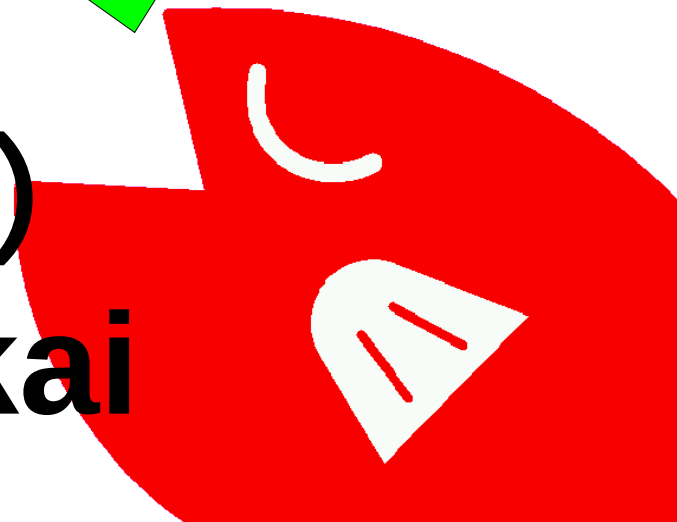


アセンブラ短歌

坂井弘亮

(KOZOSプロジェクト)

Twitter ID:kozossakai



突然ですが

SECCON

2013

SECCONとは何か

**セキュリティ競技の
コンテストです**

去年はCTFという
競技大会が
メインでしたが

今年のSECCONでは
様々なコンテストが
行われるらしい

例えば

回路設計とか

アセンブラ短歌とか

アセンブラ短歌？

アセンブラ短歌とは

5・7・5・7・7の機械語
コードでプログラムを
書いてみるという
近未来の文化的趣味

こんな感じです

68	72	6c	64	21		
68	6f	20	57	6f	90	90
68	48	65	6c	6c		
89	e5	6a	0c	55	6a	02
50	90	6a	04	58	cd	80

実行結果

Hello World!

それならば、
これでもできるだろうと
いうことで

アセンブラ川柳

6a	48	89	e5	90		
6a	01	55	6a	02	50	90
6a	04	58	cd	80		

実行結果

Н

1文字が
せいいっぱいでした

下の句が無いと、
けっこう難しい

まてよ

ということとは

こういうのも
可能ということか

アセンブラ
かるた

作ってみた

68	a5	f3	a5	b0		
68	a1	bc	a5	ea	90	90
68	a5	d5	a5	a3		
89	e5	6a	0c	55	6a	02
50	90	6a	04	58	cd	80

実行 → フィーリング

68	00	00	a5	a2		
68	a5	bb	a5	f3	90	90
68	a5	d6	a5	e9		
89	e5	6a	0c	55	6a	02
50	90	6a	04	58	cd	80

実行 → ブラセンア

しかし問題あり

バイナリダンプを
見ると

00000000	68	a5	f3	a5	b0	68	a1	bc	hングhー
00000008	a5	ea	90	90	68	a5	d5	a5	リ..hフ
00000010	a3	89	e5	6a	0c	55	6a	02	.j.Uj.
00000018	50	90	6a	04	58	cd	80		P.j.X.

00000000	68	00	00	a5	a2	68	a5	bb	h..アhセ
00000008	a5	f3	90	90	68	a5	d6	a5	ン..hブ
00000010	e9	89	e5	6a	0c	55	6a	02	.j.Uj.
00000018	50	90	6a	04	58	cd	80		P.j.X.

00000000	68	a5	f3	a5	b0	68	a1	bc
00000008	a5	ea	90	90	68	a5	d5	a5
00000010	a3	89	e5	6a	0c	55	6a	02
00000018	50	90	6a	04	58	cd	80	

	h	ン	グ	h	-	
	リ	..	h	フ		
	.	j	.	U	j	.
	P	.	j	.	X	.

00000000	68	00	00	a5	a2	68	a5	bb
00000008	a5	f3	90	90	68	a5	d6	a5
00000010	e9	89	e5	6a	0c	55	6a	02
00000018	50	90	6a	04	58	cd	80	

	h	..	ア	h	セ	
	ン	..	h	ブ		
	.	j	.	U	j	.
	P	.	j	.	X	.

つまり読み札を
読み上げた時点で
取り札が
わかってしまう

これでは
競技にできない

xorにかけて
難読化
してみる

b8	5b	0c	5a	29				
f7	d8	50	b8	5b	5d	5a	44	(字余り)
f7	d8	50	89	e5				
6a	08	55	6a	02	50	90		
90	90	6a	04	58	cd	80		

実行結果

アセンブ

4文字が
せいいっぱい
でした

バイナリダンプは

00000000	b8	5b	0c	5a	29	f7	d8	50	[.Z)P
00000008	b8	5b	5d	5a	44	f7	d8	50	[]ZDP
00000010	89	e5	6a	08	55	6a	02	50	.j.Uj.P
00000018	90	90	90	6a	04	58	cd	80	...j.X.

これなら
読めまい

ここからが
今日の本題

ふだんは趣味で
こんなことを
やっています

**GCCが対応している
CPUアーキテクチャの
アセンブラを**

片っ端から出力させて、
片っ端から読んでみる

対象アーキテクチャ

40種類

**Alpha ARC ARM ARM(Thumb) AVR
AVR(8bit) CRIS FR30 FR-V H8/300
H8/300H HP/PA i386 i960 IA-64 M32R
68HC11 68HC11(16bit) M68000 MCORE
MIPS MIPS16 MIPS64 MIST32 MMIX
MN10300 PDP-11 PowerPC PowerPC64
S/390 SH SH64 SPARC StrongARM V850
VAX x86-64 XScale Xstormy Xtensa**

この技術を
アセンブラ短歌に
活かしたい

アセンブラ短歌に
もっとも向いた
アーキテクチャは
何か？

ちなみに
他の言語では

「XX短歌」は
他の言語では
難しい

スクリプト言語は
これが邪魔

#!/bin/sh

(9 文字)

C言語は
これが邪魔

#include <stdio.h>

(18文字)

RISC系プロセッサ
→4バイト固定長命令
可変長命令でも
→偶数バイト命令が多

原理的に不可能

**アセンブラ短歌ができるのは
実は、以下くらいしか
ありませんでした**

**x86、68HC11、
MN10300、
VAX、Xtensa**

アセンブラ短歌には
OSSが
(主にGCCが)
必須

そして

現在、
アセンブラ出力環境の
GCCバージョン4対応を
やっています

以下のアーキが
利用可能に
なりました

Blackfin CR16 M32C
MicroBlaze Moxie
RL78 RX TIC6X

短歌的には
どうかか気になる

**以下のアーキで
原理的に可能な
ことが判明**

x86(Intel) M32C(三菱)

MN10300(松下)

RL78(ルネサス) RX(ルネサス)

Xtensa (テンシリカ)

やってみた

まず、RX

こんな感じか

```
mov.l #1, r1  
mov.l #10, r3  
nop
```

```
mov.l #0x6c6c6548, [r0]  
nop
```

下の句

```
mov.l #0x6c72, 8[r0]
```

```
mov.l #0x6f57206f, 4[r0]
```

上の句

```
mov.l r0, r2  
mov.l #5, r5  
int #255
```

機械語コード
にすると

こんな感じ

66	11	66	a3	03		
f8	02	48	65	6c	6c	03
f9	0a	02	72	6c		
f9	02	01	6f	20	57	6f
ef	02	66	55	75	60	ff

実行結果は

Hello Worl

(10文字)

次、RL78

こんな感じ

```
mov 0xffff10, #72  
mov a, #108
```

```
mov 0xffff10, #101  
mov 0xffff10, a  
mov 0xffff10, a  
  
mov 0xffff10, #111  
mov a, #32
```

上の句

下の句

```
mov 0xffff10, a  
mov 0xffff10, #87  
mov a, #111  
  
mov 0xffff10, a  
mov 0xffff10, #114  
nop  
nop
```

機械語コード

ce	10	48	51	6c		
ce	10	65	9e	10	9e	10
ce	10	6f	51	20		
9e	10	ce	10	57	51	6f
9e	10	ce	10	72	00	00

実行結果は

Hello Wor

(9文字)

他のは
こんな感じ

MN10300

Hello Wo (8文字)

M32C

Hello! (6文字)

i386/FreeBSD

Hello World! (12文字)

i386/Linux

Hello World!! (13文字)

比較してみる

比較の前に

定量化の
ためには
単位が必要

単位を制定する

BPT (Byte per Tanka)

... アセンブラ短歌の
やりやすさの指標

たとえばRXは

**Hello Worl で
10文字なので
10BPT**

集計すると

M32C

6BPT

MN10300

8BPT

RL78

9BPT

RX

10BPT

x86/FreeBSD

12BPT

x86/Linux

13BPT

**x86は非常に
短歌に向いている
ことが判明
(あとLinuxも)**

どうもありがとう
ございました

