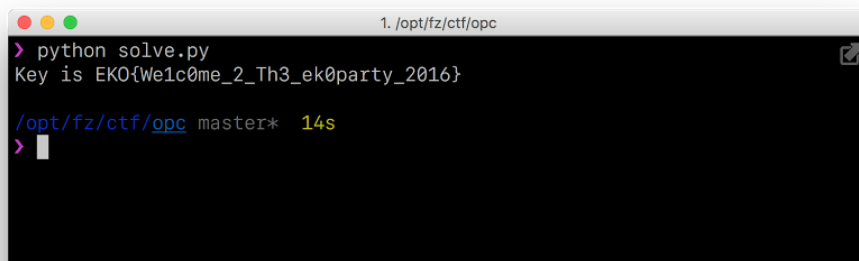


```
diff --git a/solve.py b/solve.py
index 13dbfa9..9f9acf5 100644
--- a/solve.py
+++ b/solve.py
@@ -6,7 +6,7 @@ class Checker(object):
    738095, 525756, 1332298, 1274390, 1926028, 1462800, 157737,
    1144861, 460670, 411631, 1531994, 1992766, 197800, 349871,
    2033064, 852423, 23667, 1211575, 1771461, 1727029, 86621, 805407,
-   616682, 279968, 67548
+   616682, 279968, 675489
    ]

    def checkpass(self, password):
```

- Tercero arme un script que hace brute force de todo el campo de las posibilidades utilizadas en las partes aleatorias, osea:

- El indice que usa del string => (0, 32)
- La parte del algoritmo que usa un entero entre (1, 255)
- Con el indice el algoritmo accede a esa parte del string que recibe la función por lo cual también hice brute force de esas letras (que finalmente me dieron la flag) => Para esta parte use string.printable



```
1. /opt/fz/ctf/opc
> python solve.py
Key is EK0{We1c0me_2_Th3_ek0party_2016}

/opt/fz/ctf/opc master* 14s
>
```

Codeop.py

```
class Checker(object):

    def __init__(self):
        self.password = [
            919161, 1859495, 985017, 1377995, 1659485, 1068148, 1599708,
            738095, 525756, 1332298, 1274390, 1926028, 1462800, 157737,
            1144861, 460670, 411631, 1531994, 1992766, 197800, 349871,
            2033064, 852423, 23667, 1211575, 1771461, 1727029, 86621, 805407,
            616682, 279968, 675489
        ]

    def checkpass(self, password):
        from random import shuffle
        from random import randint

        result = []
        indices = [i for i in range(len(password))]
        shuffle(indices)

        for i in indices:
            letter = (i ^ 19) << 16
            letter += (ord(password[i]) ^ 55) << 8
            letter += randint(1, 255)
            result.append(letter)

        return self.password == result

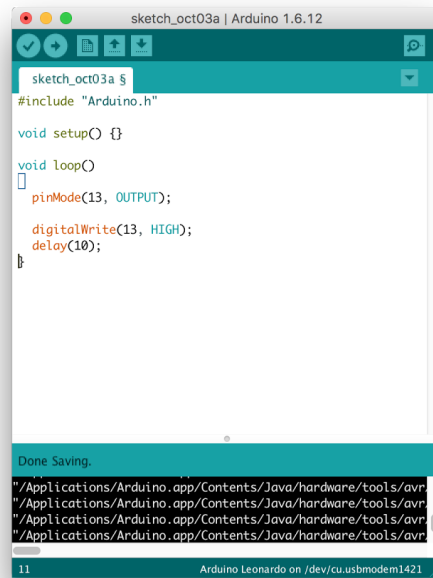
    def bruteforce(self):
        import string
        answer = ["?"] * len(self.password)
        for right in self.password:
            for idx in xrange(len(self.password)):
                for letter_ord in xrange(255):
                    for i in xrange(1, 255):
                        n = (idx ^ 19) << 16
                        n += (letter_ord ^ 55) << 8
                        n += i
                        if n == right:
                            answer[idx] = chr(letter_ord)

        return "".join(answer)

if __name__ == "__main__":
    checker = Checker()
    print "Key is", checker.bruteforce()
```


ROBOTO

- Usando strings me di cuenta que el elf/ihex eran un programa de arduino corriendo en la version Leonardo
- Con arduino IDE compile un main chiquito con un delay/write para ver como se veía el asm.



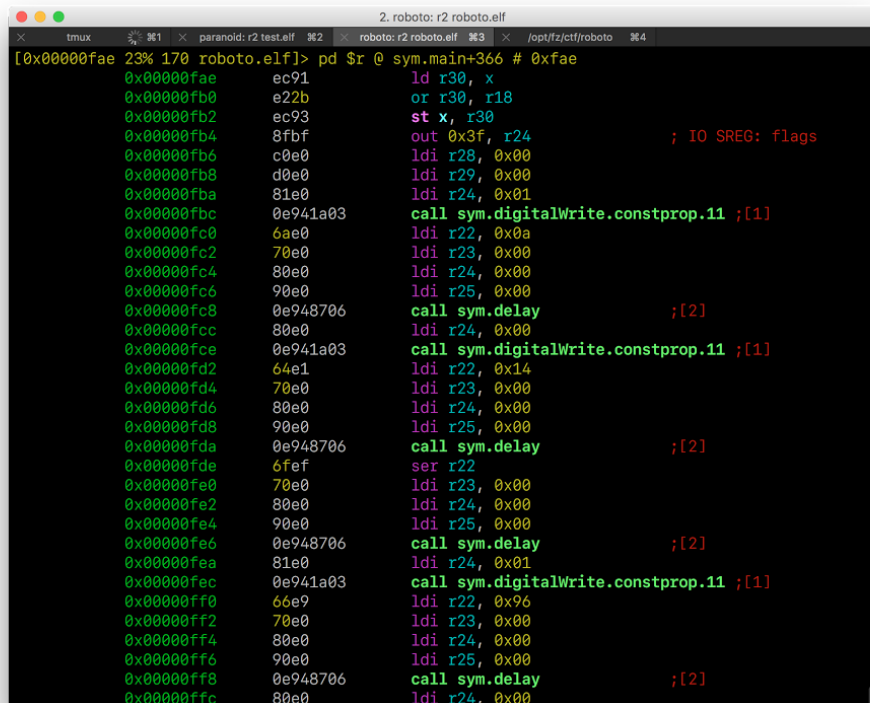
```
sketch_oct03a $  
#include "Arduino.h"  
  
void setup() {}  
  
void loop()  
{  
  pinMode(13, OUTPUT);  
  
  digitalWrite(13, HIGH);  
  delay(10);  
}
```

Done Saving.

"/Applications/Arduino.app/Contents/Java/hardware/tools/avr"
"/Applications/Arduino.app/Contents/Java/hardware/tools/avr"
"/Applications/Arduino.app/Contents/Java/hardware/tools/avr"
"/Applications/Arduino.app/Contents/Java/hardware/tools/avr"

11 Arduino Leonardo on /dev/cu.usbmodem1421

- Usando radare2 desensamble roboto.elf y con mi programa de referencia encontré donde empezaba el código (que parte de sym.main).



```
2. roboto: r2 roboto.elf  
[0x00000fae 23% 170 roboto.elf]> pd $r @ sym.main+366 # 0xfae  
ec91      ld r30, x  
e22b      or r30, r18  
ec93      st x, r30  
8fbf      out 0x3f, r24          ; IO SREG: flags  
c0e0      ldi r28, 0x00  
d0e0      ldi r29, 0x00  
81e0      ldi r24, 0x01  
0e941a03  call sym.digitalWrite.constprop.11 ;[1]  
6ae0      ldi r22, 0x0a  
70e0      ldi r23, 0x00  
80e0      ldi r24, 0x00  
90e0      ldi r25, 0x00  
0e948706  call sym.delay          ;[2]  
80e0      ldi r24, 0x00  
0e941a03  call sym.digitalWrite.constprop.11 ;[1]  
64e1      ldi r22, 0x14  
70e0      ldi r23, 0x00  
80e0      ldi r24, 0x00  
90e0      ldi r25, 0x00  
0e948706  call sym.delay          ;[2]  
6fef      ser r22  
70e0      ldi r23, 0x00  
80e0      ldi r24, 0x00  
90e0      ldi r25, 0x00  
0e948706  call sym.delay          ;[2]  
81e0      ldi r24, 0x01  
0e941a03  call sym.digitalWrite.constprop.11 ;[1]  
66e9      ldi r22, 0x96  
70e0      ldi r23, 0x00  
80e0      ldi r24, 0x00  
90e0      ldi r25, 0x00  
0e948706  call sym.delay          ;[2]  
80e0      ldi r24, 0x00
```

- Usando avr-objdump dumpie el asm del elf y manualmente removi todo excepto por todos los delay/write que se estaban haciendo (ver sequence.asm).
- Escribi un script en python que "emula" los write/delays y dibuje un PNG como si fuera un osciloscopio:

Esta fue la imagen (sorry tiene 6000px de ancho):



- Me di cuenta que era morse y agregue al script la traducción de timing a morse y de morse a ascii.

```
2. /opt/fz/ctf/roboto
tmux  第1 x paranoid: r2 test.elf 第2 x roboto: r2 roboto.elf 第3 x /opt/fz/ctf/roboto 第4 x fz: python 第5
Assigning r24 = 0x00
Assigning r25 = 0x00
Calling delay(10)
Assigning r24 = 0x00
Calling digitalWrite(0)
Assigning r22 = 0x14
Assigning r23 = 0x00
Assigning r24 = 0x00
Assigning r25 = 0x00
Calling delay(20)
Assigning r24 = 0x01
Calling digitalWrite(1)
Assigning r22 = 0x96
Assigning r23 = 0x00
Assigning r24 = 0x00
Assigning r25 = 0x00
Calling delay(150)
Assigning r24 = 0x00
Calling digitalWrite(0)
Assigning r22 = 0x14
Assigning r23 = 0x00
Assigning r24 = 0x00
Assigning r25 = 0x00
Calling delay(20)
Assigning r22 = 0xFF
Assigning r23 = 0x00
Assigning r24 = 0x00
Assigning r25 = 0x00
Calling delay(255)
The key is: EKO{OLD.IS.NEW.AGAIN}

/opt/fz/ctf/roboto
> |
```

Roboto.py

```
from operator import itemgetter
from PIL import Image

calls = {
    "0x634": ("digitalWrite", ("r24", )),
    "0xd0e": ("delay", ("r22", "r23", "r24", "r25")),
}

registers = {
    "r22": 0x0,
    "r23": 0x0,
    "r24": 0x0,
    "r25": 0x0,
}

MORSE = {
    (0, 150): "-",
    (0, 10): ".",
    (99, 255): " "
}

ALPHABET = {
    ".-": "A", "-...": "B", "-.-.": "C", "-..": "D", ".": "E", "..-": "F",
    "--": "G", "...": "H", ". .": "I", ".---": "J", "-.-": "K", "-..": "L",
    "--": "M", "-": "N", "---": "O", "-.-": "P", "--.-": "Q", ".-": "R",
    "...": "S", "-": "T", ". .-": "U", "...-": "V", ".--": "W", "-.-": "X",
    "-.-": "Y", "-...": "Z", "-.-.-": "{", "-.-.-": "}", "-.-.-": ".",
    " ": " "
}

class Plot(object):

    def __init__(self, height=100):
        self._height = height
        self._cursor = (0, height - 1)
        self._img = Image.new('RGB', (6024, height), "black")
        self._pixels = self._img.load()
        self._morse = ""

    def digitalWrite(self, bit):
        x, y = self._cursor
        y_dest = -1 if bit else self._height
        for y in xrange(y, y_dest, -1 if bit else 1):
            self._pixels[x, y] = (0xFF, 0x00, 0x00)
            self._cursor = (x, y)

    def delay(self, ms):
        x, y = self._cursor
        x_dest = x + (ms / 2)

        if (y, ms) in MORSE:
            self._morse += MORSE[(y, ms)]

        for x in xrange(x, x_dest):
            self._pixels[x, y] = (0xFF, 0x00, 0x00)
            self._cursor = (x, y)

    def draw(self):
```

```

        self._img.save("whatever.png")

    def morse(self):
        words = map(ALPHABET.__getitem__, self._morse.strip().split(" "))
        return "".join(words)

plot = Plot()
for instruction in open("sequence.asm"):
    opcode, args = instruction.strip().split("\t")
    args = args.split(", ")
    if opcode == "call":
        address = args[0]
        function, arg_regs = calls[address]

        value = 0
        for i, arg_reg in enumerate(arg_regs):
            reg = registers[arg_reg]
            value |= (reg << (i * 8))

        print "Calling %s(%d)" % (function, value)
        getattr(plot, function)(value)

    elif opcode == "ldi":
        register, value = args
        if register not in registers:
            raise ValueError("Invalid register %s" % register)

        registers[register] = int(value[2:], 16)
        print "Assigning %s = %s" % (register, value)

plot.draw()
print "The key is:", plot.morse()

```

Sequence.asm

```

ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00

```

```
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
```

```
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
```

```
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r22, 0xFF
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
```

```
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
```



```
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r22, 0xFF
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
```

```
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
```

```
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
```

```
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
```

```
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
```

```
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r22, 0xFF
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r22, 0xFF
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
```

```
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
```

```
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
```



```
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
```

```
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
```

```
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r22, 0xFF
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r22, 0xFF
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x96
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x00
call    0x634
ldi     r22, 0x14
ldi     r23, 0x00
ldi     r24, 0x00
ldi     r25, 0x00
call    0xd0e
ldi     r24, 0x01
call    0x634
ldi     r22, 0x0A
ldi     r23, 0x00
```

```
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
```

```
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x0A
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x01
call   0x634
ldi    r22, 0x96
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r24, 0x00
call   0x634
ldi    r22, 0x14
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
ldi    r22, 0xFF
ldi    r23, 0x00
ldi    r24, 0x00
ldi    r25, 0x00
call   0xd0e
```