Linux如何查看日志

典型回答

通常我们会使用ELK收集并查看日志，查询日志通过ES语法实现。
在Linux下，常用的有2种方式查看日志：

1. 可以使用`tail -f 日志文件名`滚动输出日志文件。适用于实时观察日志的场景。
2. 可以使用`less 日志文件名`查看整个日志文件，并按`空格`向下翻页，按`ctrl+b`向上翻页，按`/`并输入`关键词`向下搜索，按`?`并输入`关键词`向上搜索，按`q`退出。适用于通篇查看日志的场景。

扩展阅读

如果我们需要对日志进行分析该怎么办？考虑如下场景：

1. 统计httpClient日志文件http_access.log中，某个接口的一天总请求量是多少。
2. 统计客户端访问日志access.log中，返回http状态码为500的接口平均QPS是多少。
3. 统计access.log中，某个接口耗时大于500ms，并按照耗时排序。

在不用ELK或者其他第三方日志平台的情况下，我们使用Linux命令是完全可以做到的。但只是`tail`和`less`命令完成不了上述需求，此时我们需要使用`grep`、`awk`、`sort`、`uniq`、`wc`等命令进行组合才能达到目的。这些命令组合不只用于分析日志，也可以用到其他分析场景（如统计磁盘、CPU使用情况）。

- `grep`命令：用于查找关键词，常用玩法：

```
grep '/xxx.json' http_access.log  #查找http_access.log中包含xxx.json的接口
grep -C10 'NullPointException' warng.log  #查找warn.log中包含
NullPointException的前后10行
grep -i "asc" info.log  #查找info.log中包含asc的行，忽略字符大小写
grep "ASC|asc" info.log  #使用正则表达式匹配asc或者ASC
```

- `wc`命令：用于字词统计，常用玩法：

```
grep '/xxx.json' http_access.log | wc -l  #查找http_access.log中接口名包含
xxx.json的请求总共有多少条，wc -l起到统计行数的作用
```

- `sort`命令：用于排序，常用玩法：

```
# 若test.txt文件内容如下
aa 1
cc 2
bb 0

sort test.txt #按字符顺序排序，结果为
aa 1
bb 0
cc 2

sort -r test.txt #按字符顺序降序，结果为
cc 2
bb 0
aa 1

sort -n -k2 test.txt #按第二列排序（-k2的作用） -n按数字顺序排序，结果为
bb 0
aa 1
cc 2

# 若test.txt文件内容如下
 aa,1
 cc,2
 bb,0

sort -n -k2 -t ',' test.txt #以,作为分隔符（-t ','的作用）并按第二列排序（-k2的作用） -n按数字顺序排序 ，结果为
bb,0
aa,1
cc,2
```

- `uniq`命令：用于去重，常用玩法：

```
grep 'Exception' warn.log| uniq -c #统计warn.log中的异常，去重并显示每种异常有多少个，uniq -c起到去重并统计相同字符出现的次数

#uniq经常与sort一起使用，玩法：
# 若test.txt文件如下：
```

```
aa,1
aa,1
cc,2
bb,0
bb,0

sort -k2 -t ',' -n test.txt|uniq -D #按第二列数字排序，并且只输出重复的行（uniq -D的作用），结果为
bb,0
bb,0
aa,1
aa,1

sort -k2 -t ',' -n test.txt|uniq -D #按第二列数字排序，并且只输出重复的行，只输出一次（uniq -d的作用），结果为
bb,0
aa,1

sort -k2 -t ',' -n test.txt|uniq #按第二列数字排序，并且去重，结果为
bb,0
aa,1
cc,2

sort -u -k2 -t ',' -n test.txt #输出的结果与上面|uniq相同（sort -u的作用）

sort -k2 -t ',' -n test.txt|uniq -u #按第二列数字排序，只输出不重复的行，结果为（uniq -u的作用）
cc,2

sort -k2 -t ',' -n test.txt|uniq -c #按第二列数字排序，统计重复行数（uniq -c的作用），结果为（第一列为相同行出现几次）
2 bb,0
2 aa,1
1 cc,2
```

- `awk`命令：用于处理文本，awk是个非常强大的命令，能够处理输入的每一行字符串，玩法众多（够写一本书了），这里只介绍常用的：

```
# 若test.txt文件如下：
aa 1
bb 1
cc 2

awk '{print $1}' test.txt #按空白字（默认）符拆分每一行为多列，这里$1代表输出第1列，$0代表一整行所有列，$NF代表最后一列，以上命令结果为：
aa
bb
cc

awk -F ',' #-F参数可以修改列的分隔符，这里改为按逗号分割

awk 'BEGIN{print "开始处理每行"}{print $1}END{print "结束处理"}' test.txt #特殊模式，BEGIN在处理每一行之前执行，END在处理每一行完成后执行，以上命令结果为：
开始处理每行
aa
bb
cc
结束处理
```

我们利用以上的命令，来回答扩展阅读开头的三个问题：

> 说明：access.log和http_access.log遵循如下格式：
> `[日志级别] 请求时间年月日 请求时间时分秒 [请求线程名称] - 请求方法 响应状态码 耗时 请求URL requestID`
> 每组信息中间用空格分开。
> 如：
> `[INFO] 20231023 13:00:27.049 [abcService-exec-1] - 2 GET 200 600 http://xxx.com/abc/xxx.json d7af416bc1390dc4d9124f147bab4e53 [INFO] 20231023 13:00:27.049 [abcService-exec-1] - 2 GET 500 800 http://xxx.com/abc/yyy.json e7af416bc1390dc4d9124f147bab4e52 [INFO] 20231023 13:00:27.049 [abcService-exec-1] - 2 GET 200 810 http://xxx.com/abc/yyy.json e7af416bc1390dc4d9124f147bab4e52 [INFO] 20231023 13:00:27.050 [abcService-exec-1] - 2 GET 500 210 http://xxx.com/abc/yyy.json e7af416bc1390dc4d9124f147bab4e52 [INFO] 20231023 13:00:27.049 [abcService-exec-1] - 2 GET 200 550 http://xxx.com/abc/aaa.json f7af416bc1390dc4d9124f147bab4e51 [INFO] 20231023 13:00:27.049 [abcService-`

> exec-1] - 2 GET 200 50 http://xxx.com/abc/zzz.json f7af416bc1390dc4d9124f147bab4e51

统计httpClient日志文件http_access.log中，某个接口的一天总请求量是多少。

```
grep "/yyy.json" http_access.log|wc -l
# 结果为:
3

#解释:
1. 通过grep 查找`/yyy.json`的接口请求
2. 通过wc统计总行数，得出总量
```

统计客户端访问日志access.log中，返回http状态码为500的接口平均QPS是多少。

```
grep "/yyy.json" access.log|grep ' 500 ' |awk '{print substr($0,10,15)}'  |
uniq -c  | awk '{sum += $1} END {print sum / NR}'

#结果为:
2

#解释:
1. grep "/yyy.json" access.log: 首先，使用 grep 命令从access.log 文件中筛选出包含 "/yyy.json" 的行。
2. grep ' 500 ': 接下来，使用第二个 grep 命令从前一步的结果中筛选出包含 "500" 的行。这个命令用于查找 HTTP 状态码为 500 的行，表示服务器内部错误。
3. awk '{print substr($0,10,15)}': 使用 awk 命令提取每一行的第 10 到第 25 个字符（共 15 个字符）并打印。这个命令用于提取出包含日期和时间的部分，例如 "20231023 13:00:27"。
4. uniq -c: 接着，使用 uniq 命令计算并显示每个唯一的时间戳出现的次数。-c 选项用于在每行前面显示出现的次数。
5. awk '{sum += $1} END {print sum / NR}': 最后，使用第二个 awk 命令计算出现次数的平均值。它将每一行的第一个字段（出现次数）累加到 sum 变量中，然后在处理完所有行后，通过除以行数 NR 得到平均值，并将结果打印出来。
```

统计access.log中，某个接口耗时大于500ms，并按照耗时排序。

```
awk '$9>500{print $0}' access.log | sort -k9 -n -r |awk '!arr[$10]++ {print $9"ms",$10}'
```

```
#结果为：
810ms http://xxx.com/abc/yyy.json
600ms http://xxx.com/abc/xxx.json
550ms http://xxx.com/abc/aaa.json

#解释：
1. awk 默认将access.log每一行按空格拆分，第9列为响应耗时，$9>500为响应耗时超过500毫秒，{print $0}打印符合条件的整行
2. sort同样默认按空格拆分每一行，对第9列按数字倒序排列
3. 第10列为请求地址，arr[$10] 创建了一个关联数组，用于存储第10列的值。!arr[$10]++ 通过判断数组中的值是否已存在，实现了按第10列去重的功能。如果某个请求地址是第一次出现，则打印该行的第99列和第10列。其中第9列拼个ms打印。
```

awk命令比较复杂，有一本书叫做《sed与awk》，详细讲解俩个命令如何处理文本的。也可以多敲多练，遇到不会的拿样本数据问问chatgpt，会给出正确的命令和详细的解释。