Binary diffing and patching

Proposal for a structured binary diff format and an implementation in GNU poke

Jose E. Marchesi
jemarch@gnu.org

Binary Tools Summit 2022

Contents

cmp

- Option `-l|-verbose`

```
-l, --verbose
    output byte numbers and differing byte values
```

- `cmp -l foo.o bar.o`

```
  41   10 350
  42    2   4
  61   13  23
  63   12  22
  76    0 112
  77  107   0
  78  103   0
  79  103   0
  80   72   4
  ...
```

- Good to be used programmatically.
- Zero readability.
- ALL differences.

xxd + diff

- `diff -u <(xxd foo.o) <(xxd bar.o)`

```
--- /dev/fd/63  2022-03-05 10:58:49.994200759 +0100
+++ /dev/fd/62  2022-03-05 10:58:49.994200759 +0100
@@ -1,77 +1,155 @@
 00000000: 7f45 4c46 0201 0100 0000 0000 0000 0000  .ELF............
 00000010: 0100 3e00 0100 0000 0000 0000 0000 0000  ..>.............
-00000020: 0000 0000 0000 0000 0802 0000 0000 0000  ................
-00000030: 0000 0000 4000 0000 0000 4000 0b00 0a00  ....@.....@.....
-00000040: 5548 89e5 b800 0000 005d c300 4743 433a  UH.......]..GCC:
-00000050: 2028 4465 6269 616e 2036 2e33 2e30 2d31   (Debian 6.3.0-1
...
+00000020: 0000 0000 0000 0000 e804 0000 0000 0000  ................
+00000030: 0000 0000 4000 0000 0000 4000 1300 1200  ....@.....@.....
+00000040: 5548 89e5 b800 0000 005d c34a 0000 0004  UH.......].J....
+00000050: 0000 0000 0008 0100 0000 000c 0000 0000  ................
+00000060: 0000 0000 0000 0000 0b00 0000 0000 0000  ................
+00000070: 0000 0000 0266 6f6f 0001 0146 0000 0000  .....foo...F....
+00000080: 0000 0000 0000 000b 0000 0000 0000 0001  ................
...
```

- Bad to be used programmatically.
- Not that good readability (locality).
- ALL differences.

radiff2

- `radiff2 foo.o bar.o`

```
...
0x000000c2 0000 => 0324 0x000000c2
0x000000c5 000000000000 => 0b0b3e0b0308 0x000000c5
0x000000ce 00 => 2c 0x000000ce
0x000000d2 00 => 02 0x000000d2
0x000000d4 04 => 00 0x000000d4
0x000000d6 f1ff00 => 000008 0x000000d6
0x000000e6 00 => 0b 0x000000e6
```

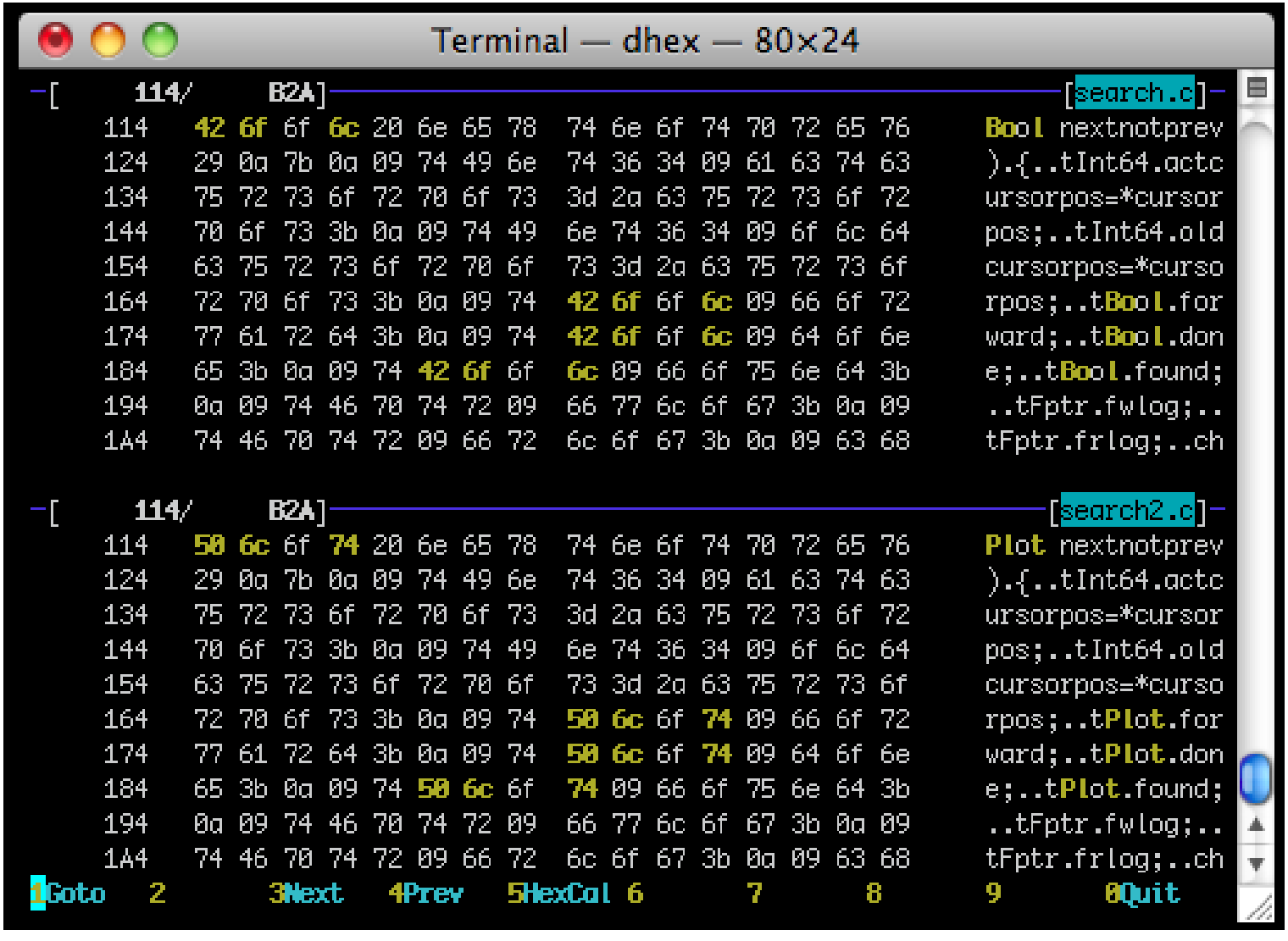
- `radiff2 -r foo.o bar.o`

```
wx 0324 @ 0x000000c2
wx 0b0b3e0b0308 @ 0x000000c5
wx 2c @ 0x000000ce
wx 02 @ 0x000000d2
wx 00 @ 0x000000d4
wx 000008 @ 0x000000d6
wx 0b @ 0x000000e6
```

- Can be restricted to particular functions.
- Oriented to disassembling and radare2.

dhex diff-mode, vbindiff and similar...

raw bytes dump differences side-to-side



A question

How useful is all this?

Structured binary diff format "sbdiff"

```
@@ [AOFF+AS],[BOFF+BS] @@
[+-]BYTES[ NAME][ VALUE]
...
```

- **AOFF** and **AS** are the the A-thunk offset and size in bytes. Ommitted if no - lines in the thunk.
- **BOFF** and **BS** are the the B-thunk offset and size in bytes. Ommitted if no + lines in the thunk.
- **BYTES** are hexadecimal digits denoting bytes.
- **NAME** is an optional symbolic name
- **VALUE** is an optional string denoting semantics of the changed portion.
- Additional white characters are ignored.

Support in poke: sdiff command

- In the master branch of poke.
- Synopsis

```
sdiff :a VAL :b VAL [:values BOOL] [:group_by OFFSET]
```

- Arguments

```
:a (any)
        First mapped value whose bytes are compared.

:b (any)
        Second mapped value whose bytes are compared.

:values (bool)
        Whether to include interpreted values in the output.

:group_by (int)
        How are bytes grouped together in the output.
```

Example: “change” thunks

```
type Elf64_Ehdr =
  struct
  {
    Elf_Ident e_ident;
    ...
    Elf64_Off e_phoff;
    Elf64_Off e_shoff;
    Elf_Word e_flags;
    offset<Elf_Half,B> e_ehsize;
    offset<Elf_Half,B> e_phentsize;
    Elf_Half e_phnum;
    offset<Elf_Half,B> e_shentsize;
    Elf_Half e_shnum;
    Elf_Half e_shstrndx : e_shnum == 0 || e_shstrndx < e_shnum;
  };
```

```
(poke) sdiff :a elf1.ehdr :b elf2.ehdr
@@ 0x28+8,0x28+8 @@
-08 02 00 00 00 00 00 00    a.e_shoff    520UL#B
+e8 04 00 00 00 00 00 00    b.e_shoff    1256UL#B
@@ 0x3c+4,0x3c+4 @@
-0b 00    a.e_shnum       11UH
+13 00    b.e_shnum       19UH
-0a 00    a.e_shstrndx    10UH
+12 00    b.e_shstrndx    18UH
```

Example: “del” thunks

```
type Foo =
  struct
  {
    byte sz;
    byte b;
    byte[sz] data;
  };

(poke) dump :size 32#B :ascii 0
76543210  0011 2233 4455 6677 8899 aabb ccdd eeff
00000000: 0004 0000 0000 0000 0000 0000 0000 0000
00000010: 0000 0000 0000 0000 0000 0000 0000 0000
(poke) sdiff :a (Foo @ 1#B) :b (Foo @ 2#B)
@@ 0x01+1,0x02+1 @@
-04     a.sz
+00     b.sz
@@ 0x03+4, @@
-00     a.data[0]
-00     a.data[1]
-00     a.data[2]
-00     a.data[3]
```

Example: “add” thunks

```
type Foo =
  struct
  {
    byte sz;
    byte b;
    byte[sz] data;
  };

(poke) dump :size 32#B :ascii 0
76543210  0011 2233 4455 6677 8899 aabb ccdd eeff
00000000: 0004 0000 0000 0000 0000 0000 0000 0000
00000010: 0000 0000 0000 0000 0000 0000 0000 0000
(poke) sdiff :a (Foo @ 2#B) :b (Foo @ 1#B)
@@ 0x02+1,0x01+1 @@
-00     a.sz
+04     b.sz
@@ ,0x03+4 @@
+00     b.data[0]
+00     b.data[1]
+00     b.data[2]
+00     b.data[3]
```

Example: combined thunks

```
type Foo =
  struct
  {
    byte sz;
    byte[sz] data;
  };

(poke) dump :size 32#B :ascii 0
76543210  0011 2233 4455 6677 8899 aabb ccdd eeff
00000000: 0004 0000 0000 0000 0000 0000 0000 0000
00000010: 0000 0000 0000 0000 0000 0000 0000 0000
(poke) sdiff :a (Foo @ 2#B) :b (Foo @ 1#B)
@@ 0x02+1,0x01+5 @@
-00     a.sz
+04     b.sz
+00     b.data[0]
+00     b.data[1]
+00     b.data[2]
+00     b.data[3]
```

Example: grouping bytes

White spaces can be interleaved in **BYTES** and are ignored.

```
(poke) sdiff :a elf1.shdr[1] :b elf2.shdr[2] :group_by 2#B
@@ 0x48+4,0x68+4 @@
-1b00 0000     a.sh_name    27U#B
+2100 0000     b.sh_name    33U#B
@@ 0x50+8,0x70+8 @@
-0600 0000 0000 0000     a.sh_flags.flags    6UL
+0300 0000 0000 0000     b.sh_flags.flags    3UL
@@ 0x60+16,0x80+16 @@
-4000 0000 0000 0000     a.sh_offset    64UL#B
+4b00 0000 0000 0000     b.sh_offset    75UL#B
-0b00 0000 0000 0000     a.sh_size      11UL#B
+0000 0000 0000 0000     b.sh_size      0UL#B
```

Example: no values

White spaces can be interleaved in **BYTES** and are ignored.

```
(poke) sdiff :a elf1.shdr[1] :b elf2.shdr[2] :values 0
@@ 0x48+4,0x68+4 @@
-1b 00 00 00     a.sh_name
+21 00 00 00     b.sh_name
@@ 0x50+8,0x70+8 @@
-06 00 00 00 00 00 00 00     a.sh_flags.flags
+03 00 00 00 00 00 00 00     b.sh_flags.flags
@@ 0x60+16,0x80+16 @@
-40 00 00 00 00 00 00 00     a.sh_offset
+4b 00 00 00 00 00 00 00     b.sh_offset
-0b 00 00 00 00 00 00 00     a.sh_size
+00 00 00 00 00 00 00 00     b.sh_size
```

Example: absent Fields

```
type Foo =
  struct
  {
    byte foo;
    int bar if foo != 0;
  };

(poke) dump :size 32#B :ascii 0
76543210  0011 2233 4455 6677 8899 aabb ccdd eeff
00000000: 0004 0000 0000 0000 0000 0000 0000 0000
00000010: 0000 0000 0000 0000 0x000 0000 0000 0000
(poke) sdiff :a (Foo @ 0#B) :b (Foo @ 1#B)
@@ 0x00+1,0x01+5 @@
-00               a.foo
+04               b.foo
+00 00 00 00      b.bar    0
(poke) sdiff :a (Foo @ 1#B) :b (Foo @ 0#B)
@@ 0x01+5,0x00+1 @@
-04               a.foo
+00               b.foo
-00 00 00 00      a.bar    0
```

Example: unions

```
var x = 0;
type Foo =
  union
  {
    byte foo;
    int bar : x != 0;
  }

(poke) dump :size 32#B :ascii 0
76543210  0011 2233 4455 6677 8899 aabb ccdd eeff
00000000: 0004 0000 0000 0000 0000 0000 0000 0000
00000010: 0000 0000 0000 0000 0x000 0000 0000 0000
(poke) sdiff :a (Foo @ 0#B) :b (Foo @ 1#B)
@@ 0x00+1,0x01+1 @@
-00     a.foo
+04     b.foo
(poke) x = 1
(poke) sdiff :a (Foo @ 0#B) :b (Foo @ 1#B)
@@ 0x00+4,0x01+4 @@
-00 04 00 00     a.bar    1024
+04 00 00 00     b.bar    4
```

Example: a string table

```
(poke) sdiff :a strtab1 :b strtab2
@@ 0xdc+40,0x6c+119 @@
-2e 63 6f 6d 6d 65 6e 74 00                                     a[7]    ".comment"
+2e 72 65 6c 61 2e 64 65 62 75 67 5f 69 6e 66 6f 00             b[7]    ".rela.debug_inf
-2e 6e 6f 74 65 2e 47 4e 55 2d 73 74 61 63 6b 00                a[8]    ".note.GNU-stack
+2e 64 65 62 75 67 5f 61 62 62 72 65 76 00                     b[8]    ".debug_abbrev"
-2e 72 65 6c 61 2e 65 68 5f 66 72 61 6d 65 00                   a[9]    ".rela.eh_frame"
+2e 72 65 6c 61 2e 64 65 62 75 67 5f 61 72 61 6e 67 65 73 00    b[9]    ".rela.debug_ara
+2e 72 65 6c 61 2e 64 65 62 75 67 5f 6c 69 6e 65 00             b[10]   ".rela.debug_lin
+2e 64 65 62 75 67 5f 73 74 72 00                               b[11]   ".debug_str"
+2e 63 6f 6d 6d 65 6e 74 00                                     b[12]   ".comment"
+2e 6e 6f 74 65 2e 47 4e 55 2d 73 74 61 63 6b 00                b[13]   ".note.GNU-stack
+2e 72 65 6c 61 2e 65 68 5f 66 72 61 6d 65 00                   b[14]   ".rela.eh_frame"
```

Open questions

- The proposed format is **byte-oriented**. Should we change this?
  - Alternative 1: support bytes and nibbles
  - Alternative 2: support bytes and bits
  - Alternative 3: support arbitrary units
  - Other approaches?
- Context support? (based on structure and/or bytes?)
- Where to discuss stuff like this? This must **not** be poke specific. Mailing list in `binary-tools.net`?

sdiff implementation

- Part of the **poke** application (`pk-diff.pk`).
- Implemented in around 300 lines of Poke.
- Will split into application-independent pickle `diff.pk`.