StarOS设施崩溃故障排除

目录

[简介](#)

[概述](#)

[崩溃场景](#)

[崩溃背后的原因](#)

[不同类型的崩溃](#)

[初始日志要求](#)

[分析步骤](#)

[会话恢复](#)

简介

本文档介绍如何查找并排除StarOs设备崩溃故障。

概述

有时，系统逻辑可能失败，导致软件任务重新启动以恢复正常功能。这可能导致进程崩溃。StarOS中经常报告任务设施崩溃，并且可以根据崩溃的根本原因采取必要的操作。要识别节点上的崩溃，可以使用以下CLI命令：

```
 ******** show crash list *******
Saturday April 15 05:05:56 SAST 2023
=== ==================== ======== ========== =============== ======================
#           Time          Process  Card/CPU/        SW          HW_SER_NUM
                                      PID         VERSION       CF / Crash Card
=== ==================== ======== ========== =============== ======================

1   2022-Dec-02+14:08:46 confdmgr 02/0/19342 21.26.13        NA
2   2022-Dec-02+14:48:08 confdmgr 02/0/31546 21.26.13        NA
3   2022-Dec-04+19:10:50 sessmgr  03/0/12321 21.26.13        NA
4   2022-Dec-21+03:34:13 sessmgr  04/0/12586 21.26.13        NA
```

类似的崩溃会合并到一个记录中。记录会显示发生此崩溃类型的次数。

```
********************* CRASH #02 ************************
SW Version          : 21.26.13
Similar Crash Count : 33 >>>>
Time of First Crash : 2022-Dec-02+14:10:05

Assertion failure at confdmgr/src/confdmgr_fsm.c:870
  Note: State machine failure, state = 3
```

```
  Function: confdmgr_fsm_state_wait_p0_handler()
  Expression: 0
  Code: CRASH
  Proclet: confdmgr (f=1900,i=0)
  Process: card=2 cpu=0 arch=X pid=31546 argv0=confdmgr
```

在 show snmp trap history verbose输出显示某个进程已崩溃：

```
Fri Dec 26 08:32:20 2014 Internal trap notification 73 (ManagerFailure) facility sessmgr
instance 188 card 7 cpu 0
Fri Dec 26 08:32:20 2014 Internal trap notification 150 (TaskFailed) facility sessmgr instance
188 on card 7 cpu 0
Fri Dec 26 08:32:23 2014 Internal trap notification 1099 (ManagerRestart) facility sessmgr
instance 139 card 4 cpu 1
Fri Dec 26 08:32:23 2014 Internal trap notification 151 (TaskRestart) facility sessmgr
instance 139 on card 4 cpu 1
```

崩溃场景

崩溃的原因可能有多种：

1.不同的呼叫流程场景

2.内存问题

3.配置问题

4.硬件故障

崩溃背后的原因

在StarOS中有多个任务工具，这些任务工具具有各自的功能，因此，每当工具遇到任何此类输入并进入有问题的状态时，都会使工具崩溃以从该错误状态恢复。

不同类型的崩溃

1.断言失败：

```
********************* CRASH #22 ***********************
SW Version          : 21.26.13
Similar Crash Count : 33
Time of First Crash : 2023-Apr-12+22:40:01

Assertion failure at sess/smgr/sessmgr_snx.c:9568 >>>>
  Function: sessmgr_snx_send_drop_call()
  Expression: result == SN_STATUS_SUCCESS
```

```
  Proclet: sessmgr (f=87000,i=261)
  Process: card=5 cpu=0 arch=X pid=12724 cpu=~0% argv0=sessmgr
```

2.分段故障：

```
******************** CRASH #69 ***********************
SW Version : 21.13.3
Similar Crash Count : 2
Time of First Crash : 2019-Nov-25+07:53:54
Fatal Signal 11: Segmentation fault >>>>
Faulty address: 0x7ff6b4801036
Signal from: kernel
Signal detail: address not mapped to object
Process: card=8 cpu=1 arch=X pid=7316 argv0=vpp
Crash time: 2020-Feb-11+04:04:23 UTC
Build_number:
```

3.致命信号：

```
******************** CRASH #01 ***********************
SW Version          : 21.23.12
Similar Crash Count : 2
Time of First Crash : 2023-Jan-27+05:22:46

Fatal Signal 11: 11  >>>>>
  PC: [04be6859/X] sessmgr_pgw_create_bearers()
  Faulty address: 0x297116e4
  Signal from: kernel
  Signal detail: address not mapped to object
  Process: card=9 cpu=1 arch=X pid=10383 cpu=~8% argv0=sessmgr
```

初始日志要求

故障日志是重要的故障事件信息来源。发生软件崩溃时，StarOS会捕获并存储有助于确定崩溃原因的相关数据。此信息可以存储在系统内存中，也可以传输并保存在网络服务器上。

Core File或Mini Core File：请注意，核心文件与发生崩溃的PID相对应。核心文件以“crash-<card no>-<cpu>-<pid>-<unixtime>-core”格式命名。您可以在“show crash list”命令输出中找到此信息。

Minicore文件：此文件包含有关失败任务的信息，包括当前堆栈跟踪、过去的分析器示例、过去内存活动示例以及其他专有文件格式的捆绑数据。

Core Dump（或Full Core）：核心转储在崩溃发生后立即提供进程的完整内存转储。此内存转储通常对于确定软件崩溃的根本原因至关重要。

崩溃签名：可以从共享的显示支持详细信息(SSD)或其他相关来源查看崩溃签名。

```
 ******** show crash list *******
Saturday April 15 05:05:56 SAST 2023
=== ==================== ======== ========== ============== ======================
#            Time         Process  Card/CPU/        SW          HW_SER_NUM
                                       PID         VERSION       CF / Crash Card
=== ==================== ======== ========== ============== ======================

1   2022-Dec-02+14:08:46 confdmgr 02/0/19342 21.26.13        NA
2   2022-Dec-02+14:48:08 confdmgr 02/0/31546 21.26.13        NA
3   2022-Dec-04+19:10:50 sessmgr  03/0/12321 21.26.13        NA
```

现在，如果您想了解崩溃1的签名，请在SSD中搜索崩溃#01，或在CLI中使用show crash number 1。

```
From SSD

********************* CRASH #01 ***********************
SW Version          : 21.26.13
Similar Crash Count : 1
Time of First Crash : 2022-Dec-02+14:08:46

Assertion failure at confdmgr/src/confdmgr_fsm.c:758
  Note: State machine failure, state = 5
  Function: confdmgr_fsm_state_wait_p1_handler()
  Expression: 0
  Code: CRASH

Using CLI

[local]abc# show  crash number 1
Friday June 09 06:41:53 CDT 2023
********************* CRASH #01 ***********************
SW Version          : 21.12.20.77760
Similar Crash Count : 1
Time of First Crash : 2021-Mar-31+15:58:06

Fatal Signal 6: Aborted
  PC: [ffffe430/X] __kernel_vsyscall()
  Note: User-initiated state dump w/core.
  Signal from: sitmain pid=6999 uid=0
  Process: card=9 cpu=0 arch=X pid=9495 cpu=~0% ar
```

在出现问题的特定时间戳期间检查显示支持详细信息(SSD)和系统日志。

分析步骤

1.需要检查故障堆栈/签名，并检查该特定故障签名是否有错误。

2.需要解析核心文件/微核心，以分析回溯并获取设备崩溃功能的线索。

3.完成核心文件调试后，您需要用软件缺陷验证症状，以及是否存在任何现有的软件缺陷，用于类

似的故障签名和回溯。

会话恢复

StarOs软件旨在处理可预见的条件/事件和不可预见的条件/事件。尽管思科努力开发完美的软件，但不可避免地存在错误，并且可能会发生故障。这正是会话恢复功能如此重要的原因。

当系统内的硬件或软件发生故障时，会话恢复功能提供无缝故障切换和用户会话信息的重建，以防止完全连接的用户会话被断开。通过在系统中镜像关键软件进程（例如，会话管理器和AAA管理器）执行会话恢复。这些镜像进程保持空闲状态（备用模式），在此状态中它们不执行任何处理，直到在软件发生故障（例如，会话管理器任务中止）时可能需要它们为止。

解复用器任务、AAA管理器和VPN管理器等任务在我们的系统中内置自动恢复机制，专门用于处理用户信息。会话恢复主要指的是sessmgr任务出现故障或卡级故障，并且需要在没有任何呼叫丢失的情况下恢复会话的情况。

- 在系统中，备用会话管理器在每个处理卡上处于活动状态，在发生故障时，可以快速接管主会话管理器。然后在处理卡上创建新的备用sessmgr。
- 当sessmgr进程意外失败时，备用sessmgr从aaamgr（AAA管理器）检索备份信息并重建其会话。
- 如果aaamgr发生故障，备用aamgr将查询sessmgr以同步相关用户信息。
- 如果demux-mgr发生故障，它将查询所有会话并重建其呼叫分布信息数据库。

- 为确保卡级冗余，一个处理卡用作备用卡，可在硬件或软件出现故障时随时进行接管。然后，备用卡从运行在其他卡上的对等aamgr恢复会话。

```
 ******** show session recovery status verbose *******
Saturday April 15 05:11:17 SAST 2023
Session Recovery Status:
  Overall Status        : Ready For Recovery  >>>>
  Last Status Update    : 5 seconds ago

              ----sessmgr---  ----aaamgr----  demux
 cpu state    active standby  active standby  active  status
---- -------  ------ -------  ------ -------  ------  -------------------------
 3/0 Active   40     1        40     1        0       Good
 4/0 Active   40     1        40     1        0       Good
 5/0 Active   40     1        40     1        0       Good
 6/0 Active   40     1        40     1        0       Good
 7/0 Active   0      0        0      0        10      Good (Demux)
 8/0 Active   40     1        40     1        0       Good
 9/0 Active   40     1        40     1        0       Good
10/0 Active   40     1        40     1        0       Good
11/0 Standby  0      40       0      40       0       Good
```

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。