Best Practices für den Betrieb von CRS-1 und IOS XR

Inhalt

Einführung

Dieses Dokument hilft Ihnen, folgende Punkte zu verstehen:

- Prozesse und Threads
- CRS-1-Fabric
- Steuern Sie Fläche
- Rommon und Monlib
- Physical Layer Interface Module (PLIM) und Modular Service Card (MSC)
- Konfigurationsverwaltung
- Sicherheit
- Out of Band
- Simple Network Management Protocol (SNMP)

Voraussetzungen

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse von Cisco IOS® XR verfügen.

Verwendete Komponenten

Die Informationen in diesem Dokument basierend auf folgenden Software- und Hardware-Versionen:

- Cisco IOS XR-Software
- CRS-1

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netz Live ist, überprüfen Sie, ob Sie die mögliche Auswirkung jedes möglichen Befehls verstehen.

Konventionen

Weitere Informationen zu Dokumentkonventionen finden Sie unter Cisco Technical Tips Conventions (Technische Tipps von Cisco zu Konventionen).

Cisco IOS XR - Überblick

Cisco IOS XR ist skalierbar. Der Kernel ist eine Microkernel-Architektur, sodass er nur grundlegende Dienste wie Prozessmanagement, Zeitplanung, Signale und Timer bereitstellt. Alle anderen Dienste wie Dateisysteme, Treiber, Protokoll-Stacks und Anwendungen werden als Ressourcenmanager betrachtet und in speichergeschütztem Benutzerbereich ausgeführt. Diese anderen Dienste können zur Laufzeit hinzugefügt oder entfernt werden, je nach Programmdesign. Der Fußabdruck von Microkernel beträgt nur 12 KB. Der Microkernel und das zugrunde liegende Betriebssystem sind von QNX Software Systems und wird Neutrino genannt. QNX ist auf das Betriebssystem-Design in Echtzeit spezialisiert. Der Microkernel ist präventiv, und der Scheduler ist prioritätsbasiert. Dadurch wird sichergestellt, dass das Context Switching zwischen Prozessen sehr schnell erfolgt und die Threads mit der höchsten Priorität bei Bedarf stets auf CPU zugreifen können. Dies sind einige der Vorteile, die Cisco IOS XR bietet. Der größte Vorteil ist jedoch das geerbte Design von Inter Process Communications innerhalb des Betriebssysteme Core.

Neutrino ist eine Nachricht, die das Betriebssystem überträgt, und Nachrichten sind das grundlegende Mittel der prozessübergreifenden Kommunikation zwischen allen Threads. Wenn ein bestimmter Server einen Dienst bereitstellen möchte, wird ein Kanal zum Austausch von Nachrichten erstellt. Clients fügen den Server-Kanal an, indem sie direkt dem entsprechenden Dateideskriptor zuordnen, um den Dienst zu nutzen. Die gesamte Kommunikation zwischen Client und Server erfolgt über denselben Mechanismus. Dies ist ein großer Vorteil für einen Supercomputer, den CRS-1 ist. Beachten Sie diese, wenn ein lokaler Lesevorgang auf einem UNIX-Standardkernel ausgeführt wird:

- Software in den Kernel unterbrechen.
- Kernel Dispatches in das Dateisystem.
- Daten werden empfangen.

Im Remote-Fall sollten Sie folgende Punkte berücksichtigen:

- Software in den Kernel unterbrechen.
- Kernel Dispatches NFS.
- NFS bezeichnet die Netzwerkkomponente.
- Remote sendet die Netzwerkkomponente.
- NFS wird aufgerufen.
- Kernel sendet das Dateisystem.

Die Semantik für das lokale Lesen und das Remote-Lesen sind nicht identisch. Argumente und Parameter für die Dateisperrung und das Festlegen von Berechtigungen sind unterschiedlich.

Berücksichtigen Sie den lokalen Fall QNX:

- Software in den Kernel unterbrechen.
- Kernel führt eine Nachricht aus, die an das Dateisystem übergeben wird.

Der nicht lokale Fall:

- Software in den Kernel unterbrechen.
- Kernel geht in QNET ein, das der IPC-Transportmechanismus ist.
- QNET geht in den Kernel.
- Kernel sendet das Dateisystem.

Alle Semantik-Parameter für die Übergabe von Argumenten und Dateisystemparameter sind identisch. Alles wurde an der IPC-Schnittstelle entkoppelt, sodass Client und Server vollständig voneinander getrennt werden können. Dies bedeutet, dass jeder Prozess jederzeit und überall ausgeführt werden kann. Wenn ein bestimmter Routingprozessor zu viele Serviceanfragen hat, können Sie diese Services problemlos auf eine andere CPU migrieren, die auf einem DRP ausgeführt wird. Ein Supercomputer, der verschiedene Dienste auf verschiedenen CPUs ausführt, die über mehrere Knoten verteilt sind und problemlos mit jedem anderen Knoten kommunizieren können. Die Infrastruktur ist so aufgebaut, dass sie skalierbar ist. Cisco hat diesen Vorteil genutzt und zusätzliche Software geschrieben, die sich in die grundlegenden Vorgänge des Kernels für die Nachrichtenübermittlung einfügt, der es dem CRS-Router ermöglicht, auf Tausende von Knoten zu skalieren, wobei ein Knoten, in diesem Fall eine CPU, eine Instanz des Betriebssystems ausführt, sei es ein Routing-Prozess (RP), ein Distributed Route-Prozessor (DRP), eine Modular Services Card (MSC) oder ein Switch-Prozessor (SP).

Prozesse und Threads

Innerhalb der Grenzen von Cisco IOS XR ist ein Prozess ein geschützter Speicherbereich, der

einen oder mehrere Threads enthält. Aus der Sicht der Programmierer führen die Threads die Arbeit aus, und jeder Thread vervollständigt einen logischen Ausführungspfad, um eine bestimmte Aufgabe auszuführen. Der Speicher, den die Threads während des Ausführungsvorgangs benötigen, gehört zu dem Prozess, innerhalb dessen sie arbeiten, geschützt vor anderen Prozessthreads. Ein Thread ist eine Ausführungseinheit mit einem Ausführungskontext, der einen Stapel enthält und registriert. Ein Prozess ist eine Gruppe von Threads, die einen virtuellen Adressbereich gemeinsam nutzen, obwohl ein Prozess einen einzelnen Thread enthalten kann, der aber häufig mehr enthält. Wenn ein anderer Thread in einem anderen Prozess versucht, in den Speicher in Ihrem Prozess zu schreiben, wird der fehlerhafte Prozess beendet. Wenn mehr als ein Thread innerhalb des Prozesses ausgeführt wird, hat dieser Thread Zugriff auf denselben Speicher innerhalb des Prozesses und kann so die Daten eines anderen Threads überschreiben. Führen Sie die Schritte in einer Prozedur aus, um die Synchronisierung mit Ressourcen beizubehalten, um diesen Thread innerhalb desselben Prozesses zu verhindern.

Ein Thread verwendet ein Objekt, das als Mutual Exclusion (MUTEX) bezeichnet wird, um den gegenseitigen Ausschluss von Diensten sicherzustellen. Der Thread, der MUTEX besitzt, ist der Thread, der als Beispiel in einen bestimmten Speicherbereich schreiben kann. Andere Threads ohne MUTEX können dies nicht. Es gibt auch andere Mechanismen, um die Synchronisierung mit Ressourcen sicherzustellen, wie Semaphores, bedingte Variablen oder Condvars, Barriers und Sleepons. Diese werden hier nicht behandelt, aber sie stellen Synchronisierungsdienste als Teil ihrer Aufgaben zur Verfügung. Wenn Sie die hier besprochenen Prinzipien mit Cisco IOS vergleichen, ist Cisco IOS ein einzelner Prozess, der viele Threads ausführt, mit allen Threads, die Zugriff auf denselben Speicherplatz haben. Cisco IOS bezeichnet diese Threadprozesse jedoch.

Prozess- und Threadstatus

Innerhalb von Cisco IOS XR gibt es Server, die die Services bereitstellen, und Clients, die die Services verwenden. Ein bestimmter Prozess kann mehrere Threads haben, die denselben Dienst bereitstellen. Ein anderer Prozess kann eine Reihe von Clients enthalten, die zu einem bestimmten Zeitpunkt einen bestimmten Service benötigen. Der Zugriff auf die Server ist nicht immer möglich, und wenn ein Client den Zugriff auf einen Dienst anfordert, befindet er sich dort und wartet darauf, dass der Server frei ist. In diesem Fall wird der Kunde als blockiert bezeichnet. Dies wird als Blockierungs-Client-Servermodell bezeichnet. Der Client kann blockiert werden, weil er auf eine Ressource wie MUTEX wartet oder weil der Server noch nicht geantwortet hat.

Führen Sie den Befehl **show process ospf aus**, um den Status der Threads im OSPF-Prozess zu überprüfen:

```
RP/0/RP1/CPU0:CWDCRS#show process ospf
                  Job Id: 250
                     PID: 110795
         Executable path: /disk0/hfr-rout-3.2.3/bin/ospf
              Instance #: 1
              Version ID: 00.00.0000
                 Respawn: ON
           Respawn count: 1
  Max. spawns per minute: 12
            Last started: Tue Jul 18 13:10:06 2006
           Process state: Run
           Package state: Normal
      Started on config: cfg/gl/ipv4-ospf/proc/101/ord_a/routerid
                    core: TEXT SHAREDMEM MAINMEM
              Max. core: 0
              Placement: ON
```

```
            startup_path: /pkg/startup/ospf.startup
                   Ready: 1.591s
               Available: 5.595s
         Process cpu time: 89.051 user, 0.254 kernel, 89.305 total
JID    TID  Stack pri state        HR:MM:SS:MSEC NAME
250    1      40K  10 Receive       0:00:11:0509 ospf
250    2      40K  10 Receive       0:01:08:0937 ospf
250    3      40K  10 Receive       0:00:03:0380 ospf
250    4      40K  10 Condvar       0:00:00:0003 ospf
250    5      40K  10 Receive       0:00:05:0222 ospf
```

Beachten Sie, dass für den ospf-Prozess eine Job-ID (JID) mit 250 angegeben wird. Dies ändert sich bei einem laufenden Router und generell bei einer bestimmten Version von Cisco IOS XR nicht. Innerhalb des OSPF-Prozesses gibt es fünf Threads mit jeweils eigener Thread-ID (TID). Listet den Stack-Speicherplatz für jeden Thread, die Priorität jedes Threads und seinen Zustand auf.

Synchrone Nachrichtenübermittlung

Es wird bereits erwähnt, dass QNX eine Meldung ist, die das Betriebssystem passiert. Es handelt sich um eine synchrone Meldung, die das Betriebssystem überträgt. Viele Probleme des Betriebssystems spiegeln sich im synchronen Messaging wider. Es wird nicht gesagt, dass die synchrone Nachrichtenübermittlung Probleme verursacht, sondern das Problem-Symptom wird in der synchronen Nachrichtenübermittlung wiedergegeben. Da es sich um eine synchrone Lösung handelt, kann der CRS-1-Operator problemlos auf Lebenszyklus- oder Statusinformationen zugreifen, was die Fehlerbehebung erleichtert. Der Life Cycle der Nachricht verläuft ähnlich wie folgt:

- Ein Server erstellt einen Meldungskanal.
- Ein Client stellt eine Verbindung zum Kanal eines Servers her (analog zum offenen POX).
- Ein Client sendet eine Nachricht an einen Server (MsgSend) und wartet auf eine Antwort und Blöcke.
- Der Server empfängt (MsgReceive) eine Nachricht von einem Client, verarbeitet die Nachricht und antwortet dem Client.
- Der Client entsperrt und verarbeitet die Antwort vom Server.

Dieses blockierende Client-Server-Modell ist die synchrone Nachrichtenübermittlung. Dies bedeutet, dass der Client eine Nachricht sendet und blockiert. Der Server empfängt die Nachricht, verarbeitet sie, antwortet zurück an den Client und entsperrt dann die Blockierung. Dies sind die Einzelheiten:

- Der Server wartet im RECEIVE-Status.
- Der Client sendet eine Nachricht an den Server und wird BLOCKIERT.
- Der Server empfängt die Meldung und hebt die Blockierung auf, wenn er im Empfangszustand wartet.
- Der Client wechselt in den Status REPLY.
- Der Server wechselt in den Status "RUNNING".
- Der Server verarbeitet die Nachricht.
- Der Server antwortet auf den Client.
- Client entsperrt.

Geben Sie den Befehl **show process** ein, um zu sehen, in welchem Zustand sich der Client und die Server befinden.

```
RP/0/RP1/CPU0:CWDCRS#show processes
JID    TID  Stack pri state        HR:MM:SS:MSEC NAME
1      1       0K  0  Ready       320:04:04:0649 procnto-600-smp-cisco-instr
1      3       0K  10 Nanosleep     0:00:00:0043 procnto-600-smp-cisco-instr
1      5       0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      7       0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      8       0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      11      0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      12      0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      13      0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      14      0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      15      0K  19 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      16      0K  10 Receive       0:02:01:0207 procnto-600-smp-cisco-instr
1      17      0K  10 Receive       0:00:00:0015 procnto-600-smp-cisco-instr
1      21      0K  10 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      23      0K  10 Running       0:07:34:0799 procnto-600-smp-cisco-instr
1      26      0K  10 Receive       0:00:00:0001 procnto-600-smp-cisco-instr
1      31      0K  10 Receive       0:00:00:0001 procnto-600-smp-cisco-instr
1      33      0K  10 Receive       0:00:00:0000 procnto-600-smp-cisco-instr
1      39      0K  10 Receive       0:13:36:0166 procnto-600-smp-cisco-instr
1      46      0K  10 Receive       0:06:32:0015 procnto-600-smp-cisco-instr
1      47      0K  56 Receive       0:00:00:0029 procnto-600-smp-cisco-instr
1      48      0K  10 Receive       0:00:00:0001 procnto-600-smp-cisco-instr
1      72      0K  10 Receive       0:00:00:0691 procnto-600-smp-cisco-instr
1      73      0K  10 Receive       0:00:00:0016 procnto-600-smp-cisco-instr
1      78      0K  10 Receive       0:09:18:0334 procnto-600-smp-cisco-instr
1      91      0K  10 Receive       0:09:42:0972 procnto-600-smp-cisco-instr
1      95      0K  10 Receive       0:00:00:0011 procnto-600-smp-cisco-instr
1      103     0K  10 Receive       0:00:00:0008 procnto-600-smp-cisco-instr
74     1       8K  63 Nanosleep     0:00:00:0001 wd-mbi
53     1      28K  10 Receive       0:00:08:0904 dllmgr
53     2      28K  10 Nanosleep     0:00:00:0155 dllmgr
53     3      28K  10 Receive       0:00:03:0026 dllmgr
53     4      28K  10 Receive       0:00:09:0066 dllmgr
53     5      28K  10 Receive       0:00:01:0199 dllmgr
270    1      36K  10 Receive       0:00:36:0091 qsm
270    2      36K  10 Receive       0:00:13:0533 qsm
270    5      36K  10 Receive       0:01:01:0619 qsm
270    7      36K  10 Nanosleep     0:00:22:0439 qsm
270    8      36K  10 Receive       0:00:32:0577 qsm
67     1      52K  19 Receive       0:00:35:0047 pkgfs
67     2      52K  10 Sigwaitinfo   0:00:00:0000 pkgfs
67     3      52K  19 Receive       0:00:30:0526 pkgfs
67     4      52K  10 Receive       0:00:30:0161 pkgfs
67     5      52K  10 Receive       0:00:25:0976 pkgfs
68     1       8K  10 Receive       0:00:00:0003 devc-pty
52     1      40K  16 Receive       0:00:00:0844 devc-conaux
52     2      40K  16 Sigwaitinfo   0:00:00:0000 devc-conaux
52     3      40K  16 Receive       0:00:02:0981 devc-conaux
52     4      40K  16 Sigwaitinfo   0:00:00:0000 devc-conaux
52     5      40K  21 Receive       0:00:03:0159 devc-conaux
65545  2      24K  10 Receive       0:00:00:0487 pkgfs
65546  1      12K  16 Reply         0:00:00:0008 ksh
66     1       8K  10 Sigwaitinfo   0:00:00:0005 pipe
66     3       8K  10 Receive       0:00:00:0000 pipe
66     4       8K  16 Receive       0:00:00:0059 pipe
66     5       8K  10 Receive       0:00:00:0149 pipe
66     6       8K  10 Receive       0:00:00:0136 pipe
71     1      16K  10 Receive       0:00:09:0250 shmwin_svr
71     2      16K  10 Receive       0:00:09:0940 shmwin_svr
61     1       8K  10 Receive       0:00:00:0006 mqueue
```

Gesperrte Prozess- und Prozessstatus

Geben Sie den Befehl **show process blocks** (**Anzeigeprozess blockiert**) ein, um festzustellen, welcher Prozess im Blockierungsstatus ist.

```
RP/0/RP1/CPU0:CWDCRS#show processes blocked
  Jid       Pid Tid                 Name State  Blocked-on
65546      4106   1                  ksh Reply    4104  devc-conaux
  105     61495   2              attachd Reply   24597  eth_server
  105     61495   3              attachd Reply    8205  mqueue
  316     65606   1          tftp_server Reply    8205  mqueue
  233     90269   2              lpts_fm Reply   90223  lpts_pa
  325    110790   1            udp_snmpd Reply   90257  udp
  253    110797   4               ospfv3 Reply   90254  raw_ip
  337    245977   2               fdiagd Reply   24597  eth_server
  337    245977   3               fdiagd Reply    8205  mqueue
65762   5996770   1                 exec Reply       1  kernel
65774   6029550   1                 more Reply    8203  pipe
65778   6029554   1       show_processes Reply       1  kernel
RP/0/RP1/CPU0:CWDCRS#
```

Durch synchronisiertes Übergeben von Nachrichten können Sie den Lebenszyklus der prozessübergreifenden Kommunikation zwischen den verschiedenen Threads leicht verfolgen. Ein Thread kann sich jederzeit in einem bestimmten Zustand befinden. Ein blockierter Zustand kann ein Symptom eines Problems sein. Dies bedeutet nicht, dass bei blockiertem Thread ein Problem vorliegt. Geben Sie daher den Befehl **show process blocks** nicht aus und öffnen Sie ein Ticket beim technischen Support von Cisco. Blockierte Threads sind ebenfalls sehr normal.

Beachten Sie die vorherige Ausgabe. Wenn Sie sich den ersten Thread in der Liste ansehen, beachten Sie, dass er der Schlüssel ist und seine Antwort auf devc-conaux blockiert wird. Der Client, in diesem Fall der ksh, hat eine Nachricht an den devc-conaux-Prozess gesendet, der Server, der devc-conaux ist, hält ksh reply blockiert, bis er antwortet. Ksh ist die UNIX-Shell, die jemand auf der Konsole oder dem AUX-Port verwendet. Ksh wartet auf die Eingabe von der Konsole, und wenn es keine gibt, weil der Operator nicht tippt, bleibt es blockiert, bis es einige Eingaben verarbeitet. Nach der Verarbeitung kehrt ksh zurück, um Antwort zu erhalten, die auf devc-conaux blockiert wurde.

Dies ist normal und veranschaulicht kein Problem. Der Punkt ist, dass blockierte Threads normal sind, und es hängt davon ab, welche XR-Version, welcher Systemtyp Sie haben, was Sie konfiguriert haben und wer tut, was dies ändert die Ausgabe des Befehls **show process blockiert**. Die Verwendung des Befehls **show process blocks** ist eine gute Methode, um Probleme mit Betriebssystemtypen zu beheben. Wenn ein Problem auftritt, z. B. wenn die CPU hoch ist, verwenden Sie den vorherigen Befehl, um festzustellen, ob etwas außerhalb des normalen Bereichs liegt.

Machen Sie sich mit den normalen Eigenschaften Ihres funktionierenden Routers vertraut. Dies stellt eine Baseline dar, die Sie bei der Fehlerbehebung von Prozesslebenszyklen als Vergleich verwenden können.

Ein Thread kann sich jederzeit in einem bestimmten Zustand befinden. Diese Tabelle enthält eine Liste der Zustände:

Ist der Staat:	Der Thread lautet:
DEFEKT	Tot. Der Kernel wartet auf die Freigabe der Threadressourcen.
AUSFÜHREN	Aktiv ausgeführt auf einer CPU

BEREIT	Nicht auf einer CPU ausgeführt, aber bereit zum Ausführen
GESPERRT	Ausgesetzt (SIGSTOP-Signal)
SENDEN	Warten, bis ein Server eine Nachricht empfängt
EMPFANGEN	Warten, bis ein Client eine Nachricht sendet
ANTWORT	Warten, bis ein Server auf eine Nachricht antwortet
STACK	Warten auf die Zuweisung weiterer Stacks
SEITE	Warten auf die Behebung eines Seitenfehlers durch den Prozessmanager
SIGNUSPEND	Warten auf ein Signal
SIGWAITINFO	Warten auf ein Signal
NANOSLEEP	Übernachten für einen bestimmten Zeitraum
MUTEX	Warten auf den Erwerb eines MUTEX
KONFERENZ	Warten auf Signalisierung einer bedingten Variablen
TEILNEHMEN	Warten auf den Abschluss eines anderen Threads
EINFÜGEN	Warten auf eine Unterbrechung
SEM	Warten auf den Erwerb eines Semaphors

Wichtige Prozesse und deren Funktionen

Cisco IOS XR bietet viele Prozesse. Dies sind einige wichtige, deren Funktionen hier erläutert werden.

WatchDog-Systemmonitor (WDSysmon)

Hierbei handelt es sich um einen Service zur Erkennung von Prozesshängen und niedrigen Speicherbedingungen. Geringer Speicher kann durch Speicherlecks oder andere Fremdumstände verursacht werden. Ein Hang kann das Ergebnis einer Reihe von Bedingungen sein, wie Prozess-Deadlocks, unbegrenzte Schleifen, Kernel-Abstürze oder Planungsfehler. In einer Multithread-Umgebung kann das System in einen Zustand versetzt werden, der als Deadlock-Zustand bezeichnet wird, oder einfach Dead Lock. Ein Deadlock kann auftreten, wenn ein oder mehrere Threads aufgrund von Ressourcenkonflikten nicht fortfahren können. Thread A kann beispielsweise eine Nachricht an Thread B senden, während gleichzeitig Thread B eine Nachricht an Thread A sendet. Beide Threads warten aufeinander und können sich im gesendeten blockierten Zustand befinden, und beide Threads warten ewig. Dies ist ein einfacher Fall, der zwei Threads umfasst. Wenn jedoch ein Server für eine Ressource verantwortlich ist, die von vielen Threads verwendet wird, in einem anderen Thread blockiert wird, können die vielen Threads, die Zugriff auf diese Ressource anfordern, blockiert werden, wenn sie auf dem Server warten.

Deadlocks können zwischen einigen Threads auftreten, können sich aber auch auf andere Threads auswirken. Deadlocks werden durch ein gutes Programmdesign vermieden, aber unabhängig davon, wie großartig ein Programm gestaltet und geschrieben wird. Manchmal kann eine bestimmte Folge von Ereignissen, die von Daten abhängig sind, mit bestimmten Timings zu einem Deadlock führen. Deadlocks sind nicht immer deterministisch und sind im Allgemeinen sehr schwer zu reproduzieren. WDSysmon hat viele Threads mit einem Thread, der mit der höchsten von Neutrino unterstützten Priorität ausgeführt wird, 63. Durch das Ausführen mit Priorität 63 wird sichergestellt, dass der Thread die CPU-Zeit in einer prioritätsbasierten präventiven Planungsumgebung erhält. WDSysmon arbeitet mit der Hardwareüberwachungsfunktion zusammen und überwacht die Softwareprozesse, die nach hängenden Bedingungen suchen. Wenn solche Bedingungen erkannt werden, sammelt WDSysmon weitere Informationen über die Bedingung, kann den Prozess oder den Kernel überschreiben, an Syslogs schreiben, Skripte ausführen und die gesperrten Prozesse beenden. Je nachdem, wie drastisch das Problem ist, kann es einen Routen-Prozessor-Switchover initiieren, um den Systembetrieb aufrechtzuerhalten.

```
RP/0/RP1/CPU0:CWDCRS#show processes wdsysmon
                  Job Id: 331
                     PID: 36908
         Executable path: /disk0/hfr-base-3.2.3/sbin/wdsysmon
              Instance #: 1
              Version ID: 00.00.0000
                 Respawn: ON
           Respawn count: 1
  Max. spawns per minute: 12
            Last started: Tue Jul 18 13:07:36 2006
           Process state: Run
           Package state: Normal
                    core: SPARSE
                Max. core: 0
                   Level: 40
                Mandatory: ON
             startup_path: /pkg/startup/wdsysmon.startup
             memory limit: 10240
                   Ready: 0.705s
        Process cpu time: 4988.295 user, 991.503 kernel, 5979.798 total

JID    TID  Stack pri state        HR:MM:SS:MSEC NAME
331    1      84K  19 Receive       0:00:00:0029 wdsysmon
331    2      84K  10 Receive       0:17:34:0212 wdsysmon
331    3      84K  10 Receive       0:00:00:0110 wdsysmon
331    4      84K  10 Receive       1:05:26:0803 wdsysmon
331    5      84K  19 Receive       0:00:06:0722 wdsysmon
331    6      84K  10 Receive       0:00:00:0110 wdsysmon
331    7      84K  63 Receive       0:00:00:0002 wdsysmon
331    8      84K  11 Receive       0:00:00:0305 wdsysmon
331    9      84K  20 Sem           0:00:00:0000 wdsysmon
```

Der Prozess WDSysmon hat neun Threads. Vier werden mit Priorität 10 ausgeführt, die anderen vier mit 11, 19, 20 und 63. Beim Entwerfen eines Prozesses berücksichtigt der Programmierer sorgfältig die Priorität, die jedem Thread innerhalb des Prozesses zugewiesen werden soll. Wie bereits erwähnt, ist der Scheduler priorienbasiert, d. h. ein Thread mit höherer Priorität hat immer Vorrang vor einem Thread mit niedrigerer Priorität. Priorität 63 ist die höchste Priorität, die ein Thread ausführen kann. In diesem Fall ist dies Thread 7. Thread 7 ist der Überwachungs-Thread, der CPU-Hogs verfolgt. Sie muss mit einer höheren Priorität ausgeführt werden als die anderen Threads, die sie überwacht, andernfalls wird sie möglicherweise überhaupt nicht ausgeführt, wodurch sie von den Schritten, die sie ausführen soll, ausgeschlossen wird.

Netio

In Cisco IOS gibt es das Konzept von Fast Switching und Process Switching. Fast Switching verwendet den CEF-Code und tritt zur Unterbrechungszeit auf. Beim Prozess-Switching wird ip_input verwendet. Dies ist der IP-Switching-Code und ein geplanter Prozess. Auf höheren Endplattformen erfolgt das CEF-Switching in der Hardware, und auf der CPU ist ip_input geplant. Die Entsprechung von ip_input in Cisco IOS XR ist Netio.

```
P/0/RP1/CPU0:CWDCRS#show processes netio
                  Job Id: 241
                     PID: 65602
         Executable path: /disk0/hfr-base-3.2.3/sbin/netio
              Instance #: 1
                    Args: d
              Version ID: 00.00.0000
                 Respawn: ON
           Respawn count: 1
  Max. spawns per minute: 12
            Last started: Tue Jul 18 13:07:53 2006
           Process state: Run
           Package state: Normal
                    core: DUMPFALLBACK COPY SPARSE
                Max. core: 0
                   Level: 56
                Mandatory: ON
             startup_path: /pkg/startup/netio.startup
                   Ready: 17.094s
        Process cpu time: 188.659 user, 5.436 kernel, 194.095 total
JID    TID  Stack pri state        HR:MM:SS:MSEC NAME
241    1     152K  10 Receive       0:00:13:0757 netio
241    2     152K  10 Receive       0:00:10:0756 netio
241    3     152K  10 Condvar       0:00:08:0094 netio
241    4     152K  10 Receive       0:00:22:0016 netio
241    5     152K  10 Receive       0:00:00:0001 netio
241    6     152K  10 Receive       0:00:04:0920 netio
241    7     152K  10 Receive       0:00:03:0507 netio
241    8     152K  10 Receive       0:00:02:0139 netio
241    9     152K  10 Receive       0:01:44:0654 netio
241    10    152K  10 Receive       0:00:00:0310 netio
241    11    152K  10 Receive       0:00:13:0241 netio
241    12    152K  10 Receive       0:00:05:0258 netio
```

Group Services-Prozess (GSP)

Es besteht ein Bedarf an Kommunikation in jedem Supercomputer mit mehreren tausend Knoten, die jeweils ihre eigene Instanz des Kernels ausführen. Im Internet erfolgt eine bis mehrere Kommunikation effizient über Multicasting-Protokolle. Das APS ist das interne Multicasting-Protokoll, das für IPC in CRS-1 verwendet wird. Das APS bietet eine bis mehrere zuverlässige Gruppenkommunikation, die ohne Verbindung mit asynchroner Semantik erfolgt. So kann das APS auf Tausende von Knoten skaliert werden.

```
RP/0/RP1/CPU0:CWDCRS#show processes gsp
                  Job Id: 171
                     PID: 65604
         Executable path: /disk0/hfr-base-3.2.3/bin/gsp
              Instance #: 1
              Version ID: 00.00.0000
                 Respawn: ON
           Respawn count: 1
  Max. spawns per minute: 12
            Last started: Tue Jul 18 13:07:53 2006
```

```
           Process state: Run
           Package state: Normal
                    core: TEXT SHAREDMEM MAINMEM
               Max. core: 0
                   Level: 80
               Mandatory: ON
            startup_path: /pkg/startup/gsp-rp.startup
                   Ready: 5.259s
               Available: 16.613s
        Process cpu time: 988.265 user, 0.792 kernel, 989.057 total
JID    TID  Stack pri state        HR:MM:SS:MSEC NAME
171    1     152K  30 Receive       0:00:51:0815 gsp
171    3     152K  10 Condvar       0:00:00:0025 gsp
171    4     152K  10 Receive       0:00:08:0594 gsp
171    5     152K  10 Condvar       0:01:33:0274 gsp
171    6     152K  10 Condvar       0:00:55:0051 gsp
171    7     152K  10 Receive       0:02:24:0894 gsp
171    8     152K  10 Receive       0:00:09:0561 gsp
171    9     152K  10 Condvar       0:02:33:0815 gsp
171    10    152K  10 Condvar       0:02:20:0794 gsp
171    11    152K  10 Condvar       0:02:27:0880 gsp
171    12    152K  30 Receive       0:00:46:0276 gsp
171    13    152K  30 Receive       0:00:45:0727 gsp
171    14    152K  30 Receive       0:00:49:0596 gsp
171    15    152K  30 Receive       0:00:38:0276 gsp
171    16    152K  10 Receive       0:00:02:0774 gsp
```

BCDL Bulk Content Downloader

BCDL wird verwendet, um zuverlässig Multicast-Daten an verschiedene Knoten wie RPs und MSCs weiterzuleiten. Dabei wird das APS als Grundlage für den Transport herangezogen. BCDL garantiert **die** Zustellung von Nachrichten. Innerhalb von BCDL gibt es einen Agenten, einen Produzenten und einen Verbraucher. Der Agent ist der Prozess, der mit dem Hersteller kommuniziert, um die Daten vor den Multicasts abzurufen und an die Verbraucher zu puffern. Der Hersteller ist der Prozess, der die Daten erzeugt, die jeder will, und der Verbraucher ist der Prozess interessiert, die Daten vom Hersteller zu erhalten. BCDL wird bei Cisco IOS XR-Software-Upgrades verwendet.

LWM (Lightweight Messaging)

LWM ist eine von Cisco entwickelte Form von Messaging, die entwickelt wurde, um eine Abstraktionsebene zwischen den Anwendungen zu schaffen, die prozessübergreifend miteinander kommunizieren, und Neutrino. Ziel ist dabei die Unabhängigkeit des Betriebssystems und der Transportschicht. Wenn Cisco den Betriebssystem-Anbieter von QNX in einen anderen Anbieter ändern möchte, hilft eine Abstraktionsebene zwischen den rudimentären Funktionen des zugrunde liegenden Betriebssystems, die Abhängigkeit vom Betriebssystem zu beseitigen, und unterstützt die Portierung auf ein anderes Betriebssystem. LWM bietet eine synchrone, garantierte Nachrichtenzustellung, die den Absender wie die Weiterleitung der nativen Neutrino-Nachricht blockiert, bis der Empfänger antwortet.

LWM ermöglicht auch die asynchrone Übermittlung von Nachrichten über 40-Bit-Impulse. Asynchrone Meldungen werden asynchron gesendet, d. h. die Nachricht wird in die Warteschlange gestellt, und der Absender blockiert nicht, wird aber nicht asynchron vom Server empfangen, wenn der Server die nächste verfügbare Nachricht abfragt. LWM ist als Client/Server strukturiert. Der Server erstellt einen Kanal, der ihm ein **Ohr** gibt, Nachrichten zu hören und sich in einer Zeit, in der die Schleife eine Nachricht empfängt Abhören auf dem Kanal, die er gerade erstellt. Wenn eine Nachricht eingeht, löst sie die Blockierung aus und erhält eine Client-ID. Dies

entspricht im Prinzip der Empfänger-ID aus der empfangenen Nachricht. Der Server führt dann eine Verarbeitung durch und antwortet später auf die Client-ID.

Auf der Clientseite wird eine Nachrichtenverbindung hergestellt. Er erhält eine Kennung, mit der er verbunden wird, und sendet dann eine Nachricht und wird blockiert. Wenn der Server die Verarbeitung abgeschlossen hat, antwortet er und der Client wird entsperrt. Dies ist die fast gleiche wie die native Nachricht von Neutrinos, sodass die Ebene der Abstraktion ist sehr dünn.

LWM ist mit einer Mindestanzahl von Systemaufrufen und Kontext-Switches für eine hohe Leistung konzipiert und die bevorzugte Methode für IPC in der Cisco IOS XR-Umgebung.

umarmen

Auf der grundlegendsten Ebene ist das Umgebungsüberwachungssystem dafür verantwortlich, zu warnen, wenn physische Parameter, z. B. Temperatur, Spannung, Gebläsegeschwindigkeit usw., außerhalb der Betriebsbereiche liegen und Hardware abgeschaltet wird, die kritische Werte erreicht, bei denen Hardware beschädigt werden könnte. Er überwacht in regelmäßigen Abständen jeden verfügbaren Hardware-Sensor, vergleicht den gemessenen Wert mit den kartenspezifischen Grenzwerten und löst bei Bedarf Alarme aus, um diese Aufgabe zu erfüllen. Ein persistenter Prozess, der bei der Systeminitialisierung gestartet wurde und regelmäßig alle Hardwaresensoren (z. B. Spannung, Temperatur und Lüftergeschwindigkeit) im Chassis abfragt und diese Daten an externe Management-Clients weiterleitet. Darüber hinaus vergleicht der periodische Prozess Sensormessungen mit Alarmschwellen und gibt Umgebungswarnungen an die Systemdatenbank weiter, damit der Fehler-Manager nachfolgende Maßnahmen ergreifen kann. Wenn die Sensormessungen gefährlich außerhalb des Bereichs liegen, kann der Umgebungsüberwachungsprozess dazu führen, dass die Karte heruntergefahren wird.

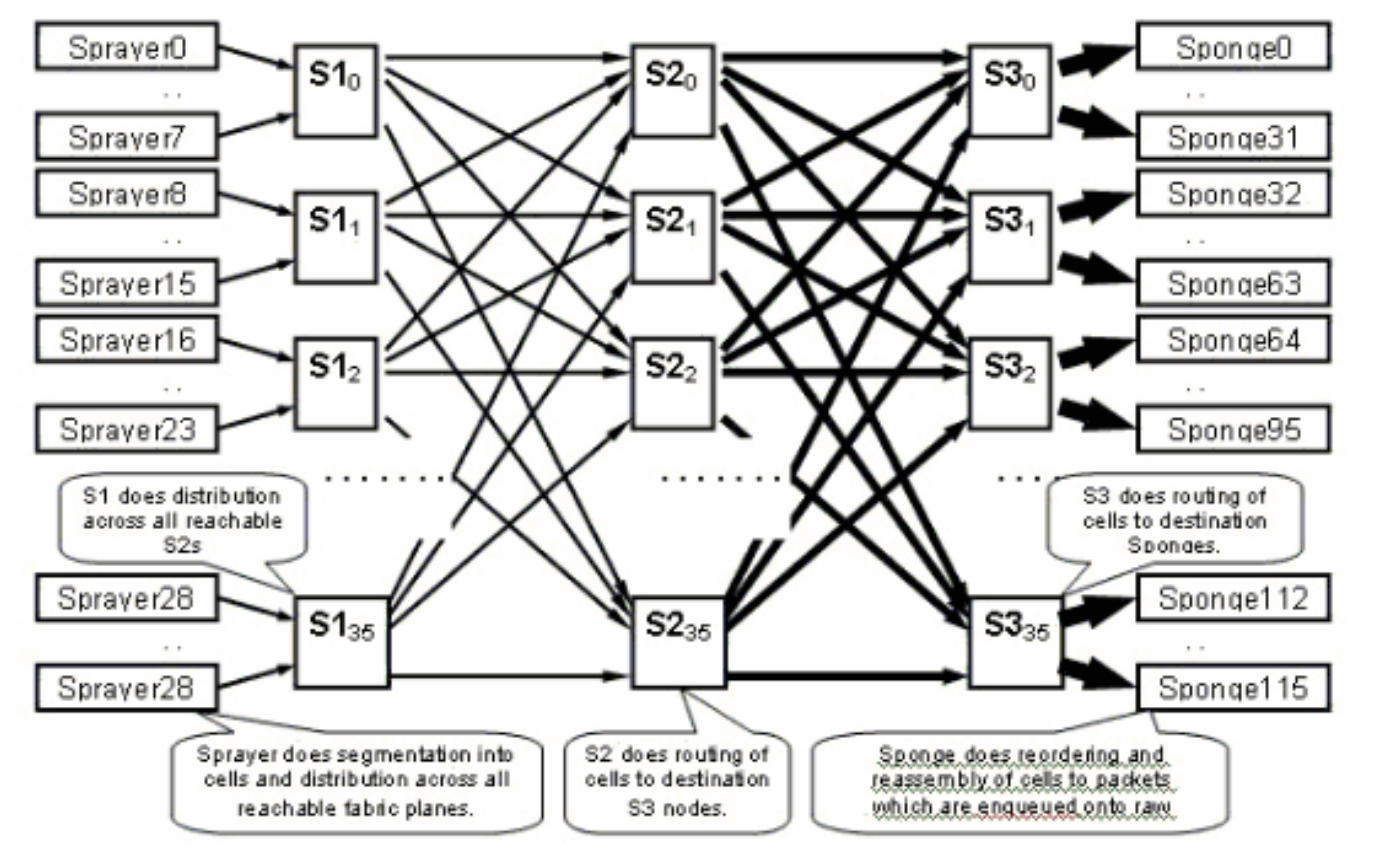
CRS-1-Fabric - Einführung

- Multistage-Fabric - Topologie mit drei Phasen
- Dynamisches Routing innerhalb der Fabric zur Minimierung von Überlastungen
- Zellbasiert: 136-Byte-Zellen, 120-Byte-Datennutzlast
- Datenflusskontrolle zur Optimierung der Datenverkehrsisolierung und Minimierung der Pufferanforderungen in der Fabric
- Bereitstellung von Phase-zu-Phase-Speedhop
- Unterstützte zwei Casts Datenverkehr (Unicast und Multicast)
- Zwei Prioritäten für pro Gast unterstützten Datenverkehr (hoch und niedrig)
- Unterstützung für 1 Mio. Fabric-Multicast-Gruppen (FGIDs)
- Kosteneffiziente Fehlertoleranz: N+1- oder N+k-Redundanz mit Fabric-Ebenen im Gegensatz zu 1+1 bei stark erhöhten Kosten

Wenn Sie im Einzel-Chassis-Modus arbeiten, befinden sich die S1-, S2- und S3-Komponenten auf denselben Fabric-Karten. Diese Karte wird auch als **S123-Karte** bezeichnet. In einer Multi-Chassis-Konfiguration ist das S2 getrennt und befindet sich im Fabric Card Chassis (FCC). Für diese Konfiguration sind zwei Fabric-Karten erforderlich, um eine Ebene, eine S2-Karte und eine S13-Karte zu bilden. Jede MSC ist mit acht Fabric-Ebenen verbunden, um Redundanz zu gewährleisten, sodass bei Loose einer oder mehrerer Ebenen der Datenverkehr in der Fabric dennoch weitergeleitet wird, obwohl der gesamte Datenverkehr, der durch die Fabric geleitet werden kann, geringer ist. Das CRS kann bei den meisten Paketgrößen mit nur sieben Ebenen weiterhin auf der Linerate betrieben werden. Über eine ungerade und gerade Ebene wird der Backdruck über das Fabric übertragen. Es wird nicht empfohlen, ein System mit weniger als zwei

Ebenen in einer ungeraden und geraden Ebene auszuführen. Bei weniger als zwei Ebenen wird keine Konfiguration unterstützt.

Die Fabric-Ebene



Das vorherige Diagramm stellt eine Ebene dar. Man muss das Diagramm mit acht multiplizieren. Das bedeutet, dass die Spritze (ingressq) einer LC mit 8 S1s (1 S1 pro Ebene) verbunden ist. Jede Fabric-Ebene verbindet S1 mit acht Spritzen:

- die 8 oberen LCs des Chassis
- die 8 unteren LCs

Es gibt 16 S1 pro LC-Chassis mit 16 Steckplätzen: 8 für die oberen LCs (1 pro Ebene) + 8 für die unteren LCs.

Auf einem Chassis mit 16 Steckplätzen verfügt eine S123 Fabric Card über 2 S1s, 2 S2 und 4 S3s. Dies ist Teil der Fabric Speedup-Berechnung. Der Datenverkehr ist doppelt so hoch, sodass er die Fabric verlassen kann, wie der Datenverkehr eintritt. Derzeit gibt es auch zwei Schwämme (fabricq) pro LC im Vergleich zu 1 Sprayer. Auf diese Weise kann der Egress-LC gepuffert werden, wenn mehrere Eingangs-LCs einen Egress-LC überladen. Der Egress-LC kann diese zusätzliche Bandbreite aus der Fabric aufnehmen.

Fabric-Überwachung

Verfügbarkeit und Konnektivität der Ebene:

```
admin show controller fabric plane all
admin show controller fabric connectivity all detail
```

Überprüfen Sie, ob die Ebenen Zellen empfangen/übertragen und einige Fehler inkrementieren:

```
admin show controllers fabric plane all statistics
```

Die Akronyme im vorherigen Befehl:

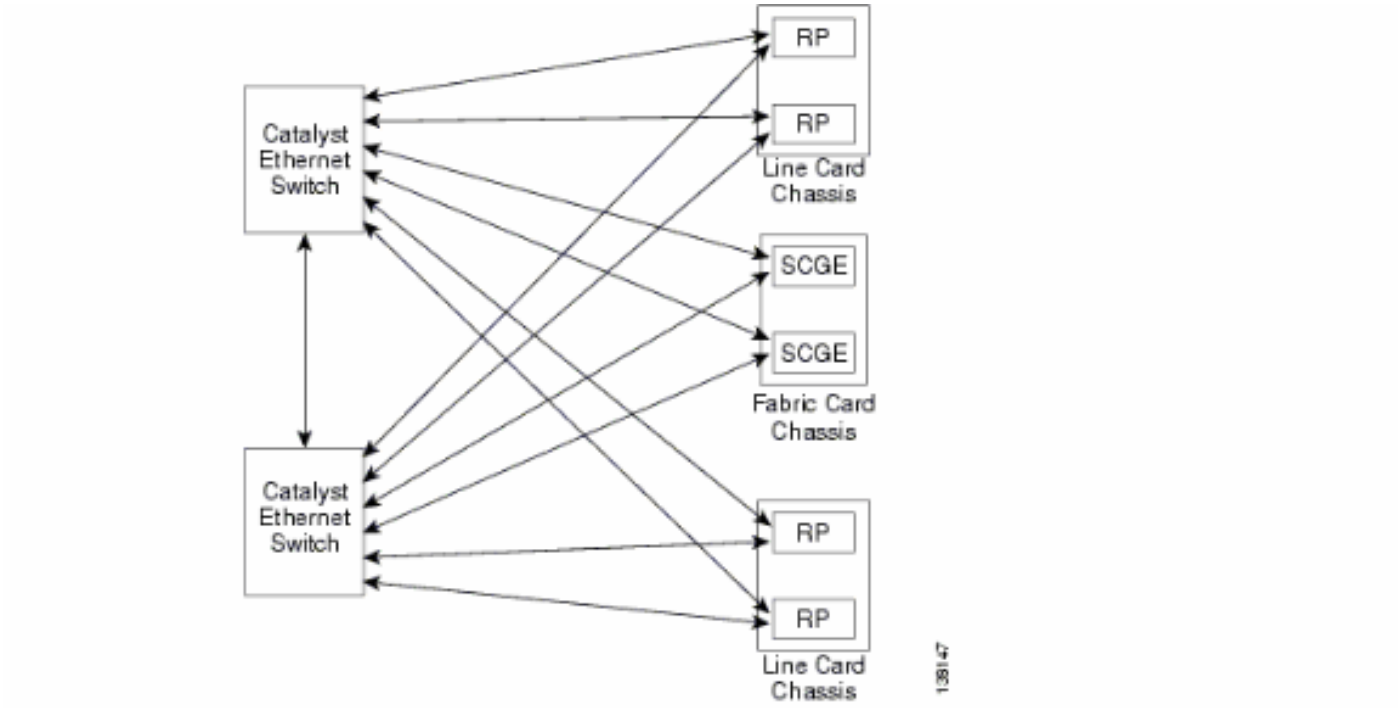
- CE: Korrigierbarer Fehler
- UCE - Nicht korrigierbarer Fehler
- PE: Paritätsfehler

Machen Sie sich keine Sorgen, wenn Sie ein paar Fehler bemerken, da dies beim Hochfahren passieren kann. Die Felder sollten zur Laufzeit nicht inkrementiert werden. Ist dies der Fall, kann dies ein Hinweis auf ein Problem in der Struktur sein. Geben Sie diesen Befehl ein, um eine Aufschlüsselung der Fehler pro Fabric-Ebene zu erhalten:

```
admin show controllers fabric plane <0-7> statistics detail
```

Übersicht über die Kontrollebene

Die Kontrollebenen-Konnektivität zwischen dem Linecard-Chassis und dem Fabric-Chassis erfolgt derzeit über Gigabit Ethernet-Ports auf den RPs (LCC) und SCGE (FCC). Die Verbindung zwischen den Ports erfolgt über ein Paar Catalyst 6500 Switches, die über zwei oder mehr Gigabit Ethernet-Ports verbunden werden können.



Catalyst 6500-Konfiguration

Dies ist die empfohlene Konfiguration für Catalyst-Switches, die für die Multi-Chassis-Kontrollebene verwendet werden:

- An allen Ports wird ein einzelnes VLAN verwendet.
- Alle Ports werden im Zugriffsmodus (ohne Trunking) ausgeführt.
- Spanning-Tree 802.1w/s wird für die Vermeidung von Schleifen verwendet.
- Zwei oder mehr Verbindungen dienen zum Cross-Connect der beiden Switches, und STP wird zur Vermeidung von Schleifen verwendet. Channel wird nicht empfohlen.
- Ports, die mit CRS-1 RP und SCGE verbunden sind, verwenden den Pre-Standard-Modus, da IOS-XR die standardbasierten 802.1s-Standards nicht unterstützt.
- UDLD sollte an den Ports aktiviert werden, die zwischen den Switches sowie zwischen den Switches und dem RP/SCGE verbunden sind.
- UDLD ist auf CRS-1 standardmäßig aktiviert.

Weitere Informationen zur Konfiguration eines Catalyst 6500 in einem Mehrfacheinschubsystem finden Sie unter Bringing Up the Cisco IOS XR Software on a Multishelf.

Management der Kontrollebene für mehrere Chassis

Das Catalyst 6504-E-Chassis, das die Konnektivität der Steuerungsebene für das Multi-Chassis-System bereitstellt, ist für folgende Management-Services konfiguriert:

- In-Band-Management über Port-Gigabit-1/2, das mit einem LAN-Switch an jedem PoP verbunden wird. Der Zugriff ist nur für eine kleine Anzahl von Subnetzen und Protokollen zulässig.
- NTP wird verwendet, um die Systemzeit festzulegen.
- Die Syslog-Funktion wird auf den Standard-Hosts ausgeführt.
- SNMP Polling und Traps können für kritische Funktionen aktiviert werden.

Hinweis: Es sollten keine Änderungen am Catalyst in Betrieb genommen werden. Vorab sollten alle geplanten Änderungen getestet werden. Es wird dringend empfohlen, dies während eines Wartungsfensters zu tun.

Dies ist ein Beispiel für eine Verwaltungskonfiguration:

```
#In-band management connectivity
interface GigabitEthernet2/1
 description *CRS Multi-chassis Management Ethernet - DO NOT TOUCH*
 ip address [ip address] [netmask]
 ip access-group control_only in
!
!
ip access-list extended control_only
 permit udp [ip address] [netmask] any eq snmp
 permit udp [ip address] [netmask] eq ntp any
 permit tcp [ip address] [netmask] any eq telnet

#NTP

ntp update-calendar
ntp server [ip address]

#Syslog
logging source-interface Loopback0
logging [ip address]
logging buffered 4096000 debugging
no logging console
```

```
#RADIUS
aaa new-model
aaa authentication login default radius enable
enable password {password}
radius-server host [ip address] auth-port 1645 acct-port 1646
radius-server key {key}

#Telnet and console access
!
access-list 3 permit [ip address]
!
line con 0
 exec-timeout 30 0
 password {password}
line vty 0 4
 access-class 3 in
 exec-timeout 0 0
password {password}
```

ROMMON und Monlib

Cisco monlib ist ein ausführbares Programm, das auf dem Gerät gespeichert und zur Ausführung durch ROMMON in den RAM geladen wird. ROMMON verwendet monlib, um auf Dateien auf dem Gerät zuzugreifen. ROMMON-Versionen können aktualisiert werden. Dies sollte auf Empfehlung des technischen Supports von Cisco erfolgen. Die neueste ROMMON-Version ist 1.40.

Upgrade-Anweisungen

Führen Sie diese Schritte aus:

1. Laden Sie die ROMMON-Binärdateien von Cisco CRS-1 ROMMON herunter (nur registrierte Kunden).
2. Entpacken Sie die TAR-Datei, und kopieren Sie die 6 BIN-Dateien in das CRS-Stammverzeichnis von Disk0.
   ```
   RP/0/RP0/Router#dir disk0:/*.bin

   Directory of disk0:

   65920       -rwx  360464      Fri Oct 28 12:58:02 2005  rommon-hfr-ppc7450-sc-dsmp-A.bin
   66112       -rwx  360464      Fri Oct 28 12:58:03 2005  rommon-hfr-ppc7450-sc-dsmp-B.bin
   66240       -rwx  376848      Fri Oct 28 12:58:05 2005  rommon-hfr-ppc7455-asmp-A.bin
   66368       -rwx  376848      Fri Oct 28 12:58:06 2005  rommon-hfr-ppc7455-asmp-B.bin
   66976       -rwx  253904      Fri Oct 28 12:58:08 2005  rommon-hfr-ppc8255-sp-A.bin
   67104       -rwx  253492      Fri Oct 28 12:58:08 2005  rommon-hfr-ppc8255-sp-B.bin
   ```
3. Verwenden Sie die **Anzeigetafel. | inc ROM|NODE|PLIM**-Befehl, um die aktuelle Version des Programmzweigs anzuzeigen.
   ```
   RP/0/RP0/CPU0:ROUTER(admin)#show diag | inc ROM|NODE|PLIM
   NODE 0/0/SP : MSC(SP)
     ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
   PLIM 0/0/CPU0 : 4OC192-POS/DPT
     ROMMON: Version 1.19b(20050216:033559) [CRS-1 ROMMON]
   NODE 0/2/SP : MSC(SP)
     ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
   PLIM 0/2/CPU0 : 8-10GbE
     ROMMON: Version 1.19b(20050216:033559) [CRS-1 ROMMON]
   NODE 0/4/SP : Unknown Card Type
   NODE 0/6/SP : MSC(SP)
     ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
   PLIM 0/6/CPU0 : 16OC48-POS/DPT
   ```

```
  ROMMON: Version 1.19b(20050216:033559) [CRS-1 ROMMON]
NODE 0/RP0/CPU0 : RP
  ROMMON: Version 1.19b(20050216:033559) [CRS-1 ROMMON]
NODE 0/RP1/CPU0 : RP
  ROMMON: Version 1.19b(20050216:033559) [CRS-1 ROMMON]
NODE 0/SM0/SP : FC/S
  ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
NODE 0/SM1/SP : FC/S
  ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
NODE 0/SM2/SP : FC/S
  ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
NODE 0/SM3/SP : FC/S
ROMMON: Version 1.19b(20050216:033352) [CRS-1 ROMMON]
```

4. Wechseln Sie in den ADMIN-Modus, und verwenden Sie den **Upgrade-Befehl auf dem Befehl "all disk0"**, um den ROMMON zu aktualisieren.

```
RP/0/RP0/CPU0:ROUTER#admin
RP/0/RP0/CPU0:ROUTER(admin)#upgrade rommon a all disk0
Please do not power cycle, reload the router or reset any nodes until
 all upgrades are completed.
Please check the syslog to make sure that all nodes are upgraded successfully.
If you need to perform multiple upgrades, please wait for current upgrade
 to be completed before proceeding to another upgrade.
Failure to do so may render the cards under upgrade to be unusable.
```

5. Beenden Sie den ADMIN-Modus, und geben Sie das **Anzeigeprotokoll ein. | inc "OK, ROMMON A"** und stellen Sie sicher, dass alle Knoten erfolgreich aktualisiert wurden. Wenn einer der Knoten ausfällt, kehren Sie zu Schritt 4 zurück und programmieren Sie neu.

```
RP/0/RP0/CPU0:ROUTER#show logging | inc "OK, ROMMON A"
RP/0/RP0/CPU0:Oct 28 14:40:57.223 PST8: upgrade_daemon[380][360]: OK, ROMMON A is
programmed successfully. SP/0/0/SP:Oct 28 14:40:58.249 PST8: upgrade_daemon[125][121]: OK,
ROMMON A is programmed successfully. SP/0/2/SP:Oct 28 14:40:58.251 PST8:
upgrade_daemon[125][121]: OK, ROMMON A is programmed successfully. LC/0/6/CPU0:Oct 28
14:40:58.336 PST8: upgrade_daemon[244][233]: OK, ROMMON A is programmed successfully.
LC/0/2/CPU0:Oct 28 14:40:58.365 PST8: upgrade_daemon[244][233]: OK, ROMMON A is programmed
successfully. SP/0/SM0/SP:Oct 28 14:40:58.439 PST8: upgrade_daemon[125][121]: OK, ROMMON A
is programmed successfully. SP/0/SM1/SP:Oct 28 14:40:58.524 PST8: upgrade_daemon[125][121]:
OK, ROMMON A is programmed successfully. LC/0/0/CPU0:Oct 28 14:40:58.530 PST8:
upgrade_daemon[244][233]: OK, ROMMON A is programmed successfully. RP/0/RP1/CPU0:Oct 28
14:40:58.593 PST8: upgrade_daemon[380][360]: OK, ROMMON A is programmed successfully.
SP/0/6/SP:Oct 28 14:40:58.822 PST8: upgrade_daemon[125][121]: OK, ROMMON A is programmed
successfully. SP/0/SM2/SP:Oct 28 14:40:58.890 PST8: upgrade_daemon[125][121]: OK, ROMMON A
is programmed successfully. SP/0/SM3/SP:Oct 28 14:40:59.519 PST8: upgrade_daemon[125][121]:
OK, ROMMON A is programmed successfully.
```

6. Wechseln Sie in den ADMIN-Modus, und verwenden Sie den Befehl **upgrade rommon b all disk0**, um den ROMMON zu aktualisieren.

```
RP/0/RP0/CPU0:ROUTER#admin
RP/0/RP0/CPU0:ROUTER(admin)#upgrade rommon b all disk0
Please do not power cycle, reload the router or reset any nodes until
 all upgrades are completed.
Please check the syslog to make sure that all nodes are upgraded successfully.
If you need to perform multiple upgrades, please wait for current upgrade
 to be completed before proceeding to another upgrade.
Failure to do so may render the cards under upgrade to be unusable.
```

7. Beenden Sie den ADMIN-Modus, und geben Sie das **Anzeigeprotokoll ein. | inc "OK, ROMMON B"** und stellen Sie sicher, dass alle Knoten erfolgreich aktualisiert wurden. Wenn einer der Knoten ausfällt, kehren Sie zu Schritt 4 zurück und programmieren Sie neu.

```
RP/0/RP0/CPU0:Router#show logging | inc "OK, ROMMON B"
RP/0/RP0/CPU0:Oct 28 13:27:00.783 PST8: upgrade_daemon[380][360]: OK,
ROMMON B is programmed successfully.
LC/0/6/CPU0:Oct 28 13:27:01.720 PST8: upgrade_daemon[244][233]: OK,
ROMMON B is programmed successfully.
```

```
SP/0/2/SP:Oct 28 13:27:01.755 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
LC/0/2/CPU0:Oct 28 13:27:01.775 PST8: upgrade_daemon[244][233]: OK,
ROMMON B is programmed successfully.
SP/0/0/SP:Oct 28 13:27:01.792 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
SP/0/SM0/SP:Oct 28 13:27:01.955 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
LC/0/0/CPU0:Oct 28 13:27:01.975 PST8: upgrade_daemon[244][233]: OK,
ROMMON B is programmed successfully.
SP/0/6/SP:Oct 28 13:27:01.989 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
SP/0/SM1/SP:Oct 28 13:27:02.087 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
RP/0/RP1/CPU0:Oct 28 13:27:02.106 PST8: upgrade_daemon[380][360]: OK,
ROMMON B is programmed successfully.
SP/0/SM3/SP:Oct 28 13:27:02.695 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
SP/0/SM2/SP:Oct 28 13:27:02.821 PST8: upgrade_daemon[125][121]: OK,
ROMMON B is programmed successfully.
```

8. Der **Upgrade**-Befehl verbrannt mit dem neuen ROMMON nur einen speziellen reservierten Abschnitt von Bootflash. Der neue ROMMON bleibt jedoch inaktiv, bis die Karte neu geladen wird. Wenn Sie die Karte neu laden, ist der neue ROMMON aktiv. Setzen Sie die einzelnen Knoten nacheinander zurück, oder setzen Sie dazu einfach den gesamten Router zurück.

```
Reload Router:
RP/0/RP0/CPU0:ROUTER#hw-module node 0/RP0/CPU0 or 0/RP1/CPU0 reload (depends on which on is
in Standby Mode.
RP/0/RP0/CPU0:ROUTER#reload
!--- Issue right after the first command. Updating Commit Database. Please wait...[OK]
Proceed with reload? [confirm] !--- Reload each Node. For Fan Controllers (FCx), !--- Alarm
Modules (AMx), Fabric Cards (SMx), and RPs (RPx), !--- you must wait until the reloaded
node is fully reloaded !--- before you reset the next node of the pair. But non-pairs !---
can be reloaded without waiting. RP/0/RP0/CPU0:ROUTER#hw-module node 0/RP0/CPU0 or
0/RP1/CPU0 reload
!--- This depends on which on is in Standby Mode. RP/0/RP0/CPU0:ROUTER#hw-module node
0/FC0/SP
RP/0/RP0/CPU0:ROUTER#hw-module node 0/AM0/SP
RP/0/RP0/CPU0:ROUTER#hw-module node 0/SM0/SP
!--- Do not reset the MSC and Fabric Cards at the same time. RP/0/RP0/CPU0:ROUTER#hw-module
node 0/0/CPU
```

9. Verwenden Sie die **Anzeigetafel. | inc ROM|NODE|PLIM**-Befehl, um die aktuelle ROMMON-Version zu überprüfen.

```
RP/0/RP1/CPU0:CRS-B(admin)#show diag | inc ROM|NODE|PLIM
NODE 0/0/SP : MSC(SP)
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
PLIM 0/0/CPU0 : 4OC192-POS/DPT
  ROMMON: Version 1.32(20050525:193559)  [CRS-1 ROMMON]
NODE 0/2/SP : MSC(SP)
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
PLIM 0/2/CPU0 : 8-10GbE
  ROMMON: Version 1.32(20050525:193559)  [CRS-1 ROMMON]
NODE 0/6/SP : MSC(SP)
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
PLIM 0/6/CPU0 : 16OC48-POS/DPT
  ROMMON: Version 1.32(20050525:193559)  [CRS-1 ROMMON]
NODE 0/RP0/CPU0 : RP
  ROMMON: Version 1.32(20050525:193559)  [CRS-1 ROMMON]
NODE 0/RP1/CPU0 : RP
  ROMMON: Version 1.32(20050525:193559)  [CRS-1 ROMMON]]
NODE 0/SM0/SP : FC/S
```

```
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
NODE 0/SM1/SP : FC/S
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
NODE 0/SM2/SP : FC/S
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
NODE 0/SM3/SP : FC/S
  ROMMON: Version 1.32(20050525:193402)  [CRS-1 ROMMON]
```

Hinweis: Auf CRS-8- und Fabric-Chassis setzt ROMMON die Lüfterdrehzahl auf die Standardgeschwindigkeit von 4000 U/min.

PLIM und MSC - Übersicht

Dies stellt den Paketfluss auf dem CRS-1-Router dar, und diese Begriffe werden synchron verwendet:

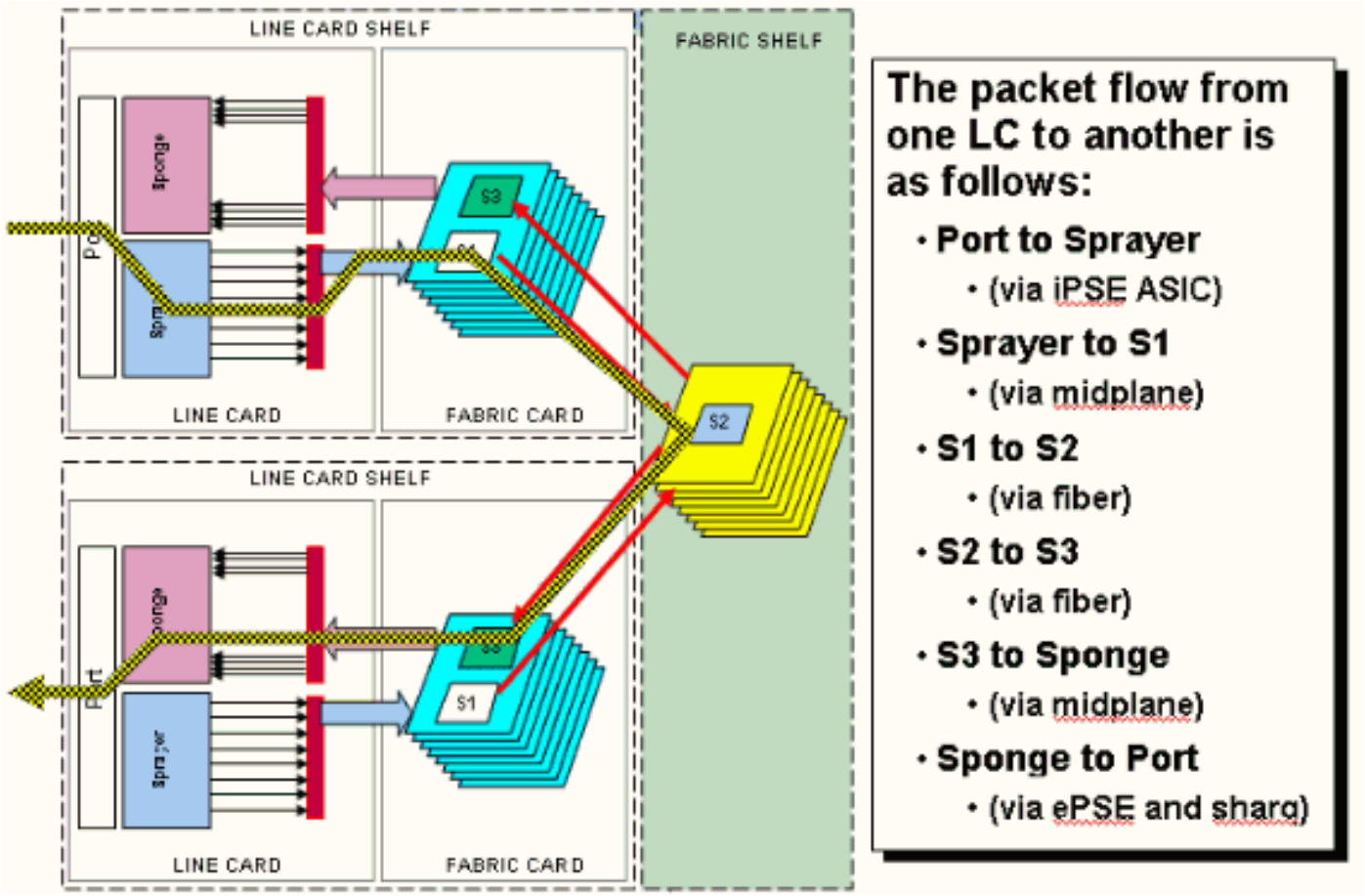
IngressQ ASIC wird auch Sprayer ASIC genannt.

FabricQ ASIC wird auch als Sponge ASIC bezeichnet.

EgressQ ASIC wird auch als Sharq ASIC bezeichnet.

SPP wird auch als PSE (Packet Switch Engine) ASIC bezeichnet.

Rx PLIM > Rx SPP > Ingress Q > Fabric > Fabric Q > Tx SPP > Egress Q > Tx PLIM (Sprayer) (Sharq)

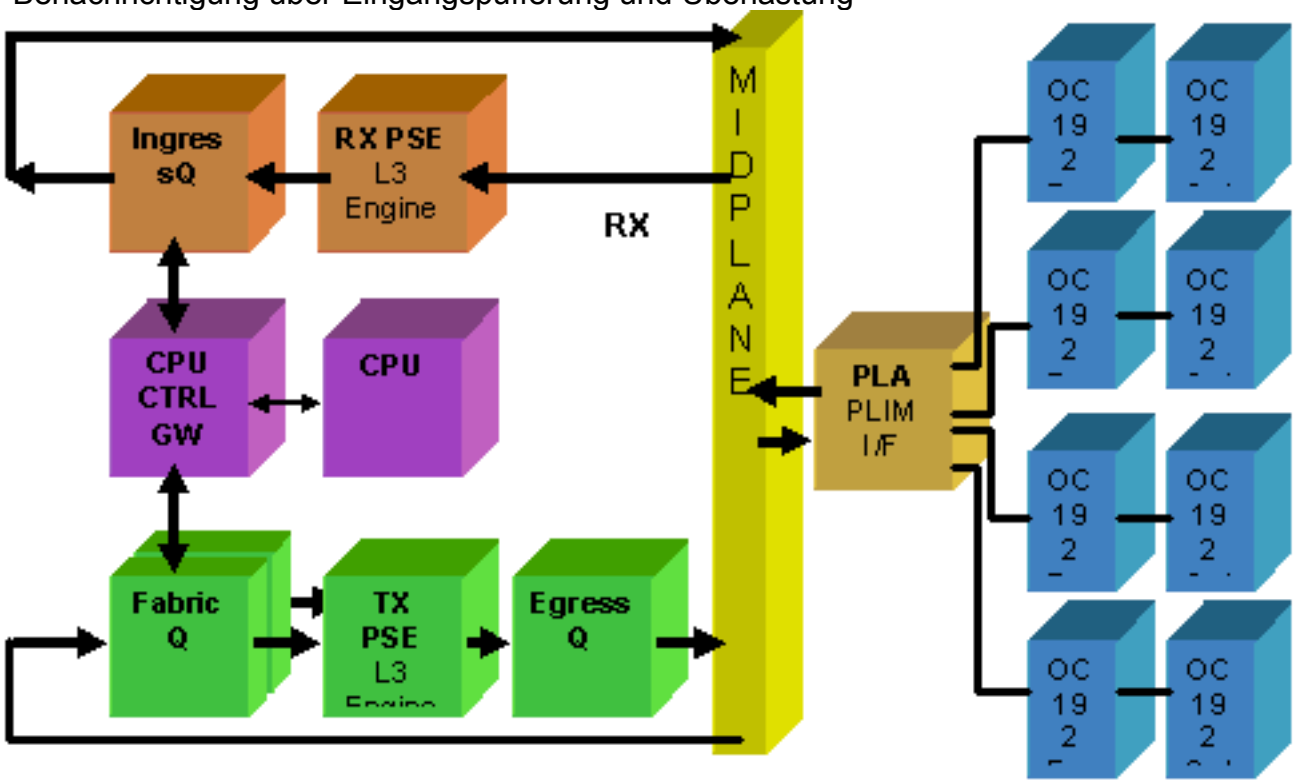


Pakete werden auf dem Physical Layer Interface Module (PLIM) empfangen.

Das PLIM enthält die physischen Schnittstellen für das MSC, mit dem es sich verbindet. PLIM und MSC sind separate Karten, die über die Chassis-Backplane verbunden sind. Daher werden die Schnittstellentypen für eine bestimmte MSC durch den Typ des PLIM definiert, mit dem sie verknüpft ist. Je nach PLIM-Typ enthält die Karte eine Reihe von ASICs, die die physischen Medien und das Framing für die Schnittstellen bereitstellen. Der Zweck der PLIM-ASICs besteht darin, die Schnittstelle zwischen dem MSC und den physischen Verbindungen bereitzustellen. Sie terminiert die Glasfaser, führt das Licht in die elektrische Konvertierung durch, terminiert das Medienframing SDH/Sonet/Ethernet/HDLC/PPP, überprüft das CRC, fügt einige Steuerungsinformationen hinzu, die als Puffer-Header bezeichnet werden, und leitet die Bits weiter, die im MSC verbleiben. Das PLIM generiert/versenkt keine HDLC- oder PPP-Keepalives. Diese werden von der CPU auf dem MSC behandelt.

Das PLIM bietet außerdem folgende Funktionen:

- MAC-Filterung für 1/10-Gigabit-Ethernet
- MAC für Eingang/Ausgang für 1/10-Gigabit-Ethernet
- VLAN-Filterung für 1/10-Gigabit-Ethernet
- VLAN für 1/10-Gigabit-Ethernet
- Benachrichtigung über Eingangspufferung und Überlastung



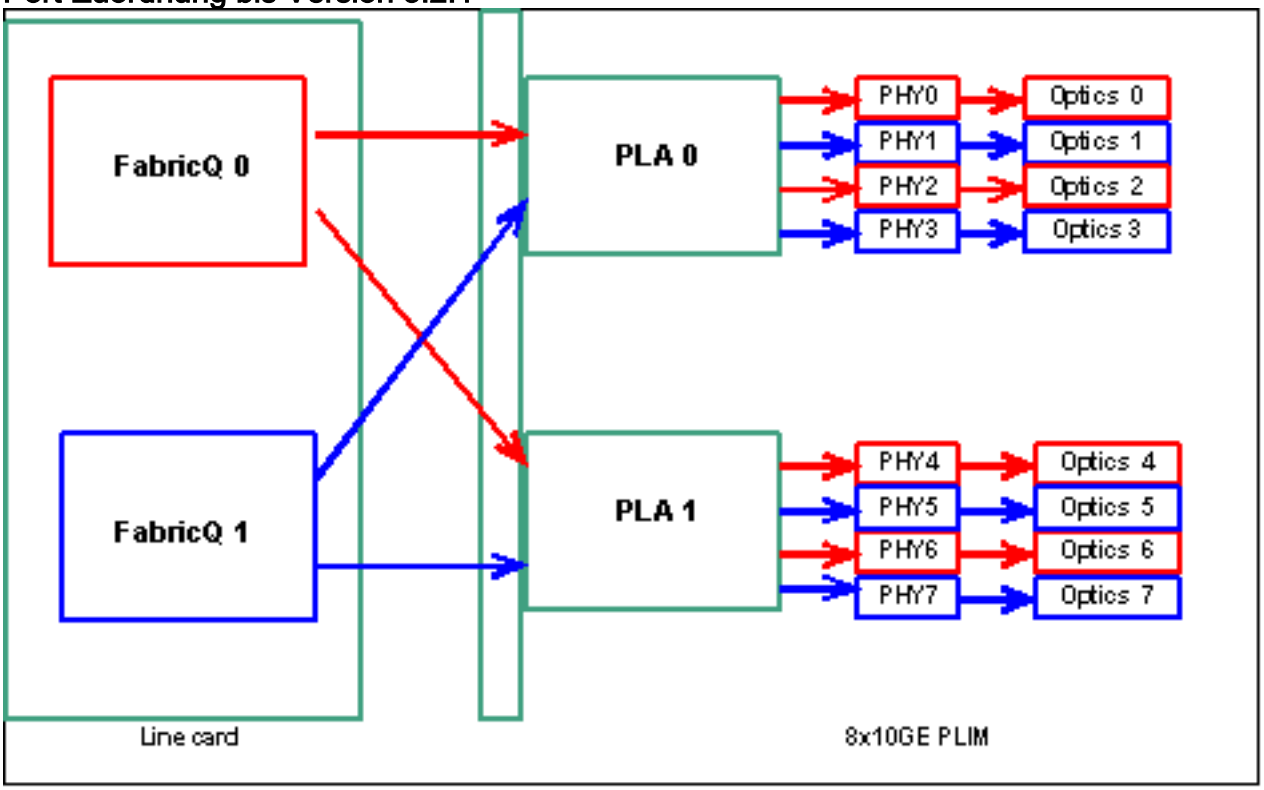
PLIM-Überbelegung

10-GE-PLIM

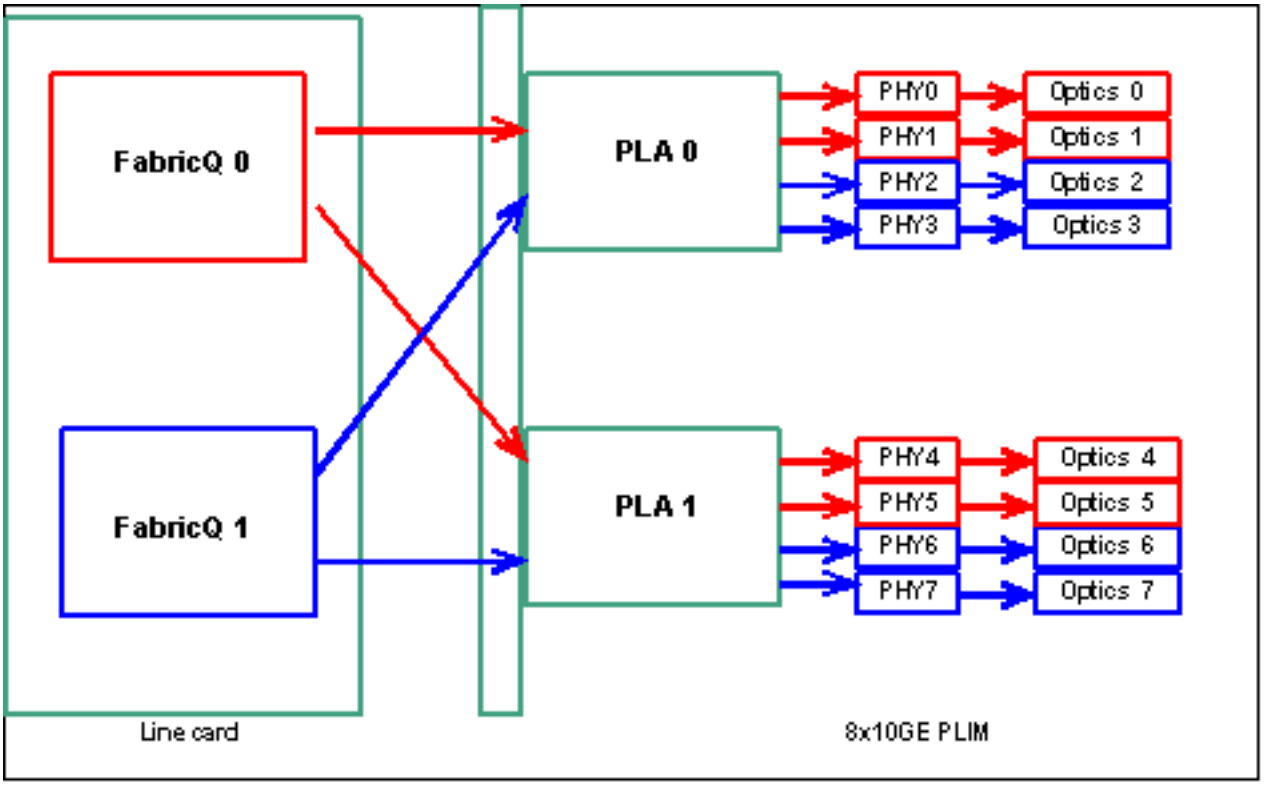
Das 8 x 10G-PLIM bietet die Möglichkeit, ca. 80 Gbit/s Datenverkehr zu terminieren, während die Weiterleitungskapazität des MSC maximal 40 Gbit/s beträgt. Wenn alle im PLIM verfügbaren Ports belegt sind, findet eine Überbelegung statt, und die QoS-Modellierung ist äußerst wichtig, um sicherzustellen, dass Premium-Datenverkehr nicht versehentlich verworfen wird. Für manche ist Überbelegung keine Option und muss vermieden werden. Dazu müssen nur vier der acht Ports verwendet werden. Darüber hinaus muss sichergestellt werden, dass die optimale Bandbreite innerhalb des MSC und PLIM für jeden der vier Ports verfügbar ist.

Hinweis: Die Port-Zuordnung ändert sich ab Version 3.2.2. Siehe diese Diagramme.

Port-Zuordnung bis Version 3.2.1



Port-Zuordnung ab Version 3.2.2



Wie bereits erwähnt, werden die physischen Ports von einem der beiden FabricQ-ASICs gewartet. Die Zuweisung von Ports zum ASIC ist statisch definiert und kann nicht geändert werden. Darüber hinaus verfügt das 8 x 10-G-PLIM über zwei PLA-ASICs. Die ersten PLA-Services Ports 0 bis 3, die zweiten Services 4 bis 7. Die Bandbreitenkapazität einer einzigen PLA auf dem 8 x 10G-PLIM beträgt ca. 24 Gbit/s. Die Switching-Kapazität eines einzelnen FabricQ ASIC beträgt ca. 62 Mpps.

Wenn Sie die Ports 0 bis 3 oder 4 bis 7 verwenden, wird die Bandbreitenkapazität der PLA (24 Gbit/s) von allen vier Ports gemeinsam genutzt, die den Gesamtdurchsatz einschränken. Wenn Sie die Ports 0, 2, 4 und 6 (bis zu 3.2.1) oder 0, 1, 4 und 5 (ab 3.2.2) füllen, da alle diese Ports von dem einen FabricQ ASIC gewartet werden, dessen Switching-Kapazität wiederum 62 Mpps beträgt, der die Durchsatzkapazität einschränkt.

Es empfiehlt sich, die Ports so zu verwenden, dass die höchste Effizienz sowohl der PLAs als auch der FabricQ-ASICs erreicht wird, um eine optimale Leistung zu erzielen.

SIP-800/SPA

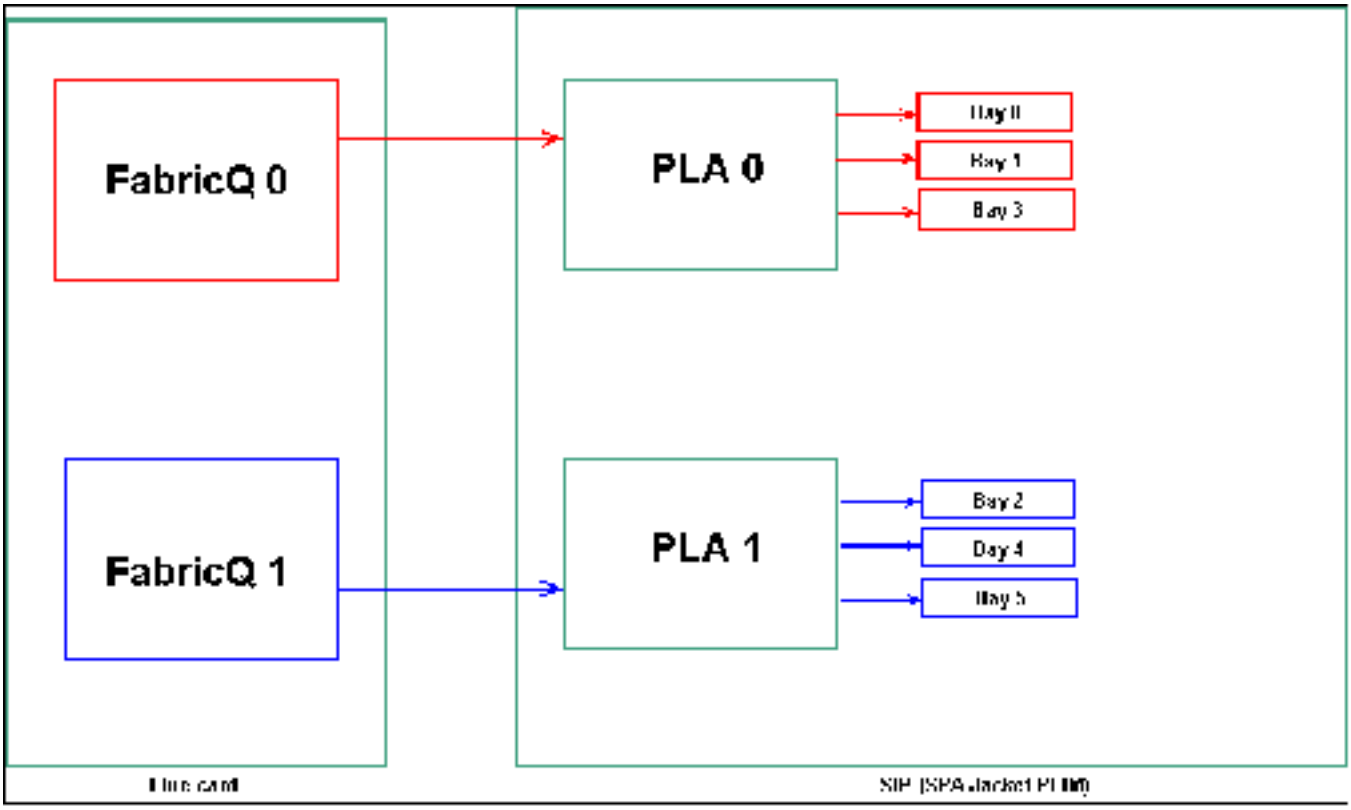
Das SIP-800-PLIM bietet die Möglichkeit, mit modularen Schnittstellenkarten zu arbeiten, die als Service Port Adapters (SPAs) bezeichnet werden. Das SIP-800 bietet 6 SPA-Einschübe mit einer theoretischen Schnittstellenkapazität von 60 Gbit/s. Die Weiterleitungskapazität des MSC beträgt maximal 40 Gbit/s. Wenn alle Einschübe des SIP-800 belegt werden sollen, ist es je nach SPA-Typ möglich, dass eine Überbelegung stattfindet und die QoS-Modellierung äußerst wichtig ist, um sicherzustellen, dass Premium-Datenverkehr nicht versehentlich verworfen wird.

Hinweis: Eine Überbelegung wird von POS-Schnittstellen nicht unterstützt. Die Platzierung des 10-

Gbit-POS-SPAs muss jedoch angemessen sein, um sicherzustellen, dass die richtige Durchsatzkapazität bereitgestellt wird. Der 10-Gbit-Ethernet-SPA wird nur in IOS-XR Version 3.4 unterstützt. Dieses SPA bietet Überbelegungsfunktionen.

Für manche ist Überbelegung keine Option und muss vermieden werden. Hierfür müssen nur vier der sechs Einschübe verwendet werden. Darüber hinaus muss sichergestellt werden, dass die optimale Bandbreite innerhalb des MSC und PLIM für jeden der vier Ports verfügbar ist.

SPA-Einschubzuordnung



Wie bereits erwähnt, werden die physischen Ports von einem der beiden FabricQ-ASICs gewartet. Die Zuweisung von Ports zum ASIC ist statisch definiert und kann nicht geändert werden. Darüber hinaus verfügt das SIP-800-PLIM über zwei PLA-ASICs. Die ersten PLA-Services Ports 0, 1 und 3, die zweiten Services 2, 4 und 5.

Die Bandbreitenkapazität einer einzigen PLA auf dem SIP-800-PLIM beträgt ca. 24 Gbit/s. Die Switching-Kapazität eines einzelnen FabricQ ASIC beträgt ca. 62 Mpps.

Wenn Sie Port 0, 1 und 3 oder die Ports 2, 4 und 5 ausfüllen, wird die Bandbreitenkapazität der PLA (24 Gbit/s) von allen drei Ports gemeinsam genutzt, die den Gesamtdurchsatz einschränken. Da ein einziger FabricQ diese Portgruppen jeweils Services, beträgt die maximale Paketrate der Portgruppe 62 Mpps. Es empfiehlt sich, die Ports so zu verwenden, dass sie die höchste Effizienz der PLAs erreichen, um eine optimale Bandbreite zu erreichen.

Empfohlene Platzierung:

	SPA-Schacht#	SPA-Schacht#	SPA-Schacht#	SPA-Schacht#
Option 1	0	1	4	5
Option 2	1	2	3	4

Wenn Sie die Karte mit mehr als vier SPAs ausfüllen möchten, wird empfohlen, eine der zuvor aufgeführten Optionen auszufüllen, die die Schnittstellen zwischen den beiden Portgruppen (0,1 & 3 & 2,4 & 5) aufteilt. Platzieren Sie dann die nächsten SPA-Module in einem der offenen Ports in den Port-Gruppen 0,1 & 3 & 2,4 & 5.

DWDM XENPACKs

Ab Version 3.2.2 können DWDM XENPACKs installiert und **einstellbare** optische Module bereitgestellt werden. Die Kühlungsanforderungen solcher XENPACK-Module erfordern einen leeren Steckplatz zwischen den installierten Modulen. Außerdem können bei Installation eines einzelnen DWDM XENPACK-Moduls maximal vier Ports verwendet werden, auch wenn es sich bei den XENPACK-Modulen nicht um DWDM-Geräte handelt. Dies hat daher direkte Auswirkungen auf die Zuordnung von FabricQ zu PLA zu Port. Diese Anforderung muss beachtet werden und wird in dieser Tabelle berücksichtigt.

Empfohlene Platzierung:

	Optische Port-Nr.	Optische Port-Nr.	Optische Port-Nr.	Optische Port-Nr.
Option 1 oder DWDM XENPACK	0	2	5	7
Option 2	1	3	4	6

Bei einer Installation ab 3.2.2 oder 3.3 vermeiden Sie die Änderung der FabricQ-Zuordnung. Für reguläre und DWDM XENPACK-Module kann daher ein einfacheres Platzierungsmuster verwendet werden.

	Optische Port-Nr.	Optische Port-Nr.	Optische Port-Nr.	Optische Port-Nr.
Option 1	0	2	4	6
Option 2	1	3	5	7

Wenn Sie die Karte mit mehr als vier Nicht-DWDM XENPACK-Ports ausfüllen möchten, wird empfohlen, eine der aufgeführten Optionen auszufüllen, bei der die optischen Schnittstellenmodule auf die beiden Portgruppen (0-3 und 4-7) aufgeteilt werden. Sie müssen dann die nächsten optischen Schnittstellenmodule in einem der offenen Ports in den 0-3- oder 4-7-Portgruppen platzieren. Wenn Sie die 0-3-Port-Gruppe für das optische Schnittstellenmodul Nr. 5 verwenden, sollten die optischen Schnittstellenmodule Nr. 6 der 4-7-Port-Gruppe angehören.

Weitere Informationen finden Sie unter DWDM XENPAK-Module.

Konfigurationsverwaltung

Die Konfiguration in IOS-XR erfolgt in einer zweistufigen Konfiguration. Die Konfiguration wird vom Benutzer in der ersten Phase eingegeben. In dieser Phase wird nur die Konfigurationssyntax von der CLI überprüft. Die in dieser Phase eingegebene Konfiguration ist nur für den Konfigurationsagentenprozess bekannt, z. B. CLI/XML. Die Konfiguration wird nicht überprüft, da sie nicht auf den sysdb-Server geschrieben wurde. Die Backend-Anwendung wird nicht benachrichtigt und kann in dieser Phase nicht auf die Konfiguration zugreifen oder sie kennen.

In der zweiten Phase wird die Konfiguration explizit vom Benutzer übernommen. In dieser Phase wird die Konfiguration auf den Sysdb-Server geschrieben, und Backend-Anwendungen überprüfen, ob die Konfigurationen und Benachrichtigungen von sysdb generiert werden. Sie können eine Konfigurationssitzung abbrechen, bevor Sie die im ersten Schritt eingegebene Konfiguration bestätigen. Daher kann nicht davon ausgegangen werden, dass alle in der ersten Phase eingegebenen Konfigurationen in Phase 2 immer übernommen werden.

Darüber hinaus kann der Betrieb und/oder die laufende Konfiguration des Routers während der ersten und zweiten Phase von mehreren Benutzern geändert werden. Daher ist jeder Test von Routern, der in Phase 1 Konfiguration und/oder Betriebsstatus ausführt, in Phase 2, in der die Konfiguration tatsächlich übernommen wurde, möglicherweise nicht gültig.

Konfigurationsdateisysteme

Configuration File System (CFS) ist ein Satz von Dateien und Verzeichnissen, die zum Speichern der Konfiguration des Routers verwendet werden. CFS wird unter dem Verzeichnis disk0:/config/ gespeichert. Dies ist der Standard-Datenträger, der auf dem RP verwendet wird. Dateien und Verzeichnisse in CFS sind für den Router intern und sollten vom Benutzer niemals geändert oder entfernt werden. Dies kann zum Verlust oder zur Beschädigung der Konfiguration führen und den Service beeinträchtigen.

Der CFS wird nach jedem Commit auf den Standby-RP überprüft. Dadurch wird die Konfigurationsdatei des Routers nach einem Failover erhalten.

Während des Router-Bootvorgangs wird die letzte aktive Konfiguration aus der in CFS gespeicherten Datenbank für den Konfigurationsbestätigung übernommen. Es ist nicht erforderlich, dass der Benutzer die aktive Konfiguration nach jedem Konfigurationstransfer manuell speichert, da dies automatisch vom Router erfolgt.

Es ist nicht ratsam, Konfigurationsänderungen vorzunehmen, während die Konfiguration beim Hochfahren angewendet wird. Wenn die Konfigurationsanwendung nicht vollständig ist, wird diese Meldung angezeigt, wenn Sie sich beim Router anmelden:

Systemkonfigurationsprozess

Die Startkonfiguration für dieses Gerät wird derzeit geladen. Dies kann einige Minuten in Anspruch nehmen. Sie werden nach Abschluss benachrichtigt. Versuchen Sie nicht, das Gerät neu zu konfigurieren, bis der Vorgang abgeschlossen ist. In seltenen Fällen ist es möglicherweise wünschenswert, die Router-Konfiguration aus einer vom Benutzer bereitgestellten ASCII-Konfigurationsdatei wiederherzustellen, anstatt die letzte aktive Konfiguration aus CFS

wiederherzustellen.

Sie können die Anwendung einer Konfigurationsdatei erzwingen, indem Sie:

```
using the “-a” option with the boot command. This option forces
    the use of the specified file only for this boot.

    rommon>boot <image> –a <config-file-path>

    setting the value of “IOX_CONFIG_FILE” boot variable to the
    path of configuration file. This forces the use of the specified file
    for all boots while this variable is set.

    rommon>IOX_CONFIG_FILE=




    rommon>boot <image>
```

Während Sie die Router-Konfiguration wiederherstellen, können ein oder mehrere Konfigurationselemente nicht wirksam werden. Alle fehlgeschlagenen Konfigurationen werden im CFS gespeichert und bis zum nächsten erneuten Laden beibehalten.

Sie können die fehlerhafte Konfiguration durchsuchen, Fehler beheben und die Konfiguration erneut anwenden.

Dies sind einige Tipps zur Behebung von Konfigurationsfehlern beim Start des Routers.

In IOX kann die Konfiguration aus drei Gründen als fehlgeschlagen klassifiziert werden:

1. Syntaxfehler - Der Parser generiert Syntaxfehler, die in der Regel darauf hindeuten, dass CLI-Befehle nicht kompatibel sind. Sie sollten die Syntaxfehler korrigieren und die Konfiguration erneut anwenden.
2. Semantische Fehler - Semantische Fehler werden von den Backend-Komponenten generiert, wenn der Konfigurationsmanager die Konfiguration beim Start des Routers wiederherstellt. Es ist wichtig zu beachten, dass cfgmgr nicht dafür verantwortlich ist, sicherzustellen, dass die Konfiguration als Teil der ausgeführten Konfiguration akzeptiert wird. Cfgmgr ist lediglich ein **Mittelsmann** und meldet nur semantische Fehler, die Backend-Komponenten generieren. Jeder Eigentümer der Backend-Komponente muss den Fehlergrund analysieren und den Fehlerursache ermitteln. Benutzer können die **beschriebenen <CLI-Befehle>** im Konfigurationsmodus ausführen, um den Besitzer des Backend-Komponentenverifizierers zu finden. Wenn z. B. der **Router bgp 217** als fehlgeschlagene Konfiguration angezeigt wird, zeigt der Befehl **description**, dass der Komponentenverifizierer die Komponente ipv4-bgp ist.

```
RP/0/0/CPU0:router#configure terminal
RP/0/0/CPU0:router(config)#describe router bgp 217
The command is defined in bgpv4_cmds.parser

Node 0/0/CPU0 has file bgpv4_cmds.parser for boot package /gsr-os-mbi-3.3.87/mbi12000-rp.vm
from gsr-rout
Package:
    gsr-rout
        gsr-rout V3.3.87[Default]  Routing Package
        Vendor : Cisco Systems
```

```
        Desc   : Routing Package
        Build  : Built on Mon Apr  3 16:17:28 UTC 2006
        Source : By ena-view3 in /vws/vpr/mletchwo/cfgmgr_33_bugfix for c2.95.3-p8
        Card(s): RP, DRP, DRPSC
        Restart information:
          Default:
            parallel impacted processes restart
Component:
    ipv4-bgp V[fwd-33/66]  IPv4 Border Gateway Protocol (BGP)

File: bgpv4_cmds.parser


User needs ALL of the following taskids:

        bgp (READ WRITE)

It will take the following actions:
  Create/Set the configuration item:
         Path: gl/ip-bgp/0xd9/gbl/edm/ord_a/running
        Value: 0x1
Enter the submode:
        bgp
RP/0/0/CPU0:router(config)#
```

3. Apply errors (Fehler anwenden): Die Konfiguration wurde erfolgreich überprüft und als Teil der ausgeführten Konfiguration akzeptiert. Der Betriebsstatus der Backend-Komponente kann jedoch aus irgendeinem Grund nicht aktualisiert werden. Die Konfiguration wird sowohl in der aktuellen Konfiguration angezeigt, da sie ordnungsgemäß verifiziert wurde, als auch als fehlerhafte Konfiguration aufgrund des Backend-Betriebsfehlers. Der Befehl **description** kann erneut auf der CLI ausgeführt werden, die nicht angewendet werden konnte, um den Besitzer der Komponentenanwendung zu finden.Gehen Sie wie folgt vor, um fehlgeschlagene Konfigurationen während des Startbetriebs zu durchsuchen und erneut anzuwenden:Für R3.2 können Operatoren dieses Verfahren verwenden, um eine fehlgeschlagene Konfiguration erneut anzuwenden:Operatoren können den Befehl **show configuration failed start** (Konfiguration anzeigen fehlgeschlagen) verwenden, um die beim Router-Start gespeicherte fehlerhafte Konfiguration zu durchsuchen.Operatoren sollten die **show configuration failed startup noerror (Konfiguration fehlgeschlagen anzeigen) ausführen | file myfailed.cfg**-Befehl ein, um die beim Start fehlgeschlagene Konfiguration in einer Datei zu speichern.Operatoren sollten in den **Konfigurationsmodus wechseln** und **Load/Commit-**Befehle verwenden, um diese fehlgeschlagene Konfiguration erneut anzuwenden:

   ```
   RP/0/0/CPU0:router(config)#load myfailed.cfg
   Loading.
   197 bytes parsed in 1 sec (191)bytes/sec
   RP/0/0/CPU0:router(config)#commit
   ```

   Für R3.3-Images können Operatoren dieses aktualisierte Verfahren verwenden:Operatoren müssen den Befehl **show configuration failed startup** (Konfiguration als fehlgeschlagen) und den Befehl **load configuration failed startup (Startkonfiguration anzeigen fehlgeschlagen)** verwenden, um fehlgeschlagene Konfigurationen zu durchsuchen und erneut anzuwenden.

```
RP/0/0/CPU0:router#show configuration failed startup
!! CONFIGURATION FAILED DUE TO SYNTAX/AUTHORIZATION ERRORS
telnet vrf default ipv4
server max-servers 5 interface POS0/7/0/3 router static
address-family ipv4 unicast
 0.0.0.0/0 172.18.189.1

!! CONFIGURATION FAILED DUE TO SEMANTIC ERRORS
```

```
router bgp 217 !!%
Process did not respond to sysmgr !
RP/0/0/CPU0:router#

RP/0/0/CPU0:router(config)#load configuration failed startup noerror
Loading.
263 bytes parsed in 1 sec (259)bytes/sec
RP/0/0/CPU0:mike3(config-bgp)#show configuration
Building configuration...
telnet vrf default ipv4 server max-servers 5 router static
address-family ipv4 unicast
  0.0.0.0/0 172.18.189.1
 !
!
router bgp 217
!
end

RP/0/0/CPU0:router(config-bgp)#commit
```

Kernel-Dumper

Standardmäßig schreibt IOS-XR einen Core Dump auf die Festplatte, wenn ein Prozess abstürzt, aber nicht, wenn der Kernel selbst abstürzt. Beachten Sie, dass diese Funktionalität für ein Multi-Chassis-System derzeit nur für das Line Card-Chassis 0 unterstützt wird. Das andere Chassis wird in einer zukünftigen Softwareversion unterstützt.

Es wird empfohlen, Kernel-Dumps sowohl für die RPs als auch für die MSCs zu aktivieren, wenn diese Konfiguration sowohl in der Standard- als auch in der Admin-Modus-Konfiguration verwendet wird:

```
exception kernel memory kernel filepath harddisk:
exception dump-tftp-route port 0 host-address 10.0.2.1/16 destination 10.0.2.1 next-hop 10.0.2.1
tftp-srvr-addr 10.0.2.1
```

Kernel-Dump-Konfiguration

Dies führt zu einem Kernel-Absturz:

1. Ein RP stürzt ab, und ein Dump wird auf die Festplatte dieses RP im Stammverzeichnis der Festplatte geschrieben.
2. Wenn ein MSC abstürzt, wird ein Dump auf die Festplatte von RP0 im Stammverzeichnis des Datenträgers geschrieben.

Dies hat keine Auswirkungen auf die RP-Failover-Zeiten, da NSF (Non-Stopp Forwarding) für die Routing-Protokolle konfiguriert wurde. Es kann einige zusätzliche Minuten dauern, bis der abgestürzte RP oder die Line Card nach einem Absturz wieder verfügbar ist, während er den Core schreibt.

Ein Beispiel für das Hinzufügen dieser Konfiguration zur Standard- und Admin-Modus-Konfiguration ist hier dargestellt. Beachten Sie, dass für die Admin-Modus-Konfiguration DRPs verwendet werden müssen.

Diese Ausgabe zeigt ein Kernel-Dump-Konfigurationsbeispiel:

```
RP/0/RP0/CPU0:crs1#configure
RP/0/RP0/CPU0:crs1(config)#exception kernel memory kernel filepat$
RP/0/RP0/CPU0:crs1(config)#exception dump-tftp-route port 0 host-$
RP/0/RP0/CPU0:crs1(config)#commit
RP/0/RP0/CPU0:crs1(config)#
RP/0/RP0/CPU0:crs1#admin
RP/0/RP0/CPU0:crs1(admin)#configure
Session                   Line        User      Date                       Lock
00000201-000bb0db-00000000  snmp        hfr-owne  Wed Apr  5 10:14:44 2006
RP/0/RP0/CPU0:crs1(admin-config)#exception kernel memory kernel f$
RP/0/RP0/CPU0:crs1(admin-config)#exception dump-tftp-route port 0$
RP/0/RP0/CPU0:crs1(admin-config)#commit
RP/0/RP0/CPU0:crs1(admin-config)#
RP/0/RP0/CPU0:crs1(admin)#
```

Sicherheit

LPTS

Local Packet Transport Services (LPTS) verarbeitet lokal bestimmte Pakete. LPTS besteht aus verschiedenen Komponenten.

1. Der Hauptprozess wird als Port-Schiedsrichterprozess bezeichnet. Er überwacht Socket-Anfragen von verschiedenen Protokollprozessen, z. B. BGP, IS-IS, und protokolliert alle Bindungsinformationen für diese Prozesse. Wenn beispielsweise ein BGP-Prozess bei der Socket-Nummer 179 abhört, ruft der PA diese Informationen von den BGP-Prozessen ab und weist diesem Prozess in einer IFIB eine Bindung zu.
2. Die IFIB ist eine weitere Komponente des LPTS-Prozesses. Es hilft dabei, ein Verzeichnis zu speichern, in dem ein Prozess eine bestimmte Portbindung überwacht. Der IFIB wird vom Port-Arbitrator-Prozess generiert und beim Port-Schiedsrichter gespeichert. Anschließend werden mehrere Teilmengen dieser Informationen generiert.Die erste Teilmenge ist ein Segment der IFIB. Dieser Abschnitt kann dem IPv4-Protokoll usw. zugeordnet werden. Anschließend werden Slices an die entsprechenden Flow-Manager gesendet, die dann mithilfe des IFIB-Slice das Paket an den richtigen Prozess weiterleiten.Die zweite Teilmenge ist eine "pre-IFIB". Sie ermöglicht dem LC, das Paket an den richtigen Prozess weiterzuleiten, wenn nur ein Prozess vorhanden ist, oder an einen geeigneten Flow Manager.
3. Flow-Manager unterstützen die weitere Verteilung der Pakete, wenn die Suche nicht trivial ist, z. B. bei mehreren Prozessen für BGP. Jeder Flow-Manager verfügt über einen oder mehrere Abschnitte der IFIB und leitet Pakete ordnungsgemäß an die entsprechenden Prozesse weiter, die dem Segment der IFIB zugeordnet sind.
4. Wenn ein Eintrag für den Zielport nicht definiert ist, kann er entweder verworfen oder an den Flow Manager weitergeleitet werden. Ein Paket wird ohne zugehörigen Port weitergeleitet, wenn eine zugeordnete Richtlinie für den Port vorliegt. Der Flow Manager hilft dann, einen neuen Sitzungseintrag zu generieren.

Wie wird ein internes Paket weitergeleitet?

Es gibt zwei Arten von Datenflüssen: Layer-2-Datenflüsse (HDLC, PPP) und Layer-4-ICMP-/PING-Datenflüsse und Routing-Datenflüsse.

1. Layer 2 HDLC/PPP - Diese Pakete werden durch die Protokoll-ID identifiziert und direkt an die CPU-Warteschlangen im Sprayer gesendet. Layer-2-Protokollpakete erhalten eine hohe Priorität und werden dann von der CPU (über den Squid) übernommen und verarbeitet. Daher werden Keepalives für Layer 2 direkt über den LC über die CPU beantwortet. So müssen Sie nicht zum RP wechseln, um Antworten zu erhalten und mit dem Thema verteilte Schnittstellenverwaltung mitzuarbeiten.
2. ICMP-Pakete (Layer 4) werden im LC empfangen und über eine Suche über das IFBI in die CPU-Warteschlangen des Sprayers gesendet. Diese Pakete werden dann (über die Squid) an die CPU gesendet und verarbeitet. Die Antwort wird dann über die Sprayer-Ausgangswarteschlangen gesendet, um durch das Fabric weitergeleitet zu werden. Dies ist der Fall, wenn eine andere Anwendung auch die Informationen benötigt (die über die Fabric repliziert werden). Sobald das Paket die Fabric durchläuft, wird es an den richtigen ausgehenden LC und die richtige Schwamm- und Steuerwarteschlange weitergeleitet.
3. Routing-Flows werden in der IFIB nachgeschlagen und dann an die Ausgabeformatwarteschlangen (8000 Warteschlangen) gesendet, von denen eine für Steuerungspakete reserviert ist. Dies ist eine nicht geformte Warteschlange und wird bei jedem vollen Laden einfach gewartet. - hohe Priorität. Das Paket wird dann über die Fabric in Warteschlangen mit hoher Priorität an eine Reihe von CPU-Warteschlangen auf dem Sponge gesendet (ähnlich den Squid-Warteschlangen auf dem Sprayer) und dann durch den richtigen Prozess, Flow Manager oder den eigentlichen Prozess verarbeitet. Eine Antwort wird über den Schwamm der Ausgangs-Linecard und dann über die Linecard gesendet. Der LC-Ausgangspuffer verfügt über eine spezielle Warteschlange für die Verarbeitung von Steuerungspaketen. Die Warteschlangen im "Sponge" sind in Pakete mit hoher Priorität, Kontrolle und niedriger Priorität pro Ausgangsport aufgeteilt.
4. Der PSE verfügt über eine Reihe von Policers, die für die Ratenbegrenzung von Layer-4-, Layer-2- und Routing-Paketen konfiguriert sind. Diese sind vordefiniert und können zu einem späteren Zeitpunkt vom Benutzer konfiguriert werden.

Eines der häufigsten Probleme bei LPTS sind Pakete, die beim Pingen des Routers verworfen werden. Die LPTS-Überwachung beschränkt diese Pakete in der Regel. Dies ist der Fall, um Folgendes zu bestätigen:

```
RP/0/RP0/CPU0:ss01-crs-1_P1#ping 192.168.3.14 size 8000 count 100
Type escape sequence to abort.
Sending 100, 8000-byte ICMP Echos to 192.168.3.14, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!
!!!!!!!!!!!!!!!.!!!!!!!!!!!!!
Success rate is 97 percent (97/100), round-trip min/avg/max = 1/2/5 ms
RP/0/RP0/CPU0:ss01-crs-1_P1#show lpts pifib hardware entry statistics location 0/5/CPU0 | excl
0/0

* - Vital; L4 - Layer4 Protocol; Intf - Interface;
DestAddr - Destination Fabric Address;
na - Not Applicable or Not Available

Local, Remote Address.Port          L4    Intf         DestAddr     Pkts/Drops
---------------------------------- ----- ------------ ------------ --- any
any   any          Punt       100/3
224.0.0.5 any                       any   PO0/5/1/0    0x3e         4/0
224.0.0.5 any                       any   PO0/5/1/1    0x3e         4/0
<further output elided>
```

IPsec

IP-Pakete sind von Natur aus unsicher. IPsec ist eine Methode zum Schutz der IP-Pakete. CRS-1 IPsec wird im Software-Weiterleitungspfad implementiert, daher wird die IPsec-Sitzung auf dem RP/DRP beendet. Es werden insgesamt 500 IPsec-Sitzungen pro CRS-1 unterstützt. Die Anzahl hängt von der CPU-Geschwindigkeit und den zugewiesenen Ressourcen ab. Hierfür gibt es keine Softwarebeschränkung. Nur lokal generierter und lokal terminierter Datenverkehr auf RP ist für die IPsec-Verarbeitung zulässig. Für die Art des Datenverkehrs kann entweder der IPsec-Transportmodus oder der Tunnel-Modus verwendet werden, wobei der erste Modus aufgrund geringerer Overhead bei der IPsec-Verarbeitung bevorzugt wird.

R3.3.0 unterstützt die Verschlüsselung von BGP und OSPFv3 über IPsec.

Weitere Informationen zur Implementierung von IPsec finden Sie im Cisco IOS XR System Security Configuration Guide.

Hinweis: IPsec erfordert Crypto-Kuchen, z. B. hfr-k9sec-p.pie-3.3.1.

Out of Band

Konsolen- und AUX-Zugriff

Die CRS-1 RP/SCs verfügen über einen Konsolen- und AUX-Port, der für Out-of-Band-Management verfügbar ist, sowie einen Ethernet-Management-Port für Out-of-Band über IP.

Die Konsole und der AUX-Port jedes RP/SCGE, zwei pro Chassis, können an einen Konsolenserver angeschlossen werden. Das bedeutet, dass das Einzel-Chassis-System vier Konsolenports benötigt, und die Multi-Chassis-Systeme 12 Ports plus zwei weitere Ports für die Supervisor Engines auf dem Catalyst 6504-E benötigen.

Die AUX-Port-Verbindung ist wichtig, da sie Zugriff auf den IOS-XR-Kernel bietet und eine Systemwiederherstellung ermöglichen kann, wenn dies nicht über den Konsolenport möglich ist. Der Zugriff über den AUX-Port ist nur für lokal im System definierte Benutzer und nur dann möglich, wenn der Benutzer über einen Root-System- oder Cisco Support-Level-Zugriff verfügt. Darüber hinaus muss für den Benutzer ein **geheimes** Kennwort definiert sein.

Virtueller Terminalzugriff

Telnet und Secure Shell (SSH) können verwendet werden, um das CRS-1 über die vty-Ports zu erreichen. Standardmäßig sind beide deaktiviert, und der Benutzer muss sie explizit aktivieren.

Hinweis: IPsec erfordert Crypto-Kuchen, z. B. hfr-k9sec-p.pie-3.3.1.

Generieren Sie zuerst die RSA- und DSA-Schlüssel, wie in diesem Beispiel gezeigt, um SSH zu aktivieren:

```
RP/0/RP1/CPU0:Crs-1#crypto key zeroize dsa
% Found no keys in configuration.
RP/0/RP1/CPU0:Crs-1#crypto key zeroize rsa
% Found no keys in configuration.

RP/0/RP1/CPU0:Crs-1#crypto key generate rsa general-keys
The name for the keys will be: the_default
```

```
  Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose
Keypair.
  Choosing a key modulus greater than 512 may take a few minutes.

How many bits in the modulus [1024]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]

RP/0/RP1/CPU0:Crs-1#crypto key generate dsa
The name for the keys will be: the_default
  Choose the size of your DSA key modulus. Modulus size can be 512, 768, or 1024 bits. Choosing
a key modulus
How many bits in the modulus [1024]:
Generating DSA keys ...
Done w/ crypto generate keypair
[OK]

!--- VTY access via SSH & telnet can be configured as shown here. vty-pool default 0 4 ssh
server ! line default secret cisco users group root-system users group cisco-support exec-
timeout 30 0 transport input telnet ssh ! ! telnet ipv4 server
```

Zugehörige Informationen

- Router-Unterstützung
- Technischer Support und Dokumentation für Cisco Systeme