Reversing Golang

George Zaytsev

Golang :about

- Created at Google in 2007 by by Robert Griesemer, Rob Pike, and Ken Thompson
- Announced in 2009
- Current stable version 1.7.3
- Go 1.0 was released at 2012
- A lot of runtime
- Mostly statically compiled

Golang :malware

- June 2016: Linux.Lady
- August 2016: Linux.Rex
- September 2016: Trojan.Encoder.6491
- ARCANUS
- Veil-evasion
- Ebowla
- Adware(Trojan).Mutabaha/Trojan.Egguard

Golang: existing work

- R2Con 2016: «Reversing Linux Malware» by Sergi Martinez
  - Linux.Lady
  - Presented script for radare2 for restoring type and function names
  - go 1.6
- «Reversing GO binaries like a pro» by Tim Strazzere
  - IDA Pro script for restoring functions and their names
  - Great go1.7 string recognition

Restoring function names

- Already described in mentioned sources
- Based on gopclntab(appeared in go1.2)
- Following format:
  - 8 byte header
  - Amount of functions
  - Array of following entry structure:
    - Function address
    - Offset from gopclntab to funcN struct (this is where we get original name)

Restoring types

- What we already know after r2con:
  - runtime_newobject creates new instance of type
  - runtime_newobject takes «type» structure pointer as argument
  - From «type» structure we can get type name

- And this is great! But…

Reflect module

- If we read some source code of Go, we can find much more interesting things (src/reflect/type.go):
  - «type» structure have an «kind» field
- Enum kind:
  - «basic» types:
    - BOOL, INT*, UINT*,FLOAT*, COMPLEX*
  - «other» types:
    - CHAN, STRING, SLICE, INTERFACE, STRUCT, MAP, FUNC

Clarify types

- According to «kind» field «type» pointer can be treated as pointer to concrete «kind type» structure:
  - StructType, InterfaceType, FuncType, …
- This structures contains very useful info:
  - Structure member names and types
  - Interface methods
  - Argument types

Problems

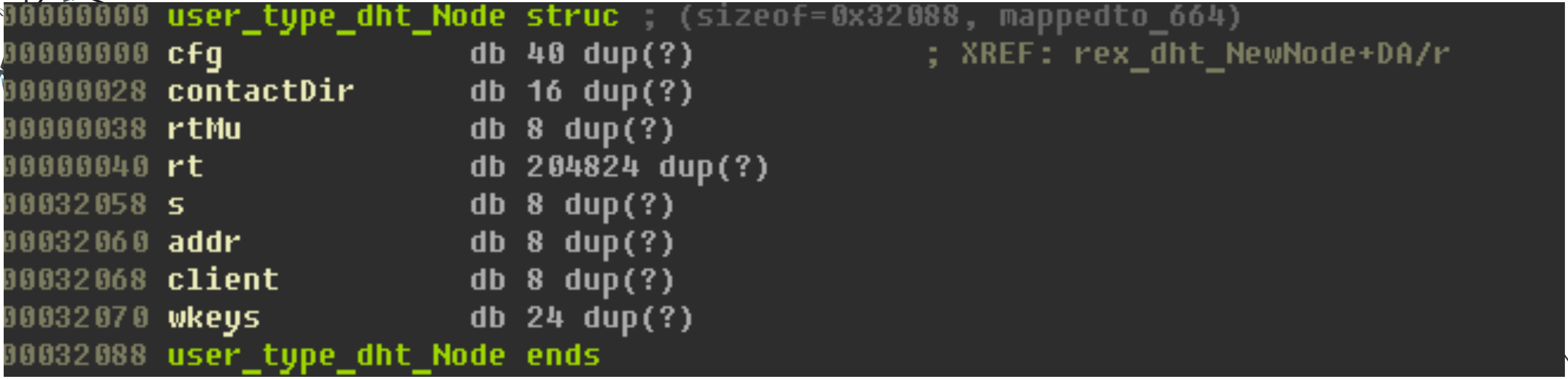
- «type» structure slightly changes in every major go release
- Example: in go1.7 instead pointer to type name we've got an offset from location where types begin:
  - .typelink section in MOST(Hi, ZN2016 HackQuest) ELF files
  - In OS X binaries it use __typelink section
  - In PE files there is no such sections - all located in .text
- We need to know which go version was used to compile binary

Goutils

- All scripts was designed mainly for ELF - and relied on existence go-specific sections
- They can be used for PE files as well, but we need to specify address where typeinfo begins and where it ends.

```
def typesGo17_win64(begin_typeinfo, end_typeinfo, rodata_addr):
    global GLOBAL_WIN_GODATA_ADDR
    GLOBAL_WIN_GODATA_ADDR = rodata_addr
    h = Go17TypesWin(begin_typeinfo, end_typeinfo, step=SizeQword)
    for i in h:
        pass
    return h
```

- So it was time to return for go sources

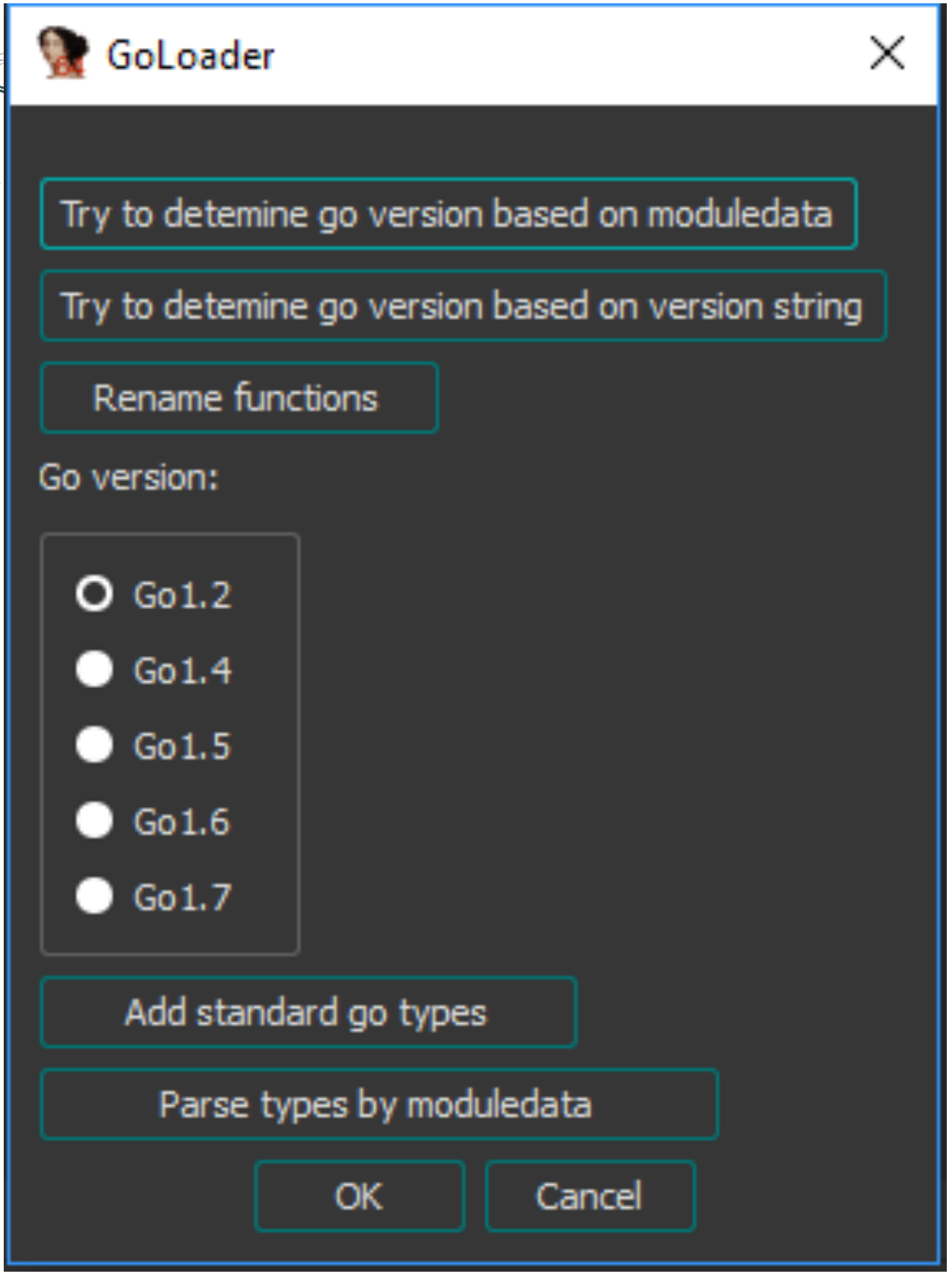


```
00000000 user_type_dht_Node struc ; (sizeof=0x32088, mappedto_664)
00000000 cfg             db 40 dup(?)                ; XREF: rex_dht_NewNode+DA/r
00000028 contactDir      db 16 dup(?)
00000038 rtMu            db 8 dup(?)
00000040 rt              db 204824 dup(?)
00032058 s               db 8 dup(?)
00032060 addr            db 8 dup(?)
00032068 client          db 8 dup(?)
00032070 wkeys           db 24 dup(?)
00032088 user_type_dht_Node ends
```

Moduledata structure

- Appeared in go1.5
- Contains pointers to gopcltab, typeinfo, and other useful fields
- This allows us use generic approach for all binaries compiled with go >= 1.5
- Unfortunately it has the same format in go1.5 and go1.6
- We still need to somehow find go version
  - Now, I just look for go1.X string :)
  - This string is used in runtime_schedinit function, so expected to be in every binary

GoUtils2.0



GoLoader

Try to detemine go version based on moduledata

Try to detemine go version based on version string

Rename functions

Go version:

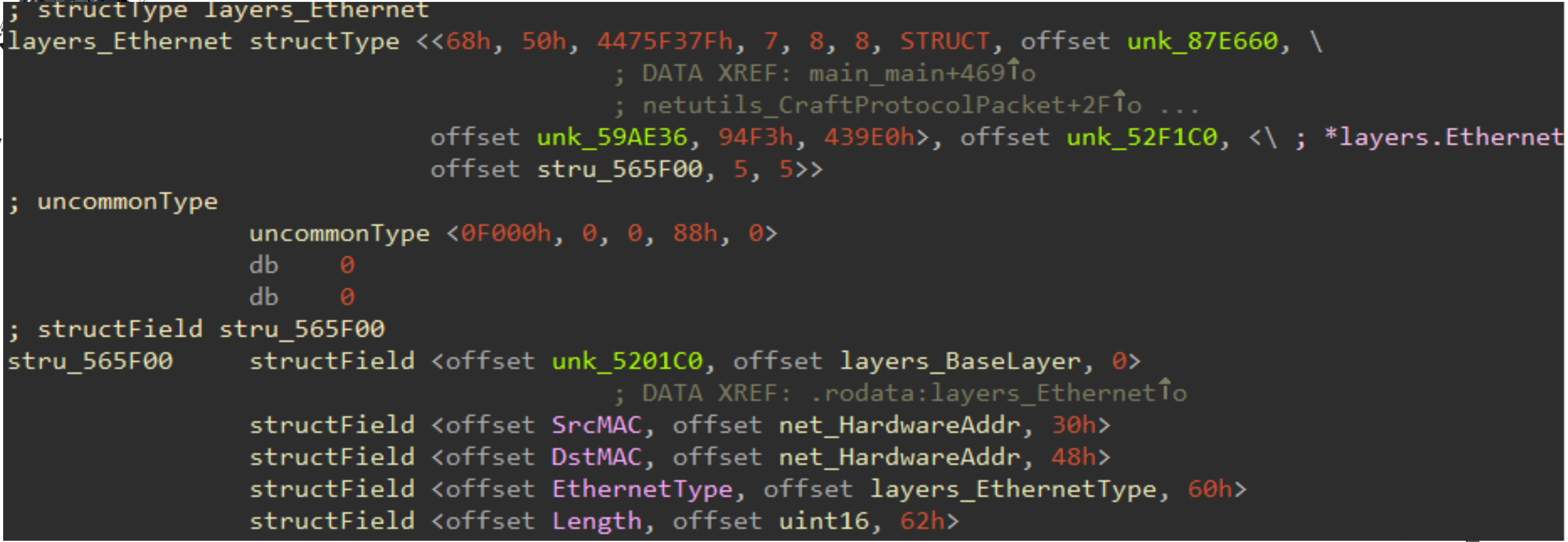
[✓ Go1.2]
Go1.4
Go1.5
Go1.6
Go1.7

Add standard go types

Parse types by moduledata

OK

Cancel



```
; structType layers_Ethernet
layers_Ethernet structType <<68h, 50h, 4475F37Fh, 7, 8, 8, STRUCT, offset unk_87E660, \
                                        ; DATA XREF: main_main+469↑o
                                        ; netutils_CraftProtocolPacket+2F↑o ...
                            offset unk_59AE36, 94F3h, 439E0h>, offset unk_52F1C0, <\ ; *layers.Ethernet
                            offset stru_565F00, 5, 5>>
; uncommonType
                uncommonType <0F000h, 0, 0, 88h, 0>
                db    0
                db    0
; structField stru_565F00
stru_565F00     structField <offset unk_5201C0, offset layers_BaseLayer, 0>
                                        ; DATA XREF: .rodata:layers_Ethernet↑o
                structField <offset SrcMAC, offset net_HardwareAddr, 30h>
                structField <offset DstMAC, offset net_HardwareAddr, 48h>
                structField <offset EthernetType, offset layers_EthernetType, 60h>
                structField <offset Length, offset uint16, 62h>
```

GoUtils2.0

- Work in progress
- Rewritten version of goutils - not all is ported yet
    - user-defined structures is not recreating for now
    - but it creates itab for go1.7 :)
- No need to manually call functions from console
- Implemented two methods of go version recognition
- Works for go >= 1.5
- Correctly parses ZN2016 HackQuest binary

GoUtils 2.0

- Wouldn't work for PE if go < 1.5 is used
  - we still need to find typeinfo location manually
  - but we can recreate standard go *Type structures and manually call handle_offset function
  - we can use IDAWalker to collect calls of runtime_newobject and then process all collected «type» pointers
- Actually maybe this is even good - Ctrl+T after auto recreating all structs sometime make me sad :)

References

- https://golang.org/
- http://rednaga.io/2016/09/21/reversing_go_binaries_like_a_pro/
- https://github.com/radareorg/r2con/blob/master/2016/talks/11-ReversingLinuxMalware/r2con_SergiMartinez_ReversingLinuxMalware.pdf
- http://vms.drweb.ru/
- http://www.slideshare.net/DefconRussia/reversing-golang-66820671
- https://gitlab.com/zaytsevgu/GoUtils2.0/
- https://gitlab.com/zaytsevgu/goutils/
- https://gitlab.com/zaytsevgu/ida-walk

That’s all

Thank you!

Feel free ask me about more details:)

zaytsevgu@gmail.com

Twitter: groke1105