



2022

# timeline 1.0 正式发布

---

一个开源的命令行时间管理大师

by 温兴森

A tool likes 'time' command, which outputs time in every line log.

# CONTENTS

## — 目录 —



time命令痛点



timeline介绍



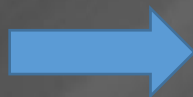
timeline进阶

# 一、time命令介绍

time是一个Linux系统中非常实用统计命令运行时间的命令，例如下面一个简单的demo-cmd.sh，可以非常方便统计出运行时间是9秒。

```
#!/bin/bash
echo "You"
sleep 1
echo "are"
sleep 1
echo "the"
sleep 1
echo "one"
sleep 1
echo "!!!"
sleep 5
```

demo-cmd.sh



```
$ time ./demo-cmd.sh
You
are
the
one
!!!

real    0m9.009s
user    0m0.003s
sys     0m0.004s
```

运行结果

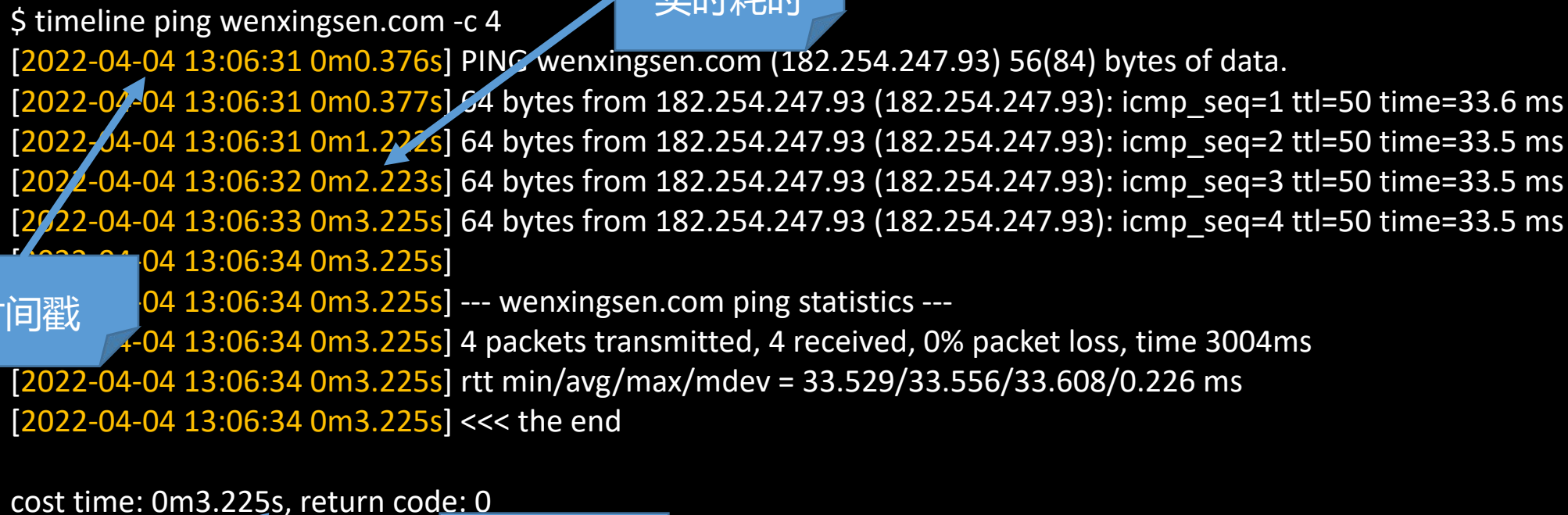
# 一、time命令痛点

但是time命令在使用过程中还是有一些痛点：

1. time命令只能等待程序运行完之后才能在屏幕中输出到运行时间，无法实时知道当前运行了多久了
2. time命令只关心整体结果，并不关心过程，统计的颗粒度太大，无法对过程进行分析和优化
3. time命令缺少对运行时间戳的统计，无法知道某一个日志在某一个时刻发生
4. 以上情况特别是在time命令在长耗时+少日志的情况下，体验特别的不佳

## 二、timeline介绍

A tool likes 'time' command, which outputs time in every line log.  
timeline是一个类似于time命令的工具，为了解决time的痛点而诞生，可以在**每一行**日志实时输出**时间戳**和**耗时**信息。



```
$ timeline ping wenxingsen.com -c 4
[2022-04-04 13:06:31 0m0.376s] PING wenxingsen.com (182.254.247.93) 56(84) bytes of data.
[2022-04-04 13:06:31 0m0.377s] 64 bytes from 182.254.247.93 (182.254.247.93): icmp_seq=1 ttl=50 time=33.6 ms
[2022-04-04 13:06:31 0m1.222s] 64 bytes from 182.254.247.93 (182.254.247.93): icmp_seq=2 ttl=50 time=33.5 ms
[2022-04-04 13:06:32 0m2.223s] 64 bytes from 182.254.247.93 (182.254.247.93): icmp_seq=3 ttl=50 time=33.5 ms
[2022-04-04 13:06:33 0m3.225s] 64 bytes from 182.254.247.93 (182.254.247.93): icmp_seq=4 ttl=50 time=33.5 ms
[2022-04-04 13:06:34 0m3.225s] --- wenxingsen.com ping statistics ---
[2022-04-04 13:06:34 0m3.225s] 4 packets transmitted, 4 received, 0% packet loss, time 3004ms
[2022-04-04 13:06:34 0m3.225s] rtt min/avg/max/mdev = 33.529/33.556/33.608/0.226 ms
[2022-04-04 13:06:34 0m3.225s] <<< the end

cost time: 0m3.225s, return code: 0
```

实时耗时

时间戳

整体耗时

timeline运行ping示意图

## 二、timeline安装

timeline是一个纯C写的一个小程序，支持Linux系统的任意个GCC编译进行编译，github开源下载地址：<https://github.com/wenxingsen/timeline>

运行sh build.sh 既可以完成编译

```
$ sh build.sh
+ gcc timeline.c -o timeline -std=gnu99 -lpthread -Wall -Werror
+ echo 'build success'
build success
```

安装可以拷贝/usr/bin 或者设置timeline目录到PATH

```
sudo cp timeline /usr/bin

# or set PATH to ~/.bashrc
export PATH=/path/to/timeline/dir:$PATH
```

### 三、timeline进阶-自定义时间戳格式

timeline工具可以自定义时间戳格式，通过设置TIMELINE\_FORMAT环境变量，默认为'%Y-%m-%d %H:%M:%S'

```
export TIMELINE_FORMAT="%H:%M:%S"
```

```
$ timeline ./demo-cmd.sh  
[21:50:21 0m00.003s] You  
[21:50:22 0m01.005s] are  
[21:50:23 0m02.006s] the  
[21:50:24 0m03.008s] one  
[21:50:25 0m04.010s] !!!  
[21:50:30 0m09.011s] >>> the end
```

```
cost time: 0m09.011s, return code: 0
```

精简时间显示示例

```
export TIMELINE_FORMAT=""
```

```
$ timeline ./demo-cmd.sh  
[0m00.002s] You  
[0m01.004s] are  
[0m02.005s] the  
[0m03.006s] one  
[0m04.008s] !!!  
[0m09.009s] >>> the end
```

```
cost time: 0m09.009s, return code: 0
```

只显示耗时示例

### 三、timeline进阶-长耗时命令优化

timeline也针对长时间无日志情况作了优化，核心还是要针对当前命令运行的耗时，通过设置PRINT\_EVERY\_SEC环境变量，当在规定的时间内没有新日志输出，空打印一些提示，补充显示耗时

```
export PRINT_EVERY_SEC=1
```

```
[2022-04-09 21:51:59 0m00.002s] You  
[2022-04-09 21:52:00 0m01.003s] are  
[2022-04-09 21:52:01 0m02.005s] the  
[2022-04-09 21:52:02 0m03.006s] one  
[2022-04-09 21:52:03 0m04.007s] !!!  
[2022-04-09 21:52:05 0m06.000s] >>>  
[2022-04-09 21:52:06 0m07.000s] >>>  
[2022-04-09 21:52:07 0m08.000s] >>>  
[2022-04-09 21:52:08 0m09.000s] >>>  
[2022-04-09 21:52:08 0m09.009s] <<< the end
```

```
cost time: 0m09.009s, return code: 0
```

示例中在6,7,8,9秒进行日志补充打印，当然这个例子很小也没有意义，当在长耗时场景是非常有用的，可以设置为60S进行一次空打印



### 三、timeline介绍-时间差值显示

timeline工具显示两个命令之间的运行差值，随时也可以通过耗时信息肉眼算一下，但是与自动打印出来更加直观，可以对定位长耗时非常有用，通过设置PRINT\_DELTA\_TIME环境变量即可。

```
export TIMELINE_FORMAT=""
export PRINT_DELTA_TIME=1

+ ./timeline ./demo-cmd.sh
[0m00.002s +0.002s] You
[0m01.003s +1.001s] are
[0m02.005s +1.002s] the
[0m03.006s +1.001s] one
[0m04.008s +1.002s] !!!
[0m09.010s +5.002s] <<< the end

cost time: 0m09.010s, return code: 0
```

可以看到每个日志的增量时间，非常直观的看到最后一条日志耗时5秒钟