gdb-tools

by Sergei Golubchik

Where to get

- PyPI

```
pip install gdb-tools
echo 'py import duel' >> ~/.gdbinit
```

- GitHub

```
git clone https://github.com/vuvova/gdb-tools
```

@PrettyPrinter

“Easy” structures

```
(gdb) p read_set
$15 = (MY_BITMAP *) 0x7fffee0601a8
(gdb) p read_set[0]
$16 = {bitmap = 0x7fffee03a1c0, last_word_mask = 4293918720, n_bits = 52}
(gdb) p/t read_set->bitmap[0] @ 2
$17 = {11111111111111111111111111111111, 10101010101011111111111111111111}

(gdb) p alias
$18 = (String *) 0x7fffee094958
(gdb) p *alias
$19 = {Ptr = 0x7fffee050370 "proc\003\005\356\377\177", str_length = 4,
  Alloced_length = 8, extra_alloc = 0, alloced = true,
  str_charset = 0x1662ce0 <my_charset_bin>}
(gdb)
```

Complex structures

```
(gdb) p thd->variables.sql_mode
$7 = 1574961152
... few minutes later ...
(gdb) p MODE_STRICT_TRANS_TABLES + MODE_STRICT_ALL_TABLES + MODE_NO_ZERO_I
MODE_NO_ZERO_DATE + MODE_ERROR_FOR_DIVISION_BY_ZERO + MODE_TRADITIONAL +
MODE_NO_AUTO_CREATE_USER + MODE_NO_ENGINE_SUBSTITUTION
$15 = 1574961152

(gdb) p ip_storage
$23 = (sockaddr_storage *) 0x7ffff503b308
(gdb) p ip_storage[0]
$24 = {ss_family = 2, __ss_padding = "\352\064\177\000\000\001",
    '\000' <repeats 111 times>, __ss_align = 0}
(gdb) p/u ip_storage->__ss_padding[2] @ 4
$25 = {127, 0, 0, 1}
```

```
import gdb.printing

class ppString:
    def __init__(self, val):
        self.val = val
    def to_string(self):
        # (gdb) p (char[val->str_length])(*val->Ptr)
        return str(self.val['Ptr'].dereference().
                   cast(gdb.lookup_type('char').array(self.val['str_length'])))

class ppMY_BITMAP:
    def __init__(self, val):
        self.val = val
    def to_string(self):
        s=''.join(reversed([format(int(val['bitmap'][i]),'032b')
                            for i in range(int(val['n_bits']+31)//32)]))
        return "b'" + s[-int(val['n_bits']):] + "'"

pp = gdb.printing.RegexpCollectionPrettyPrinter(".gdb.py")
pp.add_printer('String', '^String$', ppString)
pp.add_printer('MY_BITMAP', '^MY_BITMAP$', ppMY_BITMAP)
gdb.printing.register_pretty_printer(None, pp, True)
```

TOO MUCH BOILERPLATE!

and I've just started

DRY = Don't Repeat Yourself

```
from pretty_printer import PrettyPrinter

@PrettyPrinter
def String(val):
    # (gdb) p (char[val->str_length])(*val->Ptr)
    return str(val['Ptr'].dereference().
               cast(gdb.lookup_type('char').array(val['str_length'])))

@PrettyPrinter
def MY_BITMAP(val):
    s=''.join(reversed([format(int(val['bitmap'][i]),'032b')
                        for i in range(int(val['n_bits']+31)//32)]))
    return "b'" + s[-int(val['n_bits']):] + "'"
```

Really easy

```
(gdb) p read_set[0]
$36 = b'11111111111111111111111111111111111111111111111111'

(gdb) p *alias
$37 = "proc"

(gdb) p thd->variables.sql_mode
$38 = STRICT_TRANS_TABLES,STRICT_ALL_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,
    ERROR_FOR_DIVISION_BY_ZERO,TRADITIONAL,NO_AUTO_CREATE_USER,
    NO_ENGINE_SUBSTITUTION

(gdb) p ip_storage[0]
$39 = AF_INET://127.0.0.1:59956
```

DUEL

Debugging U (might) Even Like

DUEL: the history

- Created by Michael Golan in 1993
  - Public domain, patch for gdb-4.6
- On IRIX, in 2002
- In Gentoo, in 2008
- Reimplemented as DUEL.py in 2017

Example: Linked List

```
(gdb) dl table_list-->next_global->table_name
table_list->table_name = 0x7fffde8f8188 "t1m"
table_list->next_global->table_name = 0x7fffde8f9500 "t2i"
table_list-->next_global[[2]]->table_name = 0x7fffde8f9b40 "t3i"
table_list-->next_global[[3]]->table_name = 0x7fffde8fa180 "t4m"
table_list-->next_global[[4]]->table_name = 0x7fffde8fa7c0 "t5m"
table_list-->next_global[[5]]->table_name = 0x7fffde8fae00 "t6m"
table_list-->next_global[[6]]->table_name = 0x7fffde8fb240 "t7m"
table_list-->next_global[[7]]->table_name = 0x7fffde8fb980 "t8i"
```

Example: Linked List

```
(gdb) dl #/thd->free_list-->next
#/thd->free_list-->next = 29

(gdb) dl thd->free_list-->next ==? item
thd->free_list-->next[[5]] ==? item = 0x7fffecc4a560
```

Example: Array

```
(gdb) dl ht->table_options[0..2].name
ht->table_options[0].name = 0x7fffef0d02d5 "PAGE_COMPRESSED"
ht->table_options[1].name = 0x7fffef0d02e5 "PAGE_COMPRESSION_LEVEL"
ht->table_options[2].name = 0x7fffef0d02fc "ATOMIC_WRITES"

(gdb) dl ht->table_options[0..].name @ 0
ht->table_options[0].name = 0x7fffef0d02d5 "PAGE_COMPRESSED"
ht->table_options[1].name = 0x7fffef0d02e5 "PAGE_COMPRESSION_LEVEL"
ht->table_options[2].name = 0x7fffef0d02fc "ATOMIC_WRITES"
ht->table_options[3].name = 0x7fffef0d0319 "ENCRYPTED"
ht->table_options[4].name = 0x7fffef0d0332 "ENCRYPTION_KEY_ID"
```

Expression can have many values

Many values

```
(gdb) dl (1,2)+(100,200)
1 + 100 = 101
1 + 200 = 201
2 + 100 = 102
2 + 200 = 202

(gdb) dl ("ABC", "def")[1,2]
"ABC"[1] = 98 'B'
"ABC"[2] = 99 'C'
"def"[1] = 101 'e'
"def"[2] = 102 'f'
```

From one to many

```
(gdb) p ht->table_options[0].name
$10 = 0x7fffef0d02d5 "PAGE_COMPRESSED"

(gdb) dl ht->table_options[0..4].name
ht->table_options[0].name = 0x7fffef0d02d5 "PAGE_COMPRESSED"
ht->table_options[1].name = 0x7fffef0d02e5 "PAGE_COMPRESSION_LEVEL"
ht->table_options[2].name = 0x7fffef0d02fc "ATOMIC_WRITES"
ht->table_options[3].name = 0x7fffef0d0319 "ENCRYPTED"
ht->table_options[4].name = 0x7fffef0d0332 "ENCRYPTION_KEY_ID"
```

From one to many

```
(gdb) p ((MI_INFO*)(myisam_open_list->data))->s->file_name + 52
$2 = 0x7fffec830c58 "mysql/gtid_slave_pos.MYI"

(gdb) dl ((MI_INFO*)(myisam_open_list-->next->data))->s->file_name + 52
<...> = 0x7fffec830c8c "mysql/gtid_slave_pos.MYI"
<...> = 0x7ffff50fce1c "mysql/event.MYI"
<...> = 0x7ffff0fc0a7c "mysql/procs_priv.MYI"
<...> = 0x7ffff0fbf194 "mysql/columns_priv.MYI"
<...> = 0x7ffff0fbe654 "mysql/tables_priv.MYI"
<...> = 0x7fffeeba98b4 "mysql/roles_mapping.MYI"
<...> = 0x7ffff0fbce24 "mysql/proxies_priv.MYI"
<...> = 0x7ffff50f6b8c "mysql/host.MYI"
<...> = 0x7ffff50f50cc "mysql/db.MYI"
<...> = 0x7ffff13ff09c "mysql/user.MYI"
<...> = 0x7ffff0fb9154 "mysql/servers.MYI"
```

Checkpoint

- Any expression can return many values
- Conventional C operators work on every value
- To generate many values use
  - Enumeration: 1,2,3,a,b
  - Range: 1..5
  - Traversal: a-->b or, for example, root-->(left,right)

Until

```
(gdb) dl ht->table_options[0..].name @ 0
ht->table_options[0].name = 0x7fffef0d02d5 "PAGE_COMPRESSED"
ht->table_options[1].name = 0x7fffef0d02e5 "PAGE_COMPRESSION_LEVEL"
ht->table_options[2].name = 0x7fffef0d02fc "ATOMIC_WRITES"
ht->table_options[3].name = 0x7fffef0d0319 "ENCRYPTED"
ht->table_options[4].name = 0x7fffef0d0332 "ENCRYPTION_KEY_ID"

(gdb) dl (table->field[0..]@0)->field_name
(table->field[0])->field_name = 0x7fffdd8a1609 "id"
(table->field[1])->field_name = 0x7fffdd8a160c "code"
(table->field[2])->field_name = 0x7fffdd8a1611 "name"
```

Until (2)

```
(gdb) dl ht->table_options[0..].name @ 0
ht->table_options[0].name = 0x7fffef0d02d5 "PAGE_COMPRESSED"
ht->table_options[1].name = 0x7fffef0d02e5 "PAGE_COMPRESSION_LEVEL"
ht->table_options[2].name = 0x7fffef0d02fc "ATOMIC_WRITES"
ht->table_options[3].name = 0x7fffef0d0319 "ENCRYPTED"
ht->table_options[4].name = 0x7fffef0d0332 "ENCRYPTION_KEY_ID"

(gdb) dl ht->table_options[0..]@(name==0)
ht->table_options[0] = {type = BOOL, name = "PAGE_COMPRESSED", name_length
ht->table_options[1] = {type = ULL, name = "PAGE_COMPRESSION_LEVEL", name_
ht->table_options[2] = {type = ENUM, name = "ATOMIC_WRITES", name_length =
ht->table_options[3] = {type = ENUM, name = "ENCRYPTED", name_length = 9, 
ht->table_options[4] = {type = ULL, name = "ENCRYPTION_KEY_ID", name_lengt
```

Filters

```
(gdb) dl (table->field[0..]@0)->vcol_info !=? 0
(table->field[2])->vcol_info !=? 0 = 0x7fffa00651f8
(table->field[6])->vcol_info !=? 0 = 0x7fffa0065800
(table->field[7])->vcol_info !=? 0 = 0x7fffa0065fa0

(gdb) dl ((table->field[0..]@0)->vcol_info !=? 0)->field_name
<...> = 0x7fffa0104980 "c"
<...> = 0x7fffa0104985 "g"
<...> = 0x7fffa010498b "h"
```

Value by index

```
(gdb) dl tl->next_global->table_name
tl->table_name = "t1m"
tl->next_global->table_name = "t2i"
tl-->next_global[[2]]->table_name = "t3i"
tl-->next_global[[3]]->table_name = "t4m"
tl-->next_global[[4]]->table_name = "t5m"
tl-->next_global[[5]]->table_name = "t6m"

(gdb) dl tl-->next_global[[3]]->table_name
tl-->next_global[[3]]->table_name = "t4m"

(gdb) dl (1,2,3,4,5)[[3]]
(1,2,3,4,5)[[3]] = 4

(gdb) dl tl-->next_global[[#/tl-->next_global - 1]]->table_name
tl-->next_global[[#/tl-->next_global - 1]]->table_name = "t6m"
```

Curly Braces

```
(gdb) dl tl-->next_global[[#/tl-->next_global - 1]]->table_name
tl-->next_global[[#/tl-->next_global - 1]]->table_name = "t6m"

(gdb) dl tl-->next_global[[ {#/tl-->next_global - 1} ]]->table_name
tl-->next_global[[5]]->table_name = "t6m"
```

Aggregation

```
(gdb) dl #/thd->free_list-->next
#/thd->free_list-->next = 29

(gdb) dl +/(1..100)
+/(1..100) = 5050

(gdb) dl +/global_status_var.com_stat[0..150]
+/global_status_var.com_stat[0..150] = 17
```

Thank you!

https://github.com/vuvova/gdb-tools