

CANN
5.0.4.6

日志参考（开放态）

文档版本	01
发布日期	2023-03-17



版权所有 © 华为技术有限公司 2023。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址：深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址：<https://www.huawei.com>

客户服务邮箱：support@huawei.com

客户服务电话：4008302118

前言

概述

本文档详细的描述了日志的处理机制、日志文件的保存目录和查询方式以及日志的开启和设置，为开发和维护人员提供问题调试的参考。

读者对象

本文档主要适用于开发和维护人员。

本文档适用于使用昇腾AI处理器进行推理和训练的开发和维护人员，通过本文档您可以达成：





- 了解日志的开启关闭以及查询方式。
- 能够基于本文档中的日志设置找开发操作过程中生成的对应操作日志以进行问题定位和调试。


掌握以下经验和技能可以更好地理解本文档：

- 熟悉Linux基本命令。
- 对机器学习、深度学习有一定的了解。

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
	表示如不避免则将会导致死亡或严重伤害的具有高等级风险的危害。
	表示如不避免则可能导致死亡或严重伤害的具有中等级风险的危害。
	表示如不避免则可能导致轻微或中度伤害的具有低等级风险的危害。
	用于传递设备或环境安全警示信息。如不避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。

符号	说明
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

目 录

前言..... ii

1 日志处理机制介绍..... 1

2 环境变量说明..... 2

3 日志文件介绍..... 3

4 Host 侧驱动内核态日志介绍..... 5

5 日志类型及级别..... 6

6 日志记录格式介绍..... 8

7 日志配置文件..... 9

8 设置日志级别..... 13

9 重启日志进程..... 15

10 常见问题..... 17

10.1 日志没有正常落盘..... 17

10.2 修改昇腾 AI 处理器环境时区后日志打印的时间戳不正确..... 18

10.3 未设置日志打屏环境变量但在屏幕上仍有日志显示..... 19

11 附录..... 20

11.1 msnpureport 工具使用..... 20

11.2 连续导出 Device 侧的日志和文件..... 25

11.3 昇腾产品形态说明..... 27

1 日志处理机制介绍

概述

日志主要用于记录系统的运行过程及异常信息，为快速定位系统运行中出现的问题及开发过程中程序调试问题提供详细信息。

日志大体分为如下两大类：

- 系统类日志：系统运行产生的日志。主要包括：
 - Control CPU上的系统类日志，包括用户态日志和内核态日志。
 - 非Control CPU（例如TSDUMP/LP）上的系统类日志。
- 应用类日志：运行应用程序产生的日志，主要包括：
 - compiler中各组件（如GE、FE、AICPU、TBE、HCCL等）、runtime中各组件（如ACL、GE、Runtime等）打印的日志。
 - Device侧AICPU、HCCP打印的日志。

日志处理机制

说明

本文中提到的各个场景的详细说明请参见[昇腾产品形态说明](#)。

以下日志文件名中的`id`和`pid`分别代表Device ID和进程ID，请以实际为准。

推理场景（Ascend EP开放态）

1. 日志进程
设备完成驱动安装后日志进程自动启动。包括slogd进程、sklogd进程和log-daemon进程。
2. 采集日志
log-daemon进程采集非Control CPU上的系统类日志；slogd进程采集Control CPU上的系统类日志和应用类日志。
3. 将日志记录到文件中
slogd对接收的日志进行分类，应用类日志由slogd记录在文件名称以device-app-`pid`开头的日志文件中；Control CPU上的系统类日志由slogd记录在文件名称以device-os开头的日志文件中。
非Control CPU上的系统类日志由log-daemon进程记录在文件名称以device-`id`开头的日志文件中。

2 环境变量说明

目前支持的环境变量如表2-1所示。

表 2-1 环境变量说明

环境变量名	功能描述	详细说明
ASCEND_PROCESS_LOG_PATH	设置日志落盘路径。	日志文件介绍
ASCEND_GLOBAL_LOG_LEVEL	设置应用类日志的全局日志级别。	设置日志级别
ASCEND_GLOBAL_EVENT_ENABLE	设置应用类日志是否开启Event日志。	设置日志级别
ASCEND_SLOG_PRINT_TO_STDOUT	设置是否开启日志打屏。	日志文件介绍
ASCEND_LOG_DEVICE_FLUSH_TIMEOUT	设置业务进程退出前的延时时间，用于等待Device侧应用类日志回传到Host侧，超时后，业务进程退出。	日志文件介绍
ASCEND_HOST_LOG_FILE_NUM	设置应用类日志目录plog和device- <i>id</i> 下分别能够存储的单个进程的日志文件数量。	日志文件介绍

3 日志文件介绍

本节介绍日志的存储路径、文件名称及记录的主要信息。

日志的默认输出方式为将日志保存在log文件中，也可以通过设置环境变量ASCEND_SLOG_PRINT_TO_STDOUT，将日志打屏显示，只在当前shell下生效。设置命令如下：

```
export ASCEND_SLOG_PRINT_TO_STDOUT=1
```

ASCEND_SLOG_PRINT_TO_STDOUT取值范围如下：

- 0：表示采用日志的默认输出方式。
- 1：表示日志打屏显示。
- 其他值为非法值。

通过执行`echo $ASCEND_SLOG_PRINT_TO_STDOUT`命令可以查看设置的值，如果查询为非法值或者空，表示采用日志默认输出方式。

说明

上述日志中`id`和`pid`分别代表Device ID和进程ID，请以实际为准；日志文件中的“*”表示该日志文件创建时的时间戳。

推理场景（Ascend EP 开放态）

表 3-1 日志文件介绍

存储路径	说明
/var/log/npu/slog/device-os/ device-os_*.log	Control CPU上的系统类日志，包括用户态日志和内核态日志。主要采集以下模块的日志： <ul style="list-style-type: none">• SLOG• IDEDD• HDC• Driver•

存储路径	说明
/var/log/npu/slog/device- <i>id</i> / device- <i>id</i> _.log	非Control CPU上的系统类日志，主要采集以下模块的日志： <ul style="list-style-type: none">• TS• TSDUMP• LP
/var/log/npu/slog/device-app- <i>pid</i> / device-app- <i>pid</i> _.log	应用类日志，即运行应用程序产生的日志。 说明 <ul style="list-style-type: none">• 应用类日志整体进行老化，默认支持存储24个device-app-<i>pid</i>目录，系统每隔15秒对该目录进行一次扫描，如果日志超过配置的存储限制，将会自动删除最早的device-app-<i>pid</i>目录。存储device-app-<i>pid</i>目录的数量配置以及每个device-app-<i>pid</i>目录存储的日志文件的数量和单个日志文件的大小具体请参见日志配置文件。• 最多支持24个应用进程并发处理，如果超过该数值，可能会导致日志丢失。
/var/log/npu/slog/slogd/ slogdlog	维测日志。记录日志工具自身的运行信息，用于日志工具自身问题定位。 日志具备老化策略，当slogdlog文件达到规定大小（1MB）后，名称变更为slogdlog.old进行备份（如果已有备份文件，则删除旧的备份文件）。
注：上述日志中 <i>id</i> 和 <i>pid</i> 分别代表Device ID和进程ID，请以实际为准；日志文件中的“*”表示该日志文件创建时的时间戳。	

4 Host 侧驱动内核态日志介绍

须知

Host侧驱动日志仅适用于Ascend EP场景。

Host侧驱动产生的内核态日志，日志被记录到“/var/log/messages”文件中。日志格式增加**ascend**标记用于与其他日志进行区分。

说明

- 如果没有“/var/log/messages”文件，看不到Host侧驱动日志，可以通过**ps -ef | grep rsyslogd**命令查看是否存在rsyslogd服务。若显示类似如下信息，表示已存在；如果不存在，请先安装rsyslogd服务。

```
[root@localhost ~]# ps -ef | grep rsyslogd
root    1096    1  0 15:41 ?        00:00:02 /usr/sbin/rsyslogd -n
root    20242 20224  0 16:59 pts/2    00:00:00 grep  --color=auto rsyslogd
```
- 执行**cat /dev/null > messages**命令可以清除“/var/log/messages”文件中的日志。

5 日志类型及级别

各日志级别介绍如[表5-1](#)所示，如需更改系统日志级别，请参考级别定义描述进行选择。

日志级别等级由低到高顺序：DEBUG < INFO < WARNING < ERROR，级别越低，输出日志越详细。

表 5-1 日志级别

日志类型	日志级别	定义
运行日志	ERROR	一般错误级别，该级别的日志记录了如下错误： <ul style="list-style-type: none">• 非预期的数据或者事件。• 影响面较大且模块内部能够处理的错误。• 限制在模块内的错误。• 对其他模块有轻微影响的错误，例如统计任务创建失败。• 引起调用失败的错误。• 在业务逻辑错误的情况下记录错误状态的信息及造成错误的可能原因。
	WARNING	警告级别，系统和预期的状态不一致，但不影响整个系统的运行。
	INFO	正常级别，系统正常运行的信息。
	DEBUG	调试级别，该级别的日志记录了调试信息，便于开发人员或维护人员定位问题。
	NULL	表示NULL级别，不输出日志。
Trace日志	-	记录模块运行全生命周期中模块流程交互、状态迁移时的信息。
Oplog日志	-	记录进程任务创建或销毁资源时的日志。

日志类型	日志级别	定义
EVENT日志	-	输出全系统最关键日志，例如：整网运算启动\完成\异常终止，内存不够，单板温度过高等。

6 日志记录格式介绍

本节介绍系统记录日志的格式以及各字段的含义，便于理解日志所记录的信息。

日志样例如下：

```
[ERROR] TEFUSION(12940,atc):2021-10-17-05:54:07.599.074 [tensor_engine/te_fusion/pywrapper.cc:33]InitPyLogger Failed to import te.platform.log_util
```

日志格式如下：

```
[Level] ModuleName(PID,PName):DateTimeMS [FileName:LineNumber]LogContent
```

表 6-1 日志字段说明

字段	说明
Level	日志级别。运行日志存在5种日志级别：ERROR、WARNING、INFO、DEBUG、EVENT。
ModuleName	产生日志的模块的名称。
PID	进程ID。
PName	进程名称。
DateTimeMS	日志打印时间，格式为：yyyy-mm-dd-hh:mm:ss.SSS。
FileName:LineNumber	调用日志打印接口的文件及对应的行号。
LogContent	各模块具体的日志内容。

7 日志配置文件

本节介绍Device侧的日志配置文件。该文件中记录了Device侧系统类日志的日志级别、日志输出路径、日志数量、单个日志文件大小等配置信息。

须知

- 多用户场景下，仅支持日志进程的运行用户修改配置文件。
- 用户在修改日志配置项时，请严格按照相关配置项说明表格中的要求，根据建议值或取值范围进行配置，若不按照配置项配置要求，可能会导致系统异常。
- 手动修改Device侧配置文件后，需要重启slogd进程，使配置生效（此外需要注意：DeviceMaxFileNum和DeviceMaxFileSize字段为非Control CPU上日志配置项，需要重启log-daemon进程，使配置生效）。重启日志进程方法请参见[重启日志进程](#)。

Device侧配置文件所在路径为`/var/log/npu/conf/slog/slog.conf`，该文件中的配置字段样例如下所示。不同场景配置文件内容可能不同，请以实际为准。以推理场景（Ascend EP标准态）为例：

```
#note, 0:debug, 1:info, 2:warning, 3:error, 4:null(no output log), default(1)
global_level=3
# Event Type Log Flag, 0:disable, 1:enable, default(1)
enableEvent=1
# note, 0:debug, 1:info, 2:warning, 3:error, 4:null(no output log), 5:invalid(follow global_level)
SLOG=5                # Slog
IDEDD=5               # ascend debug device agent
DVPP=5                # DVPP
CCE=5                 # CCE
HDC=5                 # HDC
DRV=5                 # Driver
MDCDEFAULT=5          # MDC UNDEFINE
DEVMM=5               # Dlog memory managent
KERNEL=5              # Kernel
LIBMEDIA=5            # Libmedia
ASCENDDK=5            # AscendDK
ROS=5                 # ROS
HCCP=5
ROCE=5
PROFILING=5           # Profiling
APP=5                 # User Application call HIAI_ENGINE_LOG
TDT=5
MD=5
MB=5
ME=5
```

```
BBOX=5
TEEOS=5
FV=5

# note, 0:debug, 1:info, 2:warning, 3:error, 4:null(no output log), 5:invalid(follow global_level)
TS=5
TSDUMP=5
LP=5

# set device-os_XXX.log file num, range is [1, 1000], default(3)
DeviceOsMaxFileNum=3
# set device-os_XXX.log file size, range is [1048576, 104857600], default(2097152)
DeviceOsMaxFileSize=2097152
# set device-x_XXX.log file num, range is [1, 1000], default(10)
DeviceMaxFileNum=10
# set device-x_XXX.log file size, range is [1048576, 104857600], default(2097152)
DeviceMaxFileSize=2097152
# set device-app-XXX_XXX.log file num, range is [1, 1000], default(2)
DeviceAppMaxFileNum=2
# set device-app-XXX_XXX.log file size, range is [524288, 104857600], default(524288)
DeviceAppMaxFileSize=524288
# set device-app-XXX dir nums, range is [1, 96], default(24)
DeviceAppDirNums=24
# set log file root path
logAgentFileDir=/var/log/npu/slog
# set log buf size, range is [64*1024, 1024*1024], default(256*1024)
LogBufSize=262144
# set log zip(1) or not(0), default(0)
zip_switch=0
```

表 7-1 相关配置项说明

配置项	说明
global_level	设置全局日志级别。取值范围： <ul style="list-style-type: none">0：表示DEBUG级别。1：表示INFO级别。2：表示WARNING级别。3：表示ERROR级别。默认值。4：表示NULL级别，不输出日志。其他值：非法值。
enableEvent	设置是否开启Event日志。取值范围： <ul style="list-style-type: none">1：开启Event日志。默认值。0：不开启Event日志。

配置项	说明
SLOG、 IDEDD、 DVPP……	<p>设置Control CPU上各模块的日志级别。取值范围：</p> <ul style="list-style-type: none"> 0：表示DEBUG级别。 1：表示INFO级别。 2：表示WARNING级别。 3：表示ERROR级别。 4：表示NULL级别，不输出日志。 5：无效值，模块日志级别取global_level设置的级别。默认值。 <p>说明</p> <ul style="list-style-type: none"> 模块日志级别为5，而全局日志级别为正常值（0、1、2、3），则模块跟随全局日志级别。 模块日志级别为5，而全局日志级别为4，则模块日志不打印。 模块日志级别为5，而全局日志级别为非法值，则模块日志级别为ERROR。 模块日志级别为4，则模块不打印日志。 模块日志级别为正常值（0、1、2、3），则模块的日志级别取该值对应级别。
TS、 TSDUMP……	<p>设置非Control CPU上各模块的日志级别。取值范围：</p> <ul style="list-style-type: none"> 0：表示DEBUG级别。 1：表示INFO级别。 2：表示WARNING级别。 3：表示ERROR级别。 4：表示NULL级别，不输出日志。 5：无效值，模块日志级别取global_level设置的级别。默认值。 <p>说明</p> <ul style="list-style-type: none"> 模块日志级别为5，而全局日志级别为正常值（0、1、2、3），则模块跟随全局日志级别。 模块日志级别为5，而全局日志级别为4，则模块日志不打印。 模块日志级别为5，而全局日志级别为非法值，则模块日志级别为ERROR。 模块日志级别为4，则模块不打印日志。 模块日志级别为正常值（0、1、2、3），则模块的日志级别取该值对应级别。
DeviceOsMax FileNum	<p>device-os目录下保存device-os_*.log日志文件的数量，当日志文件数目大于该数目时发生滚动，新日志覆盖最早的日志。默认值为3。</p>
DeviceOsMax FileSize	<p>device-os目录下单个device-os_*.log日志文件的大小，当日志文件大小超过该值时，则生成新的日志文件。</p> <p>说明</p> <p>当前默认值为2MB，您可以根据实际情况调整大小，最大不超过100MB。如果设置值小于1MB，系统默认为1MB。</p>

配置项	说明
DeviceMaxFileNum	device- <i>id</i> 目录下保存device- <i>id</i> _.log日志文件的数量，当日志文件数目大于该数目时发生滚动，新日志覆盖最早的日志。默认值为10。
DeviceMaxFileSize	device- <i>id</i> 目录下单个device- <i>id</i> _.log日志文件的大小，当日志文件大小超过该值时，则生成新的日志文件。 说明 当前默认值为2MB，您可以根据实际情况调整大小，最大不超过100MB。如果设置值小于1MB，系统默认为1MB。
DeviceAppDirNums	保存device-app- <i>pid</i> 目录的数量，默认值EP场景为24个，取值范围为1~96。
DeviceAppMaxFileNum	每个device-app- <i>pid</i> 目录下保存device-app- <i>pid</i> _.log日志文件的数量，当device-app- <i>pid</i> _.log日志文件数目大于该数目时发生滚动，新日志覆盖最早的日志。默认值为2。
DeviceAppMaxFileSize	每个device-app- <i>pid</i> 目录下单个device-app- <i>pid</i> _.log日志文件的大小，当device-app- <i>pid</i> _.log日志文件大小超过该值时，则生成新的日志文件。 说明 默认值为0.5MB，您可以根据实际情况调整大小，最大不超过100MB。
LogBufSize	系统&应用日志（device-os_*.log和device-app- <i>pid</i> _.log）缓存大小，默认值为256KB，取值范围为64KB~1024KB。
logAgentFileDir	日志文件路径。如果修改该路径，需确保运行日志进程的用户对该路径有读写权限。 说明 该路径最长支持255字节。如果输入超过255字节长度，系统自动取截断后的路径，并在截断后的路径下保存日志。 对于推理场景（Ascend EP标准态），如果修改该路径，需确保该路径存在且属主为运行日志进程的用户；如果是container场景，还需要参考《CANN 软件安装指南》将宿主机的日志路径、日志配置文件挂载到容器。
zip_switch	是否进行日志文件压缩。 <ul style="list-style-type: none"> 0：不压缩日志文件。默认值。 1：压缩日志文件。

8 设置日志级别

日志记录了运行环境的运行情况和功能流程的处理情况，是维护人员查看系统状态、进行问题定位的重要工具和手段。日志模块根据系统设置的日志级别，记录不同详细程度的内容，满足不同系统维护需求。

说明

- 目前应用类日志不支持单独设置各个模块的日志级别。
- 应用类日志级别支持在容器内或物理机内设置；Device侧系统类日志级别不支持在容器内设置。
- 算力切分场景，不支持设置日志级别。

推理场景（Ascend EP 开放态）

应用类日志级别设置

通过环境变量设置应用类日志的全局日志级别和是否开启Event日志，具体如下：

- 设置全局日志级别：
通过ASCEND_GLOBAL_LOG_LEVEL环境变量设置全局日志级别，只在当前shell下生效，仅对当前窗口设置全局级别及各模块日志级别为该值。如果用户环境变量设置了非法值（或没有设置值），缺省设置为ERROR级别。通过执行**echo \$ASCEND_GLOBAL_LOG_LEVEL**命令可以查看设置的日志级别，如果查询为非法值或者空，表示日志级别为缺省级别。
设置ASCEND_GLOBAL_LOG_LEVEL环境变量举例：

```
export ASCEND_GLOBAL_LOG_LEVEL=1
```


ASCEND_GLOBAL_LOG_LEVEL取值范围如下：
 - 0：对应DEBUG级别。
 - 1：对应INFO级别。
 - 2：对应WARNING级别。
 - 3：对应ERROR级别。
 - 4：对应NULL级别，不输出日志。
 - 其他值为非法值。
- 设置是否开启Event日志：
通过ASCEND_GLOBAL_EVENT_ENABLE环境变量设置是否开启Event日志，只在当前shell下生效，仅对当前窗口设置为该值。如果用户环境变量设置了非法值

（或没有设置值），缺省设置为开启Event日志。通过执行**echo \$ASCEND_GLOBAL_EVENT_ENABLE**命令可以查看设置的值，如果查询为非法值或者空，表示使用缺省值。

设置ASCEND_GLOBAL_EVENT_ENABLE环境变量举例：

```
export ASCEND_GLOBAL_EVENT_ENABLE=0
```

ASCEND_GLOBAL_EVENT_ENABLE取值范围如下：

- 0：不开启Event日志。
- 1：开启Event日志。
- 其他值为非法值。

系统类日志级别设置

通过/var/log/npu/conf/slog/slog.conf配置文件设置系统类日志级别。

参考[日志配置文件](#)章节内容，修改/var/log/npu/conf/slog/slog.conf配置文件中全局日志级别、模块日志级别和是否开启Event日志。

修改配置文件中日志级别后，重启slogd进程使配置生效。重启方法请参见[重启日志进程](#)。

9 重启日志进程

须知

- 进程重启后，生成的日志信息会写入一个全新的日志文件里（即使之前的日志文件大小未达到限定的值）。
- 修改配置文件后，重启slogd进程时请使用**kill -15**停止slogd进程，这样slogd进程接收到结束信号时，会清理共享内存。如果使用**kill -9**强制停止，slogd进程接收不到结束信号，不会清理共享内存，会导致slogd进程下次启动后获取到旧的日志级别信息，导致落盘的日志级别和配置文件中的不一致。
- 如果**kill -15**反复执行失败，请使用**kill -9**强制停止日志进程。
- 用户需要有登录Device侧的权限，才能重启Device侧的日志进程。

推理场景（Ascend EP 开放态）

步骤1 执行如下命令停止日志进程。

kill -15 进程ID

其中，进程（slogd、sklogd或log-daemon）ID可以通过**ps -elf | grep log**命令查询。

步骤2 重新手动拉起日志进程。

1. 执行如下命令切换到普通用户（如HwHiAiUser）。

```
su HwHiAiUser
```

2. 执行如下命令手动拉起日志进程。

- 拉起slogd进程。

```
nohup /var/slogd > /dev/null 2>&1 &
```

- 拉起sklogd进程。

```
nohup /var/sklogd > /dev/null 2>&1 &
```

- 拉起log-daemon进程。

```
nohup /var/log-daemon > /dev/null 2>&1 &
```

3. 执行如下命令确认日志进程（slogd、sklogd或log-daemon）是否被拉起。

```
ps -elf | grep log
```

----结束

重启异常处理

如果重新拉起日志进程失败，可能为以下原因导致，请参考处理：

- slogd.pid属主异常
进入/usr/slog目录，执行**ls -l**命令查看slogd.pid属主是否为root，如果属主是root，建议删除该文件后重新手动拉起日志进程。
- /var目录所在磁盘使用率达100%
进入根目录，运行命令**df -h**，如果/var目录所在磁盘使用率达到100%导致slogd启动失败，则进入“/var/log/npu/slog”目录，手动删除一些较大的、时间较早的日志文件。删除完后重新手动拉起日志进程。

10 常见问题

[10.1 日志没有正常落盘](#)

[10.2 修改昇腾AI处理器环境时区后日志打印的时间戳不正确](#)

[10.3 未设置日志打印环境变量但在屏幕上仍有日志显示](#)

10.1 日志没有正常落盘

推理场景（Ascend EP 开放态）

应用类日志没有正常落盘

如果应用类日志没有正常落盘，请参照如下步骤处理：

步骤1 执行如下命令查看应用进程依赖的动态库是否正确。

```
ldd xxx
```

xxx为二进制应用进程。

步骤2 执行如下命令查看日志落盘路径（“/var/log/npu/slog”）所在的磁盘空间是否已满。

```
df -h
```

步骤3 执行如下命令查看slogd进程是否存在。

```
ps -elf | grep slogd
```

若返回slogd进程相关信息，说明slogd进程存在。若slogd进程不存在，可以执行如下步骤尝试手动拉起slogd进程。

1. 执行如下命令切换到普通用户（如HwHiAiUser）。

```
su HwHiAiUser
```

2. 执行如下命令手动拉起slogd进程。

```
nohup /var/slogd > /dev/null 2>&1 &
```

3. 执行如下命令确认slogd进程是否被拉起。

```
ps -elf | grep slogd
```

步骤4 若以上均无问题，但应用类日志仍没有正常落盘，可以尝试重启slogd进程，具体请参见[重启日志进程](#)。

----结束

系统类日志没有正常落盘

如果系统类日志没有正常落盘，请参照如下步骤处理：

步骤1 执行如下命令查看相关日志进程（slogd、sklogd和log-daemon）是否存在。

```
ps -elf | grep log
```

若显示进程相关信息，说明相关日志进程已存在。若不存在，可以执行如下步骤尝试手动拉起相关日志进程：

1. 执行如下命令切换到普通用户（如HwHiAiUser）。

```
su HwHiAiUser
```

2. 执行如下命令手动拉起相关日志进程。

- 拉起slogd进程。

```
nohup /var/slogd > /dev/null 2>&1 &
```

- 拉起sklogd进程。

```
nohup /var/sklogd > /dev/null 2>&1 &
```

- 拉起log-daemon进程。

```
nohup /var/log-daemon > /dev/null 2>&1 &
```

3. 执行如下命令确认相关日志进程是否被拉起。

```
ps -elf | grep log
```

步骤2 执行如下命令查看日志落盘路径（“/var/log/npu/slog”）所在的磁盘空间是否已满。

```
df -h
```

步骤3 若以上均无问题，但系统类日志仍没有正常落盘，可以尝试重启日志相关进程，具体请参见[重启日志进程](#)。

----结束

10.2 修改昇腾 AI 处理器环境时区后日志打印的时间戳不正确

异常现象

日志打印信息中时间戳与系统环境的时间不一致。

可能原因

产生该问题的可能原因是用户启动了推理任务后修改了系统环境的时区。

处理方式

参考以下方式处理：

重新拉起推理任务。

10.3 未设置日志打屏环境变量但在屏幕上仍有日志显示

异常现象

用户未设置日志打屏环境变量（`export ASCEND_SLOG_PRINT_TO_STDOUT=1`），但是在屏幕上仍有日志显示。

可能原因

日志模块收到日志消息后，首先根据环境变量判断是否打印到屏幕上，若不需要打印到屏幕上，则创建Socket与slogd进程建立连接，将日志发送给slogd进程，由slogd进程落盘。但是如果创建Socket失败，则会将日志打印到屏幕上。

处理方式

创建Socket失败的原因一般是用户在执行用例时，打开文件的数量超过了系统默认的最大数量。可以通过`ulimit -a`和`lsof | wc -l`命令分别查看系统默认打开文件的最大数量（即`open files`字段的取值）以及当前已经打开的文件数量，查看当前已经打开的文件数量是否超过了限制值。如果超过限制值，可以通过`ulimit -n <num>`命令设置打开文件的最大数量，保证当前已经打开的文件数量不超过限制值。

11 附录

- [11.1 msnpureport工具使用](#)
- [11.2 连续导出Device侧的日志和文件](#)
- [11.3 昇腾产品形态说明](#)

11.1 msnpureport 工具使用

工具介绍

msnpureport工具部署在Host侧，该工具有以下用途：

- 导出Device侧的相关日志和文件，包括slog日志、syslog日志、黑匣子和Stackcore文件。导出的日志和文件将存储到运行msnpureport工具的路径下以时间戳命名的子目录中，且slog日志、syslog日志、黑匣子和Stackcore文件分别存储到slog、message、hisi_logs和stackcore文件夹下。
- 查询和设置Device侧slog系统类日志的级别。

须知

- msnpureport工具仅适用于Ascend EP场景。
- msnpureport工具不支持容器场景，部署容器时，禁止将msnpureport工具映射到容器内。
- 不支持多个用户同时运行msnpureport工具。
- msnpureport工具不支持增量导出日志，如果用户想避免老旧日志的影响，提高问题定位的效率，可以定期通过msnpureport工具获取Device侧日志，并自行制作脚本实现日志去重处理。

导出 Device 侧的相关日志和文件

- 步骤1** 登录Host侧服务器。
- 步骤2** 获取msnpureport工具。

msnpureport工具在驱动Driver的安装目录下，路径为“*Driver安装目录*/driver/tools/msnpureport”。

步骤3 在某个有读、写、执行权限的目录（如“/var/log/npu/report”）下执行如下命令，运行msnpureport工具。

Driver安装目录/driver/tools/msnpureport [options]

须知

在加锁的目录下（使用lsattr命令查看目录属性，有“i”选项的为加锁目录），用户没有权限运行msnpureport工具。如果用户想在该目录下运行msnpureport工具，可以通过chattr -i <加锁的目录>命令，将目录的“i”选项撤销。工具运行完后建议通过chattr +i <加锁的目录>命令，将目录的“i”选项加上。为了安全起见，不建议在加锁目录中运行工具。

其中[options]支持的参数及解释请参见表11-1。

表 11-1 参数说明

参数	说明	举例
不指定任何参数	导出所有日志和文件，包括： <ul style="list-style-type: none">• slog日志。• syslog日志。• Stackcore文件。• 黑匣子日志。	<i>Driver安装目录</i> /driver/tools/msnpureport
-a或--all	导出所有日志、文件以及黑匣子设备事件信息，包括： <ul style="list-style-type: none">• slog日志。• syslog日志。• Stackcore文件。• 黑匣子日志、黑匣子设备事件信息。	<i>Driver安装目录</i> /driver/tools/msnpureport -a
-f或--force	导出所有日志、文件以及黑匣子相关信息，包括： <ul style="list-style-type: none">• slog日志。• syslog日志。• Stackcore文件。• 黑匣子日志、黑匣子设备事件信息、黑匣子存储空间中的历史维测信息。	<i>Driver安装目录</i> /driver/tools/msnpureport -f

参数	说明	举例
-t或--type	指定导出的日志类型，取值为： 0：导出所有类型日志，包括slog日志、syslog日志、Stackcore文件、黑匣子日志。 <ul style="list-style-type: none">1：slog日志和syslog日志2：黑匣子日志3：Stackcore文件	Driver安装目录 /driver/tools/ msnpureport -t 1
注：导出的日志和文件保存路径分别为：slog日志，slog目录；syslog日志，message目录；Stackcore文件，stackcore目录；黑匣子日志、黑匣子设备事件信息、黑匣子存储空间中的历史维测信息，hisi_logs目录。		

运行成功后，Device侧的相关日志和文件被导出到Host侧，并存储到当前目录（如“/var/log/npu/report”）下以时间戳命名的文件夹中，具体如下：

- slog日志：“/var/log/npu/report/*/slog”，“*”为时间戳，其具体日志目录如下：

存储目录	说明
dev-os- <i>id</i> /device-os/device-os_*.log	Device侧Control CPU上的系统类日志，包括用户态日志和内核态日志。
dev-os- <i>id</i> /device- <i>id</i> /device- <i>id</i> _.log	Device侧非Control CPU上的系统类日志。
dev-os- <i>id</i> /device-app- <i>pid</i> /device-app- <i>pid</i> _.log	Device侧应用类日志。仅当Device侧应用类日志回传到Host侧失败时，才会在Device侧存储该日志。
dev-os- <i>id</i> /slogd/slogdlog	维测日志。记录日志工具自身的运行信息，用于日志工具自身问题定位。

- syslog日志：“/var/log/npu/report/*/message”
syslog日志表示调用syslog接口记录Device侧其他组件产生的日志，“*”为时间戳。
- 黑匣子：“/var/log/npu/report/*/hisi_logs”
导出的黑匣子日志、黑匣子的设备事件信息和黑匣子存储空间中的历史维测信息均保存在该目录下，“*”为时间戳。
通过msnpureport工具导出黑匣子后，可以参考《[黑匣子日志参考_昇腾310 AI处理器](#)》或《[黑匣子日志参考_昇腾910 AI处理器](#)》对黑匣子进行后续的分析处理。
- Stackcore文件：“/var/log/npu/report/*/stackcore”

通过msnpureport工具导出Stackcore文件后，可以参考《[黑匣子日志参考_昇腾310 AI处理器](#)》或《[黑匣子日志参考_昇腾910 AI处理器](#)》中的“Stackcore文件解析定位”章节对Stackcore文件进行后续的分析处理，“*”为时间戳。

----结束

查询和设置 Device 侧 slog 系统类日志级别

步骤1 登录Host侧服务器。

步骤2 获取msnpureport工具。

msnpureport工具在驱动Driver的安装目录下，路径为“*Driver安装目录*/driver/tools/msnpureport”。

步骤3 在某个有读、写、执行权限的目录（如“/var/log/npu/report”）下，执行如下命令查询和设置日志级别。

Driver安装目录/driver/tools/msnpureport [options]

Driver安装目录/driver/tools/msnpureport [options] -d <device-id>

其中[options]为查询和设置日志级别的各个参数，[options]各个参数及其他参数的解释请参见[表11-2](#)。

表 11-2 参数说明

参数	说明	举例
-g <level>或 --global <level>	设置全局级的日志级别。 <ul style="list-style-type: none">debug：表示DEBUG级别。info：表示INFO级别。warning：表示WARNING级别。error：表示ERROR级别。null：表示NULL级别，不输出日志。	<i>Driver安装目录</i> / driver/tools/ msnpureport -g info
-m <module:level>或 --module <module:level>	设置模块级的日志级别。 <ul style="list-style-type: none">module：模块名称。例如SLOG等。level：模块级别。取值为：<ul style="list-style-type: none">debug：表示DEBUG级别。info：表示INFO级别。warning：表示WARNING级别。error：表示ERROR级别。null：表示NULL级别，不输出日志。	<i>Driver安装目录</i> / driver/tools/ msnpureport -m SLOG:error

参数	说明	举例
-e <level>或 --event <level>	设置是否开启Event日志。 <ul style="list-style-type: none">enable: 开启Event日志。disable: 不开启Event日志。	<i>Driver安装目录/</i> driver/tools/ msnpureport -e disable
-d <device-id>或 --device <device-id>	指定Device ID（逻辑ID），默认为0。通过指定Device ID设置对应Device的日志级别。 指定的Device ID是指逻辑上的Device ID而不是物理上的Device ID。须先判断逻辑ID后再设置对应的日志级别，详细步骤请参见 逻辑ID判断 。	<i>Driver安装目录/</i> driver/tools/ msnpureport -g warning -d 1
-r 或 --request	查询Device侧slog系统类日志的级别，包括全局级、模块级和是否开启Event日志。如果不指定Device ID，默认查询Device 0的日志级别。	<i>Driver安装目录/</i> driver/tools/ msnpureport -r 查询后显示的级别示例如下： The system log level of device_id:0 is as follows: [global] INFO [event] ENABLE [module] SLOG:INFO IDEDD:INFO DVPP:INFO CCE:INFO HDC:INFO DRV:INFO MDCDEFAULT:INFO DEVMM:INFO KERNEL:INFO LIBMEDIA:INFO ASCENDDK:INFO ROS:INFO HCCP:INFO ROCE:INFO PROFILING:INFO APP:INFO TDT:INFO MD:INFO MB:INFO ME:INFO BBOX:INFO TS:INFO TSDUMP:INFO LP:INFO
-h 或 --help	用于打印帮助信息。	<i>Driver安装目录/</i> driver/tools/ msnpureport -h

----结束

逻辑 ID 判断

步骤1 查询物理ID。

使用npu-smi info命令查看到的设备上的NPU Device ID即为物理ID。假设物理ID的取值范围是0~7，Device0~Device3四个为一组Device4~Device7四个为一组。由于同组的Device在同一个OS上，共用一个日志配置文件，所以日志级别也相同，只需要修改其中一个Device日志级别，那么同组其他的Device日志级别同样被修改。

物理ID的分组情况请以设备实际情况为准。

步骤2 判断逻辑ID。

查询到的物理ID会按照数字大小从小到大自上而下排列，那么对应的逻辑ID则从0开始按顺序为0~n。假设查询到的物理ID为“0, 1, 4, 5”，那么对应逻辑ID则为“0, 1, 2, 3”。根据上一步的分组情况可判断物理ID“0, 1”为一组，“4, 5”为一组，那么可以推断出逻辑ID“0, 1”为一组，“2, 3”为一组。

步骤3 设置日志级别。

根据前面判断的逻辑ID“0, 1”为一组，“2, 3”为一组。假设设置所有Device的日志级别为error，则需要配置两次：

Driver安装目录/driver/tools/msnpureport -g error -d 0

Driver安装目录/driver/tools/msnpureport -g error -d 2

----结束

11.2 连续导出 Device 侧的日志和文件

当执行模型推理时，如果Device侧出现异常，会导致无法连接Device侧，因此将不能通过msnpureport工具导出Device侧的日志和文件（msnpureport工具导出的Device侧日志和文件具体请参见[msnpureport工具使用](#)章节）。此时可以在执行模型推理之前，在Host侧运行msnpureport_auto_export.sh脚本连续导出Device侧的日志和文件，确保能够获取到Device异常前的所有日志文件，便于定位问题。

须知

- 该功能仅适用于Ascend EP场景。
- msnpureport_auto_export.sh脚本不支持容器场景，部署容器时，禁止将msnpureport_auto_export.sh脚本映射到容器内。
- 不支持多个用户同时运行msnpureport_auto_export.sh脚本。
- 在异常场景下，可能会出现连续导出Device侧日志和文件失败的情况。
- 如果用户想要终止导出Device侧日志和文件或者模型推理完成，需要使用“Ctrl +c”或kill -15 pid命令结束进程。

其中pid代表连续导出日志和文件的脚本进程ID，可以通过ps -elf | grep msnpureport_auto_export.sh命令进行查询。

操作步骤

步骤1 登录Host侧服务器。

步骤2 获取msnpureport_auto_export.sh脚本。

msnpureport_auto_export.sh脚本在驱动Driver的安装目录下，路径为“*Driver安装目录*/driver/tools/msnpureport_auto_export.sh”。

步骤3 在某个有执行权限的目录（如“/home/work”）下执行如下命令，运行脚本。

Driver安装目录/driver/tools/msnpureport_auto_export.sh <timeInterval>
<logAbsolutePathCapacity> <logAbsolutePath>

命令示例： `/usr/local/Ascend/driver/tools/msnpureport_auto_export.sh 2 10 /home/log/`

须知

在加锁的目录下（使用`lsattr`命令查看目录属性，有“i”选项的为加锁目录），用户没有权限运行`msnpureport_auto_export.sh`脚本。如果用户想在该目录下运行该脚本，可以通过`chattr -i <加锁的目录>`命令，将目录的“i”选项撤销。脚本运行完后建议通过`chattr +i <加锁的目录>`命令，将目录的“i”选项加上。为了安全起见，不建议在加锁目录中运行脚本。

其中各参数解释如表11-3所示：

表 11-3 参数说明

参数	说明
<code><timeInterval></code>	导出Device侧日志和文件的间隔时间。取值为大于0的整数，单位是s，如：2s。
<code><logAbsolutePathCapacity></code>	导出Device侧日志和文件的存储目录容量。取值为大于等于2的整数，单位是G，如：10G。
<code><logAbsolutePath></code>	导出Device侧日志和文件的存储路径（任意的绝对路径）。如：“/home/log/”。

脚本运行成功后，Device侧日志和文件将存储在运行脚本时指定的存储路径下（如“/home/log/”），如果不存在该目录，会自动进行创建；如果存在则直接存储。该目录下会自动创建如下子目录：

目录	说明
<code>msnpureport_log_new</code>	<p>导出Device侧日志和文件的存储目录。该目录下包含如下子目录：</p> <ul style="list-style-type: none">• <code>hisi_logs</code>• <code>message</code>• <code>slog</code>• <code>stackcore</code> <p>其中<code>hisi_logs</code>目录下的<code>history.log</code>文件和<code>message</code>目录下的<code>message.log</code>文件是在Device侧的同名文件中进行老化的，所以每次导出之后需要将导出内容追加到同一目录下的<code>history_new.log</code>和<code>message_new.log</code>文件中并进行去重，来获取所有日志文件。</p> <p>其他目录的日志文件在Device侧是以时间戳命名的，是通过删除较早时间戳的日志文件进行老化的，所以每次导出之后只需将导出内容拷贝覆盖就可以获取所有日志文件。</p>

目录	说明
msnpureport_log_old	老化的日志和文件的存储目录。 如果msnpureport_log_new目录存储的日志容量超过运行脚本时指定的存储目录容量的一半（如10/2=5G），将自动清空msnpureport_log_old目录下的日志和文件，再将msnpureport_log_new目录下存储的日志和文件全部移动到msnpureport_log_old目录下。

----结束

11.3 昇腾产品形态说明

以昇腾 AI 处理器的PCIe的工作模式进行区分，如果PCIe工作在主模式，可以扩展外设，则称为RC模式；如果PCIe工作在从模式，则称为EP模式。

- 昇腾 AI 处理器的工作模式如下：
 - 昇腾310 AI处理器有EP和RC两种模式。
 - 昇腾710 AI处理器只有EP模式。
- 支持RC模式的产品有：Atlas 200 AI加速模块、Atlas 200 DK 开发者套件。
产品的CPU直接运行用户指定的AI业务软件，接入网络摄像头、I²C传感器、SPI显示器等其他外挂设备作为从设备接入产品。
- 支持EP模式的产品
昇腾310 AI处理器：Atlas 200 AI加速模块、Atlas 300I 推理卡、Atlas 500 智能小站、Atlas 500 Pro 智能边缘服务器、Atlas 800 推理服务器。
昇腾710 AI处理器：Atlas 300I Pro 推理卡、Atlas 300V Pro 视频解析卡。
EP模式通常由Host侧作为主端，Device侧作为从端。客户的AI业务程序运行在Host系统中，产品作为Device系统以PCIe从设备接入Host系统，Host系统通过PCIe通道与Device系统交互，将AI任务加载到Device侧的昇腾 AI 处理器中运行。

两种模式的产品及架构如[图11-1](#)所示。

Host和Device的概念说明如下：

- Host：是指与昇腾AI处理器所在硬件设备相连接的X86服务器、ARM服务器，利用昇腾AI处理器提供的NN（Neural-Network）计算能力完成业务。
- Device：是指安装了昇腾AI处理器的硬件设备，利用PCIe接口与服务器连接，为服务器提供NN计算能力。

图 11-1 RC 和 EP 模式

