実行ファイルのスペースに
ホワイトスペースを入れる

夏のプログラミング・シンポジウム 2024 (2024/09/21)

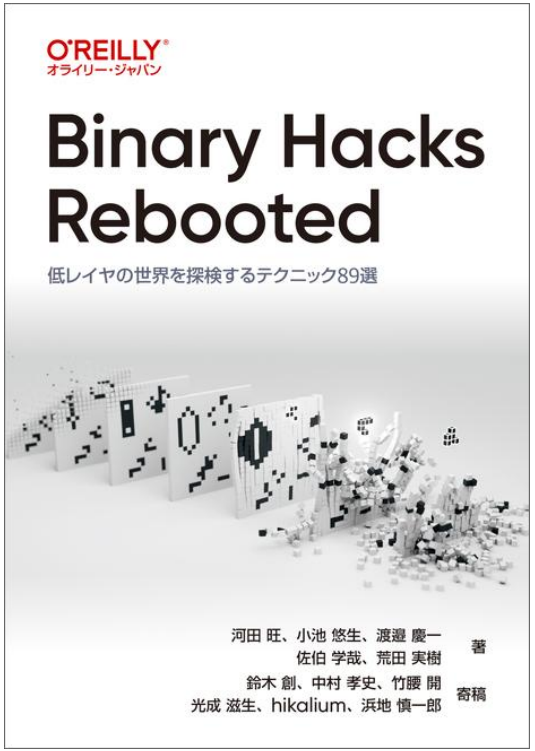
Akira Kawata
https://akawashiro.com/
@a_kawashiro

免責事項

- ここで紹介するテクニックはすでに知られているものです
- 悪用しないでください

自己紹介

- 河田 旺 (かわた あきら)
- Binary Hacks Rebooted の著者
  - 主に ELF ハックの章を担当
- 未踏 2018 年度で検索エンジンを作ったり



補足資料

```
$ git clone https://github.com/akawashiro/ps2024.git
$ cd ps2024
$ sudo docker build . --network=host -t ps2024
$ sudo docker run ps2024 /work/hello_with_lifegame
$ sudo docker run ps2024 /work/hello_with_whitespace
```

背景

スペース不足が問題になっている

- ディスク容量の消費は小さければ小さいほど嬉しい

```
$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
/dev/mapper/vgubuntu-root  912G  912G    0G 100% /
```

機能はあればあるほどよい

- 一つのバイナリができるだけ多くの機能を持っていると嬉しい
  - BusyBox には 270個のサブコマンドが存在

```
$ busybox
. . .
Currently defined functions:
    [, [[, acpid, adjtimex, ar, arch, arp, arping, ascii,
. . .
```

今求められている技術

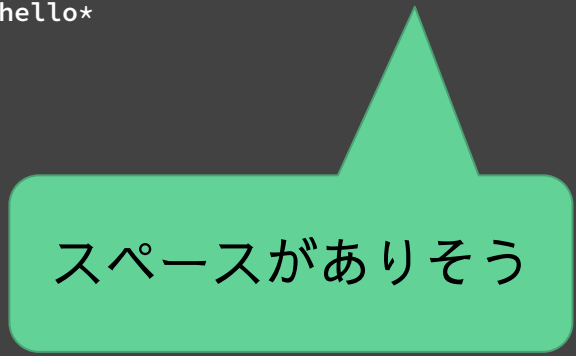
ディスクの消費量を1 byte も増やさずに
既存のプログラムに機能を足す

実践 1: Hello, World! にライフゲームを足す

Hello, World!

```
section .text

_start:

      mov rdx, 0xe                    ; 書き込むデータの長さ

      mov rsi, msg                    ; 書き込むデータの先頭アドレス

      mov rdi, 0x1                    ; 書き込み先のファイル記述子

      mov rax, 0x1                    ; writeシステムコールの番号

      syscall                         ; システムコールの呼び出し

      mov rdi, 0x0                    ; 終了ステータス

      mov rax, 0x3c                   ; exitシステムコールの番号

      syscall                         ; システムコールの呼び出し

section .rodata

      msg: db "Hello, World!", 0xa ; 0xaは改行コード
```

```
$  nasm -f elf64 ./hello.asm

$ ld -o hello ./hello.o

$ ./hello

Hello, World!

$ ls -l ./hello

-rwxrwxr-x 1 akira akira 8880 Sep  8 15:02
./hello*
```



スペースの調査

エントリポイント

```
$ xxd ./hello_syscall | grep 00001000: -A 30
00001000: ba0e 0000 0048 be00 2040 0000 0000 00bf  .....H.. @......
00001010: 0100 0000 b801 0000 000f 05bf 0000 0000  ................
00001020: b83c 0000 000f 0500 0000 0000 0000 0000  .<..............
00001030: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001040: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001050: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001060: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001070: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001080: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001090: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000010a0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000010b0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000010c0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000010d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000010e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000010f0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001100: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001110: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001120: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001130: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001140: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001150: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001160: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001170: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001180: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001190: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000011a0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000011b0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000011c0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000011d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000011e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

スペースを発見

スペースの活用手順 1: バイナリの書き込み

エントリポイント

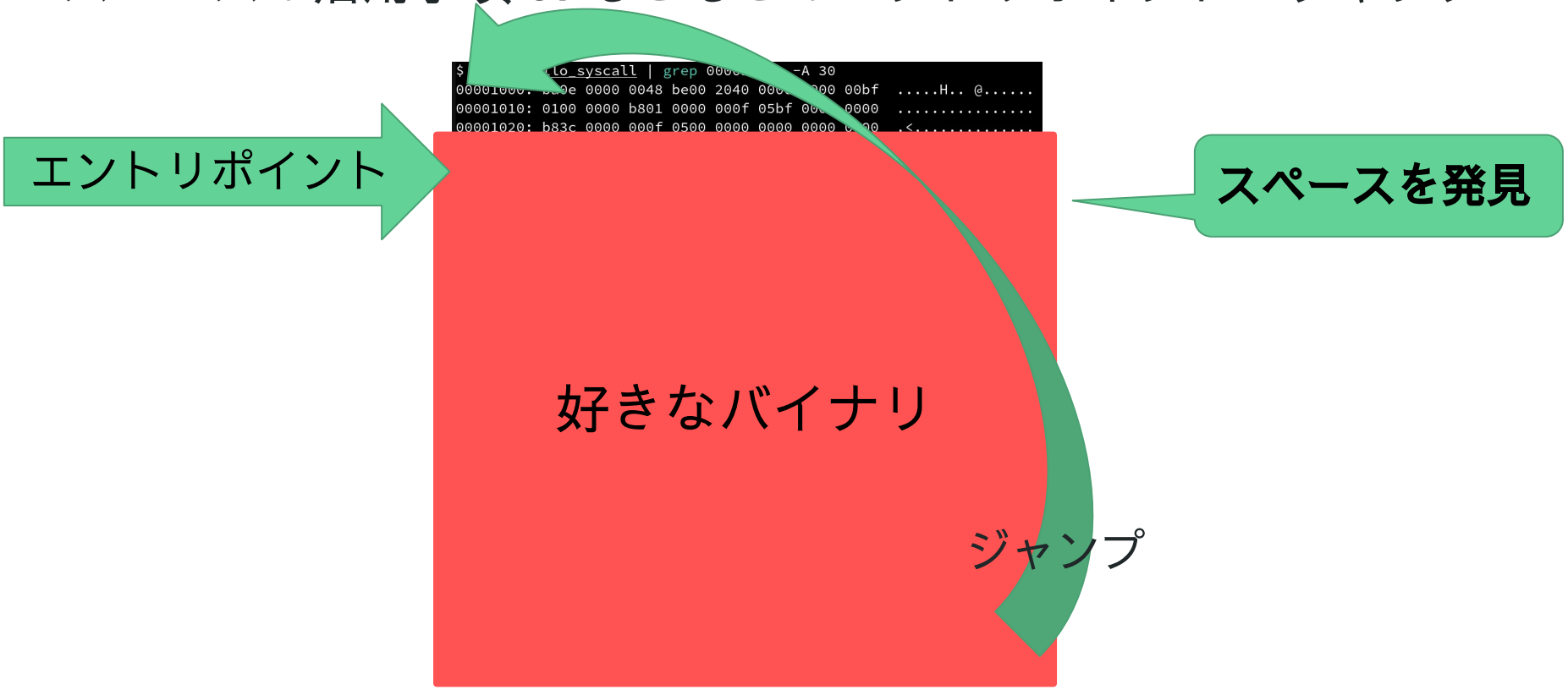
```
$ xxd ./hello_syscall | grep 00001000: -A 30
00001000: ba0e 0000 0048 be00 2040 0000 0000 00bf  .....H.. @......
00001010: 0100 0000 b801 0000 000f 05bf 0000 0000  ................
00001020: b83c 0000 000f 0500 0000 0000 0000 0000  .<..............
```

好きなバイナリ

スペースを発見

スペースの活用手順 2: エントリポイントの書き換え

スペースの活用手順 3: もともとのエントリポイントへジャンプ



埋め込むバイナリの条件

- 一切のライブラリを利用しない

- 最後にオリジナルのエントリポイントへジャンプ
- Position Independent Code
- 一塊で完結している
  - セグメントが一つ

C言語で条件を満たすバイナリを生成する方法

- 一切のライブラリを利用しない ⇒ 標準 C ライブラリを使いたいところはインラインアセンブリ
- 最後にオリジナルのエントリポイントへジャンプ ⇒ インラインアセンブリ
- Position Independent Code ⇒ 文字列リテラルは使わない
- 一塊で完結している
  - セグメントが一つ

埋め込み用ライフゲーム: lifegame.c

```c
#define BOARD_SIZE 32
#define RUNNING_TIME 10
#define ALIVE 'x'
#define DEAD '.'

void lifegame() {
  // Buffer to print without libc
  char print_buf[16];

  // Clear the screen
  print_buf[0] = '\033';
  print_buf[1] = 'c';
  print_buf[2] = '\0';
  __asm__ volatile("syscall" : : "a"(1), "D"(1), "S"(print_buf), "d"(2));

  // Welcome message
  print_buf[0] = 'L';
  print_buf[1] = 'i';
  print_buf[2] = 'f';
  print_buf[3] = 'e';
  print_buf[4] = ' ';
```

#includcde はない

syscall を直接呼び出す

文字列リテラルは使わない

エントリポイントの書き換え

```
$ readelf -h ./hello
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF64
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x401000
  Start of program headers:          64 (bytes into file)
  Start of section headers:          8488 (bytes into file)
  Flags:                             0x0
  Size of this header:               64 (bytes)
  Size of program headers:           56 (bytes)
  Number of program headers:         3
  Size of section headers:           64 (bytes)
  Number of section headers:         6
  Section header string table index: 5
```

ここを書き換える。
バイナリを
パースして書き換え
れば良い

バイナリの埋め込み

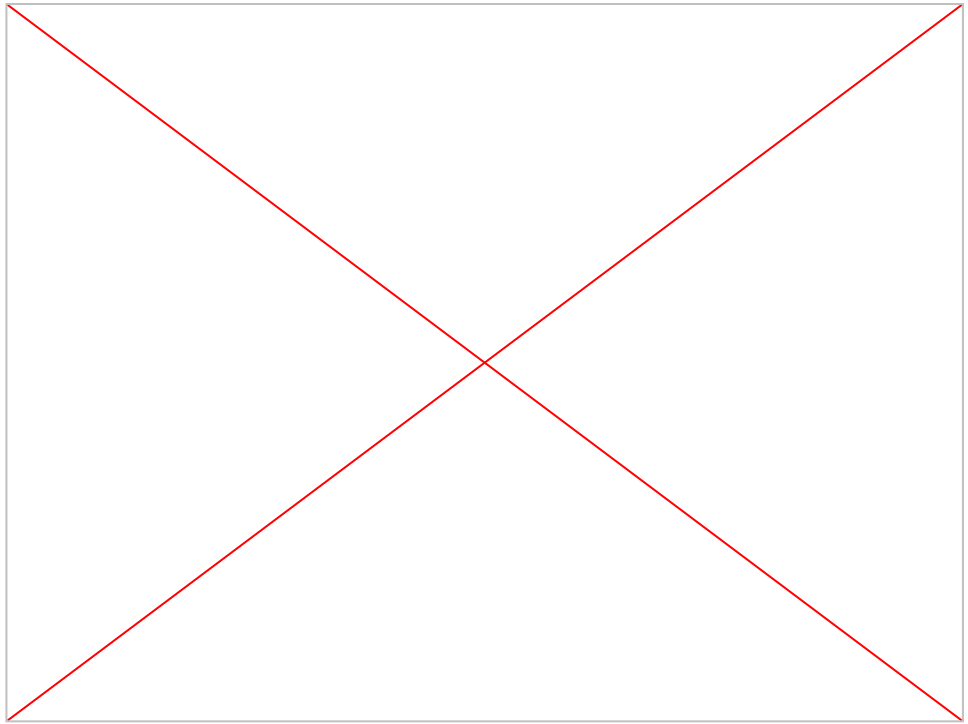
```
$ xxd hello_with_parasite | grep 00001000 -A 30
00001000: ba0e 0000 0048 be00 2040 0000 0000 00bf  .....H.. @......
00001010: 0100 0000 b801 0000 000f 05bf 0000 0000  ................
00001020: b83c 0000 000f 0500 0000 0000 0000 0000  .<..............
00001030: f30f 1efa 5548 89e5 4881 ec60 0900 0090  ....UH..H..`....
00001040: 9090 9090 9090 9090 4889 45f8 31c0 c645  ........H.E.1..E
00001050: e01b c645 e163 c645 e200 b801 0000 00bf  ...E.c.E........
00001060: 0100 0000 488d 4de0 ba02 0000 0048 89ce  ....H.M......H..
00001070: 0f05 c645 e04c c645 e169 c645 e266 c645  ...E.L.E.i.E.f.E
00001080: e365 c645 e420 c645 e547 c645 e661 c645  .e.E. .E.G.E.a.E
00001090: e76d c645 e865 c645 e90a c645 ea00 b801  .m.E.e.E...E....
000010a0: 0000 00bf 0100 0000 488d 4de0 ba09 0000  ........H.M.....
000010b0: 0048 89ce 0f05 c785 a4f6 ffff 0000 0000  .H..............
000010c0: e9e6 0000 00c7 85a8 f6ff ff00 0000 00e9  ................
000010d0: c300 0000 8b85 a8f6 ffff 4863 c88b 85a4  ..........Hc....
000010e0: f6ff ff48 63d0 4889 d048 c1e0 0448 01d0  ...Hc.H..H...H..
000010f0: 4801 c048 01e8 4801 c848 2d30 0900 00c6  H..H..H..H-0....
00001100: 002e 83bd a4f6 ffff 000f 8e81 0000 0083  ................
00001110: bda4 f6ff ff1e 7f78 83bd a8f6 ffff 007e  .......x.......~
00001120: 6f83 bda8 f6ff ff1e 7f66 8b95 a4f6 ffff  o........f......
00001130: 8b85 a8f6 ffff 01c2 4863 c248 69c0 9324  ........Hc.Hi..$
00001140: 4992 48c1 e820 01d0 c1f8 0289 d1c1 f91f  I.H.. ..........
00001150: 29c8 89c1 c1e1 0329 c189 d029 c883 f801  )......)...)....
00001160: 7f2e 8b85 a8f6 ffff 4863 c88b 85a4 f6ff  ........Hc......
00001170: ff48 63d0 4889 d048 c1e0 0448 01d0 4801  .Hc.H..H...H..H.
00001180: c048 01e8 4801 c848 2d30 0900 00c6 0078  .H..H..H-0.....x
00001190: 8385 a8f6 ffff 0183 bda8 f6ff ff21 0f8e  .............!..
000011a0: 30ff ffff 8385 a4f6 ffff 0183 bda4 f6ff  0...............
000011b0: ff21 0f8e 0dff ffff c785 acf6 ffff 0000  .!..............
000011c0: 0000 e9eb 0300 00c7 85b0 f6ff ff00 0000  ................
000011d0: 00eb 0783 85b0 f6ff ff01 81bd b0f6 ffff  ................
000011e0: ff93 3577 7eed c645 e01b c645 e163 c645  ..5w~..E...E.c.E
```

ここにバイナリを埋め込む
ちょっとしたツールを書いて
書き込み

ファイルサイズの比較

```
$ ls -la hello
-rwxr-xr-x 1 akira akira 8872 Sep 13 22:00 hello*
$ ls -la hello_with_parasite
-rwxr-xr-x 1 akira akira 8872 Sep 13 22:00 hello_with_lifegame*
```

デモ



実践 2: Hello, World! にホワイトスペースを入れる

ホワイトスペース

- 難解プログラミング言語のひとつ
- 空白だけでプログラムを構成する
  - タブ、スペース、改行
- 空白以外はコメントになる

```
$ cat helloworld.ws

Say,Hello,World!
```

更にスペースを探す

```
$ readelf -l hello

Elf file type is EXEC (Executable file)
Entry point 0x401000
There are 3 program headers, starting at offset 64

Program Headers:
  Type           Offset             VirtAddr           PhysAddr
                 FileSiz            MemSiz              Flags  Align
  LOAD           0x0000000000000000 0x0000000000400000 0x0000000000400000
                 0x00000000000000e8 0x00000000000000e8  R      0x1000
  LOAD           0x0000000000001000 0x0000000000401000 0x0000000000401000
                 0x0000000000000027 0x0000000000000027  R E    0x1000
  LOAD           0x0000000000002000 0x0000000000402000 0x0000000000402000
                 0x000000000000000e 0x000000000000000e  R      0x1000

 Section to Segment mapping:
  Segment Sections...
   00
   01     .text
   02     .rodata
```

lifegame で使っていたのはこの領域。
実行が可能(RX)。

更にスペースを探す

```
$ readelf -l hello

Elf file type is EXEC (Executable file)
Entry point 0x401000
There are 3 program headers, starting at offset 64

Program Headers:
  Type           Offset             VirtAddr           PhysAddr
                 FileSiz            MemSiz              Flags  Align
  LOAD           0x0000000000000000 0x0000000000400000 0x0000000000400000
                 0x00000000000000e8 0x00000000000000e8  R      0x1000
  LOAD           0x0000000000001000 0x0000000000401000 0x0000000000401000
                 0x0000000000000027 0x0000000000000027  R E    0x1000
  LOAD           0x0000000000002000 0x0000000000402000 0x0000000000402000
                 0x000000000000000e 0x000000000000000e  R      0x1000

 Section to Segment mapping:
  Segment Sections...
   00
   01     .text
   02     .rodata
```

この領域は使っていないが、実行不可で読み取りのみ可。

lifegame で使っていたのはこの領域。実行が可能(RX)。

スペースの有効活用

```
$ readelf -l hello

Elf file type is EXEC (Executable file)
Entry point 0x401000
There are 3 program headers, starting at offset 64

Program Headers:
  Type           Offset             VirtAddr           PhysAddr
                 FileSiz            MemSiz              Flags  Align
  LOAD           0x0000000000000000 0x0000000000400000 0x0000000000400000
                 0x00000000000000e8 0x00000000000000e8  R      0x1000
  LOAD           0x0000000000001000 0x0000000000401000 0x0000000000401000
                 0x0000000000000027 0x0000000000000027  R E    0x1000
  LOAD           0x0000000000002000 0x0000000000402000 0x0000000000402000
                 0x000000000000000e 0x000000000000000e  R      0x1000

 Section to Segment mapping:
  Segment Sections...
   00
   01     .text
   02     .rodata
```

ホワイトスペースの
プログラムを入れる

ホワイトスペースの
処理系を入れる

ホワイトスペースを埋め込んだ Hello, World! の様子

```
$ xxd hello_with_whitespace | grep 000000e0 -A 20
000000e0: 0010 0000 0000 0000 0000 0000 0000 0000  ................
000000f0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000100: 2020 2009 2020 0920 2020 0a09 0a20 2020      .  .   ...
00000110: 2020 0909 2020 0920 090a 090a 2020 2020    ..  . ....
00000120: 2009 0920 0909 2020 0a09 0a20 2020 2020   .. ..  ...
00000130: 0909 2009 0920 200a 090a 2020 2020 2009  .. ..  ...     .
00000140: 0920 0909 0909 0a09 0a20 2020 2020 0920  . .......     .
00000150: 0909 2020 0a09 0a20 2020 2020 0920 2020  ..  ...     .
00000160: 2020 0a09 0a20 2020 2020 0909 0920 0909    ...     ... ..
00000170: 090a 090a 2020 2020 2009 0920 0909 0909  ....     .. ....
00000180: 0a09 0a20 2020 2020 0909 0920 2009 200a  ...     ...  . .
00000190: 090a 2020 2020 2009 0920 0909 2020 0a09  ..     .. ..  ..
000001a0: 0a20 2020 2020 0909 2020 0920 200a 090a  .     ..  .  ...
000001b0: 2020 2020 2009 2020 2020 090a 090a 2020       .    ....
000001c0: 2020 2009 2020 2020 200a 090a 2020 2020     .     ...
000001d0: 2009 0920 2009 0920 0a09 0a20 2020 2020   ..  .. ...
000001e0: 0909 0920 2009 200a 090a 2020 2020 2009  ...  . ...     .
000001f0: 0920 0909 0909 0a09 0a20 2020 2020 0909  . .......     ..
00000200: 2009 0920 090a 090a 2020 2020 2009 2020   .. ....     .
00000210: 2020 200a 090a 2020 2020 2009 0909 2009    ...     ... .
00000220: 0909 0a09 0a20 2020 2020 0909 2009 2020  .....     .. .
```

埋め込まれた
ホワイトスペースの
プログラム

ファイルサイズの比較

```
$ ls -la hello
-rwxr-xr-x 1 akira akira 8872 Sep 13 22:00 hello*
$ ls -la hello_with_parasite
-rwxr-xr-x 1 akira akira 8872 Sep 13 22:00 hello_with_whitespace*
```

デモ

```
$ ./hello_with_whitespace

Hello, world! from whitespace!

Hello, World!
```

参考文献

- Silvio Cesare. UNIX VIRUSES. https://www.win.tue.nl/~aeb/linux/hh/virus/unix-viruses.txt
- Silvio Cesare. UNIX ELF PARASITES AND VIRUS. http://ouah.org/elf-pv.txt
- O'Neill, Ryan Elfmaster. Learning linux binary analysis. Packt Publishing, 2016.