ASR5x00 セッション マネージャ タスク - 機能、クラッシュ、リカバリ操作、クラッシュ ログについての説明

目次

概要

このドキュメントでは、Cisco アグリゲーション サービス ルータ（ASR）5x00 シリーズのソフトウェア信頼性、サービス可用性、およびフェールオーバー機能について説明します。 ASR5x00 のソフトウェア クラッシュの定義と、ソフトウェア クラッシュの影響を示します。 この記事では、予期しないソフトウェア クラッシュが発生した場合でも、ASR5x00 が持つソフトウェア復元力と可用性機能によって「キャリアクラスの」可用性という目標がどのように達成されるかを見ていきます。 モバイル サブスクライバにサービスの可用性を決して意識させるべきではありません。 シスコの目標は、システム全体の停止を含むどのようなハードウェア/ソフトウェア障害が発生した場合でも、セッションが失われないようにすること、言い換えれば音声帯域の信頼性を保つことです。 ASR5x00 のソフトウェア信頼性機能の目標は、予期しない障害がオペレータのネットワークで発生した場合でも「キャリアクラスの」サービス可用性を提供できるようにすることです。

ソフトウェア アーキテクチャ： 復元力のための設計

ASR5x00 には、特定のさまざまな機能を実行するために設計された、パケット サービス カード（PSC）やデータ処理カード（DPC）、およびシステム管理カード（SMC）や管理および I/O（MIO）カード）全般に配布されるソフトウェア タスクの集合があります。

たとえば、セッション マネージャ タスクはサブスクライバ セットに関するセッション処理を担当し、ユーザ トラフィックに対するピアツーピア（P2P）、ディープ パケット インスペクション（DPI）などのインライン サービスを実行します。 認証、認可、およびアカウンティング（AAA）マネージャ タスクは、加入者トラフィック使用状況などを記録するための課金情報イベントの生成を実行します。 セッション マネージャ タスクと AAA マネージャ タスクは PSC/DPC

カードで実行されます。

SMC/MIO カードは、オペレーション/メンテナンス (O&M) およびプラットフォーム関連タスクのために予約されています。 ASR5x00 システムは、サブスクライバ セッションを処理するセッション サブシステムや、IP アドレスの割り当て/ルーティングを担当する VPN サブシステムなど、複数のソフトウェア サブシステムに仮想的に分割されます。 各サブシステムには、制御対象のサブシステムの健全性を監視するコントローラ タスクがあります。 コントローラ タスクは SMC/MIO カードで実行されます。 セッション マネージャ タスクと AAA マネージャ タスクが組み合わされて、制御、データ トラフィック、課金目的でサブスクライバ セッションを処理します。 システムでセッション リカバリが有効になっている場合、各セッション マネージャ タスクは、セッション マネージャのクラッシュ時にリカバリされるピア AAA マネージャ タスクを使用して、一連のサブスクライバの状態をバックアップします。

クラッシュとは何か

ASR5x00 におけるタスクは、通常動作中に障害状態に遭遇するとクラッシュする可能性があります。 ASR5x00 のクラッシュまたはソフトウェア障害は、システム タスクの *予期しない* 終了または異常終了と定義されます。 クラッシュが発生するのは、禁止されているメモリ領域 (破損データ構造など) へのアクセスをソフトウェア コードが試行した場合や、コード内で予期しない状態 (無効な状態遷移など) に遭遇した場合などです。 また、タスクがシステム モニタ タスクに応答しなくなり、モニタがタスクを停止して再起動しようとした場合にも、クラッシュが発生する可能性があります。 さらに、 (予期しないクラッシュとは反対に) システムで明示的にクラッシュ イベントを発生させることもできます。これは、タスクの状態を分析する目的で CLI コマンドまたはシステム モニタによりタスクの現在の状態を強制的にダンプさせる場合です。 また、マネージャ タスクに障害が繰り返し発生する状況を修正しようとして、システム コントローラ タスクが自身を再起動する際にも、予期されるクラッシュ イベントが発生することがあります。

セッション マネージャのクラッシュの影響

通常の動作では、セッション マネージャ タスクはサブスクライバ セッションの課金情報を処理するピアリング AAA マネージャ タスクと一緒に、一連のサブスクライバ セッションおよびセッションに関連するデータ トラフィックを処理します。 セッション マネージャのクラッシュが発生すると、システム内にそれが存在しなくなります。 システムでセッション リカバリが有効になっている場合は、スタンバイ セッション マネージャ タスクが同じ PSC/DPC カード内でアクティブになります。 この新しいセッション マネージャ タスクは、ピア AAA マネージャ タスクと通信するときにサブスクライバ セッションを回復させます。 クラッシュ時にセッション マネージャでアクティブであったセッション数やカード全体の CPU 負荷などに応じて、リカバリ操作は 50 ミリ秒から数秒に及びます。 この操作では、元のセッション マネージャですでに確立されていたサブスクライバ セッションは失われません。 クラッシュ時に確立過程にあったサブスクライバ セッションもまた、プロトコル再送信などによって復元される可能性が高くなります。 クラッシュ時にシステム内で遷移中だったデータ パケットは、ネットワーク接続の通信側エンティティによってネットワーク喪失に関連付けられていると想定され、再伝送されて、新しいセッション マネージャによって接続が継続されます。 セッション マネージャによって伝送されるセッションの課金情報は、ピア AAA マネージャで保存されます。

どんなときにオペレータは注意すべきか

セッション マネージャのクラッシュが発生すると、前述のようにリカバリ手順が開始され、残りのシステムはこのイベントによる影響を受けません。 あるセッション マネージャのクラッシュは他のセッション マネージャに影響しません。 オペレータへのガイダンスとして、*同じ PSC/DPC カード上*の複数のセッション マネージャ タスクが同時に、あるいは 10 分以内の間隔でクラッシュした場合、クラッシュしたタスクに代わる新しいセッション マネージャをシステムがすばやく開始できない可能性があるため、セッションの喪失が発生することがあります。 これは二重障害のシナリオに該当し、セッション喪失が生じる可能性があります。 リカバリが可能ではない場合、セッション マネージャが単純に再起動され、新しいセッションを受け入れるようになります。

（たとえば同じ障害状態が何度も発生して）特定のセッション マネージャが繰り返しクラッシュする場合、セッション コントローラ タスクはそれを把握し、サブシステムを復元しようとして自身を再起動します。 セッション コントローラ タスクがセッション サブシステムを安定化できず、この試行で自身を継続的に再起動する場合には、次の エスカレーション段階として、システムがスタンバイ SMC/MIO カードに切り替わります。 まれにスタンバイ SMC/MIO カードがない場合や、スイッチオーバー操作で障害が発生した場合には、システムが自身を再起動します。

また、セッション マネージャは、クラッシュの発生により永久に失われる各アクセスポイント名（APN）、サービス、機能などの統計情報も保持します。 したがって、一括統計情報を定期的に収集する外部エンティティでは、1 つ以上のクラッシュが発生したときに統計上の低下が観察されます。 これは、時系列で描画される統計情報のグラフ表示に凹みとして表現されることがあります。

> **注:** 7-14 PSC または 4-10 DPC カードが実装された標準的なシャーシには、PSC/DPC カードの数に応じて約 120 ~ 160 個のセッション マネージャがあり、単一のクラッシュによって統計情報のおよそ $^{1/40}$ または $^{1/80}$ が喪失します。 スタンバイ セッション マネージャに切り替わると、再びゼロから統計情報の蓄積が開始されます。

クラッシュが発生したかどうかの判別方法

クラッシュにより、syslog イベントやイベント モニタリング サービス（EMS）などのネットワーク モニタリング ステーションへの SNMP トラップ イベントがトリガーされます。 また、システムで発生したクラッシュを **show crash list コマンド**で確認することもできます。 このコマンドは、前述したように、予期しないクラッシュ イベントと予期されたクラッシュ イベントの両方をリストすることに注意してください。 これら 2 種類のクラッシュ イベントを区別するには、各クラッシュを記述するヘッダーを確認できます。

タスク クラッシュの後に正常なセッション回復が行われた場合は、次のログ メッセージで示されます。

```
"Death notification of task <name>/<instance id> on <card#>/<cpu#> sent to parent
task <parent name>/<instance id> with failover of <task name>/<instance id>
on <card#>/<cpu#>"
```

回復できなかったタスク クラッシュは次のログ メッセージで示されます。

```
"Death notification of task <name>/<instance id> on <card#>/<cpu#> sent to parent
task <parent name>/<instance id>"
```

要約すると、セッション回復が有効な場合、ほとんどのクラッシュはサブスクライバに影響しないため、それに気づくことはありません。 クラッシュの発生を検出するには、CLI コマンドを入力するか、またはログや SNMP 通知を調べる必要があります。

次に、例を示します。

```
 ******** show crash list *******
Tuesday May 26 05:54:14 BDT 2015
=== ==================== ======== ========== =============== ======================
# Time Process Card/CPU/ SW HW_SER_NUM
PID VERSION MIO / Crash Card
=== ==================== ======== ========== =============== ======================

1 2015-May-07+11:49:25 sessmgr 04/0/09564 17.2.1 SAD171600WS/SAD172200MH
2 2015-May-13+17:40:16 sessmgr 09/1/05832 17.2.1 SAD171600WS/SAD173300G1
3 2015-May-23+09:06:48 sessmgr 03/1/31883 17.2.1 SAD171600WS/SAD1709009P
4 2015-May-25+15:58:59 sessmgr 09/1/16963 17.2.1 SAD171600WS/SAD173300G1
5 2015-May-26+01:15:15 sessmgr 04/0/09296 17.2.1 SAD171600WS/SAD172200MH

 ******** show snmp trap history verbose *******
Fri May 22 19:43:10 2015 Internal trap notification 1099 (ManagerRestart) facility
sessmgr instance 204 card 9 cpu 1
Fri May 22 19:43:29 2015 Internal trap notification 73 (ManagerFailure) facility
sessmgr instance 204 card 9 cpu 1
Fri May 22 19:43:29 2015 Internal trap notification 150 (TaskFailed) facility
sessmgr instance 204 on card 9 cpu 1
Fri May 22 19:43:29 2015 Internal trap notification 151 (TaskRestart) facility
sessmgr instance 204 on card 9 cpu 1
Fri May 22 19:43:30 2015 Internal trap notification 183 (SessMgrRecoveryComplete)
Slot Number 9 Cpu Number 1 fetched from aaa mgr 1755 prior to audit 1755 passed
audit 1754 calls recovered 1754 all call lines 1754 time elapsed ms 1108.
Fri May 22 19:43:32 2015 Internal trap notification 1099 (ManagerRestart) facility
sessmgr instance 204 card 9 cpu 1
Fri May 22 19:44:49 2015 Internal trap notification 73 (ManagerFailure) facility
sessmgr instance 236 card 7 cpu 0
Fri May 22 19:44:49 2015 Internal trap notification 150 (TaskFailed) facility
sessmgr instance 236 on card 7 cpu 0
Fri May 22 19:44:49 2015 Internal trap notification 151 (TaskRestart) facility
sessmgr instance 236 on card 7 cpu 0
Fri May 22 19:44:51 2015 Internal trap notification 183 (SessMgrRecoveryComplete)
Slot Number 7 Cpu Number 0 fetched from aaa mgr 1741 prior to audit 1741 passed audit
1737 calls recovered 1737 all call lines 1737 time elapsed ms 1047.
Fri May 22 19:44:53 2015 Internal trap notification 1099 (ManagerRestart) facility
sessmgr instance 236 card 7 cpu 0
Fri May 22 19:50:04 2015 Internal trap notification 73 (ManagerFailure) facility
sessmgr instance 221 card 2 cpu 1
: Fri May 22 19:50:04 2015 Internal trap notification 150 (TaskFailed) facility
sessmgr instance 221 on card 2 cpu 1
Fri May 22 19:50:04 2015 Internal trap notification 151 (TaskRestart) facility
sessmgr instance 221 on card 2 cpu 1
Fri May 22 19:50:05 2015 Internal trap notification 183 (SessMgrRecoveryComplete)
Slot Number 2 Cpu Number 1 fetched from aaa mgr 1755 prior to audit 1755 passed
audit 1749 calls recovered 1750 all call lines 1750 time elapsed ms 1036.

 ******** show snmp trap history verbose *******
Fri May 22 19:43:10 2015 Internal trap notification 1099 (ManagerRestart) facility
sessmgr instance 204 card 9 cpu 1
Fri May 22 19:43:29 2015 Internal trap notification 73 (ManagerFailure) facility
sessmgr instance 204 card 9 cpu 1
Fri May 22 19:43:29 2015 Internal trap notification 150 (TaskFailed) facility
sessmgr instance 204 on card 9 cpu 1
Fri May 22 19:43:29 2015 Internal trap notification 151 (TaskRestart) facility
sessmgr instance 204 on card 9 cpu 1
Fri May 22 19:43:30 2015 Internal trap notification 183 (SessMgrRecoveryComplete)
Slot Number 9 Cpu Number 1 fetched from aaa mgr 1755 prior to audit 1755 passed
audit 1754 calls recovered 1754 all call lines 1754 time elapsed ms 1108.
```

```
Fri May 22 19:43:32 2015 Internal trap notification 1099 (ManagerRestart) facility
sessmgr instance 204 card 9 cpu 1
Fri May 22 19:44:49 2015 Internal trap notification 73 (ManagerFailure) facility
sessmgr instance 236 card 7 cpu 0
Fri May 22 19:44:49 2015 Internal trap notification 150 (TaskFailed) facility
sessmgr instance 236 on card 7 cpu 0
Fri May 22 19:44:49 2015 Internal trap notification 151 (TaskRestart) facility
sessmgr instance 236 on card 7 cpu 0
Fri May 22 19:44:51 2015 Internal trap notification 183 (SessMgrRecoveryComplete)
Slot Number 7 Cpu Number 0 fetched from aaa mgr 1741 prior to audit 1741 passed
audit 1737 calls recovered 1737 all call lines 1737 time elapsed ms 1047.
Fri May 22 19:44:53 2015 Internal trap notification 1099 (ManagerRestart) facility
sessmgr instance 236 card 7 cpu 0
Fri May 22 19:50:04 2015 Internal trap notification 73 (ManagerFailure) facility
sessmgr instance 221 card 2 cpu 1
: Fri May 22 19:50:04 2015 Internal trap notification 150 (TaskFailed) facility
sessmgr instance 221 on card 2 cpu 1
Fri May 22 19:50:04 2015 Internal trap notification 151 (TaskRestart) facility
sessmgr instance 221 on card 2 cpu 1
Fri May 22 19:50:05 2015 Internal trap notification 183 (SessMgrRecoveryComplete
) Slot Number 2 Cpu Number 1 fetched from aaa mgr 1755 prior to audit 1755 passed
audit 1749 calls recovered 1750 all call lines 1750 time elapsed ms 1036.

 ******** show logs  *******
2015-May-25+23:15:53.123 [sitmain 4022 info] [3/1/4850 <sitmain:31> sittask.c:4762]
[software internal system critical-info syslog] Readdress requested for facility
sessmgr instance 5635 to instance 114
2015-May-25+23:15:53.122 [sitmain 4027 critical] [3/1/4850 <sitmain:31>
crash_mini.c:908] [software internal system callhome-crash] Process Crash Info:
time 2015-May-25+17:15:52(hex time 556358c8) card 03 cpu 01 pid 27118 procname
sessmgr crash_details
Assertion failure at acs/acsmgr/analyzer/ip/acs_ip_reasm.c:2970
Function: acsmgr_deallocate_ipv4_frag_chain_entry()
Expression: status == SN_STATUS_SUCCESS
Proclet: sessmgr (f=87000,i=114)
Process: card=3 cpu=1 arch=X pid=27118 cpu=~17% argv0=sessmgr
Crash time: 2015-May-25+17:15:52 UTC
Recent errno: 11 Resource temporarily unavailable
Stack (11032@0xffffb000):
[ffffe430/X] __kernel_vsyscall() sp=0xffffbd28
[0af1de1f/X] sn_assert() sp=0xffffbd68
[0891e137/X] acsmgr_deallocate_ipv4_frag_chain_entry() sp=0xffffbde8
[08952314/X] acsmgr_ip_frag_chain_destroy() sp=0xffffbee8
[089d87d1/X] acsmgr_process_tcp_packet() sp=0xffffc568
[089da270/X] acs_process_tcp_packet_normal_path() sp=0xffffc5b8
[089da3fd/X] acs_tcp_analyzer() sp=0xffffc638
[0892fb39/X] do_acsmgr_process_packet() sp=0xffffc668
[08940045/X] acs_ip_lean_path() sp=0xffffc6b8
[0887e309/X] acsmgr_data_receive_merge_mode() sp=0xffffc9d8
[0887f323/X] acs_handle_datapath_events_from_sm_interface() sp=0xffffca08
[037c2e1b/X] sessmgr_sef_initiate_data_packet_ind() sp=0xffffca88
[037c2f50/X] sessmgr_pcc_intf_send_data_packet_ind() sp=0xffffcaf8
[061de74a/X] sessmgr_pcc_fwd_packet() sp=0xffffcb58
[0627c6a4/X] sessmgr_ipv4_process_inet_pkt_part2_slow() sp=0xffffcf68
[06318343/X] sessmgr_ipv4_process_inet_pkt_pgw_ggsn() sp=0xffffd378
[0632196c/X] sessmgr_med_ipv4_data_received() sp=0xffffd418
[0633da9a/X] sessmgr_med_data_receive() sp=0xffffd598
[0afb977c/X] sn_epoll_run_events() sp=0xffffd5e8
[0afbdeb8/X] sn_loop_run() sp=0xffffda98
[0ad2b82d/X] main() sp=0xffffdb08

2015-May-25+23:15:53.067 [rct 13038 info] [5/0/7174 <rct:0> rct_task.c:305]
[software internal system critical-info syslog] Death notification of task
```

```
sessmgr/114 on 3/1 sent to parent task sessctrl/0 with failover of sessmgr/5635 on 3/1
2015-May-25+23:15:53.065 [evlog 2136 info] [5/0/7170 <evlogd:0> odule_persist.c:3102]
[software internal system critical-info syslog] Evlogd crashlog: Request received to
check the state of persistent crashlog.
2015-May-25+23:15:53.064 [sitmain 4099 info] [3/1/4850 <sitmain:31> crash_mini.c:765]
[software internal system critical-info syslog] have mini core, get evlogd status for
logging crash file 'crashdump-27118'
2015-May-25+23:15:53.064 [sitmain 4017 critical] [3/1/4850 <sitmain:31> sitproc.c:1544]
[software internal system syslog] Process sessmgr pid 27118 died on card 3 cpu 1
signal=6 wstatus=0x86
2015-May-25+23:15:53.048 [sitmain 4074 trace] [5/0/7168 <sitparent:50> crashd.c:1130]
[software internal system critical-info syslog] Crash handler file transfer starting
(type=2 size=0 child_ct=1 core_ct=1 pid=23021)
2015-May-25+23:15:53.047 [system 1001 error] [6/0/9727 <evlogd:1> evlgd_syslogd.c:221]
[software internal system syslog] CPU[3/1]: xmitcore[21648]: Core file transmitted to
card 5 size=663207936 elapsed=0sec:908ms
2015-May-25+23:15:53.047 [system 1001 error] [5/0/7170 <evlogd:0> evlgd_syslogd.c:221]
[software internal system syslog] CPU[3/1]: xmitcore[21648]: Core file transmitted to
card 5 size=663207936 elapsed=0sec:908ms
2015-May-25+23:15:53.047 [sitmain 4080 info] [5/0/7168 <sitparent:50> crashd.c:1091]
[software internal system critical-info syslog] Core file transfer to SPC complete,
received 8363207936/0 bytes

 ******** show session recovery status verbose *******
Tuesday May 26 05:55:26 BDT 2015
Session Recovery Status:
Overall Status : Ready For Recovery
Last Status Update : 8 seconds ago

----sessmgr--- ----aaamgr---- demux
cpu state active standby active standby active status
---- ------- ------ ------- ------ ------- ------ -------------------------
1/0 Active 24 1 24 1 0 Good
1/1 Active 24 1 24 1 0 Good
2/0 Active 24 1 24 1 0 Good
2/1 Active 24 1 24 1 0 Good
3/0 Active 24 1 24 1 0 Good
3/1 Active 24 1 24 1 0 Good
4/0 Active 24 1 24 1 0 Good
4/1 Active 24 1 24 1 0 Good
5/0 Active 0 0 0 0 14 Good (Demux)
7/0 Active 24 1 24 1 0 Good
7/1 Active 24 1 24 1 0 Good
8/0 Active 24 1 24 1 0 Good
8/1 Active 24 1 24 1 0 Good
9/0 Active 24 1 24 1 0 Good
9/1 Active 24 1 24 1 0 Good
10/0 Standby 0 24 0 24 0 Good
10/1 Standby 0 24 0 24 0 Good
```

クラッシュ ロギングのアーキテクチャ

クラッシュ ログは、ソフトウェア クラッシュに関連する、可能性のあるすべての情報を記録します（完全なコア ダンプ）。 これはサイズが大きいため、システム メモリには保存できません。したがって、これらのログが生成されるのは、ログを保存できるローカル デバイスまたはネットワーク サーバを指す URL を使ってシステムが設定されている場合だけです。

クラッシュ ログはクラッシュ イベント情報の永続的なリポジトリです。 各イベントには番号が付けられ、CPU（minicore）、ネットワーク プロセッサ ユニット（NPU）、またはカーネル クラッシュに関するテキストが含まれます。 記録されたイベントは固定長レコードに記録され、

/flash/crashlog2 に保存されます。

クラッシュが発生するたびに、次のクラッシュ情報が保存されます。

1. イベント レコードが /flash/crashlog2 ファイル (クラッシュ ログ) に保存されます。
2. 関連する minicore、NPU、カーネルのダンプ ファイルが /flash/crsh2 ディレクトリに保存されます。
3. 完全なコアダンプが、ユーザ設定のディレクトリに保存されます。

管理カード間のクラッシュ イベントと Minicore の同期

クラッシュ ログはそれぞれの管理カードに固有であるため、カード「8」がアクティブであるときにクラッシュが発生すると、それがカードは「8」に記録されます。 後続のスイッチオーバーでは、ログにクラッシュが表示されなくなります。 このクラッシュ情報を取り出すには、元のカード「8」へのスイッチ バックを実行する必要があります。 クラッシュ イベント ログとダンプは、アクティブな管理カードとスタンバイ管理カードに固有であるため、アクティブ カードでクラッシュが発生した場合、クラッシュ イベント ログおよび関連するダンプがアクティブ カードだけに保存されます。 このクラッシュ情報はスタンバイ カードでは入手できません。 アクティブ カードのクラッシュによりカードがスイッチオーバーし、引き継いだカードにクラッシュ情報が表示されなった場合、現在アクティブなカードからのみクラッシュ情報を取得できます。 他方のカードのクラッシュ リストを取得するには、再びスイッチオーバーする必要があります。 このスイッチオーバーを回避してスタンバイ カードからクラッシュ情報を取得するためには、2 つの管理カード間で同期し、最新のクラッシュ情報を維持する必要があります。

到着したクラッシュ イベントはスタンバイ SMC/MIO に送信され、同様の方法でスタンバイのクラッシュログ ファイルに保存されます。 アクティブ SMC/MIO のフラッシュ上の Minicore、NPU、またはカーネル ダンプを、**rsync** コマンドを使ってスタンバイ SMC/MMIO に同期する必要があります。 クラッシュログのエントリまたはリスト全体を CLI コマンドで削除する場合は、アクティブ/スタンバイ両方の SMC/MIO でそれを消去する必要があります。 メモリには影響がありません。 クラッシュ関連のすべての同期アクティビティはスタンバイ SMC/MIO カードの evlogd により実行されます。これは、スタンバイ evlogd の負荷の方が小さく、同期アクティビティ用の十分な容量がスタンバイ カードにあるためです。 そのため、システムのパフォーマンスは影響を受けません。

コマンド

問題をトラブルシューティングするには次のコマンドを使用できます。

```
#show support details

#show crash list

#show logs

#show snmp trap history verbose

#show session recovery status verbose

#show task resources facility sessmgr instance <>

#show task resources facility sessmgr all
```

コアファイルがクラッシュ後に生成されます。 通常、オペレータはこれらを外部サーバに格納します。 コアファイル名は通常、crash-<Cardnum>-<CPU Num>-<Hex timestamp>-coree.gcrash-09-00-5593a1b8-core のようになります。

クラッシュが発生するたびに、次のクラッシュ情報が保存されます。

- イベント レコードが /flash/crashlog2 ファイル (クラッシュ ログ) に保存されます。
- 関連する minicore、NPU、カーネルのダンプ ファイルが /flash/crsh2 ディレクトリに保存されます。

要約

すべての ASR5x00 ソフトウェアは、予期される状況/イベントと予期されない状況/イベントの両方を処理するように設計されています。 シスコは完全なソフトウェアを提供するよう努めていますが、エラーを避けることはできず、クラッシュの可能性は常にあります。 このために、セッション リカバリ機能は非常に重要です。 完璧を目指すシスコの努力でクラッシュの発生は最小限に抑えられ、セッション リカバリによりクラッシュ後もセッションを続行できます。 それでもなお、シスコにとって、完全なソフトウェアを実現する努力は重要です。 クラッシュの減少により、同時に複数のクラッシュが発生する可能性が低くなります。 セッション リカバリ機能は単一のクラッシュからシームレスに回復させますが、複数の同時クラッシュからの回復は若干異なる設計になっています。 オペレータが複数の同時クラッシュに遭遇することはほとんど (またはまったく) ありませんが、そうした状況が発生した場合、ASR5x00 は多少のサブスクライバ セッションを犠牲にしてでも、システム保全性のリカバリを最優先とするよう設計されています。