# Pratiques opérationnelles recommandées relatives à CRS-1 et IOS XR

## Contenu



## Introduction

Ce document vous aide à comprendre les éléments suivants :

- Processus et threads
- Structure CRS-1
- Plan de contrôle
- Rommon et Monlib
- Module d'interface de couche physique (PLIM) et carte de service modulaire (MSC)
- Gestion de la configuration
- Sécurité
- Hors bande
- Protocole de gestion de réseau simple (SNMP)

# Conditions préalables

## Conditions requises

Cisco vous recommande de connaître Cisco IOS® XR.

## Components Used

Les informations contenues dans ce document sont basées sur les versions de matériel et de logiciel suivantes :

- Logiciel Cisco IOS XR
- CRS-1

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

## Conventions

Pour plus d'informations sur les conventions utilisées dans ce document, reportez-vous à Conventions relatives aux conseils techniques Cisco.

# Présentation de Cisco IOS XR

Cisco IOS XR est conçu pour évoluer. Le noyau est une architecture Microkernel qui fournit uniquement des services essentiels tels que la gestion des processus, la planification, les signaux et les temporisateurs. Tous les autres services, tels que les systèmes de fichiers, les pilotes, les piles de protocoles et les applications, sont considérés comme des gestionnaires de ressources et s'exécutent dans un espace utilisateur protégé par la mémoire. Ces autres services peuvent être ajoutés ou supprimés au moment de l'exécution, selon la conception du programme. L'empreinte de Microkernel est seulement de 12 kb. Le Microkernel et le système d'exploitation sous-jacent proviennent de QNX Software Systems, et s'appelle Neutrino. QNX se spécialise dans la conception de systèmes d'exploitation en temps réel. Le micronoyau est préemptif et le planificateur est basé sur la priorité. Cela garantit que la commutation de contexte entre les processus est très rapide et que les threads de priorité la plus élevée ont toujours accès au CPU lorsque nécessaire. Voici quelques-uns des avantages de Cisco IOS XR. Mais le plus grand avantage est la conception héritée des communications interprocessus au sein du coeur des systèmes d'exploitation.

Neutrino est un système d'exploitation de passage de messages, et les messages sont les moyens de base des communications interprocessus entre tous les threads. Lorsqu'un serveur spécifique souhaite fournir un service, il crée un canal d'échange de messages. Les clients se connectent au canal des serveurs en mappant directement au descripteur de fichier approprié afin d'utiliser le service. Toutes les communications entre le client et le serveur sont effectuées par le même mécanisme. C'est un énorme avantage pour un super ordinateur, ce que CRS-1 est. Considérez ces éléments lorsqu'une opération de lecture locale est effectuée sur un noyau UNIX standard :

- Le logiciel s'interrompt dans le noyau.
- Le noyau est envoyé dans le système de fichiers.
- Les données sont reçues.

Considérez ces éléments dans le cas distant :

- Le logiciel s'interrompt dans le noyau.
- Le noyau envoie NFS.
- NFS appelle le composant réseau.
- Le composant réseau est distribué à distance.
- NFS est appelé.
- Le noyau envoie le système de fichiers.

La sémantique de la lecture locale et de la lecture distante ne sont pas les mêmes. Les arguments et les paramètres pour le verrouillage et la définition des autorisations de fichier sont différents.

Considérez le cas local de QNX :

- Le logiciel s'interrompt dans le noyau.
- Le noyau effectue le transfert de messages dans le système de fichiers.

Considérez le cas non local :

- Le logiciel s'interrompt dans le noyau.
- Le noyau va dans QNET, qui est le mécanisme de transport IPC.
- QNET entre dans le noyau.
- Le noyau envoie le système de fichiers.

Toutes les sémantiques qui concernent le passage d'arguments et les paramètres du système de fichiers sont identiques. Tout a été découplé au niveau de l'interface IPC, ce qui permet de séparer complètement le client et le serveur. Cela signifie que tout processus peut s'exécuter n'importe où à n'importe quel moment. Si un processeur de routage particulier est trop occupé pour traiter les demandes, vous pouvez facilement migrer ces services vers un processeur différent qui s'exécute sur un DRP. Un super ordinateur qui exécute différents services sur différents processeurs répartis sur plusieurs noeuds et qui peut facilement communiquer avec n'importe quel autre noeud. L'infrastructure est en place afin de fournir la possibilité d'évoluer. Cisco a utilisé cet avantage et écrit des logiciels supplémentaires qui s'intègrent aux principales opérations du noyau de transmission de messages qui permet au routeur CRS de s'étendre à des milliers de noeuds, où un noeud, dans ce cas un processeur, exécute une instance du système d'exploitation, qu'il s'agisse d'un processus de routage (RP), d'un processeur de routage distribué (DRP), d'une carte de services modulaire (MSC) ou d'un processeur de commutation (SP).

## Processus et threads

Dans les limites de Cisco IOS XR, un processus est une zone de mémoire protégée qui contient

un ou plusieurs threads. Du point de vue des programmeurs, les threads font le travail, et chacun termine un chemin d'exécution logique afin d'exécuter une tâche spécifique. La mémoire dont les threads ont besoin au cours du flux d'exécution appartient au processus dans lequel ils opèrent, protégé des autres threads de processus. Un thread est une unité d'exécution, avec un contexte d'exécution qui inclut une pile et des registres. Un processus est un groupe de threads qui partagent un espace d'adressage virtuel, bien qu'un processus puisse contenir un thread unique mais plus souvent plus. Si un autre thread d'un autre processus tente d'écrire dans la mémoire de votre processus, le processus offensant est supprimé. S'il y a plusieurs threads qui fonctionnent dans votre processus, ce thread a accès à la même mémoire dans votre processus, et par conséquent est capable de remplacer les données d'un autre thread. Exécutez les étapes d'une procédure afin de maintenir la synchronisation avec les ressources afin d'empêcher ce thread dans le même processus.

Un thread utilise un objet appelé Mutual Exclusion (MUTEX) afin d'assurer l'exclusion mutuelle aux services. Le thread qui a le MUTEX est le thread qui peut écrire dans une zone particulière de mémoire comme exemple. Les autres threads qui n'ont pas le MUTEX ne peuvent pas. Il existe également d'autres mécanismes permettant d'assurer la synchronisation avec les ressources, à savoir les sémaphores, les variables conditionnelles ou les condvars, les barrières et les sleepons. Ils ne sont pas abordés ici, mais ils fournissent des services de synchronisation dans le cadre de leurs fonctions. Si vous comparez les principes présentés ici à Cisco IOS, Cisco IOS est un processus unique qui exécute de nombreux threads, avec tous les threads qui ont accès au même espace mémoire. Mais Cisco IOS appelle ces processus de threads.

## États de processus et de thread

Dans Cisco IOS XR, il y a des serveurs qui fournissent les services et des clients qui utilisent les services. Un processus particulier peut avoir un certain nombre de threads qui fournissent le même service. Un autre processus peut avoir un certain nombre de clients qui peuvent nécessiter un service particulier à tout moment. L'accès aux serveurs n'est pas toujours disponible, et si un client demande l'accès à un service, il est installé et attend que le serveur soit libre. Dans ce cas, on dit que le client est bloqué. Il s'agit d'un modèle de serveur client bloquant. Le client peut être bloqué parce qu'il attend une ressource telle qu'un MUTEX, ou parce que le serveur n'a pas encore répondu.

Émettez une commande **show process ospf** afin de vérifier l'état des threads dans le processus ospf :

```
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
```

Notez que le processus ospf reçoit un ID de tâche (JID), qui est 250. Cela ne change jamais sur un routeur en cours d'exécution et généralement sur une version particulière de Cisco IOS XR. Dans le processus ospf, il y a cinq threads chacun avec leur propre ID de thread (TID). L'espace de pile est indiqué pour chaque thread, la priorité de chaque thread et son état.

## Passage de messages synchrones

Il est mentionné précédemment que QNX est un système d'exploitation de transmission de messages. Il s'agit en fait d'un système d'exploitation de passage de messages synchrones. La plupart des problèmes de système d'exploitation se reflètent dans la messagerie synchrone. Il n'est pas dit que le passage de messages synchrones entraîne des problèmes, mais plutôt que le symptôme du problème se reflète dans le passage de messages synchrones. Comme il est synchrone, les informations de cycle de vie ou d'état sont facilement accessibles à l'opérateur CRS-1, ce qui facilite le processus de dépannage. Le cycle de vie des messages est similaire à celui-ci :

- Un serveur crée un canal de message.
- Un client se connecte au canal d'un serveur (analogue à posix open).
- Un client envoie un message à un serveur (MsgSend) et attend une réponse et bloque.
- Le serveur reçoit (MsgReceive) un message d'un client, traite le message et répond au client.
- Le client débloque et traite la réponse du serveur.

Ce modèle client-serveur de blocage est le message synchrone qui passe. Cela signifie que le client envoie un message et bloque. Le serveur reçoit le message, le traite, répond au client, puis le client se débloque. Voici les détails spécifiques :

- Le serveur attend à l'état RECEIVE.
- Le client envoie un message au serveur et devient BLOQUÉ.
- Le serveur reçoit le message et débloque, en cas d'attente dans l'état de réception.
- Le client passe à l'état REPLY.
- Le serveur passe à l'état EXÉCUTIF.
- Le serveur traite le message.
- Le serveur répond au client.
- Le client se débloque.

Émettez la commande **show process** afin de voir dans quels états le client et les serveurs sont.

```
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
```

## États de processus et de processus bloqués

Émettez la commande **show process bloqué** afin de voir quel processus est en état bloqué.

```
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#
```

Le passage de messages synchronisés vous permet de suivre facilement le cycle de vie des communications interprocessus entre les différents threads. À tout moment, un thread peut être dans un état spécifique. Un état bloqué peut être le symptôme d'un problème. Cela ne signifie pas que si un thread est en état bloqué alors il y a un problème, alors n'émettez pas la commande **show process block** et ouvrez un dossier avec l'assistance technique Cisco. Les threads bloqués sont également très normaux.

Notez le résultat précédent. Si vous regardez le premier thread de la liste, notez qu'il s'agit du ksh et que sa réponse est bloquée sur devc-conaux. Le client, le ksh dans ce cas, a envoyé un message au processus devc-conaux, le serveur, qui est devc-conaux, bloque la réponse ksh jusqu'à ce qu'il réponde. Ksh est le shell UNIX que quelqu'un utilise sur la console ou le port AUX. Ksh attend l'entrée de la console, et s'il n'y en a aucune parce que l'opérateur ne tape pas, alors il reste bloqué jusqu'à ce qu'il traite une entrée. Après le traitement, ksh retourne à la réponse bloquée sur devc-conaux.

C'est normal et n'illustre pas un problème. Le fait est que les threads bloqués sont normaux, et cela dépend de la version XR, du type de système que vous avez, de ce que vous avez configuré et de qui fait ce qui modifie la sortie de la commande **show process block**. L'utilisation de la commande **show process block** est un bon moyen de commencer à dépanner les problèmes de type de système d'exploitation. En cas de problème, par exemple, le CPU est élevé, alors utilisez la commande précédente afin de voir si quelque chose semble en dehors de la normale.

Comprendre ce qui est normal pour votre routeur en fonctionnement. Ceci fournit une ligne de base que vous pouvez utiliser comme comparaison lorsque vous dépannez des cycles de vie de processus.

Àtout moment, un thread peut être dans un état particulier. Ce tableau répertorie les états suivants :

| Si l'État est : | Le thread est : |
|---|---|
| MORT | Mort. Le noyau attend de libérer les ressources de threads. |
| EXÉCUTION | Exécution active sur un processeur |
| PRÊT | Non exécuté sur un processeur mais prêt à être exécuté |
| ARRÊTÉ | Suspendu (signal SIGSTOP) |
| ENVOYER | En attente de réception d'un message par un serveur |
| RECEVOIR | En attente d'envoi d'un message par un client |
| RÉPONSE | En attente de réponse d'un serveur à un message |

| | |
|---|---|
| PILE | En attente de l'allocation d'une pile supplémentaire |
| PAGE D'ATTENTE | Attente du gestionnaire de processus pour résoudre un problème de page |
| SIGSUSPENDRE | En attente d'un signal |
| SIGWAITINFO | En attente d'un signal |
| NANOSLER | Dormir pendant un certain temps |
| MUTEX | En attente d'acquisition d'un MUTEX |
| CONDVAR | En attente de signalisation d'une variable conditionnelle |
| REJOINDRE | Attente de la fin d'un autre thread |
| ENTRÉE | Attente d'une interruption |
| SEM | En attente d'acquisition d'un sémaphore |

## Processus importants et leurs fonctions

Cisco IOS XR comporte de nombreux processus. Voici quelques-uns des plus importants dont les fonctions sont expliquées ici.

### Moniteur système WatchDog (WDSysmon)

Il s'agit d'un service fourni pour la détection des blocages de processus et des conditions de mémoire insuffisante. Une mémoire insuffisante peut se produire en raison d'une fuite de mémoire ou d'autres circonstances extérieures. Un blocage peut être le résultat d'un certain nombre de conditions telles que des blocages de processus, des boucles infinies, des verrouillages du noyau ou des erreurs de planification. Dans n'importe quel environnement multithread, le système peut se trouver dans un état appelé blocage, ou simplement blocage. Un blocage peut se produire lorsqu'un ou plusieurs threads ne peuvent pas continuer en raison d'un conflit de ressources. Par exemple, le thread A peut envoyer un message au thread B tandis que le thread B envoie simultanément un message au thread A. Les deux threads s'attendent l'un l'autre et peuvent être à l'état d'envoi bloqué, et les deux threads attendent indéfiniment. Il s'agit d'un cas simple qui implique deux threads, mais si un serveur est responsable d'une ressource utilisée par de nombreux threads est bloqué sur un autre thread, alors les nombreux threads qui demandent l'accès à cette ressource peuvent être envoyés bloqués en attente sur le serveur.

Des blocages peuvent se produire entre quelques threads, mais peuvent avoir un impact sur d'autres threads. Les blocages sont évités par une bonne conception de programme, mais quelle que soit la magnificence de la conception et de l'écriture d'un programme. Parfois, une séquence particulière d'événements dépendant de données avec des temporisations spécifiques peut provoquer un blocage. Les blocages ne sont pas toujours déterministes et sont généralement très difficiles à reproduire. WDSysmon a de nombreux threads avec un qui s'exécute à la priorité la plus élevée prise en charge par Neutrino, 63. L'exécution à la priorité 63 garantit que le thread obtient le temps processeur dans un environnement de planification préemptive basé sur les priorités. WDSysmon fonctionne avec la fonctionnalité de surveillance du matériel et surveille les processus logiciels qui recherchent les conditions de blocage. Lorsque de telles conditions sont détectées, WDSysmon recueille des informations supplémentaires sur la condition, peut coredump le processus ou le noyau, écrire dans syslogs, exécuter des scripts et tuer les processus bloqués. En fonction de la gravité du problème, il peut initier un commutateur de

processeur de routage afin de maintenir le fonctionnement du système.

```
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
```

Le processus WDSysmon comporte neuf threads. Quatre sont exécutés à la priorité 10, les quatre autres à la priorité 11, 19, 20 et 63. Lorsqu'un processus est conçu, le programmeur considère soigneusement la priorité que chaque thread du processus doit être donnée. Comme nous l'avons vu précédemment, le planificateur est basé sur la priorité, ce qui signifie qu'un thread de priorité supérieure prévient toujours un thread de priorité inférieure. La priorité 63 est la priorité la plus élevée à laquelle un thread peut s'exécuter, qui est le thread 7 dans ce cas. Thread 7 est le thread d'observation, le thread qui suit les bogues du processeur. Il doit être exécuté à une priorité plus élevée que les autres threads qu'il observe, sinon il pourrait ne pas avoir la possibilité de s'exécuter du tout, ce qui l'empêche des étapes qu'il a été conçu pour effectuer.

## Netio

Dans Cisco IOS, il existe le concept de commutation rapide et de commutation de processus. La commutation rapide utilise le code CEF et se produit au moment de l'interruption. La commutation de processus utilise ip_input, qui est le code de commutation IP, et est un processus planifié. Sur les plates-formes supérieures, la commutation CEF est effectuée dans le matériel et ip_input est planifié sur le processeur. L'équivalent d'ip_input dans Cisco IOS XR est 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
```

## Processus de services de groupe (GSP)

Il y a un besoin de communication dans n'importe quel super-ordinateur avec plusieurs milliers de noeuds qui exécutent chacun sa propre instance du noyau. Sur Internet, une à plusieurs communications sont efficacement effectuées via des protocoles de multidiffusion. GSP est le protocole de multidiffusion interne utilisé pour IPC dans CRS-1. Le protocole GSP fournit une communication de groupe fiable à plusieurs, sans connexion avec la sémantique asynchrone. Cela permet à GSP de s'adapter aux milliers de noeuds.

```
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
```

## Téléchargeur de contenu en masse BCDL

BCDL est utilisé afin de multicast fiable aux différents noeuds tels que les RP et les MSC. Il utilise le SPG comme transport sous-jacent. BCDL garantit **la** livraison des messages. BCDL compte un agent, un producteur et un consommateur. L'agent est le processus qui communique avec le producteur afin de récupérer et de mettre en mémoire tampon les données avant ses multidiffusions aux consommateurs. Le producteur est le processus qui produit les données que tout le monde veut, et le consommateur est le processus intéressé à recevoir les données fournies par le producteur. BCDL est utilisé lors des mises à niveau du logiciel Cisco IOS XR.

## Messagerie légère (LWM)

LWM est une forme de messagerie créée par Cisco et conçue pour créer une couche d'abstraction entre les applications qui intertraitent communiquent entre elles et avec Neutrino, dans le but d'assurer l'indépendance du système d'exploitation et de la couche transport. Si Cisco souhaite changer le fournisseur du système d'exploitation de QNX à quelqu'un d'autre, une couche d'abstraction entre les fonctions rudimentaires du système d'exploitation sous-jacent permet d'éliminer la dépendance au système d'exploitation et facilite le portage vers un autre système d'exploitation. LWM fournit une remise de message garantie synchrone, qui, comme le passage de message natif Neutrino, provoque le blocage de l'expéditeur jusqu'à ce que le destinataire réponde.

LWM assure également la remise asynchrone des messages via des impulsions de 40 bits. Les messages asynchrones sont envoyés de manière asynchrone, ce qui signifie que le message est mis en file d'attente et que l'expéditeur ne bloque pas, mais qu'il n'est pas reçu par le serveur de manière asynchrone, mais lorsque le serveur interroge le prochain message disponible. LWM est structuré comme client/serveur. Le serveur crée un canal qui lui donne une **oreille** pour écouter les messages et s'assied pendant un temps où la boucle fait un message de réception d'écoute sur le canal, qu'il vient de créer. Lorsqu'un message arrive, il se débloque et obtient un identificateur de client, qui est en fait la même chose que l'ID de réception du message reçu. Le serveur effectue ensuite un certain traitement, puis répond par message à l'identificateur client.

Du côté client, un message se connecte. Il reçoit un identificateur auquel il se connecte, puis envoie un message et est bloqué. Une fois le traitement terminé, le serveur répond et le client est débloqué. C'est pratiquement la même chose que le message natif Neutrinos qui passe, donc la couche d'abstraction est très mince.

LWM est conçu avec un nombre minimum d'appels système et de commutateurs de contexte pour des performances élevées et est la méthode préférée de IPC dans l'environnement Cisco IOS XR.

## Évangile

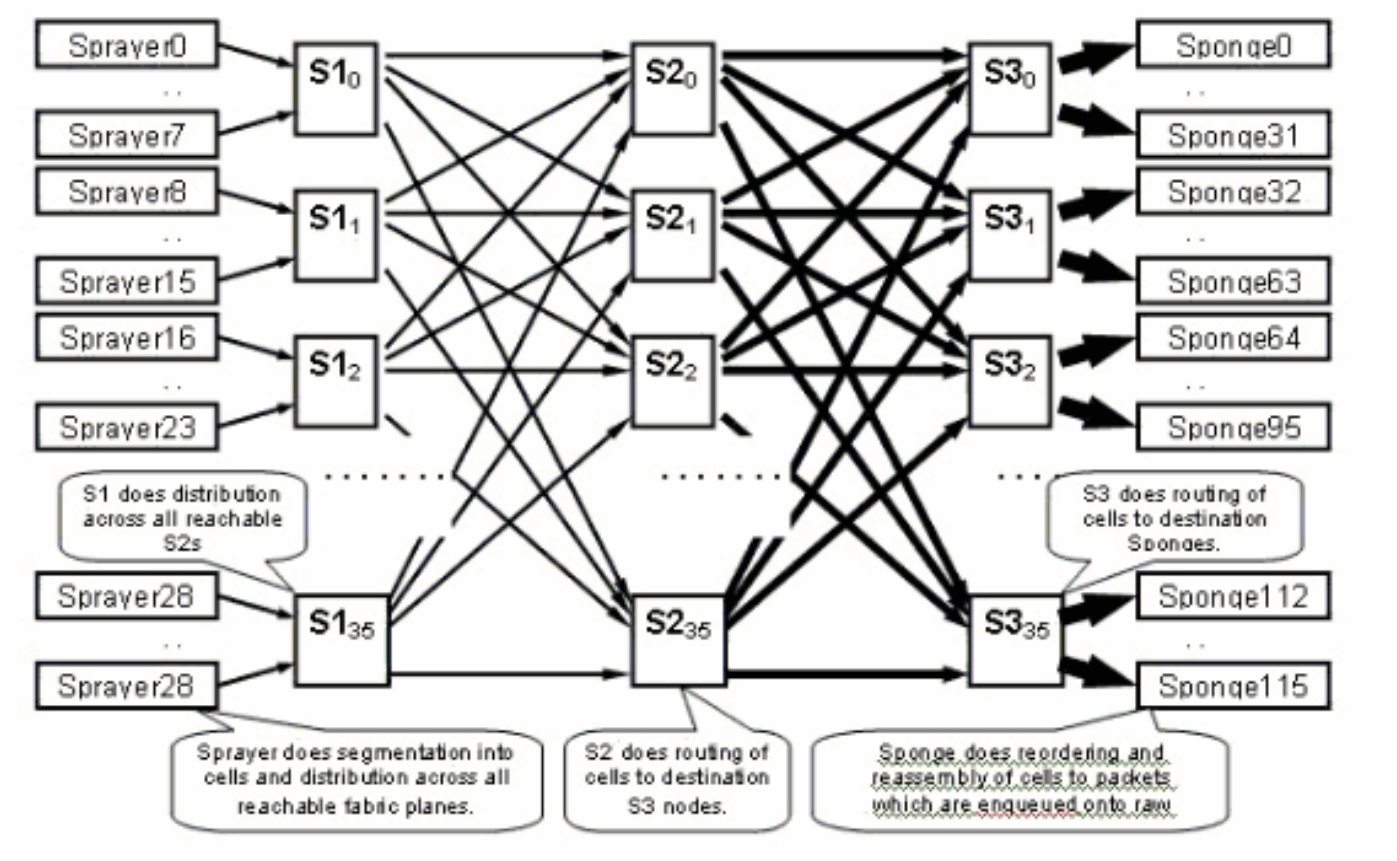
Au niveau le plus fondamental, le système de surveillance de l'environnement est chargé d'avertir lorsque les paramètres physiques, tels que la température, la tension, la vitesse du ventilateur, etc., ne sont pas compris dans les plages de fonctionnement et d'arrêter le matériel qui approche des niveaux critiques où le matériel peut être endommagé. Il surveille régulièrement chaque capteur matériel disponible, compare la valeur mesurée aux seuils spécifiques à la carte et déclenche des alarmes si nécessaire pour accomplir cette tâche. Un processus persistant, lancé lors de l'initialisation du système, qui interroge périodiquement tous les capteurs matériels, par exemple la tension, la température et la vitesse du ventilateur, dans le châssis et fournit ces données aux clients de gestion externes. En outre, le processus périodique compare les lectures des capteurs aux seuils d'alarme et publie des alertes environnementales dans la base de données système pour une action ultérieure par le gestionnaire de pannes. Si les relevés des capteurs sont dangereusement hors limites, le processus de surveillance de l'environnement peut entraîner l'arrêt de la carte.

# Présentation du fabric CRS-1

- Fabric multiétape : topologie à trois niveaux
- Routage dynamique au sein du fabric pour réduire l'encombrement
- Basé sur les cellules : Cellules de 136 octets, données utiles de 120 octets
- Contrôle de flux pour améliorer l'isolation du trafic et minimiser les exigences de mise en mémoire tampon dans le fabric
- Vitesse de l'étape à l'étape de la livraison
- Deux casques de trafic pris en charge (monodiffusion et multidiffusion)
- Deux priorités de trafic prises en charge par lot (élevé et faible)
- Prise en charge des groupes de multidiffusion de fabric 1M (FGID)
- Tolérance de panne économique : Redondance N+1 ou N+k utilisant des plans de fabric par opposition à 1+1 à un coût considérablement plus élevé

Lorsque vous utilisez le mode châssis unique, les cartes de base S1, S2 et S3 sont situées sur les mêmes cartes de fabric. Cette carte est également appelée **carte S123**. Dans une configuration multichâssis, le commutateur S2 est séparé et se trouve sur le châssis FCC (Fabric Card Chassis). Cette configuration nécessite deux cartes de fabric pour former un plan, une carte S2 et une carte S13. Chaque MSC se connecte à huit plans de fabric afin d'assurer la redondance de sorte que si vous perdez un ou plusieurs plans, votre fabric continue de passer le trafic bien que le trafic agrégé, qui peut passer par le fabric, soit plus faible. Le CRS peut toujours fonctionner en ligne pour la plupart des tailles de paquets avec seulement sept plans. La contre-pression est envoyée sur le tissu sur un plan impair et pair. Il n'est pas recommandé d'exécuter un système avec moins de deux plans, dans un plan impair et pair. Toute configuration inférieure à deux plans n'est pas prise en charge.

## Le plan de fabric




Le schéma précédent représente un plan. Vous devez multiplier ce diagramme par huit. Cela signifie que l'asic de pulvérisateur (ingressq) d'une LC se connecte à 8 S1 (1 S1 par plan). Le commutateur S1 de chaque plan de fabric se connecte à 8 pulvérisateurs :

- les 8 LC supérieures du châssis
- les 8 LC inférieurs

Il y a 16 S1 par châssis LC à 16 logements : 8 pour les LC supérieures (1 par plan) + 8 pour les LC inférieures.

Sur un seul châssis à 16 logements, une carte de fabric S123 comporte 2 S1, 2 S2 et 4 S3. Cela fait partie du calcul de la vitesse du fabric. Il y a deux fois plus de trafic, qui peut quitter le fabric que le trafic qui peut y entrer. Il y a actuellement deux éponges (fabriq) par CL comparativement à 1 pulvérisateur. Cela permet la mise en mémoire tampon sur le LC de sortie lorsque plusieurs LC d'entrée surchargent un LC de sortie. Le LC de sortie est capable d'absorber cette bande passante supplémentaire à partir du fabric.

## Surveillance du fabric

Disponibilité et connectivité du plan :

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

Vérifiez si les plans reçoivent/transmettent des cellules et si certaines erreurs s'incrémentent :

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

Les acronymes de la commande précédente :

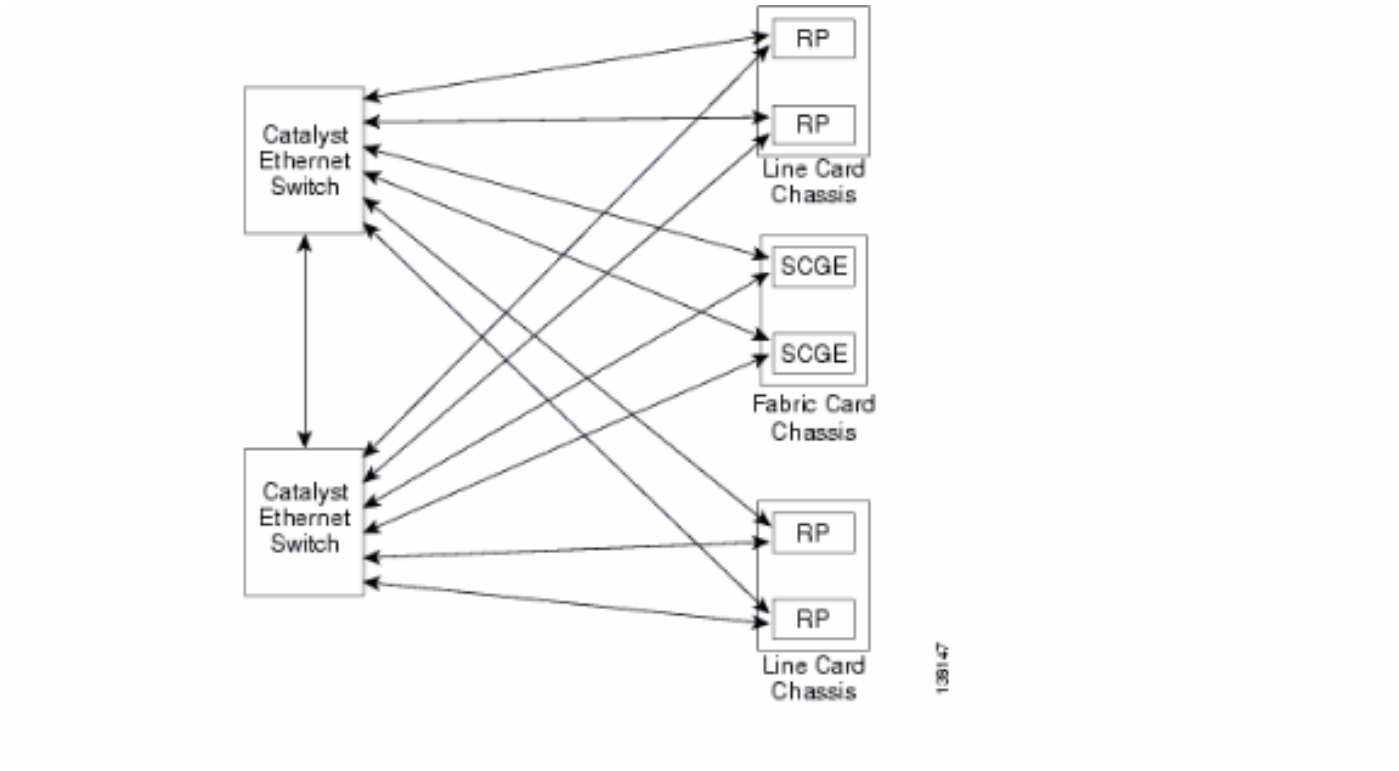
- CE : erreur corrigée
- UCE : erreur non corrigible
- PE : erreur de parité

Ne vous inquiétez pas s'ils remarquent quelques erreurs, car cela peut se produire au démarrage. Les champs ne doivent pas être incrémentés au moment de l'exécution. Si tel est le cas, cela peut indiquer un problème dans le tissu. Émettez cette commande afin d'obtenir une ventilation des erreurs par plan de fabric :

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

# Présentation du plan de contrôle

La connectivité du plan de contrôle entre le châssis de carte de ligne et le châssis de fabric se fait actuellement via des ports Gigabit Ethernet sur les RP (LCC) et SCGE (FCC). L'interconnexion entre les ports est assurée par une paire de commutateurs Catalyst 6500, qui peut être connectée via deux ports Gigabit Ethernet ou plus.




## Configuration du Catalyst 6500

Cette configuration est recommandée pour les commutateurs Catalyst utilisés pour le plan de contrôle multichâssis :

- Un VLAN unique est utilisé sur tous les ports.
- Tous les ports fonctionnent en mode d’accès (pas de jonction).
- Le protocole Spanning Tree 802.1w/s est utilisé pour la prévention des boucles.
- Deux liaisons ou plus sont utilisées afin d'interconnecter les deux commutateurs et STP est utilisé pour empêcher les boucles. La canalisation n'est pas recommandée.
- Les ports qui se connectent au RP CRS-1 et au SCGE utilisent le mode pré-standard, car

IOS-XR ne prend pas en charge les normes 802.1s.
- UDLD doit être activé sur les ports qui se connectent entre les commutateurs et entre les commutateurs et le RP/SCGE.
- UDLD est activé par défaut sur CRS-1.

Référez-vous à Mise en route du logiciel Cisco IOS XR sur un système multi-étagère pour plus d'informations sur la façon de configurer un Catalyst 6500 dans un système multi-étagère.

## Gestion du plan de contrôle multichâssis

Le châssis Catalyst 6504-E, qui fournit la connectivité du plan de contrôle pour le système multichâssis, est configuré pour les services de gestion suivants :

- Gestion intrabande via le port gigabit 1/2, qui se connecte à un commutateur LAN à chaque PoP. L'accès est uniquement autorisé pour une petite plage de sous-réseaux et de protocoles.
- NTP est utilisé afin de définir l'heure système.
- Syslogging est exécuté sur les hôtes standard.
- L'interrogation et les interruptions SNMP peuvent être activées pour les fonctions critiques.

**Remarque :** Aucune modification ne doit être apportée au Catalyst en cours d'utilisation. Des tests préalables doivent être effectués sur toute modification prévue et il est fortement recommandé de le faire pendant une période de maintenance.

Voici un exemple de configuration de gestion :

```
#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 et Monlib

Cisco monlib est un programme exécutable qui est stocké sur le périphérique et chargé dans la mémoire vive pour être exécuté par ROMMON. ROMMON utilise monlib pour accéder aux fichiers du périphérique. Les versions de ROMMON peuvent être mises à niveau et doivent l'être sur recommandation de l'assistance technique Cisco. La dernière version de ROMMON est 1.40.

## Instructions de mise à niveau

Procédez comme suit :

1. Téléchargez les binaires ROMMON à partir de Cisco CRS-1 ROMMON (clients enregistrés uniquement).
2. Déballez le fichier TAR et copiez les fichiers 6 BIN dans le répertoire racine CRS de 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. Utiliser la **boîte de dialogue show | inc ROM|NODE|PLIM** afin de voir la version rommon actuelle.
   ```
   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. Passez en mode ADMIN et utilisez la commande **upgrade rommon a all disk0** afin de mettre à niveau le ROMMON.

```
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. Quittez le mode ADMIN et entrez **show log | inc « OK, ROMMON A »** et assurez-vous que tous les noeuds ont bien été mis à niveau. Si l'un des noeuds échoue, revenez à l'étape 4 et reprogrammez.

```
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. Passez en mode ADMIN et utilisez la commande **upgrade rommon b all disk0** afin de mettre à niveau le ROMMON.

```
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. Quittez le mode ADMIN et entrez **show log | inc « OK, ROMMON B »** et assurez-vous que tous les noeuds ont bien été mis à niveau. Si l'un des noeuds échoue, revenez à l'étape 4 et reprogrammez.

```
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. La commande **upgrade** brûle une section réservée spéciale de bootflash avec le nouveau ROMMON. Mais le nouveau ROMMON reste inactif jusqu'à ce que la carte soit rechargée. Ainsi, lorsque vous rechargez la carte, le nouveau ROMMON est actif. Réinitialisez chaque noeud un par un ou réinitialisez simplement l'ensemble du routeur afin d'effectuer cette opération.

```
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. Utiliser la **boîte de dialogue show | inc ROM|NODE|PLIM** afin de vérifier la version actuelle de ROMMON.

```
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]
```

**Remarque :** sur le CRS-8 et le châssis de fabric, ROMMON définit également les vitesses de

ventilateur sur la vitesse par défaut de 4 000 tr/min.

## Présentation du PLIM et du MSC

Il s'agit du flux de paquets sur le routeur CRS-1 et les termes suivants sont utilisés de manière interchangeable :

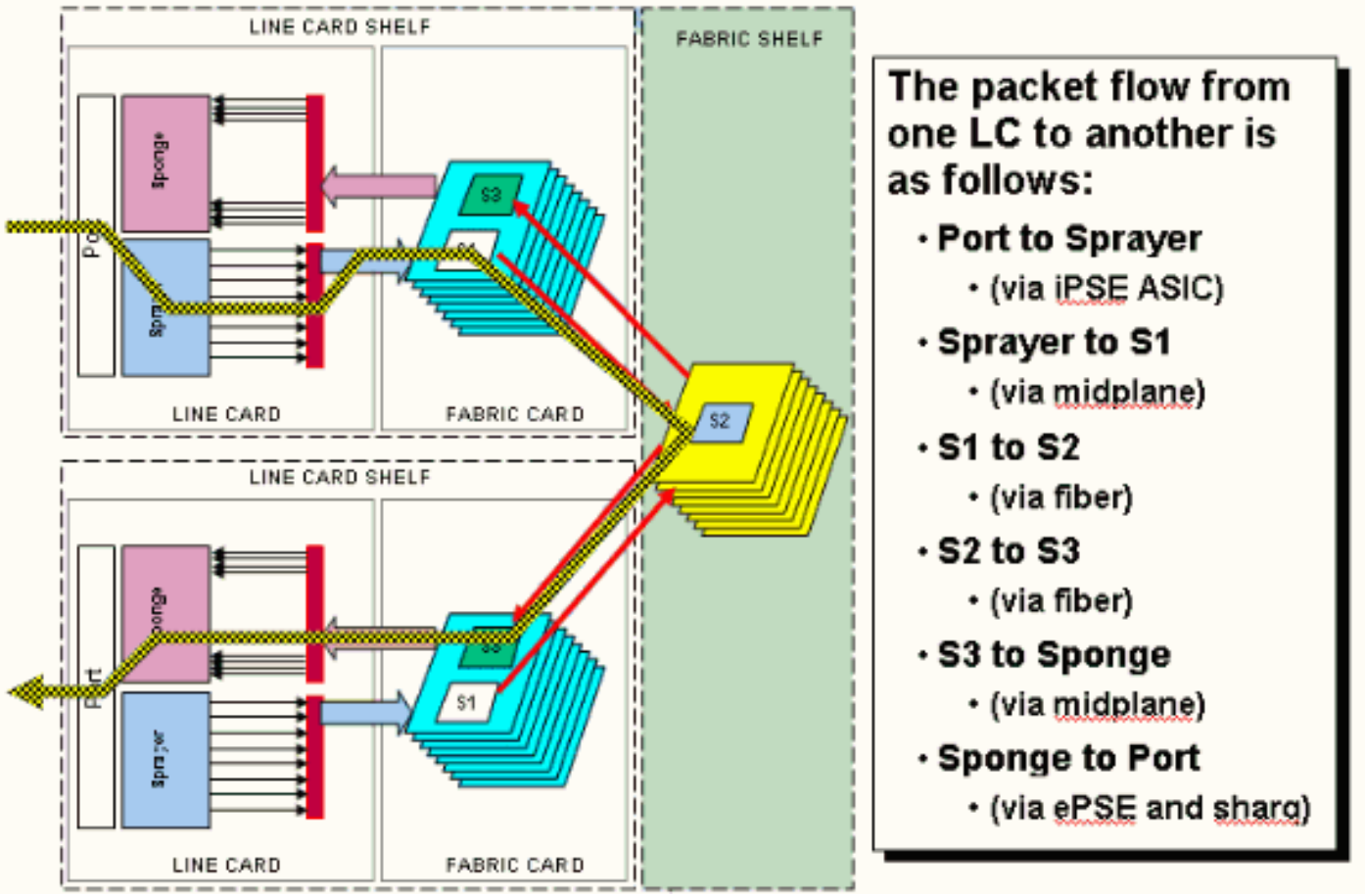
**L'ASIC InboundQ est également appelé ASIC Sprayer.**

**L'ASIC FabricQ est également appelé l'ASIC Sponge.**

**L'ASIC EentryQ est également appelé ASIC Sharq.**

**SPP est également appelé ASIC PSE (Packet Switch Engine).**

**Rx PLIM > Rx SPP > Insent Q > Fabric > Fabric Q > Tx SPP > Eient Q > Tx PLIM (Sprayer) (Éponge) (Sharq)**



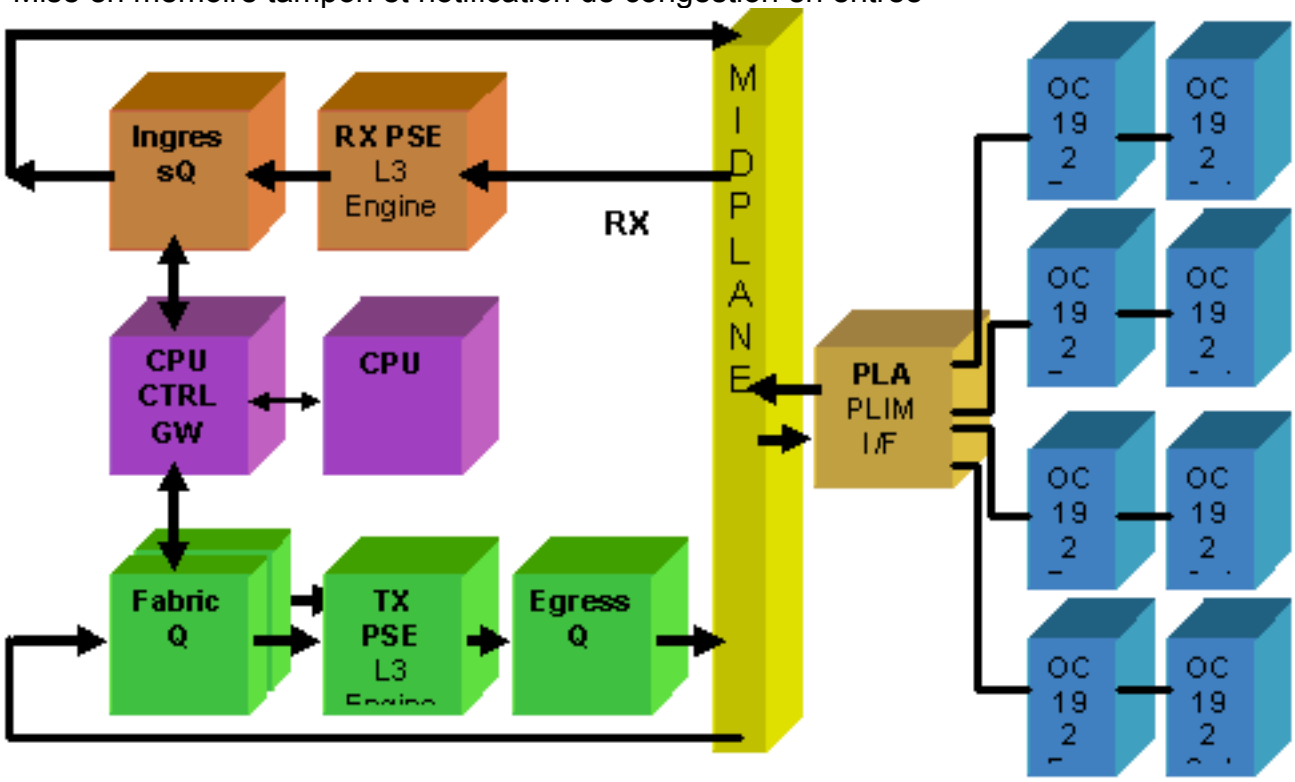

Les paquets sont reçus sur le module PLIM (Physical Layer Interface Module).

Le PLIM contient les interfaces physiques du MSC avec lequel il se joint. Les cartes PLIM et MSC sont des cartes séparées connectées via le fond de panier du châssis. Par conséquent, les types d'interface pour un MSC particulier sont définis par le type de PLIM avec lequel il a été associé. Selon le type de PLIM, la carte contient un certain nombre d'ASIC qui fournissent le support physique et le tramage des interfaces. L'objectif des ASIC PLIM est de fournir l'interface entre le MSC et les connexions physiques. Il termine la fibre, effectue la conversion de la lumière en électricité, termine le tramage du support en SDH/Sonet/Ethernet/HDLC/PPP, vérifie le CRC,

ajoute des informations de contrôle appelées en-tête de tampon et transmet les bits qui restent sur le MSC. Le module PLIM ne source pas/n'enfonce pas les keepalives HDLC ou PPP. Celles-ci sont gérées par le processeur sur le MSC.

Le PLIM assure également les fonctions suivantes :

- Filtrage MAC pour 1/10 Gigabit Ethernet
- Comptabilisation MAC en entrée/sortie pour 1/10 Gigabit Ethernet
- Filtrage VLAN pour 1/10 Gigabit Ethernet
- Comptabilisation VLAN pour 1/10 Gigabit Ethernet
- Mise en mémoire tampon et notification de congestion en entrée



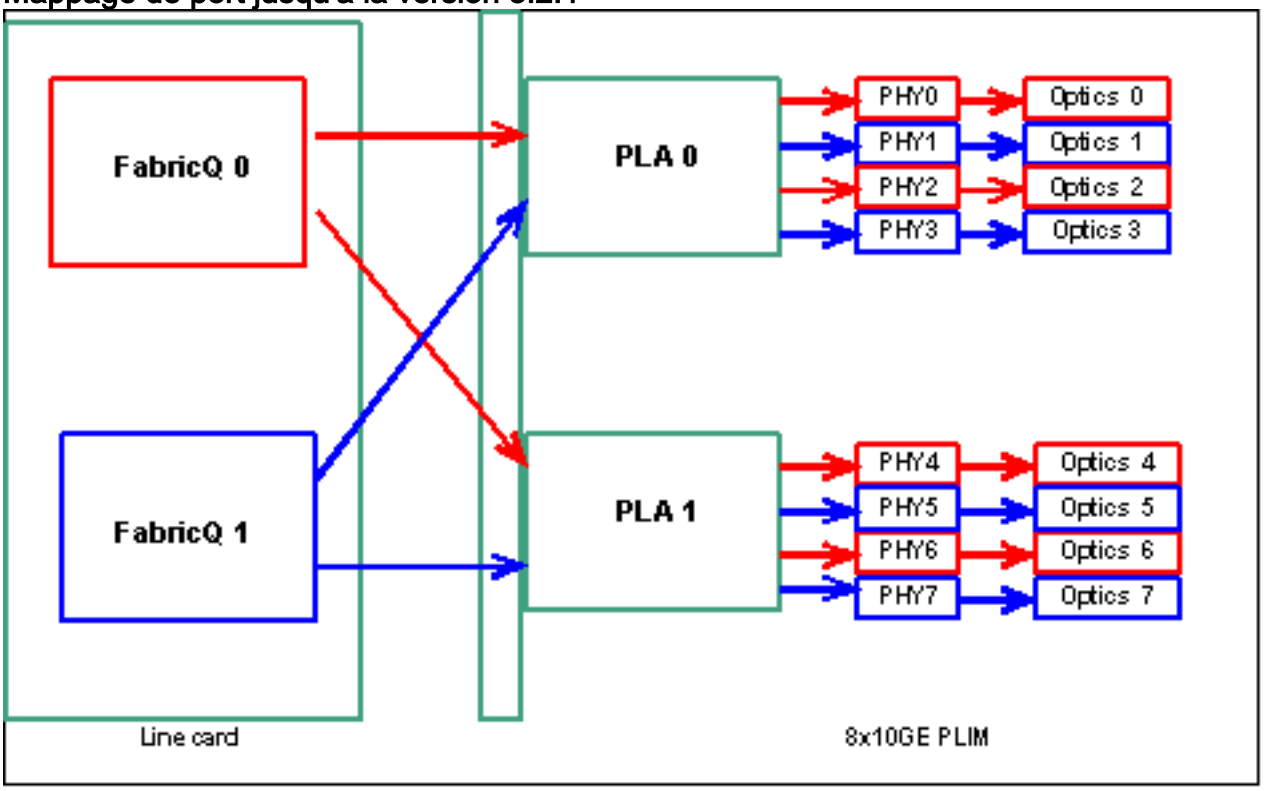

## Surabonnement PLIM

### PLIM 10 GE

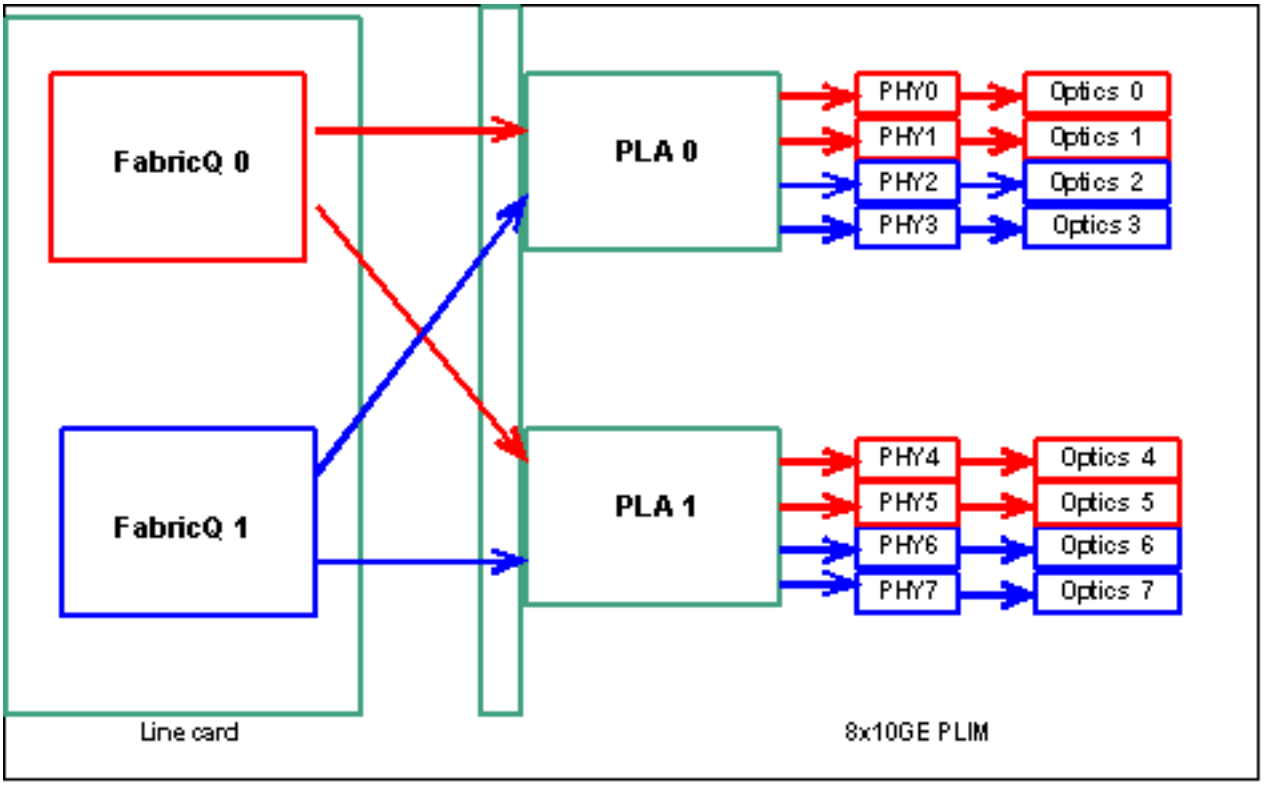
Le module PLIM 8 X 10G offre la possibilité de terminer environ 80 Gbits/s de trafic tandis que la capacité de transfert du MSC est maximale de 40 Gbits/s. Si tous les ports disponibles sur le module PLIM sont renseignés, la sursouscription se produit et la modélisation de la qualité de service devient extrêmement importante pour s'assurer que le trafic premium n'est pas abandonné par inadvertance. Pour certains, la sursouscription n'est pas une option et doit être évitée. Seuls quatre des huit ports doivent être utilisés pour cela. En outre, il faut veiller à ce que la bande passante optimale au sein du MSC et du PLIM soit disponible pour chacun des quatre ports.

**Remarque :** le mappage des ports change à partir de la version 3.2.2. Voir ces diagrammes.

Mappage de port jusqu'à la version 3.2.1




Mappage des ports à partir de la version 3.2.2




Comme indiqué précédemment, les ports physiques sont desservis par l'un des deux ASIC FabricQ. L'affectation des ports à l'ASIC est définie de manière statique et ne peut pas être modifiée. En outre, le module PLIM 8 X 10G dispose de deux ASIC PLA. Les premiers ports de services PLA sont compris entre 0 et 3, les seconds entre 4 et 7. La capacité de bande passante d'un seul PLA sur le PLIM 8 X 10 G est d'environ 24 Gbit/s. La capacité de commutation d'un seul ASIC FabricQ est d'environ 62 Mpps.

Si vous remplissez le port 0 à 3 ou les ports 4 à 7, la capacité de bande passante du PLA (24 Gbits/s) est partagée entre les quatre ports qui limitent le débit global. Si vous renseignez les ports 0, 2, 4 et 6 (jusqu'à 3.2.1) ou 0, 1, 4 et 5 (3.2.2 à partir de) car tous ces ports sont desservis par l'ASIC FabricQ, dont la capacité de commutation est de 62 Mpps, une fois de plus, ce qui limite la capacité de débit.

Il est préférable d'utiliser les ports de manière à obtenir la plus grande efficacité des PLA et des ASIC FabricQ afin d'obtenir des performances optimales.

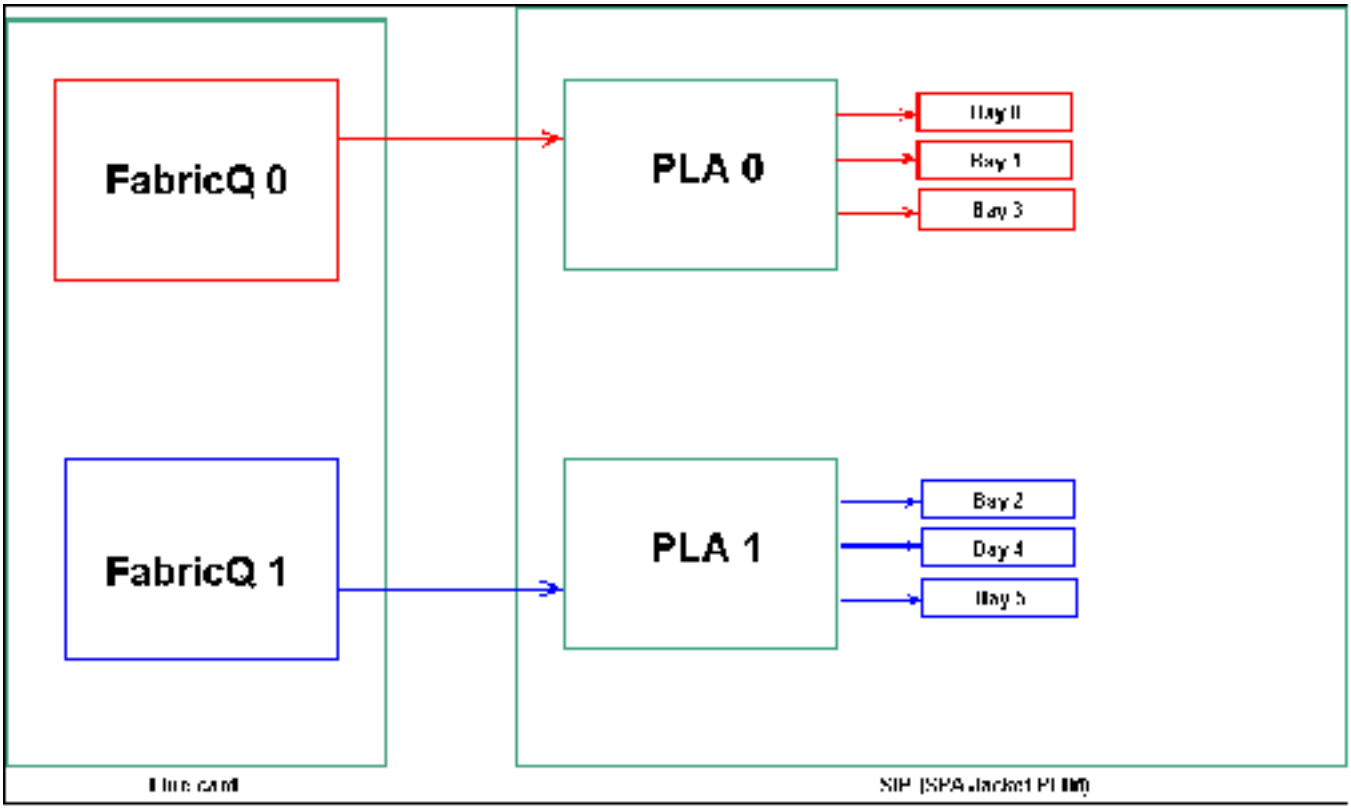
## SIP-800/SPA

Le module PLIM SIP-800 offre la possibilité de fonctionner avec des cartes d'interface modulaires appelées adaptateurs de port de service (SPA). Le SIP-800 fournit 6 baies SPA avec une capacité d'interface théorique de 60 Gbit/s. La capacité de transfert du MSC est maximale de 40 Gbit/s. Si toutes les baies du SIP-800 devaient être remplies, alors, en fonction du type de SPA, il est possible que la sursouscription ait lieu et que la modélisation de la qualité de service devienne extrêmement importante afin de s'assurer que le trafic premium ne soit pas abandonné par inadvertance.

**Remarque :** La sursouscription n'est pas prise en charge avec les interfaces POS. Mais le positionnement du SPA POS 10 Gbit doit être approprié afin de garantir une capacité de débit correcte. Le SPA Ethernet 10 Gb est uniquement pris en charge dans IOS-XR version 3.4. Ce SPA offre des fonctionnalités de surabonnement.

Pour certains, la sursouscription n'est pas une option et doit être évitée. Seules quatre des six baies doivent être utilisées pour cela. En outre, il convient de veiller à ce que la bande passante optimale au sein du MSC et du PLIM soit disponible pour chacun des quatre ports.

[Mappage de baie SPA]




Comme indiqué précédemment, les ports physiques sont desservis par l'un des deux ASIC FabricQ. L'affectation des ports à l'ASIC est définie de manière statique et ne peut pas être modifiée. En outre, le module PLIM SIP-800 dispose de deux ASIC PLA. Les premiers ports de services PLA 0, 1 et 3, les seconds services 2, 4 et 5.

La capacité de bande passante d'un seul PLA sur le PLIM SIP-800 est d'environ 24 Gbit/s. La capacité de commutation d'un seul ASIC FabricQ est d'environ 62 Mpps.

Si vous renseignez les ports 0, 1 et 3 ou les ports 2, 4 et 5, la capacité de bande passante du PLA (24 Gbits/s) est partagée entre les trois ports qui limitent le débit global. Étant donné qu'un seul FabricQ dessert chacun ces groupes de ports, le débit maximal de paquets du groupe de ports est de 62 Mpps. Il est préférable d'utiliser les ports de manière à obtenir la plus grande efficacité des PLA afin d'obtenir une bande passante optimale.

## Position suggérée :

| | N° de baie SPA | N° de baie SPA | N° de baie SPA | N° de baie SPA |
|---|---|---|---|---|
| Option 1 | 0 | 1 | 4 | 5 |
| Option 2 | 1 | 2 | 3 | 4 |

Si vous voulez remplir la carte avec plus de quatre SPA, il est recommandé de compléter l'une des options précédemment répertoriées, qui répartit les interfaces entre les deux groupes de ports (0,1 & 3 & 2,4 & 5). Vous devez ensuite placer les prochains modules SPA dans l'un des ports ouverts dans les groupes de ports 0,1 & 3 & 2,4 & 5.

## XENPACK DWDM

Àpartir de la version 3.2.2, les XENPACK DWDM peuvent être installés et fournir des modules optiques **réglables**. Les exigences de refroidissement de ces modules XENPACK exigent qu'il y ait un emplacement vide entre les modules installés. En outre, si un seul module DWDM XENPACK est installé, un maximum de quatre ports peut être utilisé, même si les modules XENPACK ne sont pas des périphériques DWDM. Cela a donc un impact direct sur le mappage entre FabricQ et PLA et les ports. Il faut tenir compte de cette exigence et elle est prise en compte dans ce tableau.

**Position suggérée :**

| | N° de port optique | N° de port optique | N° de port optique | N° de port optique |
|---|---|---|---|---|
| Option 1 ou DWDM XENPACK | 0 | 2 | 5 | 7 |
| Option 2 | 1 | 3 | 4 | 6 |

Pour une installation 3.2.2 ou ultérieure ou 3.3, évitez la modification du mappage FabricQ. Un modèle de placement plus simple peut donc être utilisé pour les modules XENPACK standard et DWDM.

| | N° de port optique | N° de port optique | N° de port optique | N° de port optique |
|---|---|---|---|---|
| Option 1 | 0 | 2 | 4 | 6 |
| Option 2 | 1 | 3 | 5 | 7 |

Si vous voulez remplir la carte avec plus de quatre ports XENPACK non DWDM, il est recommandé de compléter l'une des options répertoriées, qui répartit les modules d'interface optique entre les deux groupes de ports (0-3 et 4-7). Vous devez ensuite placer les prochains modules d'interface optique dans l'un des ports ouverts dans les groupes de ports 0-3 ou 4-7. Si vous utilisez le groupe de ports 0-3 pour le module d'interface optique n°5, les modules d'interface optique n°6 doivent être placés dans le groupe de ports 4-7.

Référez-vous à Modules XENPAK DWDM pour plus de détails.

# Gestion de la configuration

La configuration dans IOS-XR s'effectue par le biais d'une configuration en deux étapes, la configuration est entrée par l'utilisateur dans la première étape. Il s'agit de l'étape où seule la syntaxe de configuration est vérifiée par l'interface de ligne de commande. La configuration entrée dans cette étape est connue uniquement du processus de l'agent de configuration, par exemple, CLI/XML. La configuration n'est pas vérifiée car elle n'est pas écrite sur le serveur sysdb. L'application principale n'est pas notifiée et ne peut pas accéder à la configuration ou en avoir connaissance à cette étape.

Dans la deuxième étape, la configuration est explicitement validée par l'utilisateur. Au cours de cette étape, la configuration est écrite sur le serveur sysdb, les applications principales vérifient que les configurations et les notifications sont générées par sysdb. Vous pouvez annuler une session de configuration avant de valider la configuration entrée dans la première étape. Par conséquent, il n'est pas sûr de supposer que toute configuration entrée dans l'étape 1 est toujours validée dans l'étape 2.

En outre, le fonctionnement et/ou la configuration en cours du routeur peuvent être modifiés par plusieurs utilisateurs au cours des étapes 1 et 2. Ainsi, tout test de routeur qui exécute la configuration et/ou l'état opérationnel dans la première étape peut ne pas être valide dans la deuxième étape où la configuration est réellement validée.

## Systèmes de fichiers de configuration

Le système de fichiers de configuration (CFS) est un ensemble de fichiers et de répertoires utilisés pour stocker la configuration du routeur. CFS est stocké sous le répertoire disk0:/config/, qui est le support par défaut utilisé sur le RP. Les fichiers et les répertoires de CFS sont internes au routeur et ne doivent jamais être modifiés ou supprimés par l'utilisateur. Cela peut entraîner une perte ou une corruption de la configuration et affecter le service.

Le CFS est coché sur le RP de secours après chaque validation. Cela permet de conserver le fichier de configuration du routeur après un basculement.

Au démarrage du routeur, la dernière configuration active est appliquée à partir de la base de données de validation de configuration stockée dans CFS. Il n'est pas nécessaire que l'utilisateur enregistre manuellement la configuration active après chaque validation de configuration, car cela est fait automatiquement par le routeur.

Il n'est pas recommandé d'apporter des modifications à la configuration lorsque celle-ci est appliquée au démarrage. Si l'application de configuration n'est pas terminée, ce message s'affiche lorsque vous vous connectez au routeur :

## Processus de configuration du système

La configuration de démarrage de ce périphérique est en cours de chargement. Cela peut prendre quelques minutes. Vous êtes averti à la fin. N'essayez pas de reconfigurer le périphérique tant que ce processus n'est pas terminé. Dans quelques rares cas, il peut être souhaitable de restaurer la configuration du routeur à partir d'un fichier de configuration ASCII fourni par l'utilisateur au lieu de restaurer la dernière configuration active à partir de CFS.

Vous pouvez forcer l'application d'un fichier de configuration en procédant comme suit :

```
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>
```

Lorsque vous restaurez la configuration du routeur, un ou plusieurs éléments de configuration risquent de ne pas prendre effet. Toute configuration ayant échoué est enregistrée dans le CFS et est conservée jusqu'au prochain rechargement.

Vous pouvez parcourir la configuration qui a échoué, corriger les erreurs et réappliquer la configuration.

Voici quelques conseils pour résoudre les problèmes de configuration lors du démarrage du routeur.

Dans IOX, la configuration peut être classée en échec pour trois raisons :

1. Erreurs de syntaxe : l'analyseur génère des erreurs de syntaxe, qui indiquent généralement qu'il existe une incompatibilité avec les commandes CLI. Vous devez corriger les erreurs de syntaxe et réappliquer la configuration.
2. Erreurs sémantiques : les erreurs sémantiques sont générées par les composants principaux lorsque le gestionnaire de configuration restaure la configuration au démarrage du routeur. Il est important de noter que cfgmgr n'est pas responsable de s'assurer que la configuration est acceptée dans le cadre de la configuration en cours. Cfgmgr n'est qu'un **intermédiaire** et ne signale que les défaillances sémantiques générées par les composants principaux. Il appartient à chaque propriétaire de composant principal d'analyser la raison de l'échec et de déterminer la raison de l'échec. Les utilisateurs peuvent exécuter les **commandes <CLI>** de **description** à partir du mode de configuration afin de trouver facilement le propriétaire du vérificateur du composant principal. Par exemple, si le **routeur bgp 217** apparaît comme échec de configuration, la commande **description** montre que le vérificateur de composant est le composant ipv4-bgp.

```
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. Erreurs d'application : la configuration a été vérifiée et acceptée avec succès dans le cadre de la configuration en cours, mais le composant principal n'est pas en mesure de mettre à jour son état opérationnel pour une raison quelconque. La configuration s'affiche dans la configuration en cours, car elle a été correctement vérifiée et comme une configuration en échec en raison de l'erreur opérationnelle du serveur principal. La commande **description** peut à nouveau être exécutée sur l'interface de ligne de commande qui n'a pas pu être appliquée afin de trouver le propriétaire d'application de composant.Complétez ces étapes afin de parcourir et de réappliquer la configuration ayant échoué pendant les opérateurs de démarrage :Pour R3.2, les opérateurs peuvent utiliser cette procédure afin de réappliquer la configuration en échec :Les opérateurs peuvent utiliser la commande **show configuration fail startup** afin de parcourir la configuration en échec enregistrée lors du démarrage du routeur.Les opérateurs doivent exécuter la **commande show configuration fail startup noerror | file myfail.cfg** afin d'enregistrer la configuration de démarrage échouée dans un fichier.Les opérateurs doivent passer en mode **configuration** et utiliser les commandes **load/commit** afin de réappliquer cette configuration échouée :

```
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
```

Pour R3.3, les opérateurs d'images peuvent utiliser cette procédure mise à jour :Les opérateurs doivent utiliser la commande **show configuration fail startup** et la commande **load configuration fail startup** afin de parcourir et de réappliquer toute configuration ayant échoué.

```
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
```

## Dumper de noyau

Par défaut, IOS-XR écrit un vidage de noyau sur le disque dur en cas de panne du processus, mais pas si le noyau lui-même tombe en panne. Notez que pour un système multichâssis, cette fonctionnalité n'est actuellement prise en charge que pour le châssis de carte de ligne 0. L'autre châssis est pris en charge dans une version ultérieure du logiciel.

Il est suggéré que les vidages de noyau pour les RP et MSC soient activés avec l'utilisation de ces configurations dans les configurations standard et en mode admin :

```
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
```

### Configuration du vidage du noyau

Ceci entraîne cette occurrence pour un plantage du noyau :

1. Un RP tombe en panne et un dump est écrit sur le disque dur de ce RP dans le répertoire racine du disque.
2. Si un MSC tombe en panne, un vidage est écrit sur le disque dur de RP0 dans le répertoire racine du disque.

Cela n'a aucun impact sur les temps de basculement du RP puisque le transfert sans arrêt (NSF) est configuré pour les protocoles de routage. Il peut prendre quelques minutes supplémentaires pour que le RP ou la carte de ligne en panne redevienne disponible après une panne pendant qu'il écrit le noyau.

Un exemple d'ajout de cette configuration à la configuration standard et au mode admin est présenté ici. Notez que la configuration du mode admin nécessite l'utilisation de DRP.

Ce résultat montre un exemple de configuration de vidage du noyau :

```
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)#
```

# Sécurité

## LPTS

Les services de transport de paquets locaux (LPTS) gèrent les paquets destinés localement. LPTS est composé de différents composants.

1. La principale est appelée le processus d'arbitrage des ports. Il écoute les requêtes de socket provenant de différents processus de protocole, par exemple BGP, IS-IS, et garde trace de toutes les informations de liaison pour ces processus. Par exemple, si un processus BGP écoute au numéro de socket 179, le PA obtient ces informations des processus BGP, puis attribue une liaison à ce processus dans un IFIB.
2. La FIB est un autre élément du processus LPTS. Il permet de conserver un répertoire de l'emplacement d'un processus qui écoute une liaison de port spécifique. L'IFIB est généré par le processus d'arbitrage de port et est conservé avec l'arbitre de port. Il génère ensuite plusieurs sous-ensembles de ces informations.Le premier sous-ensemble est une tranche de l'IFIB. Cette tranche peut être associée au protocole IPv4, etc. Les tranches sont ensuite envoyées aux gestionnaires de flux appropriés, qui utilisent ensuite la tranche IFIB afin de transférer le paquet au processus approprié.Le deuxième sous-ensemble est un pré-IFIB, qui permet au LC de transférer le paquet au processus approprié si un seul processus existe ou à un gestionnaire de flux approprié.
3. Les gestionnaires de flux permettent de distribuer davantage les paquets si la recherche n'est pas triviale, par exemple, plusieurs processus pour BGP. Chaque gestionnaire de flux possède une ou plusieurs tranches de l'IFIB et transfère correctement les paquets aux processus appropriés associés à la tranche de l'IFIB.
4. Si une entrée n'est pas définie pour le port de destination, elle peut être abandonnée ou transférée au gestionnaire de flux. Un paquet est transféré sans port associé s'il existe une stratégie associée pour le port. Le gestionnaire de flux aide ensuite à générer une nouvelle entrée de session.

# Comment un paquet interne est-il transféré ?

Il existe deux types de flux : les flux de couche 2 (HDLC, PPP) et les flux de routage et les flux ICMP/PING de couche 4.

1. HDLC/PPP de couche 2 : ces paquets sont identifiés par l'identificateur de protocole et sont envoyés directement aux files d'attente du processeur dans le pulvérisateur. Les paquets de protocole de couche 2 reçoivent une priorité élevée et sont ensuite récupérés par le CPU (via le Squid) et traités. Par conséquent, les keepalives pour la couche 2 sont directement traités via le LC via le CPU. Cela évite d'avoir à aller au RP pour les réponses et joue avec le thème de la gestion d'interface distribuée.

2. Les paquets ICMP (couche 4) sont reçus dans le LC et sont envoyés via la recherche via l'IFBI dans les files d'attente du processeur sur le pulvérisateur. Ces paquets sont ensuite envoyés au processeur (via le Squid) et traités. La réponse est ensuite envoyée via les files d'attente de sortie de pulvérisateur afin d'être transmise via le fabric. Cela est possible si une autre application a également besoin des informations (répliquées via le fabric). Une fois le paquet traversé dans le fabric, il est destiné au LC de sortie approprié et à la file d'attente de contrôle et d'éponge appropriée.
3. Les flux de routage sont recherchés dans l'IFIB, puis envoyés aux files d'attente de formatage de sortie (8 000 files d'attente) dont l'une est réservée aux paquets de contrôle. Il s'agit d'une file d'attente sans forme et elle est simplement traitée chaque fois qu'elle est pleine. - haute priorité. Le paquet est ensuite envoyé via le fabric sur les files d'attente de priorité élevée dans un ensemble de files d'attente CPU sur l'éponge (comme les files d'attente Squid sur le pulvérisateur), puis traite par le processus approprié, le gestionnaire de flux ou le processus réel. Une réponse est renvoyée par l'éponge de la carte de ligne de sortie, puis par la carte de ligne. L'éponge LC de sortie a une file d'attente spéciale réservée pour gérer les paquets de contrôle. Les files d'attente dans l'Eponge sont divisées en paquets de priorité élevée, de contrôle et de priorité faible, par port de sortie.
4. Le PSE dispose d'un ensemble de régulateurs configurés pour la limitation de débit des paquets de couche 4, de couche 2 et de routage. Ces paramètres sont prédéfinis et modifiés pour être configurables par l'utilisateur à une date ultérieure.

L'un des problèmes les plus courants avec le protocole LPTS est celui des paquets abandonnés lorsque vous essayez d'envoyer une requête ping au routeur. Les contrôleurs LPTS limitent généralement le débit de ces paquets. C'est le cas pour confirmer :

```
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

Les paquets IP sont intrinsèquement non sécurisés. IPsec est une méthode utilisée pour protéger les paquets IP. L'IPsec CRS-1 est implémenté dans le chemin de transfert logiciel, par conséquent la session IPsec est interrompue sur le RP/DRP. Un nombre total de 500 sessions IPsec par CRS-1 sont prises en charge. Le nombre dépend de la vitesse du processeur et des ressources allouées. Il n'y a aucune limitation logicielle à cela, seul le trafic local et le trafic de terminaison locale sur RP sont éligibles pour la gestion IPsec. Le mode de transport IPsec ou le mode de tunnel peut être utilisé pour le type de trafic, bien que le premier soit préféré en raison de moins

de surcharge dans le traitement IPsec.

R3.3.0 prend en charge le chiffrement de BGP et OSPFv3 sur IPsec.

Référez-vous au Guide de configuration de la sécurité du système Cisco IOS XR pour plus d'informations sur la mise en oeuvre d'IPsec.

**Remarque :** IPsec requiert crypto pie, par exemple, hfr-k9sec-p.pie-3.3.1.

# Hors bande

## Console et accès AUX

Les RP/SC CRS-1 disposent à la fois d'une console et d'un port AUX disponibles pour la gestion hors bande, ainsi que d'un port Ethernet de gestion pour l'hors bande via IP.

La console et le port AUX de chaque RP/SCGE, deux par châssis, peuvent être connectés à un serveur de console. Cela signifie que le système à châssis unique nécessite quatre ports de console et que les systèmes multichâssis nécessitent 12 ports plus deux ports supplémentaires pour les modules Supervisor Engine du Catalyst 6504-E.

La connexion au port AUX est importante car elle permet d'accéder au noyau IOS-XR et permet la récupération du système lorsque cela n'est pas possible via le port de console. L'accès via le port AUX n'est disponible que pour les utilisateurs définis localement sur le système, et uniquement lorsque l'utilisateur dispose d'un accès au niveau root-system ou cisco-support. En outre, l'utilisateur doit avoir un mot de passe **secret** défini.

## Accès au terminal virtuel

Telnet & Secure Shell (SSH) peut être utilisé pour atteindre le CRS-1 via les ports vty. Par défaut, les deux sont désactivés et l'utilisateur doit les activer explicitement.

**Remarque :** IPsec requiert crypto pie, par exemple, hfr-k9sec-p.pie-3.3.1.

Commencez par générer les clés RSA et DSA comme indiqué dans cet exemple afin d'activer SSH :

```
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
```

# Informations connexes

- Prise en charge des routeurs
- Support et documentation techniques - Cisco Systems