Передовой опыт эксплуатации CRS-1 и IOS XR

Содержание

Введение

Этот документ знакомит читателя со следующими понятиями:

- Процессы и потоки выполнения
- Коммутирующая матрица CRS-1
- Плоскость управления
- Монитор ROMMON и библиотека Monlib
- Интерфейсный модуль физического уровня (PLIM) и модульная обслуживающая плата (MSC)
- Управление конфигурацией
- Безопасность
- Внеполосное управление
- Упрощенный протокол управления сетью (SNMP)

Предварительные условия

Требования

Компания Cisco считает желательным предварительное знакомство читателя с операционной системой Cisco IOS® XR.

Используемые компоненты

Сведения, содержащиеся в данном документе, касаются следующих версий программного обеспечения и оборудования:

- Cisco IOS XR Software
- CRS-1

Сведения, представленные в этом документе, были получены от устройств, работающих в специальной лабораторной среде. Все устройства, описанные в этом документе, были запущены с чистой (стандартной) конфигурацией. В рабочей сети необходимо изучить потенциальное воздействие всех команд до их использования.

Условные обозначения

Дополнительные сведения об условных обозначениях см. в документе Условные обозначения технических терминов Cisco.

Обзор Cisco IOS XR

Операционная система Cisco IOS XR спроектирована в расчете на масштабируемость. Ядро системы основано на микроядерной архитектуре: оно реализует только важнейшие службы, в числе которых управление процессами, планирование, механизмы сигналов и таймеров. Все прочие службы, такие как файловые системы, драйверы, протокольные стеки и приложения, классифицируются как диспетчеры ресурсов и работают в защищенной пользовательской области памяти. Эти прочие службы можно добавлять и удалять во время работы, если программа спроектирована соответствующим образом. Микроядро занимает всего 12 кбайт. В качестве микроядра и базовой операционной системы используется ПО Neutrino компании QNX Software Systems. QNX специализируется на

разработке операционных систем реального времени. Микроядро использует принцип вытесняющей многозадачности с планировщиком на основе приоритетов. За счет этого переключение контекста выполнения процессов происходит чрезвычайно быстро, а потоки выполнения с наивысшим приоритетом при необходимости всегда получают доступ к центральному процессору. Это лишь некоторые из преимуществ, эффективно используемых в Cisco IOS XR. Однако главным преимуществом является структура межпроцессного взаимодействия, положенная в основу операционной системы.

В операционной системе Neutrino используется механизм передачи сообщений. Сообщения –основные средства межпроцессного взаимодействия для всех потоков выполнения. Сервер, предоставляющий ту или иную службу, создает канал для обмена сообщениями. Клиенты для пользования службами подключаются к серверам через непосредственную привязку к соответствующему дескриптору файла. Для любого обмена данными между клиентом и сервером используется один и тот же механизм. Это огромное преимущество для суперкомпьютера, каковым является система CRS-1. Например, в стандартном ядре UNIX операция локального чтения реализуется следующим образом:

- Ядро вызывается через программное прерывание.
- Ядро передает вызов в файловую систему.
- Происходит получение данных.

В случае удаленного доступа реализация будет следующей:

- Ядро вызывается через программное прерывание.
- Ядро передает вызов в сетевую файловую систему (NFS).
- NFS обращается к сетевому компоненту.
- Удаленная сторона передает вызов сетевому компоненту.
- Вызывается сетевая файловая система.
- Ядро передает вызов в файловую систему.

Семантика в случае локального и удаленного чтения неодинакова. Различаются аргументы и параметры функций блокировки файла и установки полномочий.

Теперь рассмотрим случай локального доступа в ОС QNX:

- Ядро вызывается через программное прерывание.
- Ядро передает сообщение файловой системе.

В случае удаленного доступа:

- Ядро вызывается через программное прерывание.
- Ядро входит в QNET — транспортный механизм для межпроцессного взаимодействия.
- QNET входит в ядро.
- Ядро передает вызов в файловую систему.

Вся семантика, связанная с передачей аргументов и параметров файловой системы, идентична. Развязка обеспечивается интерфейсом IPC, который позволяет полностью разделить клиентскую и серверную части. Это означает, что любой процесс в конкретное время может выполняться где угодно. Если один процессор маршрутизации перегружен обслуживанием запросов, то службы можно перенести с него на другой центральный процессор, функционирующий в составе распределенного процессора маршрутизации (DRP). Суперкомпьютер, в котором разные службы функционируют на различных центральных процессорах, распределенных по нескольким узлам с простым механизмом межузлового взаимодействия. Предоставляет необходимую инфраструктуру для масштабируемости. Компания Cisco воспользовалась этим преимуществом и написала

дополнительное программное обеспечение, которое встраивается в основные операции ядра передачи сообщений и позволяет масштабировать маршрутизатор CRS до нескольких тысяч узлов, где узел, в данном случае –процессор, исполняет свой экземпляр операционной системы, будь то процессор маршрутизации (RP), распределенный процессор маршрутизации (DRP), модульная плата служб (MSC) или процессор коммутации (SP).

Процессы и потоки выполнения

В системе Cisco IOS XR процессом именуется защищенная область памяти, содержащая один или несколько потоков выполнения. С точки зрения программиста эти потоки делают непосредственную работу; при этом каждый из них решает определенную задачу, следуя логически описанному пути выполнения. Память, необходимая потокам выполнения в течение их работы, принадлежит содержащему их процессу и защищена от потоков выполнения прочих процессов. Поток является единицей выполнения и имеет свой контекст выполнения, содержащий стек и регистры. Процесс –это группа потоков выполнения, совместно использующих виртуальное адресное пространство. Процесс может содержать только один поток выполнения, но чаще содержит их несколько. Процесс, пытающийся осуществить запись в память другого процесса, уничтожается системой. Если один процесс содержит несколько потоков выполнения, то потоки имеют доступ к одной и той же памяти в рамках процесса и, как следствие, могут затирать данные друг друга. Чтобы избежать подобных конфликтов между потоками выполнения в рамках процесса, следует реализовывать процедуры последовательно, с синхронизацией доступа к ресурсам.

Для исключительного доступа к службам в потоках выполнения используется объект, именуемый взаимоисключающим семафором (MUTEX). Например, потоку выполнения, имеющему MUTEX-семафор, может быть разрешена запись в определенную область памяти. Другие потоки выполнения, не имеющие MUTEX-семафора, не смогут этого делать. Существуют также другие механизмы обеспечения синхронизации доступа к ресурсам: семафоры состояния, условные переменные (condvar), барьеры и условия ожидания (sleepon). В настоящем документе они не рассматриваются, но они также предназначены для синхронизации. Применив обсуждаемые здесь принципы к системе Cisco IOS, можно рассматривать последнюю как одиночный процесс, управляющий несколькими потоками выполнения, все из которых имеют доступ к одному и тому же пространству памяти. Отличие состоит в том, что в Cisco IOS подобные потоки выполнения именуются «процессами».

Состояния процессов и потоков выполнения

В системе Cisco IOS XR имеются серверы, предоставляющие службы, а также клиенты, которые их используют. Отдельно взятый процесс может иметь множество потоков выполнения, реализующих одну и ту же службу. У другого процесса может иметься множество клиентов, которым в произвольное время может потребоваться конкретная служба. Серверы доступны не всегда: клиент, запросивший доступ к службе, переходит в состояние ожидания до тех пор, пока сервер не освободится. Такое состояние называется блокировкой клиента. А сама модель –блокируемой моделью «клиент-сервер». Клиент может быть блокирован по причине ожидания ресурса, например MUTEX-семафора, или из-за того, что сервер еще не ответил.

Для проверки состояния потоков выполнения в процессе ospf наберите команду show process 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
```

Обратите внимание на то, что процессу ospf присвоен идентификатор задания (JID), равный 250. Он никогда не меняется в работающем маршрутизаторе и в общем случае не меняется в пределах конкретной версии Cisco IOS XR. В процессе ospf имеется пять потоков выполнения, каждый из которых имеет собственный идентификатор потока выполнения (TID). В списке приводятся объем стека для каждого потока выполнения, приоритет каждого потока и его состояние.

Синхронная передача сообщений

Ранее упоминалось, что QNX –это операционная система на основе передачи сообщений. Фактически она относится к системам с синхронной передачей сообщений. Принцип синхронного обмена сообщениями наделяет операционную систему рядом особенностей. Которые нельзя отнести к недостаткам, но которые легко выдают существующие проблемы. Благодаря синхронности оператору системы CRS-1 легко доступна информация о жизненном цикле и состояниях процессов, что облегчает диагностику неполадок. Жизненный цикл передачи сообщения имеет следующий общий вид:

- Сервер создает канал сообщения.
- Клиент подключается к каналу сервера (аналогично команде open в POSIX).
- Клиент посылает сообщение серверу (MsgSend) и ждет ответа в состоянии блокировки.
- Сервер получает (MsgReceive) сообщение от клиента, обрабатывает сообщение и отвечает клиенту.
- Клиент выходит из блокировки и обрабатывает ответ сервера.

Эта блокирующая модель «клиент-сервер» описывает синхронную передачу сообщений. Клиент отправляет сообщение и блокируется. Сервер получает сообщение, обрабатывает его и отвечает клиенту, после чего клиент разблокируется. Вот как этот механизм выглядит в деталях:

- Сервер ожидает обращения в состоянии RECEIVE.
- Клиент посылает сообщение серверу и переходит в состояние блокировки (BLOCKED).
- Сервер получает сообщение и выходит из блокировки, если до этого пребывал в состоянии ожидания RECEIVE.
- Клиент переходит в состояние ответа (REPLY).
- Сервер переходит в состояние выполнения (RUNNING).
- Сервер обрабатывает сообщение.
- Сервер отвечает клиенту.
- Клиент выходит из состояния блокировки.

Для просмотра состояний клиента и серверов выполните команду show process.

```
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
```

Заблокированные процессы и состояния процессов

Команда show process blocked показывает процессы, находящиеся в заблокированном состоянии.

```
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#
```

Благодаря синхронной передаче сообщений можно легко отследить жизненный цикл межпроцессного взаимодействия между различными потоками выполнения. В конкретный момент поток выполнения может иметь одно из определенных состояний. Иногда состояние блокировки указывает на наличие проблемы. **Это не означает, что блокировка любого потока выполнения непременно является признаком проблемы, поэтому не следует на основании данных, сообщаемых командой show process blocked, регистрировать обращения в Службу технической поддержки Cisco.** Заблокированные потоки выполнения –совершенно обычная ситуация.

Обратите внимание на приведенный выше вывод команды. Можно заметить, что первым потоком выполнения в списке является программа ksh, заблокированная в состоянии ответа процессом devc-conaux. Клиент, которым в данном случае является ksh, отправил сообщение серверному процессу devc-conaux, который удерживает процесс ksh в заблокированном состоянии до получения ответа. Ksh –интерпретатор команд UNIX, с которым работает оператор, подключенный через консоль или порт AUX. Ksh ждет ввода в консоли. Если оператор ничего не вводит в консоли, то в это время процесс остается заблокированным, поскольку данные, которые требуется обрабатывать, отсутствуют. По завершении обработки поступивших данных ksh возвращается в состояние блокирования

процессом devc-conaux.

Это нормальное явление, и оно не указывает на существование проблемы. **Таким образом, заблокированные потоки выполнения следует считать нормой, а вывод команды show process blocked будет зависеть от версии XR, типа имеющейся системы и действий операторов. Команда show process blocked–хорошая отправная точка для диагностики проблем, связанных с операционной системой.** При наличии проблемы, например высокой загрузки центрального процессора, следует воспользоваться приведенной выше командой, чтобы определить, нет ли отклонений от нормы.

Важно получить представление о нормальном рабочем состоянии маршрутизатора. Чтобы в дальнейшем руководствоваться им при диагностике жизненного цикла процессов.

В каждый момент поток выполнения может иметь одно из определенных состояний. Эти состояния перечислены в следующей таблице:

Обозначение:	**Соответствующее состояние потока выполнения:**
Отключено	Отключено. Ядро ожидает высвобождения ресурсов потока выполнения.
РАБОТАЕТ	Активное состояние–поток выполняется центральным процессором
ГОТОВО	Поток не выполняется центральным процессором, но готов к выполнению
ОСТАНОВЛЕНО	Поток приостано

	влен (сигнал SIGSTOP)
Send _____ _____	Ожидание приема сообщения сервером
ПОЛУЧИТЬ	Ожидание отправки сообщения клиентом
ОТВЕТ	Ожидание ответа на сообщение со стороны сервера
СТЕК	Ожидание выделения дополнительного объема стека
WAITPAGE	Ожидание диспетчера процессов для разрешения ситуации с отсутствующей страницей
SIGSUSPEND	Ожидание сигнала
SIGWAITINFO	Ожидание сигнала
NANOSLEEP	Бездействие в течение заданного времени
MUTEX	Ожидание получения MUTEX-семафора

CONDVAR	Ожидание сигнала условной переменной
JOIN	Ожидание завершения другого потока выполнения
INTR	Ожидание прерывания
SEM	Ожидание получения семафора

Важнейшие процессы и их функции

В системе IOS Cisco XR имеется множество процессов. Ниже перечислены важнейшие из них с пояснением их функций.

Контрольный системный монитор (WDSysmon)

Эта служба предназначена для обнаружения зависших процессов и исчерпания памяти. Память может быть исчерпана в результате утечки памяти или некоторого другого постороннего обстоятельства. Зависание может быть вызвано множеством причин: неразрешимая блокировка процессов, бесконечный цикл, остановка выполнения в ядре или ошибки планирования. В любой многопоточной среде система может войти в состояние неразрешимой блокировки (deadlock). Неразрешимая блокировка возникает при невозможности продолжения работы одного или нескольких потоков из-за конфликта пользования ресурсами. Например, поток выполнения A может отправить сообщение потоку B, в то время как поток B отправляет сообщение потоку A. Оба потока выполнения будут ждать друг друга и могут оказаться в состоянии блокировки отправки (SEND), которое будет длиться бесконечно. Это простой случай с участием двух потоков выполнения, но если сервер отвечает за ресурс, используемый многими потоками, то блокировка на одном из потоков выполнения может привести к бесконечной блокировке в состоянии SEND сразу нескольких потоков, пытающихся получить доступ к этому ресурсу.

Возникнув с участием нескольких потоков выполнения, неразрешимая блокировка может в результате отразиться на других потоках выполнения. Избежать неразрешимой блокировки помогает грамотная разработка ПО, но даже при квалифицированном подходе к проектированию и написанию программы. Особая последовательность событий при конкретном характере данных и хронологии событий иногда может вызвать неразрешимую блокировку. Неразрешимые блокировки не всегда детерминированы и в общем случае трудновоспроизводимы. Процесс WDSysmon имеет несколько потоков выполнения, один из которых выполняется с наивысшим приоритетом, поддерживаемым ядром Neutrino, 63. Выполнение с приоритетом 63 гарантирует получение потоком своей доли процессорного времени в среде с вытесняющей многозадачностью на основе приоритетов. WDSysmon

взаимодействует с аппаратной контрольной схемой и управляет программными процессами, выявляющими состояния зависания. При обнаружении подобных состояний WDSysmon собирает дополнительную информацию о возникшем состоянии и может сформировать дамп памяти процесса или ядра, оставить запись в журнале SYSLOG, выполнить сценарий или уничтожить неразрешимо заблокированный процесс. В зависимости от серьезности проблемы он также может инициировать аварийное переключение процессора маршрутизации для поддержания работоспособности системы.

```
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
```

Процесс WDSysmon имеет девять потоков выполнения. Четыре потока работают с приоритетом 10, остальные –с приоритетами 11, 19, 20 и 63. При разработке процесса программист тщательно взвешивает приоритеты, которые должны быть предоставлены каждому потоку выполнения в процессе. Как обсуждалось ранее, планировщик руководствуется системой приоритетов: поток выполнения с более высоким приоритетном всегда вытесняет поток с более низким приоритетом. Приоритет 63 –наивысший приоритет, который может назначаться потоку выполнения. В данном примере он назначен потоку 7. Поток выполнения 7 соответствует контрольному механизму, выявляющему источники высокой нагрузки на центральный процессор. Чтобы выполнять возложенные на него функции в любых условиях, он должен всегда выполняться с более высоким приоритетом, чем другие потоки выполнения.

Netio

В системе Cisco IOS различаются понятия быстрой коммутации (fast switching) и коммутации в контексте процесса (process switching). Быстрая коммутация использует код экспресс-пересылки (CEF) и выполняется в контексте прерывания. Коммутация в контексте процесса использует код IP-коммутации ip_input, представляющий собой процесс под управлением планировщика. В более мощных платформах коммутация CEF выполняется аппаратно, а процесс ip_input планируется центральным процессором. Эквивалентом ip_input в системе Cisco IOS XR является 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
```

Процесс групповых служб (GSP)

В любом суперкомпьютере с несколькими тысячами узлов с собственными экземплярами

ядра возникает задача обмена данными. В Интернете обмен данными по схеме «один ко многим» эффективно реализуется протоколами многоадресной рассылки. GSP –внутренний протокол многоадресной рассылки, используемый для межпроцессного взаимодействия в рамках системы CRS 1. GSP обеспечивает надежный бессеансовый групповой обмен данными по схеме «один ко многим» с асинхронной семантикой. Это является залогом масштабируемости GSP до тысячи узлов.

```
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
```

Механизм массовой загрузки содержимого (BCDL)

BCDL используется для надежной многоадресной рассылки данных на различные узлы, в т.ч. процессоры маршрутизации и модульные обслуживающие платы. В качестве основного транспорта он использует протокол GSP. **BCDL обеспечивает доставку сообщений с сохранением последовательности.** В BCDL различаются понятия агента, источника и потребителя. Агент –это процесс, который взаимодействует с источником для извлечения и буферизации данных перед их рассылкой потребителям через многоадресный механизм. Источник –это процесс, который выдает данные, интересующие всех получателей, а потребитель –это процесс, заинтересованный в получении данных от производителя. BCDL используется при обновлении программного обеспечения Cisco IOS XR.

Облегченный механизм обмена сообщениями (LWM)

LWM –изобретенный компанией Cisco способ обмена сообщениями, предусматривающий создание уровня абстракции между приложениями, которые через средства межпроцессного взаимодействия обмениваются данными друг с другом и с ядром Neutrino. LWM призван обеспечить независимость от операционной системы и транспортного уровня. Если компания Cisco пожелает сменить поставщика операционной системы, отказавшись от QNX, то наличие уровня абстракции между элементарными функциями базовой операционной системы поможет предотвратить зависимость от операционной системы и упростит адаптацию программ для другой операционной системы. LWM обеспечивает синхронную гарантированную доставку сообщений, которая, как и собственный механизм передачи сообщений в ядре Neutrino, переводит отправителя в состояние блокировки до получения ответа от получателя.

LWM также предусматривает асинхронную доставку сообщений посредством 40-битовых «импульсов». Асинхронные сообщения характеризуются асинхронным механизмом отправки, при котором сообщение помещается в очередь и отправитель не блокируется, но не асинхронным получением: сервер опрашивает наличие следующего доступного сообщения. С точки зрения структуры LWM относится к системе «клиент-сервер». **Сервер создает канал –«ухо»—для прослушивания сообщений и не завершает работу, в то время как цикл прослушивает созданный канал для приема сообщений.** Прибывшее сообщение разблокирует сервер с присвоением идентификатора клиента, который фактически соответствует идентификатору приема из полученного сообщения. Далее сервер выполняет

обработку, после чего формирует ответное сообщение на идентификатор клиента.

На стороне клиента выполняется подключение к каналу сообщений. Передается идентификатор стороны, к которой произошло подключение, и затем после отправки сообщения происходит блокировка. По завершении обработки сервер отвечает и клиент разблокируется. Такой механизм фактически не отличается от собственного механизма передачи сообщений Neutrino — иными словами, прослойка абстракции весьма тонкая.

Для повышения производительности в LWM число системных вызовов и переключений контекста сведено к минимуму. LWM –предпочтительный метод межпроцессного взаимодействия в среде Cisco IOS XR.

Envmon

На базовом уровне системный монитор окружающей среды отвечает за предупреждение о выходе за границы допустимого диапазона физических параметров, например температуры, напряжения, скорости вентиляторов и т.п., а также за останов оборудования, приближающегося к критическим уровням, при которых возможен его выход из строя. Этот процесс периодически контролирует каждый доступный аппаратный датчик, сравнивает взвешенное значение с определенными пороговыми уровнями для конкретных плат и при необходимости инициирует события сигнализации в соответствии со своим предназначением. Постоянно действующий процесс, запускаемый во время инициализации системы, периодически опрашивает все аппаратные датчики в шасси, например датчики напряжения, температуры и скорости вентиляторов, и сообщает эти данные внешним клиентам управления. Другой, периодический, процесс сравнивает показания датчиков с порогами сигнализации и публикует оповещения о параметрах окружающей среды в системной базе данных для последующей обработки диспетчером обработки отказов (Fault Manager). Если показания датчиков выходят за пределы диапазона на опасную величину, то процесс контроля параметров окружающей среды может остановить работу платы.

Обзор коммутирующей матрицы CRS-1

- Многоступенчатая коммутирующая матрица на основе топологии Бенеша с 3-мя ступенями
- Динамическая маршрутизация в коммутирующей матрице для минимизации перегрузок в сети
- Принцип ячеек: 136-байтовые ячейки, содержащие 120 байт полезных данных
- Квитирование для эффективного разделения трафика и сведения к минимуму потребности в буферизации внутри коммутирующей матрицы
- Ускоренная межступенчатая доставка трафика
- Поддерживаются два вида доставки трафика (одноадресная и многоадресная)
- Для каждого вида доставки поддерживаются два приоритета трафика (высокий и низкий)
- Поддерживаются группы многоадресной рассылки коммутирующей матрицы 1M (FGID)
- Экономичная отказоустойчивость: Резервирование по схеме N+1 или N+k с использованием плоскостей коммутирующей матрицы в противоположность намного более дорогостоящей схеме 1+1

При эксплуатации в режиме одиночного шасси микросхемы S1, S2 и S3 расположены на одних и тех же платах коммутирующей матрицы. **Такую плату принято называть платой S123.** В конфигурации с несколькими шасси микросхема S2 обособлена и находится в

составе шасси плат коммутирующей матрицы (FCC). Для этой конфигурации необходима плоскость, образованная двумя платами коммутирующей матрицы: платой S2 и платой S13. Каждая модульная обслуживающая плата соединяется с восемью плоскостями коммутирующей матрицы. Благодаря такой избыточности в случае выхода из строя одной или нескольких плоскостей коммутирующая матрица продолжает передавать трафик, хотя ее совокупная пропускная способность снижается. CRS сохраняет физическую пропускную способность линии, если действуют всего семь плоскостей. Сигналы противодавления передаются по четной и нечетной плоскостям коммутирующей матрицы. Не рекомендуется эксплуатировать систему, если в ней нет хотя бы двух плоскостей –четной и нечетной. Конфигурации, число плоскостей в которых менее двух, не поддерживаются.

Плоскость коммутирующей матрицы

На приведенной выше диаграмме показана одна плоскость. Эту диаграмму нужно представить себе в восьми экземплярах. Это означает, что микросхема сегментации (ingressq) линейной платы (LC) соединяется с восемью микросхемами S1 (по одной микросхеме S1 для каждой плоскости). Микросхема S1 в каждой плоскости коммутирующей матрицы соединяется с восемью микросхемами сегментации:

- 8 верхних линейных плат (LC) в шасси
- 8 нижних линейных плат

Итого в 16-слотовом шасси LC имеется 16 микросхем S1: 8 для верхних линейных плат (по одной для каждой плоскости) + 8 для нижних линейных плат.

В одиночном 16-слотовом шасси плата коммутирующей матрицы S123 имеет 2 микросхемы S1, 2 микросхемы S2 и 4 микросхемы S3. Это одна из составляющих ускорения производительности коммутирующей матрицы. Объем трафика, выходящего из матрицы, увеличивается вдвое по сравнению с трафиком, входящим в коммутирующую матрицу. Кроме того, в настоящее время для каждой линейной платы —две микросхемы сборки и упорядочивания (fabricq) по сравнению с 1 микросхемой сегментации (Sprayer). Это делает возможной буферизацию на выходной линейной плате, когда несколько входных линейных плат перегружают выходную линейную плату. Выходная линейная плата, таким образом, справляется с увеличенной полосой пропускания коммутирующей матрицы.

Контроль коммутирующей матрицы

Доступность плоскостей и состояние их подключения:

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

Проверка приема/передачи ячеек плоскостями и увеличения счетчиков ошибок:

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

Сокращения в выводе предыдущей команды:

- CE –исправимая ошибка
- UCE –неисправимая ошибка
- PE –ошибка четности

Присутствие нескольких ошибок –не повод для опасений, поскольку подобные ситуации часто возникают при первоначальном запуске. Во время работы значения полей не должны увеличиваться. Иначе это может указывать на проблемы с коммутирующей матрицей. Для

получения картины ошибок по отдельным плоскостям коммутирующей матрицы выполните следующую команду:

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

Обзор плоскости управления

В плоскости управления связь между шасси линейной платы и шасси коммутирующей матрицы в настоящее время осуществляется через порты Gigabit Ethernet на процессорах маршрутизации (LCC) и плате SCGE (FCC). Соединение между портами реализуется парой коммутаторов Catalyst 6500, которые могут быть подключены через два или несколько портов Gigabit Ethernet.

Конфигурация Catalyst 6500

Для коммутаторов Catalyst, используемых в плоскости управления с несколькими шасси рекомендуется следующая конфигурация:

- На всех портах используется одна и та же сеть VLAN.
- Все порты работают в режиме доступа (без группообразования).
- Для предотвращения закольцовывания применяется остовное дерево 802.1w/s.
- Перекрестное соединение двух коммутаторов осуществляется посредством двух или нескольких линий, а для предотвращения закольцовывания применяется протокол STP. Устанавливать каналы не рекомендуется.
- Для портов, соединяемых с процессором маршрутизации и SCGE CRS-1, используется режим, основанный на предварительной редакции стандарта, поскольку стандартизованная версия 802.1s в системе IOS-XR не поддерживается.
- На портах, через которые соединяются друг с другом коммутаторы или через которые коммутаторы соединяются с процессорами маршрутизации (RP)/платами стоечных контроллеров Gigabit Ethernet (SCGE), необходимо включить обнаружение однонаправленных соединений (UDLD).
- По умолчанию в системе CRS-1 функция UDLD включена.

Процедура настройки коммутатора Catalyst 6500 в системе с несколькими стойками описана в документе Подготовка к использованию программного обеспечения Cisco IOS XR в системе с несколькими стойками.

Сопровождение плоскости управления в системе с несколькими шасси

Шасси Catalyst 6504-E, обеспечивающее соединение плоскости управления в системе с несколькими шасси, настроено для работы со следующими службами управления:

- Внутриполосное управление через гигабитный порт 1/2, подключаемый к сетевому коммутатору в каждой точке присутствия. Доступ разрешается только для узкого диапазона подсетей и протоколов.
- Для задания системного времени используется протокол NTP.
- Ведение системных журналов (SYSLOG) осуществляется на стандартных хостах.
- Для критических функций могут быть включены опрос и прерывания SNMP.

Примечание: Не допускается внесение каких-либо изменений в операции Catalyst. Любые планируемые изменения должны предварительно тестироваться; при этом настоятельно

рекомендуется выполнять подобные операции в период регламентного обслуживания.

Пример конфигурации управления:

```
#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}
```

Монитор ROMMON и библиотека Monlib

Monlib –исполняемая программа компании Cisco, которая хранится в устройстве и загружается в оперативную память для выполнения монитора ROMMON. ROMMON использует библиотеку Monlib для доступа к файлам на устройстве. Предусмотрена возможность обновления версий ROMMON. Обновление должно выполняться согласно рекомендациям Службы технической поддержки Cisco. Последняя версия ROMMON –1.40.

Указания по обновлению

Выполните следующие действия:

1. Загрузите двоичные файлы ROMMON на странице ROMMON для Cisco CRS-1 (только для зарегистрированных заказчиков).
2. Распакуйте файл TAR и скопируйте 6 двоичных файлов в корневой каталог CRS 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. **Проверьте текущую версию ROMMON при помощи команды show diag | inc ROM|NODE|PLIM.**`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. **Войдите в режим администрирования (ADMIN) и при помощи команды upgrade rommon a all disk0 произведите обновление 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. **Выйдите из режима ADMIN и введите show log | inc "OK, ROMMON A", после чего убедитесь в выполнении модернизации всех узлов.** Если с каким-либо узлом возникли затруднения, возвратитесь к шагу 4 и повторите запись программы.
```
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. **Войдите в режим администрирования (ADMIN) и при помощи команды upgrade rommon a all disk0 выполните обновление 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. **Выйдите из режима ADMIN и введите show log | inc "OK, ROMMON B", после чего убедитесь в выполнении модернизации всех узлов.** Если с каким-либо узлом возникли затруднения, возвратитесь к шагу 4 и повторите запись программы.
```
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. **Команда upgrade осуществляет только запись новой версии ROMMON в специально зарезервированный загрузочный раздел флеш-памяти.** При этом новая версия ROMMON остается неактивной до тех пор, пока плата не будет перезагружена. Новый модуль ROMMON активируется при перезагрузке платы. Для этого необходимо сбросить каждый узел по отдельности или произвести сброс всего маршрутизатора.
```
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. **Проверьте текущую версию ROMMON при помощи команды show diag | inc ROM|NODE|PLIM.**
```
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]
```
Примечание: На CRS 8 и шасси коммутационной фабрики ROMMON также настраивает скорость вентилятора на скорость по умолчанию, равную 4000 об/мин.

Краткое описание интерфейсного модуля физического уровня и модульной обслуживающей платы

Для заказных микросхем (ASIC), отвечающих за движение пакетов в маршрутизаторе CRS-1, используются следующие взаимозаменяемые термины:

Микросхема входной очереди (IngressQ) также именуется микросхемой сегментации (Sprayer).

Микросхема очереди коммутирующей матрицы (FabricQ) также именуется микросхемой сборки и упорядочивания (Sprayer).

Микросхема выходной очереди (EgressQ) также именуется микросхемой Sharq.

Процессор пакетов SPP также именуется микросхемой ядра коммутации пакетов (PSE).

Приемный PLIM > приемный SPP > входная очередь > коммутирующая матрица > очередь матрицы > передающий SPP > выходная очередь > передающий PLIM (Sprayer) (Sponge) (Sharq)

Пакеты принимаются интерфейсным модулем физического уровня (PLIM).

Интерфейсный модуль физического уровня содержит физические интерфейсы для модульной обслуживающей платы, с которой он стыкуется. Интерфейсный модуль физического уровня и модульная обслуживающая плата –отдельные платы, подключаемые через соединительную плату шасси. Как следствие, типы интерфейсов для конкретной платы MSC диктуются типом модуля PLIM, с которым стыкуется эта плата. В зависимости от типа модуля PLIM, плата содержит различное число заказных микросхем, которые обеспечивают для интерфейса взаимодействие с физической средой и передачу кадров.

Назначение заказных микросхем модуля PLIM –обеспечение интерфейса между модульной обслуживающей платой и физическими соединениями. Этот интерфейс является точкой подключения к оптоволокну, осуществляет преобразование световых сигналов в электрические, выполняет оконечную обработку формата кадров в передающей среде SDH/Sonet/Ethernet/HDLC/PPP, проверяет контрольные поля CRC, добавляет сведения для управления, именуемые «заголовком буфера», и пересылает остальные биты плате MSC. Модуль PLIM не является источником или потребителем пакетов поддержания активности (keepalive) в протоколах HDLC и PPP. Обработка этих пакетов осуществляется центральным процессором на плате MSC.

Модуль PLIM также реализует следующие функции:

- Фильтрация MAC-адресов для 1/10 Gigabit Ethernet
- Учет входящих/исходящих MAC-адресов для 1/10 Gigabit Ethernet
- Фильтрация по сетям VLAN для 1/10 Gigabit Ethernet
- Учет входящих/исходящих сетей VLAN для 1/10 Gigabit Ethernet
- Входная буферизация и уведомление о перегрузке

Выделение ресурсов интерфейсного модуля физического уровня сверх физического предела

Интерфейсный модуль физического уровня 10GE

Интерфейсный модуль физического уровня 8 X 10G позволяет осуществлять оконечную обработку трафика с пропускной способностью приблизительно 80 Гбит/с, в то время как пропускная способность модульной обслуживающей платы ограничена 40 Гбит/с. Если заняты все порты, доступные на модуле, то будет превышена физическая пропускная способность. В этом случае большую важность приобретает моделирование QoS, позволяющее избежать непреднамеренного удаления ценного трафика. В некоторых случаях превышение физической пропускной способности неприемлемо и должно быть исключено. Для этого должны использоваться только четыре из восьми указанных портов. Кроме того, необходимо следить за тем, чтобы каждому из четырех портов была доступна оптимальная полоса пропускания на плате MSC и в модуле PLIM.

Примечание: Сопоставление портов меняется, начиная с версии 3.2.2 и так далее. См. приведенные диаграммы.

Назначение портов до выпуска 3.2.1 Назначение портов в выпуске 3.2.2 и последующих выпусках

Как отмечалось ранее, физические порты обслуживаются одной из двух заказных микросхем FabricQ. Назначение портов микросхемам определено статически и не допускает возможности изменения. Кроме того, интерфейсный модуль физического уровня 8 X 10G имеет две заказных микросхемы PLA. Первая микросхема PLA обслуживает порты с 0 по 3, вторая –с 4 по 7. Полоса пропускания одной микросхемы PLA в конфигурации 8 X 10G PLIM составляет приблизительно 24 Гбит/с. Коммутационная емкость одной заказной микросхемы FabricQ –приблизительно 62 млн пакетов в секунду.

Если заняты порты с 0 по 3 или с 4 по 7, то полоса пропускания PLA (24 Гбит/с) будет распределена между всеми четырьмя портами, что ограничит общую пропускную способность. Если заняты порты 0, 2, 4 и 6 (до выпуска 3.2.1) или 0, 1, 4 и 5 (начиная с

выпуска 3.2.2), то, поскольку все эти порты обслуживаются одной микросхемой FabricQ, коммутационная емкость которой равна 62 млн пакетов в секунду, пропускная способность снова будет ограничена.

В интересах оптимизации производительности лучше выбирать используемые порты таким образом, чтобы платы PLA и микросхемы FabricQ работали наиболее эффективно.

SIP-800/SPA

Интерфейсный модуль физического уровня SIP 800 рассчитан на работу с модульными интерфейсными платами, именуемыми адаптерами портов служб (SPA). SIP 800 предусматривает 6 отсеков SPA с теоретической пропускной способностью интерфейса 60 Гбит/с. Максимальная пропускная способность платы MSC в режиме пересылки составляет 40 Гбит/с. Если заняты все отсеки на SIP 800, то, в зависимости от типа платы SPA, возможно превышение физической пропускной способности. В этом случае большую важность приобретает моделирование QoS, позволяющее избежать непреднамеренного удаления ценного трафика.

Примечание: Превышение лимита не поддерживается с интерфейсами POS. Тем не менее необходимо правильно подобрать место размещения платы SPA POS 10 Гбит/с, чтобы обеспечить требуемую пропускную способность. Плата SPA с интерфейсом 10 Гбит/с Ethernet поддерживается только в выпуске IOS-XR 3.4. Эта плата предусматривает возможность превышения физической пропускной способности.

В некоторых случаях превышение физической пропускной способности неприемлемо и должно быть исключено. Для этого должны использоваться только четыре из шести отсеков. Кроме того, необходимо следить за тем, чтобы каждому из четырех портов была доступна оптимальная полоса пропускания на плате MSC и в модуле PLIM.

Назначение отсеков SPA

Как отмечалось ранее, физические порты обслуживаются одной из двух заказных микросхем FabricQ. Назначение портов микросхемам определено статически и не допускает возможности изменения. Кроме того, интерфейсный модуль физического уровня SIP-800 имеет две заказных микросхемы PLA. Первая микросхема PLA обслуживает порты 0, 1 и 3, вторая –2, 4 и 5.

Полоса пропускания одной микросхемы PLA в составе модуля PLIM SIP-800 –приблизительно 24 Гбит/с. Коммутационная емкость одной заказной микросхемы FabricQ –приблизительно 62 млн пакетов в секунду.

Если заняты порты 0, 1 и 3 или 2, 4 и 5, то полоса пропускания PLA (24 Гбит/с) будет распределена между всеми тремя портами, что ограничит общую пропускную способность. Поскольку эти порты обслуживаются одной микросхемой FabricQ, то максимальная скорость передачи пакетов для группы портов составляет 62 млн пакетов в секунду. В интересах оптимизации производительности лучше выбирать используемые порты таким образом, чтобы платы PLA работали наиболее эффективно.

Рекомендуемая схема размещения:

	№	№	№ отсека	№ отсека

	отсека SPA	отсека SPA	SPA	SPA
Вариант 1	0	1	4	5
Вариант 2	1	2	3	4

Если требуется установить на плату более четырех модулей SPA, то рекомендуется прибегнуть к одному из изложенных выше вариантов, где интерфейсы распределены между двумя группами портов (0, 1 и 3, а также 2, 4 и 5). Последующие модули SPA в этом случае размещаются в одном из открытых портов любой из групп: 0, 1 и 3 или 2, 4 и 5.

Модули DWDM XENPACK

Начиная с выпуска 3.2.2, предусмотрена возможность установки настраиваемых оптоволоконных модулей DWDM XENPACK. Для охлаждения модулей XENPACK необходим свободный слот между установленными модулями. Кроме того, при наличии одного установленного модуля DWDM XENPACK может использоваться не более четырех портов, даже если модули XENPACK не являются устройствами DWDM. Это прямым образом влияет на взаимную привязку микросхем FabricQ, плат PLA и портов. Это требование необходимо принять во внимание. Оно отражено в следующей таблице.

Рекомендуемая схема размещения:

	№ оптоволоконного порта	№ оптоволоконного порта	№ оптоволоконного порта	№ оптоволоконного порта
Вариант 1 или DWDM XENPACK	0	2	5	7
Вариант 2	1	3	4	6

Если установлен выпуск 3.2.2 или более поздний либо выпуск 3.3, не изменяйте назначений FabricQ. Таким образом, для обычных модулей и модулей DWDM XENPACK можно применять более простую схему размещения.

	№ оптоволоконного порта	№ оптоволоконного порта	№ оптоволоконного порта	№ оптоволоконного порта
Вариант 1	0	2	4	6
Вариант 2	1	3	5	7

Если требуется установить на плату более четырех модулей XENPACK без DWDM, то

рекомендуется прибегнуть к одному из изложенных выше вариантов, где модули оптоволоконных интерфейсов распределены между двумя группами портов (с 0 по 3 и с 4 по 7). Последующие модули оптоволоконных интерфейсов в этом случае размешаются в одном из открытых портов любой из групп: 0–3 или 4–7. Если для модуля оптоволоконного интерфейса № 5 используется группа портов 0–3, то модуль оптоволоконного интерфейса № 6 должен быть размещен в группе портов 4–7.

Дополнительные сведения см. в документе Модули DWDM XENPAK.

Управление конфигурацией

Конфигурация в IOS-XR задается в два этапа. На первом этапе конфигурацию вводит пользователь. На этом этапе интерфейс командной строки (CLI) проверяет только синтаксис конфигурации. Конфигурация, введенная на этом этапе, известна только процессу агента конфигурации, например CLI/XML. Конфигурация не проверяется, поскольку в сервер sysdb она не записывается. Внутреннее приложение на этом этапе не уведомляется и не имеет доступа к конфигурации или сведений о ней.

На втором этапе конфигурация в явном виде сохраняется пользователем. При этом конфигурация записывается на сервер sysdb, внутреннее приложение проверяет конфигурацию, и для sysdb формируются уведомления. Сеанс настройки можно прервать до сохранения конфигурации, введенной на первом этапе. Поэтому не следует исходить из того, что любая конфигурация, введенная на первом этапе, всегда сохраняется на втором этапе.

Кроме того, рабочая и/или эксплуатационная конфигурация маршрутизатора может быть на первом и втором этапах изменена несколькими пользователями. Поэтому проверка маршрутизатора с рабочей конфигурацией и/или рабочим состоянием, пройденная на первом этапе, может на втором этапе при непосредственном сохранении конфигурации оказаться недействительной.

Файловые системы конфигурации

Файловая система конфигурации (CFS) представляет собой набор файлов и каталогов, используемых для хранения конфигурации маршрутизатора. CFS хранится в каталоге disk0:/config/, который соответствует носителю по умолчанию в процессоре маршрутизации. Файлы и каталоги CFS являются внутренними данными маршрутизатора и не подлежат изменению или удалению пользователем. Такое вмешательство может привести к потере или искажению конфигурации и скажется на работе устройства.

При каждом сохранении конфигурации на резервном процессоре маршрутизации сохраняется контрольная версия CFS. Это помогает сохранять файл конфигурации маршрутизатора после аварийного переключения.

При начальной загрузке маршрутизатора вводится в действие последняя активная конфигурация из базы данных, хранящейся в CFS. Пользователю не требуется вручную сохранять активную конфигурацию после записи каждой конфигурации, поскольку это автоматически делает маршрутизатор.

Не рекомендуется выполнять изменения конфигурации в то время, когда конфигурация вводится в действие в ходе начальной загрузки. Если конфигурация применена не

полностью, то при входе в маршрутизатор появится следующее сообщение:

System Configuration Process (Процесс конфигурации системы)

В это время подгружается загрузочная конфигурация устройства. Этот процесс может занять несколько минут. О его завершении вы будете уведомлены. Не пытайтесь изменять конфигурацию устройства до тех пор, пока этот процесс не будет завершен. В ряде редких случаев может быть желательно восстановить конфигурацию маршрутизатора из пользовательского файла конфигурации ASCII вместо восстановления последней активной конфигурации из системы CFS.

Можно принудительно ввести в действие файл конфигурации одним из следующих способов:

```
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=<config-file-path> rommon>boot <image>
```

Во время восстановления конфигурации маршрутизатора могут не вступить в силу один или несколько элементов конфигурации. Все ошибочные элементы конфигурации сохраняются в CFS до следующей перезагрузки.

Вы можете просмотреть ошибочную конфигурацию, исправить ошибки и повторно применить конфигурацию.

Ниже приведено несколько советов по устранению ошибок в конфигурации во время запуска маршрутизатора.

В системе IOX конфигурация может быть классифицирована как ошибочная по трем причинам:

1. Синтаксические ошибки. Синтаксический анализатор выдает синтаксические ошибки, которые обычно указывают на несовместимость с командами CLI. Необходимо самостоятельно исправить синтаксические ошибки и повторно применить конфигурацию.
2. Семантические ошибки. Семантические ошибки генерируются внутренними компонентами во время восстановления конфигурации соответствующим диспетчером при запуске маршрутизатора. Важно заметить, что диспетчер cfgmgr не отвечает за введение конфигурации в действие в качестве элемента рабочей конфигурации. **Cfgmgr –лишь посредник, который сообщает только о семантических ошибках, обнаруженных внутренними компонентами.** Анализ сбоя и диагностика его причины возлагаются на владельца внутреннего компонента. **Пользователи могут легко определить проверяющий компонент внутреннего компонента командой describe <команды CLI> в режиме конфигурации. Например, если в качестве ошибочной конфигурации показывается router bgp 217, то команда describe сообщит, что проверяющим компонентом является 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. Ошибки применения конфигурации. Конфигурация успешно проверена и принята в составе рабочей конфигурации, однако внутренний компонент по определенной причине не смог обновить свое рабочее состояние. Конфигурация показывается одновременно и как рабочая, поскольку она прошла проверку, и как ошибочная –из-за ошибки при работе внутреннего компонента. **При помощи команды describe в интерфейсе командной строки снова возможно установить владельца компонента, который не смог применить конфигурацию.**Для просмотра и повторного применения ошибочной конфигурации при запуске операторам следует выполнить следующие действия:Для повторного применения ошибочной конфигурации в выпуске R3.2 операторы могут придерживаться следующего порядка действий:**Операторы могут при помощи команды show configuration failed startup просмотреть ошибочную конфигурацию, сохраненную при запуске маршрутизатора.Операторы должны выполнить команду show configuration failed startup noerror | file myfailed.cfg для сохранения ошибочной загрузочной конфигурации в файле.Операторам следует войти в режим конфигурации (configuration) и при помощи команд load/commit повторно применить ошибочную конфигурацию:** `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` Для образов R3.3 операторы могут использовать следующую обновленную методику:**Операторы должны просмотреть и применить ошибочную конфигурацию (если таковая есть) при помощи команд show configuration failed startup и load configuration failed startup.**`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`

Генератор дампа ядра

По умолчанию система IOS XR записывает дамп ядра на жесткий диск при сбое процесса, но не при сбое внутри самого ядра. Следует учесть, что для системы с несколькими шасси эти функции в настоящее время поддерживаются только для шасси линейной платы 0. Поддержка других шасси будет введена в одном из последующих выпусков программного обеспечения.

Рекомендуется включить дампы ядра как для процессоров маршрутизации, так и для модульных обслуживающих плат, применяя эту конфигурацию как в стандартном режиме, так и в режиме администрирования:

```
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
```

Конфигурация дампа ядра

Сбой в ядре приводит к следующим событиям:

1. При сбое процессора маршрутизации дамп записывается в корневом каталоге жесткого диска соответствующего процессора маршрутизации.
2. При сбое модульной обслуживающей платы дамп записывается в каталоге RP0 жесткого диска.

Это никоим образом не влияет на время аварийного переключения процессора маршрутизации, поскольку для протоколов маршрутизации настроена безостановочная пересылка (NSF). После сбоя восстановление работы процессора маршрутизации или линейной платы может занять на несколько минут больше из-за времени, которое уйдет на запись дампа ядра.

Здесь показан пример добавления этой конфигурации к стандартной и к конфигурации режима администрирования. Следует учесть, что для конфигурации режима администрирования должны применяться распределенные процессоры маршрутизации.

В выводе команды показан пример конфигурации дампа ядра:

```
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)#
```

Безопасность

LPTS

Локальные пакетные транспортные службы (LPTS) обрабатывают пакеты для локальных получателей. Стек служб LTPS состоит из трех компонентов.

1. Главный компонент –арбитражный процесс портов. Он слушает запросы к сокету от процессов различных протоколов, например BGP и IS-IS, и учитывает все параметры привязок для этих процессов. Например, если процесс BGP прослушивает сокет с номером 179, то арбитражный процесс получает эти сведения от процессов BGP и затем назначает соответствующему процессу привязку в IFIB.
2. IFIB –компонент процесса LPTS. Помогающий вести каталог местонахождений процессов, прослушивающих определенную привязку к порту. IFIB создается арбитражным процессом портов и хранится неразрывно от него. Далее IFIB создает несколько подмножеств полученных сведений.Первое подмножество –срез IFIB. Срез может быть связан с протоколом IPv4 и т.п. Срезы направляются соответствующим диспетчерам потоков, которые используют срезы IFIB для передачи пакетов соответствующему процессу.Второе подмножество называется pre-IFIB. Оно позволяет линейной плате пересылать пакет надлежащему процессу, если существует только один процесс, или соответствующему диспетчеру потока.
3. Диспетчеры потоков помогают распределять пакеты далее, если их поиск нетривиален, например при наличии нескольких процессов для BGP. Каждый диспетчер потока имеет один или несколько срезов IFIB и надлежащим образом передает пакеты соответствующим процессам, связанным со срезом IFIB.

4. Если запись для целевого порта не определена, то пакет может быть либо отброшен, либо направлен диспетчеру потока. Если с портом связана политика, то пакет передается без связанного порта. Диспетчер потока затем помогает сформировать новую запись сеанса.

Как происходит передача внутренних пакетов?

Существует два типа потоков: потоки 2-го уровня (HDLC, PPP) и потоки ICMP/PING и маршрутизации 4-го уровня.

1. Потоки HDLC/PPP 2-го уровня. Эти пакеты обозначаются идентификаторами протокола и направляются непосредственно очередям центрального процессора в микросхеме Sprayer. Пакеты протокола 2-го уровня получают высокий приоритет, принимаются центральным процессором (через Squid) и обрабатываются. Вследствие этого ответ на сообщения поддержания активности (keepalive) для 2-го уровня непосредственно формируется через линейную плату и центральный процессор. При этом для получения ответа не требуется вызывать процессор маршрутизации. Кроме того, такая схема хорошо согласуется с распределенным управлением интерфейсами.
2. Пакеты ICMP (4-й уровень) принимаются линейной платой и после отыскания маршрута передаются через IFBI в очередь центрального процессора на микросхеме Sprayer. Эти пакеты затем посылаются центральному процессору (через Squid) и обрабатываются. После этого через выходные очереди микросхемы Sprayer передается ответ, который подлежит пересылке через коммутирующую матрицу. Подобная реализация учитывает и случаи, в которых информация также необходима другому приложению (путем копирования через коммутирующую матрицу). Пройдя через коммутирующую матрицу, пакет направляется соответствующей выходной линейной плате через соответствующую микросхему сборки и упорядочивания (Sponge) и очередь управления.
3. Потоки маршрутизации отыскиваются в IFIB и затем передаются очередям формирования выходного трафика (8000 очередей), одна из которых зарезервирована для управляющих пакетов. Для этой очереди механизм формирования не действует, она обслуживается каждый раз при заполнении. и имеет высокий приоритет. Пакет далее передается через коммутирующую матрицу очередей высокого приоритета в набор очередей центрального процессора на микросхеме Sponge (аналогичный очередям Squid на микросхеме Sprayer) и затем обрабатывается соответствующим процессом: диспетчером потока или непосредственным процессом. Ответ возвращается через выходную микросхему сборки и упорядочивания (Sponge) линейной платы и линейную плату. Выходная микросхема Sponge линейной платы имеет специальную очередь для обработки управляющих пакетов. Очереди в микросхеме Sponge на уровне отдельных выходных портов делятся на высокоприоритетные (управляющие) и низкоприоритетные (для пакетов).
4. В PSE имеется ряд контрольных условий, настраиваемых для ограничения пакетов 4-го уровня, 2-го уровня и пакетов маршрутизации. Они заданы изначально и допускают возможность последующего изменения пользователем.

Одна из наиболее распространенных проблем с LPTS –пропадание пакетов при попытке отправки эхозапроса маршрутизатору. Контрольные условия LPTS обычно ограничивают скорость трафика этих пакетов. Пример и подтверждение этой проблемы:

```
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

Протокол IP сам по себе не предусматривает защиты пакетов. Для защиты пакетов IP применяется метод IPsec. Протокол IPSec реализован в программном пути пересылки CRS-1, поэтому оконечная обработка сеанса IPsec осуществляется на процессоре маршрутизации или распределенном процессоре маршрутизации. Одна система CRS-1 поддерживает в общей сложности 500 сеансов IPsec. Конкретное число зависит от скорости центрального процессора и выделенных ресурсов. Программного ограничения в данном случае не существует –обработке механизмами IPSec подлежит только трафик от локальных источников и для локальных получателей в процессоре маршрутизации. Для этого вида трафика может применяться транспортный или туннельный режим IPsec. Первый тип предпочтителен ввиду меньших накладных расходов на обработку IPsec.

Выпуск R3.3.0 поддерживает шифрование протоколов BGP и OSPFv3 поверх IPsec.

Подробное описание реализации IPSec см. в Руководстве по настройке параметров безопасности в системе Cisco IOS XR.

Примечание: IPSec требует крипто-вставки, например, hfr-k9sec-p.pie-3.3.1.

Внеполосное управление

Доступ через консоль и порт AUX

Процессоры маршрутизации и обслуживающие платы снабжены для внеполосного управления консольным портом и портом AUX, а также портом управления Ethernet для внеполосного управления по протоколу IP.

Консольный порт и порт AUX каждого процессора маршрутизации или платы SCGE (по две единицы в каждом шасси) можно соединить с сервером консоли. Это означает, что для одного системного шасси требуются четыре консольных порта, а в системе с несколькими шасси требуется 12 портов плюс два дополнительных порта для диспетчерских ядер Catalyst 6504-E.

Подключение к порту AUX важно тем, что оно обеспечивает доступ к ядру IOS-XR и делает возможным восстановление системы в тех случаях, когда этого нельзя сделать через консольный порт. Доступ через порт AUX предоставляется только пользователям, заданным в системе локально, и только в том случае, когда пользователь имеет уровень доступа root-system или cisco-support. **Кроме того, для пользователя должен быть задан секретный пароль (secret).**

Доступ через виртуальный терминал

Для доступа к CRS-1 через порты vty можно использовать протоколы Telnet и SSH. По умолчанию оба протокола отключены, и пользователь должен разрешить их в явном виде.

Примечание: IPSec требует крипто-вставки, например, hfr-k9sec-p.pie-3.3.1.

Для включения поддержки SSH вначале следует сгенерировать ключи RSA и DSA, как показано в следующем примере:

```
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
```

Дополнительные сведения

- Поддержка маршрутизаторов
- Cisco Systems – техническая поддержка и документация