容器网络IO测试工具说明文档

本IO测试工具是基于LINUX下的FIO软件，对FIO进行了二次封装，接口改良，输出优化。本说明文档将包括以下几方面信息：

- 源码地址
- 安装说明
- 接口说明

源码地址

地址：https://github.com/xianglizhi123/fioservice
仓库文件：

文件说明：fioProject,fioTool 这两个可执行程序是本项目的必须程序，两者缺一不可。fioproject.go 是fioProject 执行程序的源码，fioTool.go是fioTool 执行程序的源码（统一采用GO语言编写）。fio_2.2.10-1ubuntu1_amd64.deb 是本项目内置使用的fio版本，使用其它fio版本有可能导致程序运行失败，因此强力建议使用我们提供的fio版本，该版本的deb安装包已在文件仓库中。Dokerfile里面包含了如何构造这个工具的具体步骤。

安装说明

将仓库文件全部拷贝到本地文件夹中，然后输入命令 docker build -t iotestservice:1.0 . 进行镜像构建。

接口说明

提交测试请求接口：

提交测试请求接口 139.9.53.88:32764/ExecuteFio （IP地址以及端口号以实际部署为准），请求方式POST。使用样例：

IO只读模式（提交如下json格式数据到测试请求接口）

```
{
"filename":"/go/src/fioProject/test",
"direct":"1",
"rw":"randread",
"bs":"16k",
"size":"1g",
"numjobs":"1",
"runtime":"30",
"ioengine":"psync",
"iodepth":"1",
"name":"test2"
}
```

IO只写模式

```
{
"filename":"/go/src/fioProject/test",
"direct":"1",
"rw":"randwrite",
"bs":"16k",
"size":"1g",
"numjobs":"1",
"runtime":"30",
"ioengine":"psync",
"iodepth":"1",
"name":"test2"
}
```

IO读写混合模式

{
"filename":"/go/src/fioProject/test",
"direct":"1",
"rw":"randrw",
"bs":"16k",
"size":"1g",
"numjobs":"1",
"runtime":"30",
"ioengine":"psync",
"iodepth":"1",
"rwmixwrite":"70",
"name":"test2"
}

参数详解：

"filename":"/go/src/fioProject/test" 测试文件名称，需要绝对路径。

"direct":"1" 测试过程绕过机器自带的buffer。使测试结果更真实。

"rw":"randwrite" 测试随机写的I/O

"rw":"randrw" 测试随机写和读的I/O

"rw":"randread" 测试随机读的I/O

"bs":"16k" 单次io的块文件大小为16k

"size":"1g" 本次的测试文件大小为1g。

"numjobs":"1" 本次的测试线程为1.

"runtime":"30" 测试时间为30秒。

"ioengine":"psync" io引擎使用pync方式

"rwmixwrite":"70" 在混合读写的模式下，写占70%

"name":"test2" 本次测试工程的名字（自定义）

接口返回样例：

{"fioPid":34}

因为对IO的测试操作是一个持续的过程，如果等待测试完成再返回，则客户端有可能会超时。所以测试请求接口在设计的时候采用了异步的操作，

调用测试请求接口，立马会返回一个fioPid。我们可以利用返回的fioPid进行测试任务的查询，通过测试任务查询接口来查看任务是否已经执行完成。

测试任务查询接口：

139.9.53.88:32764/CheckStatus，将下面的json数据格式提交到本接口，进行测试任务查询（该接口为POST类型）。

{
"fioPid":34
}

返回样例：

{"status":"Finished"}， {"status”:"Running"}

当返回状态为“Finished”时候表示测试任务已经执行完毕，“ Running”状态则表示测试任务还在执行。

测试报告获取接口：

139.9.53.88:32764/GetReport 将下面的json数据格式提交到本接口，进行测试报告获取（该接口为POST类型）。

{
"fioPid":34,
"task":"test2"
}

fioPid 为测试请求接口返回的fioPid,task为本次测试工程名字。

返回结果样例：

{"fioOutcome":
{"eachJob":
[
{"read_bw":"1954.5KB/s","read_bw_details_avg":"1956.42","read_bw_details_max":"2592KB/s",
"read_bw_details_min":"1225KB/s","read_bw_details_per":"100.00%",
"read_bw_details_stdev":"266.75","read_clat_avg":"8180.92usec","read_clat_max":"43806usec",
"read_clat_min":"479usec","read_clat_stdev":"5874.55","read_io":"58656KB",
"read_iops":"122","read_lat_avg":"8181.40usec","read_lat_max":"43807usec",
"read_lat_min":"479usec","read_lat_stdev":"5874.56","read_runt":"30012msec","write_bw":"",
"write_bw_details_avg":"","write_bw_details_max":"","write_bw_details_min":"",
"write_bw_details_per":"","write_bw_details_stdev":"","write_clat_avg":"",

"write_clat_max":"","write_clat_min":"","write_clat_stdev":"","write_io":"","write_iops":"",
"write_lat_avg":"","write_lat_max":"","write_lat_min":"","write_lat_stdev":"","write_runt":""
}
],
"overAll":
{"disk_stats_in_queue":"29685","disk_stats_ios":"3655/1","disk_stats_merge":"0/0",
"disk_stats_ticks":"29676/1","disk_stats_util":"98.96%","total_read_aggrb":"1954KB/s",
"total_read_io":"58656KB","total_read_maxb":"1954KB/s","total_read_maxt":"30012msec",
"total_read_minb":"1954KB/s","total_read_mint":"30012msec","total_write_aggrb":"",
"total_write_io":"","total_write_maxb":"","total_write_maxt":"","total_write_minb":"",
"total_write_mint":""
}
},"code":200,"message":"success"
}

参数说明：read_bw(读带宽），write_bw（写带宽），read_bw_details_avg(平均读带宽），write_bw_details_avg(平均写带宽），read_bw_details_max(最大读带宽），write_bw_details_max(最大写带宽），read_bw_details_per(读占操作比例的多少），write_bw_details_per(写占操作比例的多少），read_bw_details_stdev（读标准差偏移量），write_bw_details_stdev（写标准差便宜量），read_clat_avg（读平均提交延迟，该延迟为内核接受请求到操作完成这个区间的时间），write_clat_avg(写平均提交延迟，该延迟为内核接受请求到操作完成这个区间的时间），read_clat_max（读最大提交延迟，该延迟为内核接受请求到操作完成这个区间的时间），write_clat_max(写最大提交延迟，该延迟为内核接受请求到操作完成这个区间的时间），read_clat_min(读最小提交延迟，该延迟为内核接受请求到操作完成这个区间的时间），write_clat_min(写最小提交延迟，该延迟为内核接受请求到操作完成这个区间的时间），read_io(读IO操作数据量），write_io(写IO操作数据量），read_iops(每秒钟读操作数），write_iops(每秒钟写操作数），read_lat_avg（平均一次读操作时间），write_lat_avg(平均一次写操作时间），read_lat_max(一次读操作最大时间），write_lat_max(一次写操作最大时间）,read_lat_min(一次读操作最小时间），write_lat_min(一次写操作最小时间），read_lat_stdev（读延迟标准偏移量），write_lat_stdev(写延迟标准偏移量），read_runt(整个测试工程中读操作时间），write_runt(整个测试工程中写操作时间）,disk_stats_in_queue(硬盘队列总共花费时间），disk_stats_ios（总共硬盘IO操作），disk_stats_merge（IO调度程序中融合次数），disk_stats_ticks（用来保持硬盘繁忙的时钟数量）,total_read_aggrb(所有线程的总共读带宽），total_write_aggrb(所有线程的总共写带宽），total_read_io（所有线程读操作数据量的和），total_write_io（所有线程写操作数据量的和），total_read_maxb（所有线程中最大的读带宽），total_write_maxb(所有线程中最大的写带宽），total_read_maxt（所有读线程运行时间中最长的那个），total_write_maxt（所有写线程运行时间中最长的那个），total_write_mint(所有写线程运行时间中最短的那个），total_read_mint(所有读线程运行时间中最短的那个）