Building Go Programs for MIPS

5 min read

February 10, 2020

I’ve got a Lenovo router last year, and I’ve been always using it as a room AP to make the WiFi better. Recently, it’s getting harder to bypass the Great Firewall so I flashed the router to OpenWrt and try to make it as a VPN client.

Device Info

After everything is done, the router now is a mini Linux server and I can SSH onto it. Here are some infomation of the device.

```
openwrt:# uname -a
Linux OpenWrt 4.14.151 #0 Tue Nov 5 14:12:18 2019 mips GNU/Linux

openwrt:# cat /proc/cpuinfo
system type             : MediaTek MT7620A ver:2 eco:6
machine                 : Lenovo Y1
processor               : 0
cpu model               : MIPS 24KEc V5.0
BogoMIPS                : 385.84
wait instruction        : yes
microsecond timers      : yes
tlb_entries             : 32
extra interrupt vector  : yes
hardware watchpoint     : yes, count: 4, address/irw mask: [0x0ffc, 0x0ffc, 0x0ffb
isa                     : mips1 mips2 mips32r1 mips32r2
ASEs implemented        : mips16 dsp
shadow register sets    : 1
kscratch registers      : 0
package                 : 0
core                    : 0
VCED exceptions         : not available
VCEI exceptions         : not available

openwrt:# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 2.5M      2.5M         0 100% /rom
tmpfs                    61.2M      1.3M     60.0M   2% /tmp
/dev/mtdblock6           12.0M      1.5M     10.6M  12% /overlay
overlayfs:/overlay       12.0M      1.5M     10.6M  12% /
```

```
tmpfs                   512.0K         0    512.0K   0% /dev
```

It has a MIPS architected CPU *MIPS 24KEc V5.0* and that's new for me. Next step I want to install ***v2ray*** on it, a proxy client. But I found the official released package is too big for my device. It takes about 20MB of a zipped binary. The *opkg* package manager doesn't provide v2ray either.

Build A Smaller v2ray

So, it's time to do myself. I cloned the source code of v2ray on my macOS, it's written in Go and must be easy to cross-compile.

```
macbook:$ mkdir -p $GOPATH/src/v2ray.com/core
macbook:$ git clone https://github.com/v2ray/v2ray-core $GOPATH/src/v2ray.com/core
macbook:$ cd $GOPATH/src/v2ray.com/core/main
```

(For go1.12+ with go mod, you can clone it to anywhere.)

According to the GoMips wiki, I ran the following command to compile.

```
macbook:$ GOOS=linux GOARCH=mips GOMIPS=softfloat go build -o v2ray
```

After a few minutes of retrieving packages, everything's OK, I got the MIPS binary.

```
macbook:$ ls -alh v2ray
-rwxr-xr-x  1 ferdi  staff    21M Feb 10 21:34 v2ray
macbook:$ file v2ray
v2ray: ELF 32-bit MSB executable, MIPS, MIPS32 version 1 (SYSV), statically linked
```

It's 21M and I guess this is the official way of building. To make the output smaller, I retry the build command with some parameters. (Although the router supports external USB stick, I don't have spare one and it's unnecessary.)

```
macbook:$ GOOS=linux GOARCH=mips GOMIPS=softfloat go build -trimpath -ldflags="-s
macbook:$ ls -alh v2ray
-rwxr-xr-x  1 ferdi  staff    15M Feb 10 21:36 v2ray
```

It's smaller now as 15M, but still can't fit my device since the router has only 10.6M left. Then I found a tool named *upx*, which can compress a ELF

file.

```
macbook:$ upx -9 v2ray
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96        Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
  15728640 ->   5461384   34.72%   linux/mips    v2ray

Packed 1 file.

macbook:$ ls -alh v2ray
-rwxr-xr-x   1 ferdi  staff   5.2M Feb 10 21:36 v2ray
```

It’s acceptable now so I copied it to my router.

```
macbook:$ scp v2ray root@192.168.99.1:/tmp
```

On the router, it’s there! The sad news is, I can’t execute it.

```
openwrt:$ cd /tmp && ls -alh v2ray
-rwxr-xr-x    1 root     root        5.2M Feb 10 13:44 v2ray
openwrt:$ ./v2ray
openwrt:$ ./v2ray: line 2: syntax error: unterminated quoted string
```

Anything wrong? Do the *upx* make it corrupted? Or any of the build parameter wrong? I checked the build steps and try to figure that out what’s going on.

Try With A Helloworld Program

Then I tried a hello-world program without any build parameter and no upx, it also failed.

```
package main

import (
    "fmt"
)

func main() {
    fmt.Println("hello, mips")
```

```
}
```

```
macbook:$ GOOS=linux GOARCH=mips GOMIPS=softfloat go build -o hello
macbook:$ scp hello root@192.168.99.1:/tmp

# on the router
openwrt:/tmp# ./hello
./hello: line 1: syntax error: unexpected "("
```

I started googling the problem, and here is my go info.

```
macbook:$ go version
go version go1.13.4 darwin/amd64

macbook:$ go env
GO111MODULE=""
GOARCH="amd64"
GOBIN=""
GOCACHE="/Users/ferdi/Library/Caches/go-build"
GOENV="/Users/ferdi/Library/Application Support/go/env"
GOEXE=""
GOFLAGS=""
GOHOSTARCH="amd64"
GOHOSTOS="darwin"
GONOPROXY=""
GONOSUMDB=""
GOOS="darwin"
GOPATH="/Users/ferdi/GOPATH"
GOPRIVATE=""
GOPROXY="https://proxy.golang.org,direct"
GOROOT="/usr/local/go"
GOSUMDB="sum.golang.org"
GOTMPDIR=""
GOTOOLDIR="/usr/local/go/pkg/tool/darwin_amd64"
GCCGO="gccgo"
AR="ar"
CC="clang"
CXX="clang++"
CGO_ENABLED="1"
GOMOD="/Users/ferdi/GOPATH/src/v2ray.com/core/go.mod"
CGO_CFLAGS="-g -O2"
CGO_CPPFLAGS=""
CGO_CXXFLAGS="-g -O2"
CGO_FFLAGS="-g -O2"
CGO_LDFLAGS="-g -O2"
PKG_CONFIG="pkg-config"
GOGCCFLAGS="-fPIC -m64 -pthread -fno-caret-diagnostics -Qunused-arguments -fmessag
```

The Solution

I even looked through the offical MIPS32® 24KE™ Core spec and it's strange there's no indication of Big-Endian or Little-Endian.

At last I solved the problem by setting **GOARCH=mipsle**, it works now.

```
macbook:$ GOOS=linux GOARCH=mipsle GOMIPS=softfloat go build -o hello
macbook:$ scp hello root@192.168.99.1:/tmp

openwrt:/tmp# ./hello
hello, mips
```

The v2ray binary also works now,

```
macbook:$ GOOS=linux GOARCH=mipsle GOMIPS=softfloat go build -trimpath -ldflags="-
macbook:$ scp v2ray root@192.168.99.1:/tmp

openwrt:/tmp# ./v2ray --help
Usage of ./v2ray:
  -config string
        Config file for V2Ray.
  -format string
        Format of input file. (default "json")
  -test
        Test config file only, without launching V2Ray server.
  -version
        Show current version of V2Ray.
```

But the upx compressed one hang on command line, let's skip this.

```
root@OpenWrt:/tmp# ./v2ray --help
<hang, nothing prints>
```

Conslusion

If you also have trouble building programs for a different operating system and architecture, remember the first step is to determine the exact target device information. Some tools may be helpful.

```
# check big-endian/little-endian
$ lscpu | grep "Byte Order"

# check go supported OS/Arch
```

```
$ go tool dist list | grep mips
linux/mips
linux/mips64
linux/mips64le
linux/mipsle
```

The router above is *newifi mini Y1* and labeled *Model R6830* on the backside. It has 16M ROM along with 128MB of memory, and it's Okay to install and run OpenWrt. I searched the device on openwrt.org and luckily it's fully supported. So I followed the instruction page and get the router flashed! If you're insterested on how to flash the router, just leave a comment below and I may write another post on how to install and configure OpenWrt.

References

[1] StackOverflow: Writing and Compiling Program For OpenWrt. https://stackoverflow.com/questions/55878437/writing-and-compiling-program-for-openwrt/60161561#60161561

[2] Wikipedia: Endianness https://en.wikipedia.org/wiki/Endianness

喜欢这篇文章吗？

5 Responses

赞一个 有趣 想要 生气 难过