**Battery Historian**

说明:本文是对github中battery-historian的理解翻译整理（并为实践操作）

Battery Historian是Android上一个工具，可以用它检测电池的相关信息和事件，支持android5.0及后续版本，当设备不在充电的时候。它允许应用开发人员在一个具有平移和缩放功能的时间轴上可视化系统及应用层事件。很容易看各种各样汇总统计的数据，在设备停止充电之后。它可以选择一个应用程序并检查被选择的应用对电池产生影响的细节。它还允许对两个bugreports进行对比，对不同的关键电池指标进行highlight

**入门指南**

使用 Docker（相关链接：<https://docs.docker.com/engine/installation/>）

安装 Docker(for Windows https://download.docker.com/win/stable/InstallDocker.msi)

安装运行Battery Historian镜像，选择一个端口号，并在下面的命令中替换到<port>

docker -- run -p <port>:9999 gcr.io/android-battery-historian/stable:3.0 --port 9999

**对于linux和Mac OS X平台：**

* 替换端口号后，运行[http://localhost:<port](http://localhost:%3cport)>就可以运行Historian工具

**对于Windows平台：**

* 需要在BIOS中设置[Virtualization为enable](http://www.itworld.com/article/2981515/virtualization/virtualbox-diagnose-and-fix-vt-xamd-v-hardware-acceleration-errors.html)
* 如果你开始启动Docker，它会告诉你正在使用的设备的IP地址（例如：IP地址是123.456.78.90，Historian工具将会在[http://123.456.78.90:<port](http://123.456.78.90:%3cport)>中运行）

关于端口转发的更多信息，请参阅[Docker文档](https://docs.docker.com/engine/reference/run/#/expose-incoming-ports)

从源代码构建

一、确保安装了Golang的最低版本1.8.1：

1. <http://golang.org/doc/install>遵循链接中说明下载和安装Go语言的编译器和工具以及函数库
2. <http://golang.org/doc/code.html#Organization>按照链接中的指令创建一个worksapce目录
3. 把GOPATH和GOBIN环境变量正确的设置添加到$PATH环境变量中（$GOBIN should be set to $GOPATH/bin）

* Windows设置环境变量：计算机->属性->高级系统设置->环境变量->系统变量->path 中添加文件路径
* Linux和MAC OS X：将下面的命令添加到你的 ~/.bashrc或者 ~/.profile中（假设你的workspace是$HOME/work）:

export GOPATH=$HOME/work

export GOBIN=$GOPATH/bin

export PATH=$PATH:$GOBIN

二、[安装git](https://git-scm.com/downloads)

三、安装[python2.7](https://www.python.org/downloads/)（注意不是python 3），并配置环境变量

四、安装[JAVA JDK](http://www.oracle.com/technetwork/java/javase/downloads/index.html)，并配置环境变量

五、下载Battery Historian的代码和它的依赖关系

六、最后运行 Battery Historian！

$ cd $GOPATH/src/github.com/google/battery-historian

# Compile Javascript files using the Closure compiler

$ go run setup.go

# Run Historian on your machine (make sure $PATH contains $GOBIN)

$ go run cmd/battery-historian/battery-historian.go [--port <default:9999>]

注意！一定要在$GOPATH/src/github.com/google/battery-historian目录下运行

battery-historian

cd $GOPATH/src/github.com/google/battery-historian

go run cmd/battery-historian/battery-historian.go [--port <default:9999>]

如何获取一个bugreport

使用android4.2或更高的版本。首先，你需要开启usb调试模式（设置->系统->开发者选项），~~默认情况下“开发者选项”是隐藏的~~。

你可以使用下面的命令启用它（bugreport）

从运行Android 7.0和更高版本的开发设备中获得一个bug report：

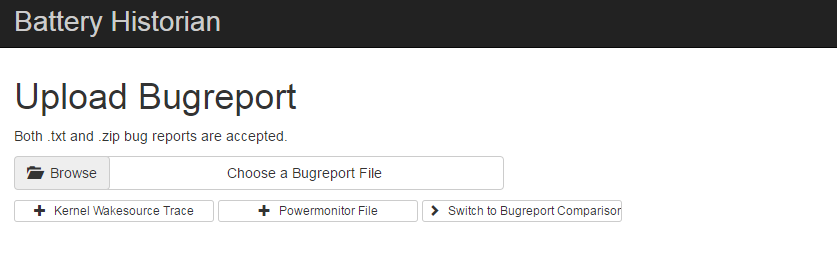
$ adb bugreport bugreport.zip

从android6.0或者以下的版本：

$ adb bugreport > bugreport.txt

**开始分析！**

一切准备就绪！浏览器访问 [http://localhost:9999](http://localhost:9999/)运行historian,上传bugreport.txt文件开始分析



截图：略

高级

清楚电池的数据和历史：

adb shell dumpsys batterystats –reset

Wakelock分析

默认情况下，android不会记录特定于应用程序的用户空间到唤醒转换的时间戳；即使汇总统计是在运行的基础的进行的。如果你想Historian在时间轴上显示每一个wakelock的详细信息。在开始测试前，你需要执行下面的命令开启完整的wakelock报告

adb shell dumpsys batterystats --enable full-wake-history

注意！开启完整的wakelock报告后，电池记录的log几个小时后就会溢出，短时间的测试可以使用这个功能（3-4小时）

**内核跟踪分析**

需要生成一个跟踪文件，该文件记录内核唤醒源和唤醒的活动：

首先，设备连接到电脑，执行下面的命令开启kernel trace log：

$ adb root

$ adb shell

# Set the events to trace.

$ echo "power:wakeup\_source\_activate" >> /d/tracing/set\_event

$ echo "power:wakeup\_source\_deactivate" >> /d/tracing/set\_event

# The default trace size for most devices is 1MB, which is relatively low and might cause the logs to overflow.

# 8MB to 10MB should be a decent size for 5-6 hours of logging.

$ echo 8192 > /d/tracing/buffer\_size\_kb

$ echo 1 > /d/tracing/tracing\_on

然后，使用该设备执行预期的test case

最后，取出kernel trace log

$ echo 0 > /d/tracing/tracing\_on

$ adb pull /d/tracing/trace <some path>

# Take a bug report at this time.

$ adb bugreport > bugreport.txt

**注意：**

Historian的情节和相关的事件都是实时的，