

前言

Battery Historian是谷歌推出的一款专门分析bugreport的开源工具，具体使用事项可以阅读[《使用 Battery Historian 分析耗电情况》](#)

前置环境安装

本文是基于代码编译，使用Docker的环境存在三个问题：

- 1.几乎所有的镜像都是外网的；
- 2.依赖别人的环境指不定哪天就崩了；
- 3.别人得环境不一定实时更新，可能不兼容最新版本。

需要安装的环境有：Go、git、python、java。

一、Ubuntu环境下

1.安装Go语言

- 1) sudo add-apt-repository ppa:gophers/go（用于配置下载源）
- 2) sudo apt-get update（将下载源部署）
- 3) sudo apt-get install golang（安装golang）
- 4) 安装完之后配置环境变量 `gedit ~/.bashrc`，如果没有这个文件可以使用 `cp /etc/skel/.bashrc ~/` 从系统中拷贝一份出来，在文档末尾添加

```
1 export GOPATH=~/go
2 export PATH=$GOPATH/bin:$PATH
```

- 5) source .bashrc

2.安装git

sudo apt-get install git

3.安装python2.7

sudo apt-get install python2.7

4.安装java

- 1) 从[官网](#)下载java.tar.gz;
- 2) 使用 `tar -zxvf java.tar.gz -C xxx`
- 3)

```

1 xxx就是上一步中的文件路径
2 sudo update-alternatives --install /usr/bin/java java xxx/bin/java 1000
3 sudo update-alternatives --install /usr/bin/javac javac xxx/bin/javac 1000
4 sudo update-alternatives --install /usr/bin/javaws javaws xxx/bin/javaws 1000

```

二、windows环境下

1. 安装Go语言

1) 前往官网<https://golang.google.cn/dl/> 下载

Microsoft Windows

Windows 7 or later, Intel 64-bit processor

 [go1.20.3.windows-amd64.msi](#)

Apple macOS (ARM64)

macOS 11 or later, Apple 64-bit processor

 [go1.20.3.darwin-arm64.pkg](#)

Apple macOS (x86-64)

macOS 10.13 or later, Intel 64-bit processor

 [go1.20.3.darwin-amd64.pkg](#)

Stable versions

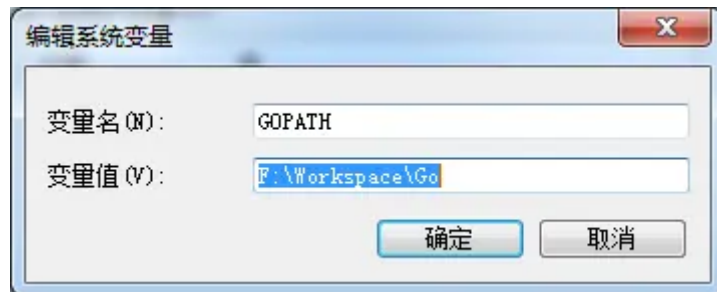
go1.20.3 二选一即可						
File name	Kind	OS	Arch	Size	SHA-256	SHA-512
go1.20.3.src.tar.gz	Source			25MB	e44	
go1.20.3.darwin-amd64.tar.gz	Archive	macOS	x86-64	95MB	c1e	
go1.20.3.darwin-amd64.pkg	Installer	macOS	x86-64	96MB	7at	
go1.20.3.darwin-arm64.tar.gz	Archive	macOS	ARM64	92MB	86b	
go1.20.3.darwin-arm64.pkg	Installer	macOS	ARM64	93MB	b37	
go1.20.3.linux-386.tar.gz	Archive	Linux	x86	96MB	e12	
go1.20.3.linux-amd64.tar.gz	Archive	Linux	x86-64	96MB	979	
go1.20.3.linux-arm64.tar.gz	Archive	Linux	ARM64	91MB	eb1	
go1.20.3.linux-armv6l.tar.gz	Archive	Linux	ARMv6	93MB	b42	
go1.20.3.windows-386.zip	Archive	Windows	x86	109MB	37e	
go1.20.3.windows-386.msi	Installer	Windows	x86	96MB	9b7	
go1.20.3.windows-amd64.zip	Archive	Windows	x86-64	109MB	143	
go1.20.3.windows-amd64.msi	Installer	Windows	x86-64	95MB	233	
Other Ports						
go1.20.3.freebsd-386.tar.gz	Archive	FreeBSD	x86	96MB	340	
go1.20.3.freebsd-amd64.tar.gz	Archive	FreeBSD	x86-64	95MB	216	

2) 配置GOROOT和GOPATH（注意环境变量的配置）

a. GOROOT的作用是告诉Go 命令和其他相关工具，在哪里去找到安装在你系统上的Go包,所以这里配置的是GO的安装目录



b.GOPATH可以简单理解为是工程的目录，所以创建一个GO的工程路径



C.最后配置一下环境变量，把Go的bin目录放到path环境变量中



2.安装python2.7

- 1) 去[官网](#)下在安装包；
- 2) 剩下得步骤略；

3.安装java

4.安装git

正式安装

在正式开始安装前，最好配备一个VPN，因为下面所有的下载都需要上github，众所周知的原因，或者使用 `go env -w GOPROXY=https://goproxy.cn,direct` 命令改成国内下载源。如果只想使用 battery historian，且不需要随时迭代的话，可以直接前往笔者的[github](#)下载，将其中的 `src` 文件夹放入“GOPATH”文件夹下。这个文件夹中是已经全编译的环境，可以直接进入 `src/github.com/google/battery-historian/` 中使用 `go run cmd/battery-historian/battery-historian.go` 运行 battery historian。

1.使用 `go env` 查看环境，如果 `GO111MODULE` 不为 "off"，则使用 `go env -w GO111MODULE=off` 更改为 `GO111MODULE="off"`，如果没有 `GO111MODULE` 变量，则表明 go 版本较低，是用 `go version` 查看版本，只要 1.10 以上就可以不用关注，如果版本过低，可以参考 [《Ubuntu安装go/升级go版本【转载】》](#) 进行版本升级；

```
hoperun@hoperun-ThinkCentre-E77: ~$ go env
GO111MODULE="on"
GOARCH="amd64"
GOBIN="/home/hoperun/go/bin"
GOCACHE="/home/hoperun/.cache/go-build"
GOENV="/home/hoperun/.config/go/env"
GOEXE=""
GOFLAGS=""
GOHOSTARCH="amd64"
GOHOSTOS="linux"
GONOPROXY=""
GONOSUMDB=""
GOOS="linux"
GOPATH="/home/hoperun/go"
GOPRIVATE=""
GOPROXY="https://goproxy.io,direct"
GOROOT="/usr/lib/go-1.13"
GOSUMDB="sum.golang.org"
GOTMPDIR=""
GOTOOLDIR="/usr/lib/go-1.13/pkg/tool/linux_amd64"
GCCGO="gccgo"
AR="ar"
CC="gcc"
CXX="g++"
CGO_ENABLED="1"
GOMOD="/dev/null"
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 -fmessage-length=0 -fdebug-prefix-map=/tmp/go-build"
hoperun@hoperun-ThinkCentre-E77:~$ go get -u github.com/golang/protobuf/proto;go
go: finding google.golang.org/protobuf v1.30.0
go: downloading google.golang.org/protobuf v1.30.0
go: extracting google.golang.org/protobuf v1.30.0
go: finding github.com/google/battery-historian latest
hoperun@hoperun-ThinkCentre-E77:~$
```

2.使用如下命令下载Battery Historian源码及其编译所需的环境（如果无法正常下载，可以前往作者的[github](#)上手动下载，将其中得src.zip文件解压后放入“GOPATH”文件夹下）；

```
1 go get -u github.com/golang/protobuf/proto
2 go get -u github.com/golang/protobuf/protoc-gen-go
3 go get -u github.com/google/battery-historian/...
```

3.进入到前面配置的“GOPATH”文件目录中，并进入其中的src/github.com/google/battery-historian；

4.使用 go env -w GO111MODULE=on 更改GO111MODULE为on；

5.使用 go run setup.go 开始编译（出现下图中类似得 i/o timeout 表明无法正常下载，可以前往[closure-library](#)自行下载，并放入src/github.com/google/battery-historian/third_party中，如果还是报相同错误，说明你的电脑环境完全不能上github，还请下载src文件夹进行替换）；

```
hoperun@hoperun-HP-EltteDesk-880-G6-tower-PC:~/go/src/github.com/google/battery-historian$ go run setup.go
build command-line-arguments: cannot load github.com/google/battery-historian/bugreportutils: module github.com/google/battery-historian/bugreportutils/@v/list: dial tcp 172.217.163.49:443: i/o timeout
```

6.出现下图后就表明编译成功，可以使用 go run cmd/battery-historian/battery-historian.go 运行battery historian。

```
hoperun@hoperun-ThinkCentre-E77:~/go/src/github.com/google/battery-historian/third_party/closure-library$ go run setup.go
go: finding github.com/google/battery-historian latest

Generating JS runfiles...

Generating optimized JS runfiles...
```

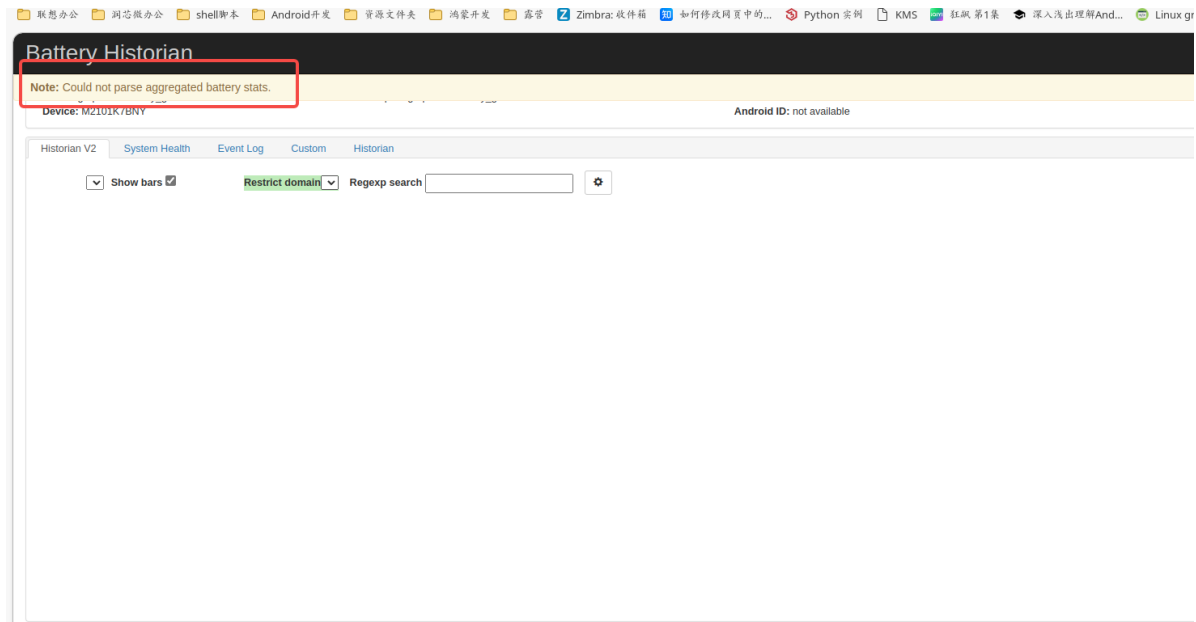
可能存在的问题

1.使用 `go run setup.go` 编译battery historian可能报错js error，如下图所示，不一定完全一样：

```
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/testing/csp_test.js:111: WARNING - Parse error. this language feature is only supported for ECMAScript8 mode or better: async function
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/ui/ac/richmotearraymatcher_test.js:68: WARNING - Parse error. unknown @suppress parameter: strictMissingProperties
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/ui/ac/richmotearraymatcher_test.js:82: WARNING - Parse error. unknown @suppress parameter: strictMissingProperties
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/ui/keyboardeventdata.js:146: WARNING - Parse error. unknown @suppress parameter: strictMissingProperties
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/url/url.js:616: WARNING - Parse error. unknown @suppress parameter: strictMissingProperties
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/i18n/durationformat.js:91: ERROR - Parse error. primary expression expected
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/labs/useragent/test_agentdata.js:67: ERROR - Parse error. ')' expected
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/storage/mechanism/mechanismtests.js:31: ERROR - Parse error. ')' expected
/home/hoperun/go/src/github.com/google/battery-historian/third_party/closure-library/closure/goog/streams/full_test_cases.js:633: ERROR - Parse error. '(' expected
4 error(s), 106 warning(s)
hoperun@hoperun-ThinkCentre-E77:~/go/src/github.com/google/battery-historian$
```

可以前往[no-ssr-battery-historian](https://github.com/google/battery-historian)下载必要的环境（如果下载的是笔者的代码，no-ssr-battery-historian存放在 `src/github.com/google/` 中），将其中的cdn文件夹放入 `src/github.com/google/battery-historian/third_party` 中，将其中的 `base.html` 替换掉 `src/github.com/google/battery-historian/templates` 中的 `base.html`，如果进行完以上操作后仍然报js编译异常，可使用 `go env` 查看GO111MODULE是否为on，若为on，则使用 `go env -w GO111MODULE=off` 改为off。

2.打开bugreport后，没有折线图，并且网页弹窗提醒 `Note: Could not parse aggregated battery stats.`，可以参考[lilydjwg/battery-historian](https://lilydjwg.github.io/battery-historian)的提交：



```

10  checkinparse/checkin_parse.go
1979 1979 //
1980 1980 // format: <idle_time>, <rx_time>, <power_ma_ms>, tx_time...
1981 1981 func parseControllerData(pc checkinutil.Counter, section string, record []string) (*bspb.BatteryStats_ControllerActivity, error) {
1982 -   var idle, rx, pwr int64
1982 +   var idle, rx, pwr float64
1983 1983 rem, err := parseSlice(pc, section, record, &idle, &rx, &pwr)
1984 1984 if err != nil {
1985 1985     return nil, err
1986 1986 }
1987 1987 return nil, fmt.Errorf("%s didn't contain any transmit level data: \"%v\"", section, record)
1988 1988 }
1989 1989 c := &bspb.BatteryStats_ControllerActivity{
1990 1990     IdleTimeMsec: proto.Int64(idle),
1991 -   RxTimeMsec:    proto.Int64(rx),
1992 -   PowerMah:      proto.Int64(pwr),
1993 +   IdleTimeMsec: proto.Int64(int64(idle)),
1993 +   RxTimeMsec:  proto.Int64(int64(rx)),
1993 +   PowerMah:    proto.Int64(int64(pwr)),
1994 1994 }
1995 1995 for i, t := range rem {
1996 -   tm, err := strconv.Atoi(t)
1996 +   tm, err := strconv.ParseFloat(t, 64)
1997 1997 if err != nil {
1998 1998     return nil, fmt.Errorf("%s contained invalid transmit value: %v", section, err)
1999 1999 }

```

```

2  packageutils/packageutils.go
51 51 )
52 52
53 53 // abrUIDRE is a regular expression to match an abbreviated uid (ie u0a2). Based on the format printed in frameworks/base/core/java/android/os/UserHan
54 - var abrUIDRE = regexp.MustCompile("u(?P<userId>\\d+)(?P<aidType>[ias])(?P<appId>\\d+)")
54 + var abrUIDRE = regexp.MustCompile("u(?P<userId>\\d+)(?P<aidType>[ias#])(?P<appId>\\d+)")
55 55
56 56 // This list is not comprehensive but it will cover the most common cases. The list was curated
57 57 // from the output of running both 'adb shell dumpsys activity providers' and

```