Práticas recomendadas operacionais do CRS-1 e do IOS XR

Contents

Introduction

Este documento ajuda a compreender o seguinte:

- Processos e Threads
- Estrutura CRS-1
- Controle o plano
- Rommon e Monlib
- Módulo de Interface de Camada Física (PLIM - Physical Layer Interface Module) e Placa de Serviço Modular (MSC - Modular Service Card)
- Gerenciamento de configuração
- Security
- Fora da banda
- Simple Network Management Protocol

Prerequisites

Requirements

A Cisco recomenda que você tenha conhecimento do Cisco IOS® XR.

Componentes Utilizados

As informações neste documento são baseadas nestas versões de software e hardware:

- Cisco IOS XR Software
- 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

Consulte as Convenções de Dicas Técnicas da Cisco para obter mais informações sobre convenções de documentos.

Visão geral do Cisco IOS XR

O Cisco IOS XR foi projetado para escalar. O kernel é uma arquitetura de microkernel, portanto, fornece somente serviços essenciais como gerenciamento de processos, agendamento, sinais e temporizadores. Todos os outros serviços, como sistemas de arquivos, drivers, pilhas de protocolos e aplicativos, são considerados gerenciadores de recursos e executados em espaço de usuário protegido por memória. Esses outros serviços podem ser adicionados ou removidos em tempo de execução, dependendo do projeto do programa. O volume de microkernel é de apenas 12 kb. O microkernel e o sistema operacional subjacente são de QNX Software Systems e são chamados de Neutrino. O QNX é especializado em projeto de sistema operacional em tempo real. O Microkernel é preventivo e o agendador é baseado em prioridade. Isso garante que a comutação de contexto entre os processos seja muito rápida e que os segmentos de prioridade mais alta sempre tenham acesso à CPU quando necessário. Esses são alguns dos benefícios que o Cisco IOS XR aproveita. Mas, o maior benefício é o projeto herdado de Inter Process Communications dentro do núcleo dos sistemas operacionais.

Neutrino é uma mensagem que passa pelo sistema operacional e as mensagens são o meio básico das comunicações entre processos entre todos os processos. Quando um servidor específico deseja fornecer um serviço, ele cria um canal para a troca de mensagens. Os clientes se conectam ao canal Servidores mapeando diretamente o descritor de arquivo relevante para utilizar o serviço. Todas as comunicações entre cliente e servidor são feitas pelo mesmo mecanismo. Esse é um grande benefício para um supercomputador, que é o CRS-1. Considere isso quando uma operação de leitura local é executada em um kernel UNIX padrão:

- Interrupção de software no kernel.
- O kernel é enviado para o sistema de arquivos.
- Os dados são recebidos.

Considere isso no caso remoto:

- Interrupção de software no kernel.
- O kernel despacha o NFS.
- O NFS chama o componente de rede.
- Remoto despacha o componente de rede.
- NFS é chamado.
- O kernel despacha o sistema de arquivos.

A semântica da leitura local e da leitura remota não são iguais. Os argumentos e parâmetros para bloqueio de arquivos e definição de permissões são diferentes.

Considere o caso local do QNX:

- Interrupção de software no kernel.
- O kernel executa a mensagem que passa para o sistema de arquivos.

Considere o caso não local:

- Interrupção de software no kernel.
- O kernel entra no QNET, que é o mecanismo de transporte IPC.
- O QNET entra no kernel.
- O kernel despacha o sistema de arquivos.

Todas as semânticas que dizem respeito à passagem de argumentos e aos parâmetros do sistema de arquivos são idênticas. Tudo foi desacoplado na interface IPC que permite que o cliente e o servidor sejam completamente segregados. Isso significa que qualquer processo pode ser executado em qualquer lugar a qualquer momento. Se um processador de rota específico estiver muito ocupado atendendo solicitações, você poderá facilmente migrar esses serviços para uma CPU diferente que seja executada em um DRP. Um supercomputador que executa diferentes serviços em diferentes CPUs espalhadas por vários nós que podem se comunicar facilmente com qualquer outro nó. A infraestrutura está em vigor para oferecer a oportunidade de escalar. A Cisco utilizou essa vantagem e escreveu software adicional que se conecta às operações principais do kernel de passagem de mensagem que permite que o roteador CRS expanda para milhares de nós, onde um nó, neste caso uma CPU, executa uma instância do SO, seja um Route Process (RP), um Distributed Route Processor (DRP), uma Modular Services Card (MSC) ou um Switch Processor (SP).

Processos e Threads

Dentro dos limites do Cisco IOS XR, um processo é uma área protegida de memória que contém um ou mais segmentos. Do ponto de vista dos programadores, os threads fazem o trabalho e

cada um conclui um caminho de execução lógica para executar uma tarefa específica. A memória que os processos exigem durante o fluxo de execução pertence ao processo em que operam, protegida de quaisquer outros processos. Um thread é uma unidade de execução, com um contexto de execução que inclui uma pilha e registros. Um processo é um grupo de threads que compartilham um espaço de endereço virtual, embora um processo possa conter um único thread, mas com mais frequência contém mais. Se outro thread em um processo diferente tentar gravar na memória em seu processo, o processo ofensivo será encerrado. Se houver mais de um thread que opera em seu processo, esse thread terá acesso à mesma memória em seu processo e, como resultado, poderá sobrescrever os dados de outro thread. Conclua as etapas em um procedimento para manter a sincronização com os recursos a fim evitar esse segmento no mesmo processo.

Um thread utiliza um objeto denominado MUTEX (Exclusão Mútua) para garantir a exclusão mútua dos serviços. O thread que tem o MUTEX é o thread que pode gravar em uma área específica da memória como exemplo. Outros segmentos que não têm o MUTEX não podem. Há também outros mecanismos para garantir a sincronização com os recursos, que são Semáforos, Variáveis Condicionais ou Condores, Barreiras e Sonhos. Estes não são discutidos aqui, mas fornecem serviços de sincronização como parte das suas funções. Se você equipara os princípios discutidos aqui ao Cisco IOS, o Cisco IOS é um único processo operando muitos processos, com todos os processos que têm acesso ao mesmo espaço de memória. Mas o Cisco IOS chama esses processos de processos.

Estados de processos e segmentos

No Cisco IOS XR existem servidores que fornecem os serviços e os clientes que usam os serviços. Um processo específico pode ter vários segmentos que fornecem o mesmo serviço. Outro processo pode ter vários clientes que podem exigir um determinado serviço a qualquer momento. O acesso aos servidores nem sempre está disponível e, se um cliente solicitar acesso a um serviço, ele fica lá e espera que o servidor esteja livre. Neste caso, diz-se que o cliente está bloqueado. Isso é chamado de modelo de servidor de cliente de bloqueio. O cliente pode ser bloqueado porque espera por um recurso como um MUTEX, ou porque o servidor ainda não respondeu.

Emita um comando **show process ospf** para verificar o status dos segmentos no processo 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
```

Observe que o processo ospf recebe uma ID de trabalho (JID), que é 250. Isso nunca muda em um roteador em execução e geralmente em uma versão específica do Cisco IOS XR. No processo ospf, há cinco segmentos cada um com seu próprio ID de thread (TID). Listado é o espaço da pilha para cada thread, a prioridade de cada thread e seu estado.

Passagem de mensagem síncrona

É mencionado anteriormente que o QNX é uma mensagem que passa pelo sistema operacional. Na verdade, é um sistema operacional de passagem de mensagem síncrona. Muitos dos problemas do sistema operacional são refletidos na mensagem síncrona. Não se diz que a passagem de mensagem síncrona causa qualquer problema, mas o sintoma do problema é refletido na passagem de mensagem síncrona. Por ser síncrona, as informações de ciclo de vida ou estado são facilmente acessíveis ao operador do CRS-1, o que ajuda no processo de solução de problemas. A mensagem que passa o ciclo de vida é semelhante a esta:

- Um servidor cria um canal de mensagem.
- Um cliente se conecta ao canal de um servidor (análogo ao posix open).
- Um cliente envia uma mensagem a um servidor (MsgSend) e espera uma resposta e bloqueia.
- O servidor recebe (MsgReceive) uma mensagem de um cliente, processa a mensagem e responde ao cliente.
- O cliente desbloqueia e processa a resposta do servidor.

Esse modelo de cliente-servidor de bloqueio é a passagem de mensagem síncrona. Isso significa que o cliente envia uma mensagem e bloqueia. O servidor recebe a mensagem, processa-a, responde ao cliente e o cliente desbloqueia. Estes são os detalhes específicos:

- O servidor aguarda no estado RECEIVE.
- O cliente envia uma mensagem ao servidor e torna-se BLOQUEADO.
- O servidor recebe a mensagem e desbloqueia, se estiver aguardando no estado de recebimento.
- O cliente passa para o estado REPLY.
- O servidor é movido para o estado EM EXECUÇÃO.
- O servidor processa a mensagem.
- O servidor responde ao cliente.
- O cliente desbloqueia.

Execute o comando **show process** para ver em que estados o cliente e os servidores estão.

```
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
```

Processos e Estados de Processo Bloqueados

Emita o comando **show process blocking** para ver qual processo está no estado bloqueado.

```
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#
```

A passagem sincronizada de mensagens permite que você controle facilmente o ciclo de vida da comunicação entre processos entre os diferentes segmentos. A qualquer momento, um thread pode estar em um estado específico. Um estado bloqueado pode ser um sintoma de um problema. Isso não significa que, se um thread estiver no estado bloqueado, há um problema, portanto, não emita o comando **show process blocking** e abra um caso no Suporte Técnico da Cisco. Os segmentos bloqueados também são muito normais.

Observe a saída anterior. Se você observar o primeiro segmento na lista, observe que é o ksh, e sua resposta está bloqueada em devc-conaux. O cliente, o ksh nesse caso, enviou uma mensagem ao processo devc-conaux, o servidor, que é devc-conaux, mantém a resposta do ksh bloqueada até que ela responda. O Ksh é o shell UNIX que alguém usa no console ou na porta AUX. O Ksh aguarda a entrada do console e, se não houver nenhum porque o operador não está digitando, ele permanecerá bloqueado até que ele processe alguma entrada. Após o processamento, o ksh retorna para responder bloqueado em devc-conaux.

Isso é normal e não ilustra um problema. O ponto é que os threads bloqueados são normais, e depende da versão XR, do tipo de sistema que você tem, do que você configurou e de quem faz o que altera a saída do comando **show process locked**. O uso do comando **show process blocking** é uma boa maneira de começar a solucionar problemas de tipo de SO. Se houver um problema, por exemplo, a CPU está alta, use o comando anterior para ver se algo parece fora do normal.

Entenda o que é normal para o seu roteador em funcionamento. Isso fornece uma linha de base para você usar como comparação ao solucionar problemas de ciclos de vida do processo.

A qualquer momento, um thread pode estar em um estado específico. Esta tabela fornece uma lista dos estados:

Se o Estado for:	O segmento é:
MORTO	Morto. O kernel está esperando para liberar os recursos de threads.
EXECUTANDO	Em execução ativa em uma CPU
PRONTO	Não está em execução em uma CPU, mas está pronto para ser executado
PARADO	Suspenso (sinal SIGSTOP)
ENVIAR	Aguardando que um servidor receba uma mensagem
RECEBER	Aguardando que um cliente envie uma mensagem
REPLY	Aguardando um servidor responder a uma mensagem
PILHA	Aguardando a alocação de mais pilha

WAITPAGE	Aguardando que o gerenciador de processos resolva uma falha de página
SIGSUSPEND	Aguardando sinal
SIGWAITINFO	Aguardando sinal
NANOSLEEP	Dormindo por um período de tempo
MUTEX	Aguardando a aquisição de um MUTEX
CONDVAR	Aguardando a sinalização de uma variável condicional
UNIÃO	Aguardando a conclusão de outro thread
INTR	Aguardando interrupção
SEM	À espera de adquirir um semáforo

Processos importantes e suas funções

O Cisco IOS XR tem muitos processos. Estes são alguns importantes com suas funções explicadas aqui.

Monitor de sistema WatchDog (WDSysmon)

Este é um serviço fornecido para a detecção de travamentos de processo e condições de memória baixa. A memória baixa pode ocorrer como resultado de um vazamento de memória ou de alguma outra circunstância estranha. Um travamento pode ser o resultado de várias condições, como bloqueios de processo, loops infinitos, travamentos de kernel ou erros de agendamento. Em qualquer ambiente multisegmentado, o sistema pode entrar em um estado conhecido como condição de impasse ou simplesmente um impasse. Um impasse pode ocorrer quando um ou mais segmentos não podem continuar devido à contenção de recursos. Por exemplo, o thread A pode enviar uma mensagem para o thread B enquanto o thread B simultaneamente envia uma mensagem para o thread A. Ambos os processos esperam um do outro e podem estar no estado de envio bloqueado, e ambos os processos esperam para sempre. Este é um caso simples que envolve dois processos, mas se um servidor for responsável por um recurso que é usado por muitos processos está bloqueado em outro thread, então os vários processos que solicitam acesso a esse recurso podem ser enviados bloqueados aguardando no servidor.

Dadlocks podem ocorrer entre alguns segmentos, mas podem afetar outros processos como resultado. Os prazos são evitados pelo bom design do programa, mas independentemente do quão magnífico um programa é projetado e escrito. Às vezes, uma sequência específica de eventos que dependem de dados com temporizações específicas pode causar um impasse. Os prazos nem sempre são deterministas e geralmente são muito difíceis de se reproduzir. O WDSysmon tem muitos segmentos com um que é executado com a prioridade mais alta suportada por Neutrino, 63. A execução na prioridade 63 garante que o thread obtenha tempo de CPU em um ambiente de programação preventiva com base em prioridade. O WDSysmon trabalha com o recurso de watchdog de hardware e observa os processos de software que buscam condições de travamento. Quando tais condições são detectadas, o WDSysmon coleta mais informações sobre a condição, pode jogar o processo no despejo do kernel, gravar em syslogs, executar scripts e matar os processos bloqueados. Dependendo do quão drástico o problema é, ele pode iniciar um switch do processador de rota para manter a operação do

sistema.

```
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
```

O processo WDSysmon tem nove segmentos. Quatro corridas na prioridade 10, as outras quatro são 11, 19, 20 e 63. Quando um processo é projetado, o programador considera cuidadosamente a prioridade que cada thread no processo deve ser dada. Conforme discutido anteriormente, o agendador é baseado em prioridade, o que significa que um segmento de prioridade mais alta sempre antecipa um de prioridade mais baixa. A prioridade 63 é a prioridade mais alta na qual um thread pode ser executado, que é o thread 7 neste caso. O thread 7 é o thread do observador, o thread que rastreia os porcos da CPU. Ele deve ser executado com uma prioridade mais alta do que os outros segmentos que ele observa, caso contrário, ele pode não ter a chance de funcionar, o que o impede das etapas que ele foi projetado para executar.

Netio

No Cisco IOS, há o conceito de switching rápida e switching de processo. A comutação rápida usa o código CEF e ocorre no momento da interrupção. A comutação de processos usa ip_input, que é o código de comutação IP, e é um processo programado. Em plataformas superiores, a comutação CEF é feita em hardware, e ip_input é agendado na CPU. O equivalente de ip_input no Cisco IOS XR é 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
```

Processo de serviços de grupo (GSP)

Há necessidade de comunicação em qualquer supercomputador com vários milhares de nós que executam cada um sua própria instância do kernel. Na Internet, uma para muitas comunicações é feita com eficiência através de protocolos de multicast. GSP é o protocolo de multicast interno usado para IPC no CRS-1. O GSP oferece uma comunicação de grupo confiável, sem conexão, com semântica assíncrona. Isso permite que o GSP escale até os milhares de nós.

```
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

O BCDL é usado para transmitir dados de multicast confiável para vários nós, como RPs e MSCs. Utiliza o SPG como transporte subjacente. Garantias BCDL **em ordem** de entrega de mensagens. Dentro do BCDL há um agente, um produtor e um consumidor. O agente é o processo que se comunica com o produtor para recuperar e armazenar em buffer os dados antes de seus multicasts para os consumidores. O produtor é o processo que produz os dados que todos querem, e o consumidor é o processo interessado em receber os dados fornecidos pelo produtor. O BCDL é usado durante atualizações do software Cisco IOS XR.

Mensagens leves (LWM)

O LWM é uma forma de mensagem criada pela Cisco e projetada para criar uma camada de abstração entre os aplicativos que interprocessos se comunicam entre si e o Neutrino, com o objetivo de independência do sistema operacional e da camada de transporte. Se a Cisco desejar mudar o fornecedor do SO do QNX para outra pessoa, uma camada de abstração entre as funções rudimentares do sistema operacional subjacente ajuda a remover a dependência do sistema operacional e ajuda na migração para outro sistema operacional. O LWM fornece entrega de mensagem síncrona garantida, o que, como a passagem de mensagem Neutrino nativa, faz com que o remetente bloqueie até que o receptor responda.

O LWM também fornece entrega de mensagem assíncrona através de pulsos de 40 bits. As mensagens assíncronas são enviadas de forma assíncrona, o que significa que a mensagem está na fila e o remetente não bloqueia, mas não são recebidas pelo servidor de forma assíncrona, mas quando o servidor pesquisa a próxima mensagem disponível. O LWM é estruturado como cliente/servidor. O servidor cria um canal que lhe dá um **ouvido** para escutar mensagens e se senta em um momento em que o loop faz uma mensagem de recebimento ouvindo no canal, que ele acabou de criar. Quando uma mensagem chega, ela é desbloqueada e obtém um identificador de cliente, que é efetivamente a mesma coisa que a ID de recebimento da mensagem recebida. Em seguida, o servidor executa algum processamento e, posteriormente, envia uma mensagem ao identificador do cliente.

No lado do cliente, uma mensagem é conectada. Ele recebe um identificador ao qual se conecta e, em seguida, envia uma mensagem e é bloqueado. Quando o servidor termina o processamento, ele responde e o cliente é desbloqueado. Isso é praticamente o mesmo que a passagem de mensagem nativa de Neutrinos, então a camada de abstração é muito fina.

O LWM foi projetado com um número mínimo de chamadas de sistema e switches de contexto para alto desempenho e é o método preferido de IPC no ambiente Cisco IOS XR.

Envmon

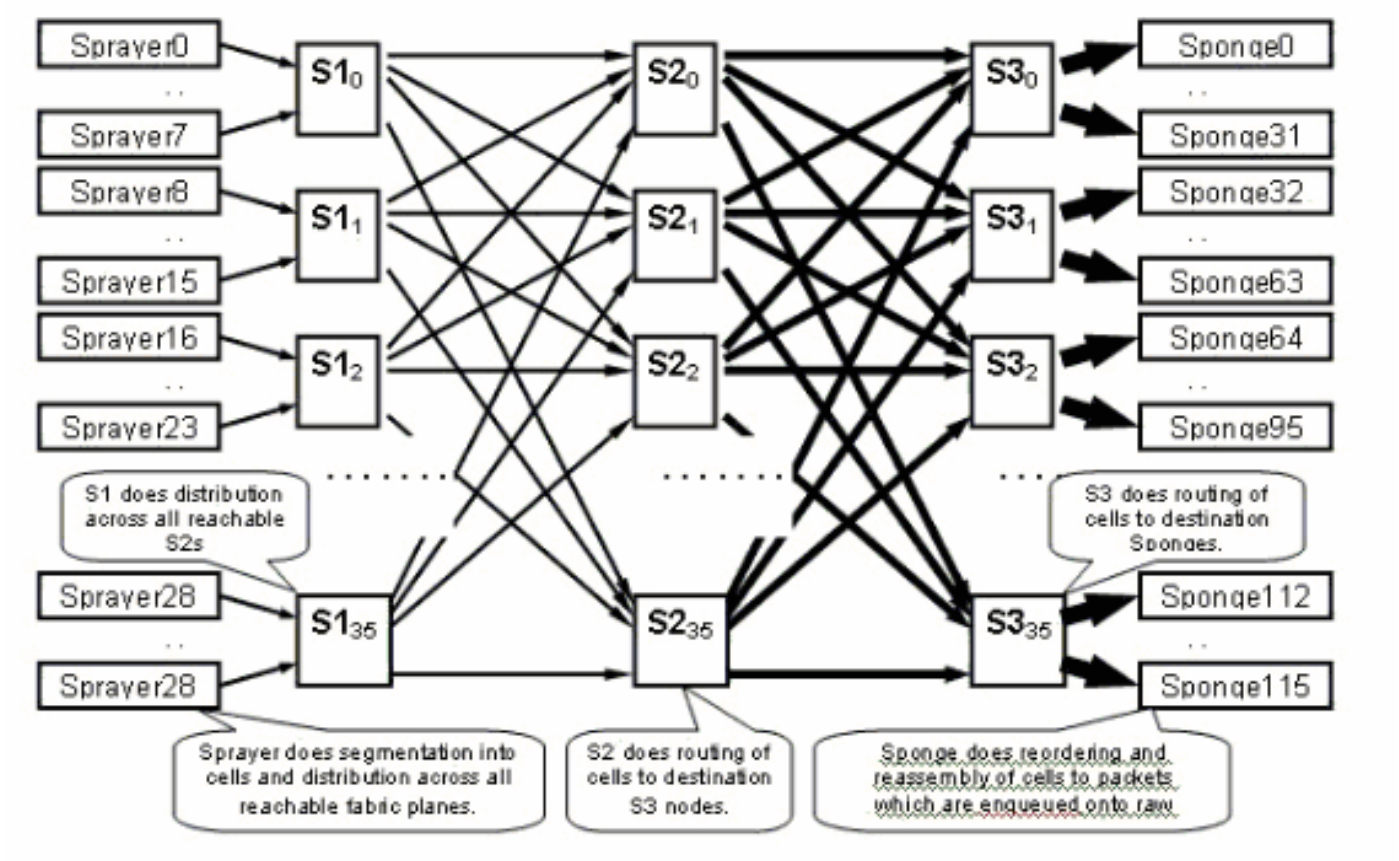
No nível mais fundamental, o sistema de monitoramento ambiental é responsável por avisar quando parâmetros físicos, como temperatura, voltagem, velocidade do ventilador e assim por diante, estão fora dos intervalos operacionais e desligam o hardware que se aproxima de níveis críticos onde o hardware pode ser danificado. Monitora periodicamente cada sensor de hardware disponível, compara o valor medido com relação aos limites específicos da placa e eleva os alarmes conforme necessário para realizar essa tarefa. Um processo persistente, iniciado na inicialização do sistema, que pesquisa periodicamente todos os sensores de hardware, por exemplo, voltagem, temperatura e velocidade do ventilador, no chassi e fornece esses dados aos clientes de gerenciamento externo. Além disso, o processo periódico compara as leituras de sensores com limiares de alarme e publica alertas ambientais no banco de dados do sistema para ação subsequente do Gerente de falhas. Se as leituras do sensor estiverem perigosamente fora do alcance, o processo de monitoramento ambiental pode fazer com que a placa seja desligada.

Introdução à estrutura CRS-1

- Malha em vários estágios—topologia em 3 estágios de Benes
- Roteamento dinâmico dentro da estrutura para minimizar o congestionamento
- Baseado em célula: 136 células de byte, payload de dados de 120 bytes
- Controle de fluxo para melhorar o isolamento de tráfego e minimizar os requisitos de buffer na estrutura
- Prepare-se para a aceleração de estágio
- Dois Casts de tráfego suportados (Unicast e Multicast)
- Duas prioridades de tráfego suportadas por cast (alta e baixa)
- Suporte para grupos multicast de estrutura (FGIDs) de 1 M
- Tolerância a falhas econômica: Redundância N+1 ou N+k usando planos de estrutura em vez de 1+1 a um custo muito maior

Quando você executa no modo de chassi único, os asics S1, S2 e S3 estão localizados nas mesmas placas de estrutura. Esta placa também é comumente conhecida como **placa S123**. Em uma configuração de vários chassis, o S2 é separado e está no Fabric Card Chassis (FCC). Essa configuração exige duas placas de estrutura para formar um plano, uma placa S2 e uma placa S13. Cada MSC conecta-se a oito planos de estrutura para fornecer redundância, de modo que, se você soltar um ou mais planos, sua estrutura ainda passe pelo tráfego, embora o tráfego agregado, que pode passar pela estrutura, seja menor. O CRS ainda pode operar em linerate para a maioria dos tamanhos de pacote com apenas sete planos. A contrapressão é enviada sobre o tecido sobre um plano ímpar e par. Não é recomendado executar um sistema com menos de dois planos em um plano ímpar e par. Qualquer configuração com menos de dois planos não é suportada.

O plano da estrutura



O diagrama anterior representa um plano. Você tem que multiplicar esse diagrama por oito. Isso significa que o asic de pulverização (ingressq) de um LC se conecta a 8 S1s (1 S1 por plano). O S1 em cada plano de estrutura se conecta a 8 pulverizadores:

- os 8 principais LCs do chassi
- os 8 LCs inferiores

Há 16 S1s por chassi LC de 16 slots: 8 para LCs superiores (1 por plano) + 8 para LCs inferiores.

Em um único chassi de 16 slots, uma placa de estrutura S123 tem 2 S1s, 2 S2 e 4 S3s. Isso é parte da computação de aceleração da estrutura. Há o dobro de tráfego, que pode sair da estrutura quanto o tráfego pode entrar. Atualmente, existem duas esponjas (fabricq) por LC em comparação com 1 Sprayer. Isso permite o buffer no LC de saída quando mais de um LC de entrada sobrecarrega um LC de saída. O LC de saída é capaz de absorver essa largura de banda extra da estrutura.

Monitoramento de estrutura

Disponibilidade e conectividade de plano:

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

Verifique se os planos estão recebendo/transmitindo células e se alguns erros estão aumentando:

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

Os acrônimos do comando anterior:

- CE — Erro corrigível

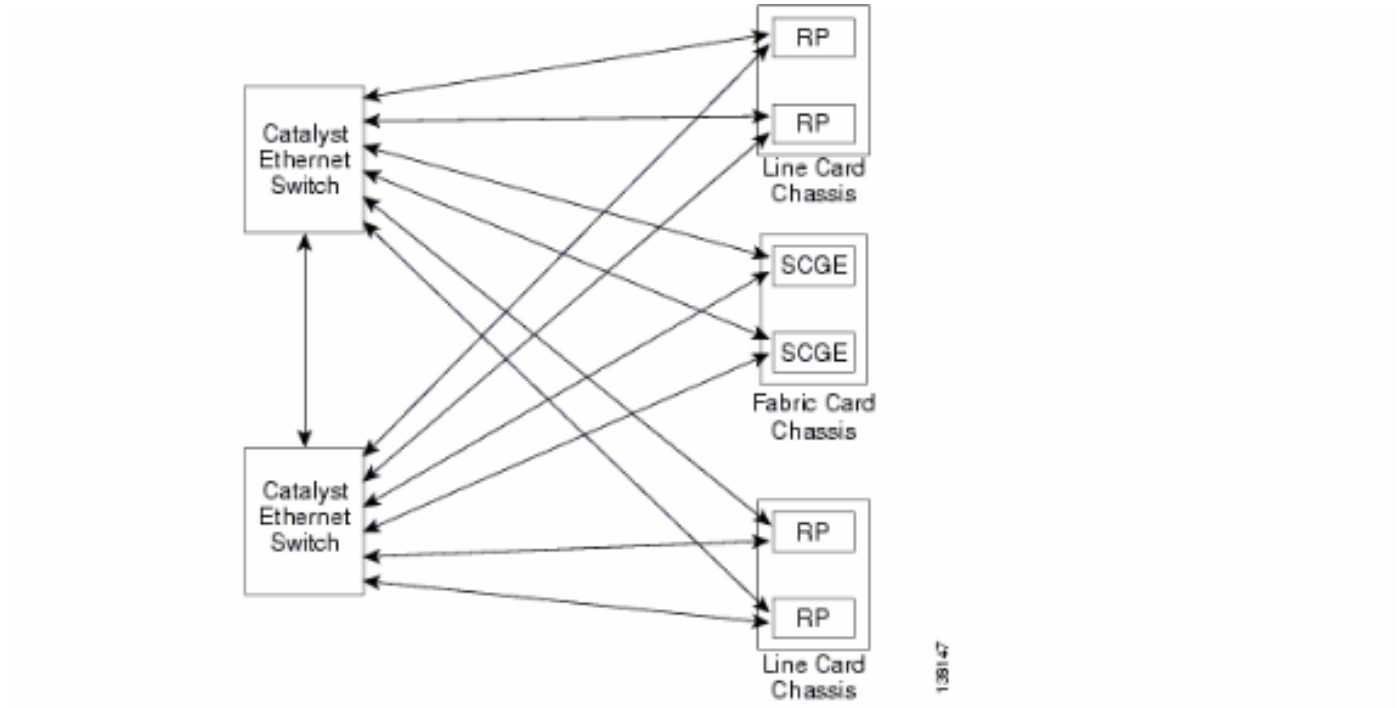
- UCE — Erro incorrigível
- PE — Erro de paridade

Não se preocupe se eles perceberem alguns erros, pois isso pode acontecer na inicialização. Os campos não devem ser incrementados em tempo de execução. Se estiverem, pode ser uma indicação de um problema na estrutura. Execute este comando para obter uma análise dos erros por plano de estrutura:

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

Visão geral do plano de controle

A conectividade do plano de controle entre o chassi da placa de linha e o chassi de estrutura é atualmente via portas Gigabit Ethernet nos RPs (LCC) e SCGE (FCC). A interconexão entre as portas é fornecida por meio de um par de switches Catalyst 6500, que podem ser conectados por duas ou mais portas Gigabit Ethernet.



Configuração do Catalyst 6500

Esta é a configuração recomendada para os switches Catalyst usados para o plano de controle multichassi:

- Uma única VLAN é usada em todas as portas.
- Todas as portas são executadas no modo de acesso (sem entroncamento).
- O Spanning-Tree 802.1w/s é usado para prevenção de loop.
- Dois ou mais links são usados para conectar os dois switches em uma conexão cruzada e o STP é usado para evitar loop. Canalização não é recomendado.
- As portas que se conectam ao CRS-1 RP e SCGE usam o modo pré-padrão, já que o IOS-XR não suporta os padrões baseados em 802.1s.

- O UDLD deve ser ativado nas portas que se conectam entre os switches e entre eles e o RP/SCGE.
- O UDLD é ativado por padrão no CRS-1.

Consulte Ativação do Cisco IOS XR Software em um Sistema Multishelf para obter mais informações sobre como configurar um Catalyst 6500 em um sistema Multishelf.

Gerenciamento plano de controle de vários chassis

O chassi do Catalyst 6504-E, que fornece a conectividade do plano de controle para o sistema multichassi, é configurado para estes serviços de gerenciamento:

- Gerenciamento em banda através da porta gigabit 1/2, que se conecta a um switch LAN em cada PoP. O acesso é permitido somente para uma pequena faixa de sub-redes e protocolos.
- O NTP é usado para definir a hora do sistema.
- O syslogging é executado nos hosts padrão.
- A interrogação e as interceptações SNMP podem ser ativadas para funções críticas.

Observação: não devem ser feitas alterações no Catalyst em operação. Os testes anteriores devem ser feitos em qualquer alteração planejada e é altamente recomendável que isso seja feito durante uma janela de manutenção.

Este é um exemplo de configuração de gerenciamento:

```
#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 e Monlib

O Cisco monlib é um programa executável que é armazenado no dispositivo e carregado na RAM para execução pelo ROMMON. O ROMMON usa monlib para acessar arquivos no dispositivo. As versões do ROMMON podem ser atualizadas e devem ser feitas sob recomendação do Suporte Técnico da Cisco. A versão mais recente do ROMMON é 1.40.

Instruções de atualização

Conclua estes passos:

1. Baixe os binários ROMMON do Cisco CRS-1 ROMMON (somente clientes registrados).
2. Descompacte o arquivo TAR e copie os 6 arquivos BIN no diretório raiz do CRS do 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. Use o comando **show diag | inc ROM|NODE|** comando **PLIM** para ver a versão atual do rommon.

```
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. Entre no modo ADMIN e use o comando **upgrade rommon a all disk0** para atualizar o 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. Saia do Modo ADMIN e digite **show log | inc "OK, ROMMON A"** e certifique-se de que todos os nós foram atualizados com êxito. Se algum dos nós falhar, volte para a etapa 4 e reprograme.

```
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. Entre no modo ADMIN e use o comando **upgrade rommon b all disk0** para atualizar o 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. Saia do Modo ADMIN e digite **show log | inc "OK, ROMMON B"** e certifique-se de que todos os nós foram atualizados com êxito. Se algum dos nós falhar, volte para a etapa 4 e reprograme.

```
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. O comando **upgrade** simplesmente grava uma seção reservada especial de bootflash com o novo ROMMON. Mas o novo ROMMON permanece inativo até que a placa seja recarregada. Assim, quando você recarregar a placa, o novo ROMMON estará ativo. Redefina cada nó um de cada vez ou reinicie o roteador inteiro para fazer isso.

```
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. Use o comando **show diag | inc ROM|NODE|PLIM** para verificar a versão atual do 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]
```

Observação: no CRS-8 e no chassi de malha, o ROMMON também define as velocidades do ventilador para a velocidade padrão de 4000 RPM.

Visão geral de PLIM e MSC

Isso representa o fluxo de pacotes no roteador CRS-1, e estes termos são usados de forma intercambiável:

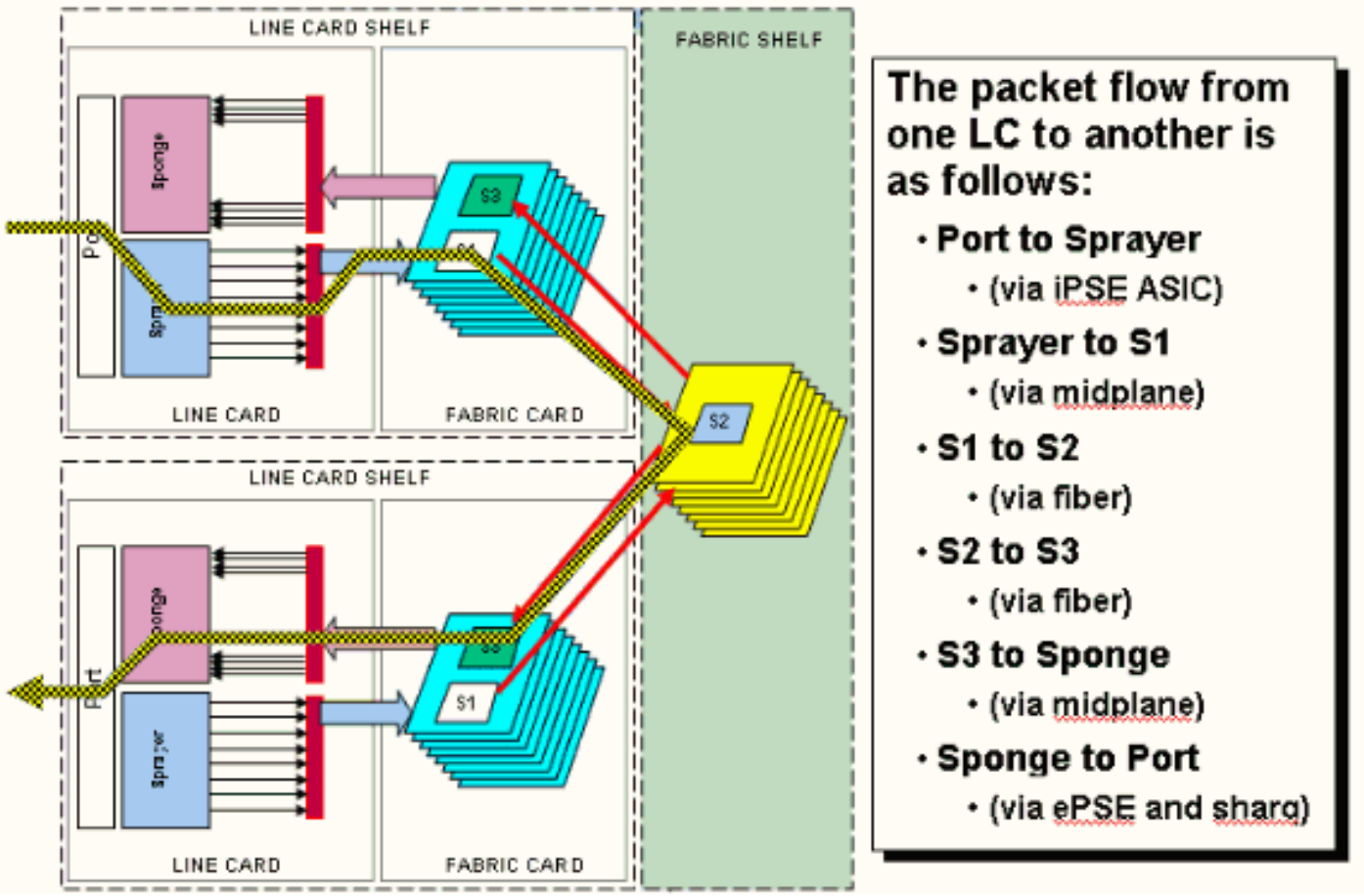
O ASIC de IngressQ também é chamado de ASIC de Sprayer.

O ASIC de FabricQ também é chamado de ASIC de Esponja.

O ASIC de EgressQ também é chamado de ASIC Sharq.

O SPP também é chamado de ASIC do PSE (Packet Switch Engine).

Rx PLIM > Rx SPP > Fila de Entrada > Estrutura > Fila > Tx SPP > Fila de Saída > Tx PLIM (Sprayer) (Esponge) (Sharq)

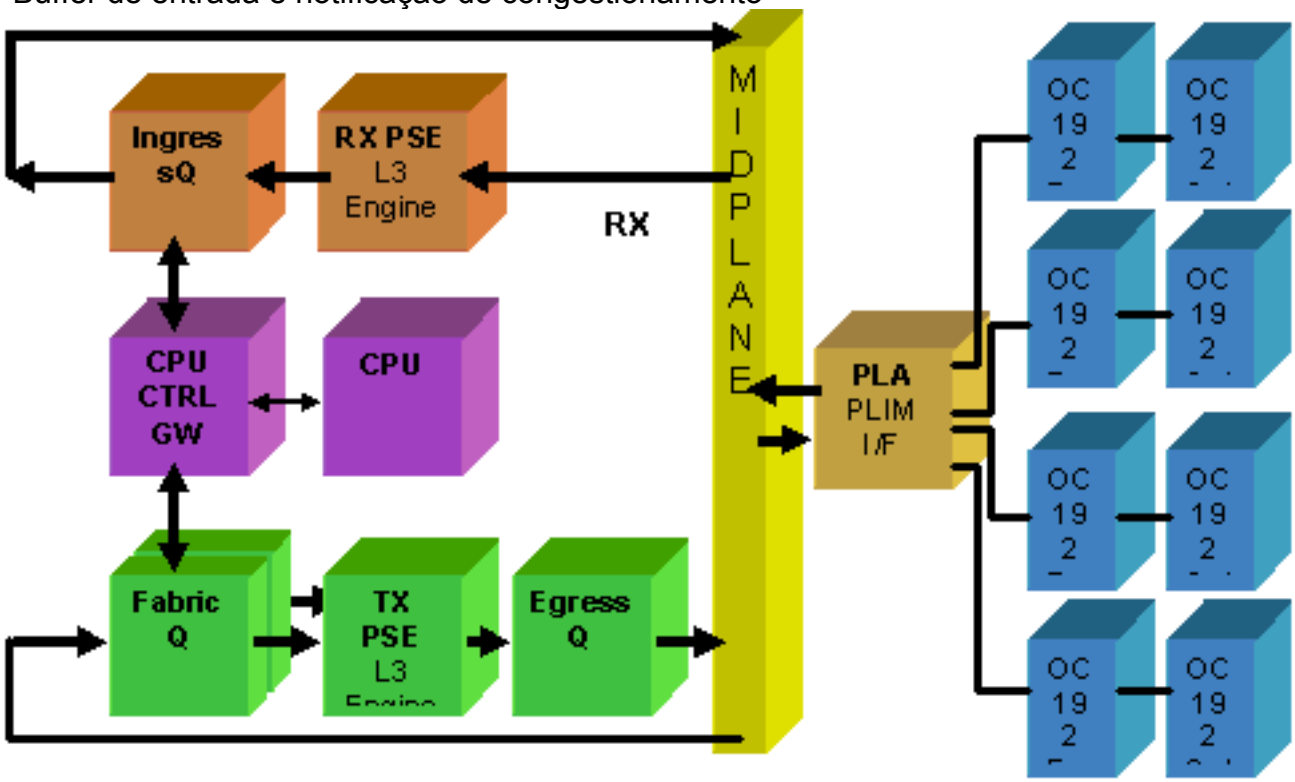


Os pacotes são recebidos no Módulo de Interface de Camada Física (PLIM - Physical Layer Interface Module).

O PLIM contém as interfaces físicas para o MSC com o qual ele se encontra. PLIM e MSC são placas separadas conectadas através do backplane do chassi. Como resultado, os tipos de interface para um MSC específico são definidos pelo tipo de PLIM com o qual ele tratou. Dependendo do tipo de PLIM, a placa contém vários ASICs que fornecem a mídia física e o enquadramento para as interfaces. A finalidade dos ASICs PLIM é fornecer a interface entre o MSC e as conexões físicas. Termina a fibra, faz a conversão elétrica da luz, termina o enquadramento de mídia sendo SDH/Sonet/Ethernet/HDLC/PPP, verifica o CRC, adiciona algumas informações de controle chamadas de cabeçalho do buffer e encaminha os bits que permanecem no MSC. O PLIM não origina/sinaliza os keepalives HDLC ou PPP. Eles são tratados pela CPU no MSC.

O PLIM também fornece estas funções:

- Filtragem de MAC para 1/10 Gigabit Ethernet
- Contabilidade MAC de entrada/saída para 1/10 Gigabit Ethernet
- Filtragem de VLAN para 1/10 Gigabit Ethernet
- Contabilização de VLAN para 1/10 Gigabit Ethernet
- Buffer de entrada e notificação de congestionamento



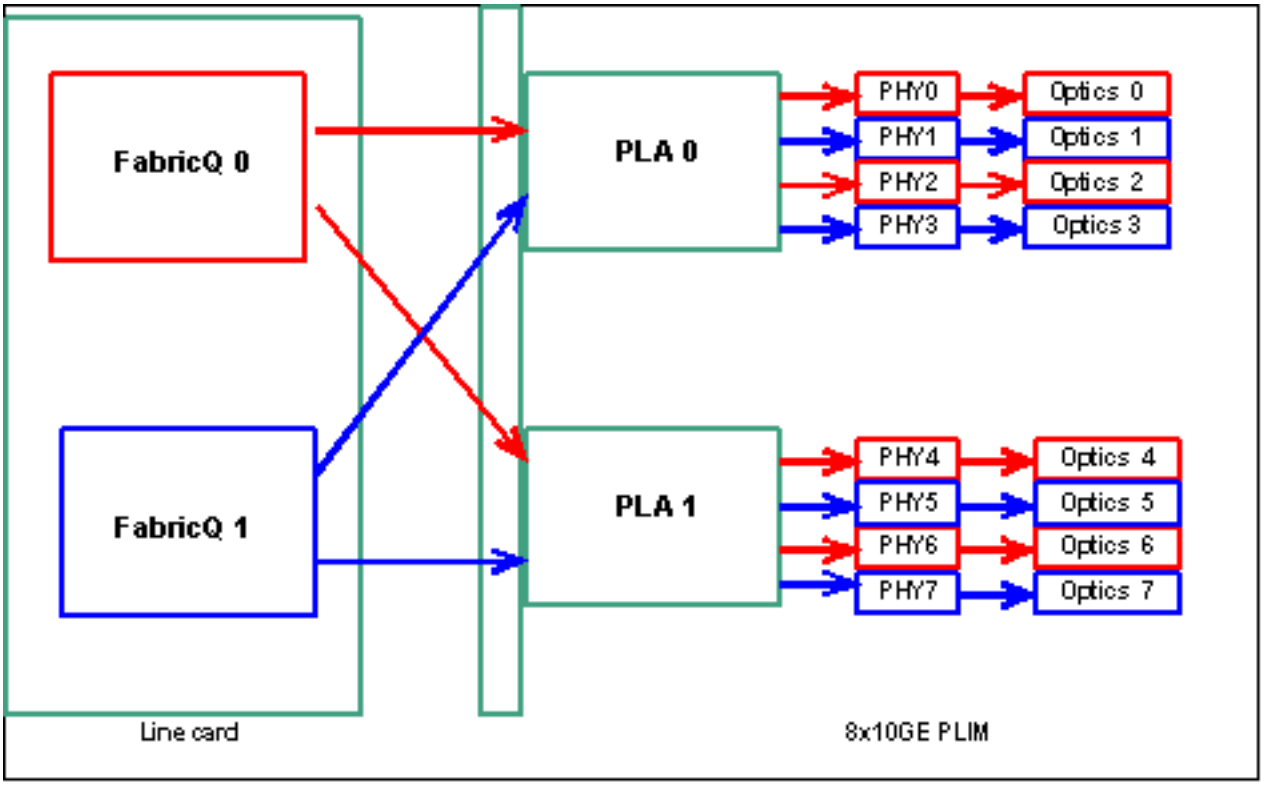
Excesso de assinatura PLIM

PLIM 10GE

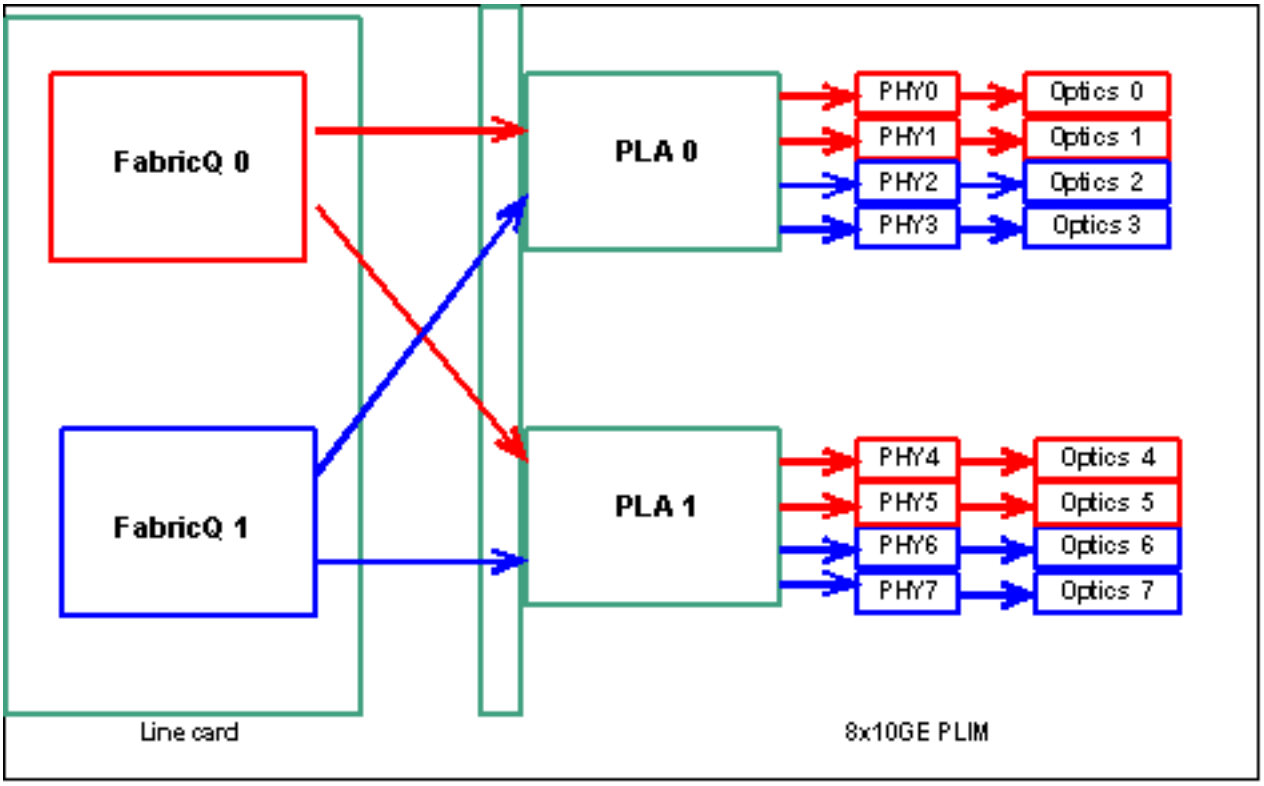
O PLIM 8 X 10G oferece a capacidade de terminar aproximadamente 80 Gbps de tráfego, enquanto a capacidade de encaminhamento do MSC é no máximo de 40 Gbps. Se todas as portas disponíveis no PLIM forem preenchidas, a sobreassinatura ocorrerá e a modelagem de QoS se tornará extremamente importante para garantir que o tráfego premium não seja descartado inadvertidamente. Para alguns, o excesso de assinaturas não é uma opção e deve ser evitado. Apenas quatro das oito portas devem ser usadas para fazer isso. Além disso, é preciso ter cuidado para garantir que a largura de banda ideal dentro do MSC e do PLIM esteja disponível para cada uma das quatro portas.

Observação: o mapeamento de porta é alterado a partir da versão 3.2.2. Veja estes diagramas.

Mapeamento de portas até a versão 3.2.1



Mapeamento de portas a partir da versão 3.2.2



Como mencionado anteriormente, as portas físicas são atendidas por um dos dois ASICs de FabricQ. A atribuição de portas ao ASIC está definida estaticamente e não pode ser alterada. Além disso, o PLIM 8 X 10G tem dois ASICs PLA. As primeiras portas de serviços PLA são 0 a 3, e os segundos serviços 4 a 7. A capacidade de largura de banda de um único PLA no PLIM 8 X 10G é de aproximadamente 24 Gbps. A capacidade de switching de um único ASIC de FabricQ é de aproximadamente 62 Mpps.

Se você preencher a porta 0 a 3 ou as portas 4 a 7, a capacidade de largura de banda do PLA (24 Gbps) será compartilhada entre as quatro portas que restringem o throughput geral. Se você preencher as portas 0,2,4 & 6 (até 3.2.1) ou 0,1,4 & 5 (3.2.2 em diante), pois todas essas portas são atendidas pelo ASIC FabricQ, cuja capacidade de switching é de 62 Mpps, novamente, o que restringe a capacidade de throughput.

É melhor utilizar as portas de uma maneira que obtenha a mais alta eficiência dos PLAs e dos ASICs do FabricQ para alcançar o desempenho ideal.

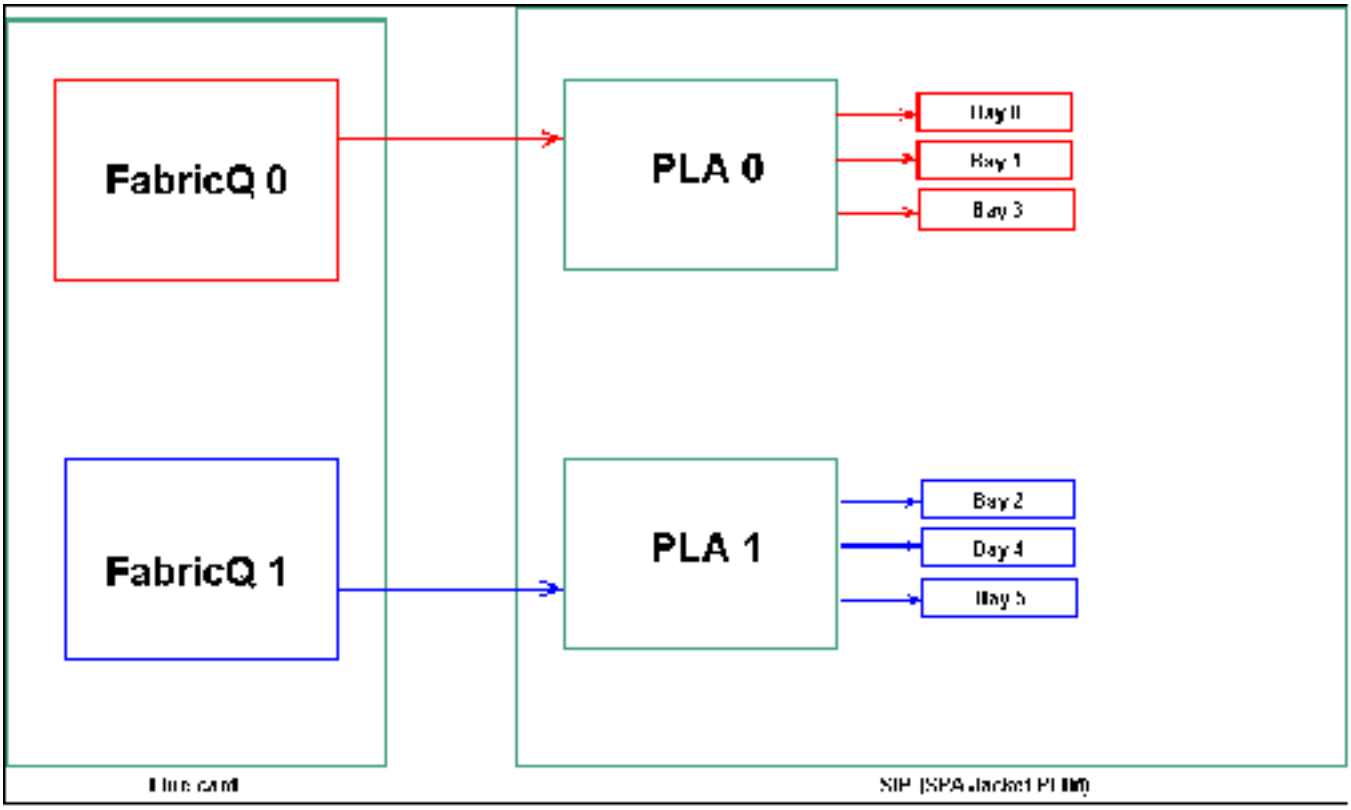
SIP-800/SPA

O SIP-800 PLIM oferece a capacidade de operar com placas de interface modulares conhecidas como Adaptadores de Porta de Serviço (SPAs - Service Port Adapters). O SIP-800 fornece 6 compartimentos de SPA com uma capacidade de interface teórica de 60 Gbps. A capacidade de encaminhamento do MSC é de no máximo 40 Gbps. Se todos os compartimentos do SIP-800 forem preenchidos, então, dependendo do tipo de SPA, é possível que ocorra excesso de assinaturas e a modelagem de QoS se torne extremamente importante para garantir que o tráfego premium não seja descartado inadvertidamente.

Observação: não há suporte para excesso de assinatura com interfaces POS. No entanto, a colocação da SPA POS de 10 Gb deve ser apropriada para garantir que a capacidade de throughput correta seja fornecida. A SPA Ethernet de 10 Gb só é suportada na versão IOS-XR 3.4. Esse SPA oferece recursos de assinatura excessiva.

Para alguns, o excesso de assinaturas não é uma opção e deve ser evitado. Para isso, só devem ser utilizados quatro dos sessenta. Além disso, deve-se tomar cuidado para garantir que a largura de banda ideal dentro do MSC e do PLIM esteja disponível para cada uma das quatro portas.

Mapeamento de compartimento de SPA



Como mencionado anteriormente, as portas físicas são atendidas por um dos dois ASICs de FabricQ. A atribuição de portas ao ASIC está definida estaticamente e não pode ser alterada. Além disso, o SIP-800 PLIM tem dois ASICs PLA. As primeiras portas de serviços PLA 0,1 e 3, os segundos serviços 2, 4 e 5.

A capacidade de largura de banda de um único PLA no SIP-800 PLIM é de aproximadamente 24 Gbps. A capacidade de switching de um único ASIC de FabricQ é de aproximadamente 62 Mpps.

Se você preencher as portas 0,1 e 3 ou 2, 4 e 5, a capacidade de largura de banda do PLA (24 Gbps) será compartilhada entre as três portas que restringem o throughput geral. Como um único FabricQ atende a esses grupos de portas, a taxa máxima de pacotes do grupo de portas é de 62 Mpps. É melhor utilizar as portas de uma maneira que obtenha a mais alta eficiência dos PLAs para alcançar a largura de banda ideal.

Posicionamento sugerido:

	Nº do compartimento de SPA	Nº do compartimento de SPA	Nº do compartimento de SPA	Nº do compartimento de SPA
Opção 1	0	1	4	5
Opção 2	1	2	3	4

Se você quiser preencher a placa com mais de quatro SPA, a recomendação é concluir uma das opções listadas anteriormente, que distribui as interfaces entre os dois grupos de portas (0,1 e 3 e 2,4 e 5). Em seguida, você deve colocar os próximos módulos SPA em uma das portas abertas nos grupos de portas 0,1 e 3 e 2,4 e 5.

DWDM XENPACKs

A partir da versão 3.2.2, os DWDM XENPACKs podem ser instalados e fornecer módulos ópticos **ajustáveis**. Os requisitos de resfriamento desses módulos XENPACK exigem que haja um slot em branco entre os módulos instalados. Além disso, se um único módulo DWDM XENPACK estiver instalado, um máximo de quatro portas poderá ser usado, mesmo que os módulos XENPACK não sejam dispositivos DWDM. Isso, portanto, tem um impacto direto no mapeamento de FabricQ para PLA para porta. É necessário prestar atenção a este requisito e é considerado neste quadro.

Posicionamento sugerido:

	Nº da porta óptica	Nº da porta óptica	Nº da porta óptica	Nº da porta óptica
Opção 1 ou DWDM XENPACK	0	2	5	7
Opção 2	1	3	4	6

Para uma instalação 3.2.2 ou posterior ou 3.3, evite a alteração do mapeamento de FabricQ. Um padrão de posicionamento mais simples pode, portanto, ser usado para módulos regulares e DWDM XENPACK.

	Nº da porta óptica	Nº da porta óptica	Nº da porta óptica	Nº da porta óptica
Opção 1	0	2	4	6
Opção 2	1	3	5	7

Se você quiser preencher a placa com mais de quatro portas XENPACK não-DWDM, recomenda-se concluir uma das opções listadas, que espalha os módulos de interface óptica entre os dois grupos de portas (0-3 e 4-7). Em seguida, você precisa colocar os próximos módulos de interface

óptica em uma das portas abertas nos grupos de portas 0-3 ou 4-7. Se você usar o grupo de portas 0-3 para o módulo de interface óptica nº 5, os módulos de interface óptica nº 6 devem ser colocados no grupo de 4-7 portas.

Consulte os módulos DWDM XENPAK para obter mais detalhes.

Gerenciamento de configuração

A configuração no IOS-XR é feita através de uma configuração de dois estágios, a configuração é inserida pelo usuário no primeiro estágio. Este é o estágio em que somente a sintaxe de configuração é verificada pelo CLI. A configuração inserida neste estágio é conhecida somente pelo processo do agente de configuração, por exemplo, CLI/XML. A configuração não é verificada, pois não está gravada no servidor sysdb. O aplicativo backend não é notificado e não pode acessar ou ter nenhum conhecimento da configuração nesta etapa.

No segundo estágio, a configuração é explicitamente confirmada pelo usuário. Nesta etapa, a configuração é gravada no servidor sysdb, os aplicativos de backend verificam se as configurações e notificações são geradas pelo sysdb. Você pode abortar uma sessão de configuração antes de confirmar a configuração inserida no primeiro estágio. Portanto, não é seguro supor que toda configuração inserida no estágio um é sempre comprometida no estágio dois.

Além disso, a operação e/ou a configuração de execução do roteador podem ser modificadas por vários usuários durante o estágio um e o estágio dois. Portanto, qualquer teste de roteador que executa a configuração e/ou o estado operacional no estágio um pode não ser válido no estágio dois em que a configuração é realmente comprometida.

Sistemas de arquivos de configuração

O Configuration File System (CFS) é um conjunto de arquivos e diretórios usados para armazenar a configuração do roteador. O CFS é armazenado no diretório disk0:/config/, que é a mídia padrão usada no RP. Os arquivos e diretórios no CFS são internos ao roteador e nunca devem ser modificados ou removidos pelo usuário. Isso pode resultar em perda ou corrupção da configuração e afetar o serviço.

O CFS é apontado para o standby-RP após cada confirmação. Isso ajuda a preservar o arquivo de configuração do roteador após um failover.

Durante a inicialização do roteador, a última configuração ativa é aplicada do banco de dados de confirmação de configuração armazenado no CFS. Não é necessário que o usuário salve manualmente a configuração ativa após cada confirmação de configuração, pois isso é feito automaticamente pelo roteador.

Não é aconselhável fazer alterações de configuração enquanto a configuração está sendo aplicada durante a inicialização. Se o aplicativo de configuração não estiver concluído, você verá esta mensagem quando fizer logon no roteador:

Processo de configuração do sistema

A configuração de inicialização deste dispositivo está sendo carregada no momento. Isso pode levar alguns minutos. Você será notificado após a conclusão. Não tente reconfigurar o dispositivo

até que esse processo esteja concluído. Em alguns casos raros, pode ser desejável restaurar a configuração do roteador de um arquivo de configuração ASCII fornecido pelo usuário em vez de restaurar a última configuração ativa do CFS.

Você pode forçar a aplicação de um arquivo de configuração:

```
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>
```

Enquanto você restaura a configuração do roteador, um ou mais itens de configuração podem não entrar em vigor. Toda configuração com falha é salva no CFS e é mantida até o próximo recarregamento.

Você pode navegar pela configuração com falha, corrigir os erros e reaplicar a configuração.

Estas são algumas dicas para endereçar a configuração com falha durante a inicialização do roteador.

No IOX, a configuração pode ser classificada como configuração com falha por três motivos:

1. Erros de sintaxe—O analisador gera erros de sintaxe, que geralmente indicam que há uma incompatibilidade com comandos CLI. Você deve corrigir os erros de sintaxe e reaplicar a configuração.
2. Erros semânticos—Erros semânticos são gerados pelos componentes de back-end quando o gerenciador de configuração restaura a configuração durante a inicialização do roteador. É importante observar que o cfgmgr não é responsável por garantir que a configuração seja aceita como parte da configuração em execução. O Cfgmgr é apenas um **intermediário** e apenas relata quaisquer falhas semânticas que os componentes de back-end geram. Cabe a cada proprietário do componente de back-end analisar o motivo da falha e determinar o motivo da falha. Os usuários podem executar os **comandos <CLI>** no modo de configuração para localizar facilmente o proprietário do verificador de componente de back-end. Por exemplo, se o **roteador bgp 217** mostrar a configuração com falha, o comando **description** mostra que o verificador do componente é o componente 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. Aplicar erros—A configuração foi verificada e aceita com êxito como parte da configuração em execução, mas o componente de back-end não pode atualizar seu estado operacional por algum motivo. A configuração é mostrada na configuração em execução, pois foi verificada corretamente e como falha na configuração devido ao erro operacional de back-end. O comando **description** pode ser executado novamente na CLI que não foi aplicada para localizar o proprietário da aplicação do componente.Conclua estes passos para navegar e reaplicar a configuração com falha durante os operadores de inicialização:Para os operadores R3.2 podem usar este procedimento para reaplicar a configuração com falha:Os operadores podem usar o comando **show configuration failed startup** para navegar pela configuração com falha salva durante a inicialização do roteador.Os operadores devem executar o **comando show configuration failed startup noerror** comando **| file myfailed.cfg** para salvar a configuração com falha na inicialização em um arquivo.Os operadores devem ir para o modo **de configuração** e usar comandos **load/commit** para reaplicar esta configuração com falha:

```
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
```

Para operadores de imagens R3.3, use este procedimento atualizado:Os operadores devem usar o comando **show configuration failed startup** e o comando **load configuration failed startup** para navegar e reaplicar qualquer configuração com falha.

```
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 Kernel

Por padrão, o IOS-XR grava um dump central no disco rígido caso haja um travamento do processo, mas não se o kernel em si falhar. Observe que para um sistema multichassi, essa funcionalidade é atualmente suportada apenas para o chassi 0 da placa de linha. O outro chassi é suportado em uma versão futura do software.

Sugere-se que os dumps de kernel para RPs e MSCs sejam habilitados com o uso dessas configurações nas configurações padrão e no modo de administração:

```
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
```

Configuração de despejo de kernel

Isso resulta nessa ocorrência para um travamento de kernel:

1. Um RP trava e um dump é gravado no disco rígido desse RP no diretório raiz do disco.
2. Se um MSC travar, um dump será gravado no disco rígido de RP0 no diretório raiz do disco.

Isso não tem impacto nos tempos de failover do RP, pois o encaminhamento ininterrupto (NSF) está configurado para os protocolos de roteamento. Pode levar alguns minutos a mais para que o RP ou a placa de linha com falha se tornem disponíveis novamente depois de um travamento enquanto ele escreve o núcleo.

Um exemplo da adição dessa configuração à configuração do modo padrão e admin é mostrado aqui. Observe que a configuração do modo admin exige que os DRPs sejam usados.

Esta saída mostra um exemplo de configuração de despejo de kernel:

```
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)#
```

Security

LPTS

O Local Packet Transport Services (LPTS) lida com pacotes destinados localmente. O LPTS é feito de vários componentes diferentes.

1. O principal é chamado de processo de arbitragem de portas. Ele ouve solicitações de soquete de diferentes processos de protocolo, por exemplo, BGP, IS-IS e rastreia todas as informações de vinculação para esses processos. Por exemplo, se um processo BGP escuta no soquete número 179, o PA obtém essas informações dos processos BGP e, em seguida, atribui uma associação a esse processo em um IFIB.
2. O IFIB é outro componente do processo LPTS. Ele ajuda a manter um diretório de onde está um processo que está ouvindo uma ligação de porta específica. O IFIB é gerado pelo processo de arbitragem de portas e é mantido com o árbitro de portas. Em seguida, gera vários subconjuntos dessas informações.O primeiro subconjunto é uma fatia do IFIB. Essa fatia pode ser associada ao protocolo IPv4 e assim por diante. As fatias são então enviadas aos gerentes de fluxo apropriados, que usam a fatia IFIB para encaminhar o pacote ao processo apropriado.O segundo subconjunto é um pré-IFIB, permite que o LC encaminhe o pacote para o processo apropriado se houver apenas um processo ou para um gerenciador de fluxo apropriado.
3. Os gerenciadores de fluxo ajudam a distribuir ainda mais os pacotes se a pesquisa não for trivial, por exemplo, vários processos para o BGP. Cada gerenciador de fluxo tem uma fatia ou várias fatias do IFIB e encaminha corretamente pacotes para os processos apropriados associados à fatia do IFIB.
4. Se uma entrada não for definida para a porta de destino, ela poderá ser liberada ou encaminhada para o gerenciador de fluxo. Um pacote é encaminhado sem porta associada se houver uma política associada para a porta. O gerenciador de fluxo ajuda a gerar uma nova entrada de sessão.

Como um pacote interno é encaminhado?

Há dois tipos de fluxos: fluxos de Camada 2 (HDLC, PPP) e fluxos e fluxos de roteamento ICMP/PING de Camada 4.

1. Camada 2 HDLC/PPP—Esses pacotes são identificados pelo identificador do protocolo e enviados diretamente às filas da CPU no Sprayer. Os pacotes de protocolo da camada 2 têm alta prioridade e são capturados pela CPU (via Squid) e processados. Portanto, os keepalives da camada 2 são respondidos diretamente via LC através da CPU. Isso evita a necessidade de ir ao RP para respostas e se reproduz com o tema de gerenciamento de interface distribuída.

2. Os pacotes ICMP (Camada 4) são recebidos no LC e são enviados através de pesquisa pelo IFBI para as filas de CPU no Sprayer. Esses pacotes são enviados à CPU (via Squid) e processados. A resposta é então enviada através das filas de saída do Sprayer para ser encaminhada através da estrutura. Nesse caso, outro aplicativo também precisa das informações (replicadas através da estrutura). Uma vez na estrutura, o pacote é destinado ao LC de saída apropriado e através da esponja e fila de controle apropriadas.
3. Os fluxos de roteamento são pesquisados no IFIB e enviados às filas de modelagem de saída (8000 filas), uma das quais é reservada para pacotes de controle. Esta é uma fila não modelada e é simplesmente atendida toda vez que está cheia. - alta prioridade. O pacote é então enviado através da estrutura em filas de alta prioridade em um conjunto de filas de CPU no Espongue (semelhante às filas Squid no Sprayer) e processado pelo processo apropriado, pelo gerenciador de fluxo ou pelo processo real. Uma resposta é enviada de volta pela esponja da placa de linha de saída e, em seguida, pela placa de linha. A esponja LC de saída tem uma fila especial reservada para tratar pacotes de controle. As filas no Espongue são divididas em pacotes de alta prioridade, controle e baixa prioridade, por porta de saída.
4. O PSE tem um conjunto de vigilantes configurados para limitar a taxa de pacotes de Camada 4, Camada 2 e roteamento. Eles são predefinidos e alterados para serem configuráveis pelo usuário posteriormente.

Um dos problemas mais comuns com o LPTS são os pacotes que são descartados quando você tenta fazer ping no roteador. Os vigilantes de LPTS geralmente estão limitando a taxa desses pacotes. Este é o caso para confirmar:

```
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

Os pacotes IP são inerentemente inseguros. O IPsec é um método usado para proteger os pacotes IP. O CRS-1 IPsec é implementado no caminho de encaminhamento de software, portanto, a sessão IPsec é terminada no RP/DRP. Há suporte para um número total de 500 sessões IPsec por CRS-1. O número depende da velocidade da CPU e dos recursos alocados. Não há limitação de software para isso, somente o tráfego originado localmente e terminado localmente no RP são elegíveis para tratamento de IPsec. O modo de transporte IPsec ou o modo de túnel podem ser usados para o tipo de tráfego, embora o primeiro seja preferido devido à menor sobrecarga no processamento IPsec.

R3.3.0 suporta a criptografia de BGP e OSPFv3 sobre IPsec.

Consulte o Guia de Configuração de Segurança do Sistema do Cisco IOS XR para obter mais informações sobre como implementar o IPsec.

Observação: o IPsec requer a pizza de criptografia, por exemplo, hfr-k9sec-p.pie-3.3.1.

Fora da banda

Console e acesso AUX

Os RP/SCs CRS-1 têm uma porta de console e AUX disponíveis para fins de gerenciamento fora da banda, bem como uma porta Ethernet de gerenciamento para fora da banda via IP.

O console e a porta AUX de cada RP/SCGE, dois por chassi, podem ser conectados a um servidor de console. Isso significa que o sistema de chassi único requer quatro portas de console, e os sistemas de vários chassis exigem 12 portas mais duas portas adicionais para os Supervisor Engines no Catalyst 6504-E.

A conexão da porta AUX é importante, pois fornece acesso ao kernel IOS-XR e pode permitir a recuperação do sistema quando isso não é possível através da porta de console. O acesso através da porta AUX só está disponível para usuários definidos localmente no sistema e somente quando o usuário tiver acesso de nível de suporte do sistema raiz ou da Cisco. Além disso, o usuário deve ter uma senha **secreta** definida.

Acesso ao terminal virtual

O Telnet e o Secure Shell (SSH) podem ser usados para acessar o CRS-1 através das portas vty. Por padrão, ambos estão desativados e o usuário precisa ativá-los explicitamente.

Observação: o IPsec requer a pizza de criptografia, por exemplo, hfr-k9sec-p.pie-3.3.1.

Primeiro, gere chaves RSA e DSA como mostrado neste exemplo para ativar o 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
```

Informações Relacionadas

- Suporte a roteadores
- Suporte Técnico e Documentação - Cisco Systems