Рекомендация по настройке параметров оптимизации производительности высокопроизводительных платформ

Введение

Данная инструкция актуальна для двухпроцессорных аппаратных платформ (АП) «С-Терра Шлюз» (далее по тексту Продукт или криптошлюз). Документ содержит настройки параметров оптимизации производительности для следующих устройств: «С-Терра Шлюз» 8000HE (на базе платформы Aquarius T50 с двумя процессорами «Intel Xeon Gold 6226R 2.90GHz») и «С-Терра Шлюз» 7000HE (на базе платформы Lanner FW-8894B с двумя процессорами «Intel Xeon 2643v4 3.40GHz»).

Настройка параметров оптимизации производительности

Настройки параметров оптимизации производительности применимы для «С-Терра Шлюз» моделей 7000HE и 8000HE, которые, как правило, размещаются в центральных офисах и предоставляют защищенный доступ большому количеству филиальных шлюзов (или VPN клиентов) к ресурсам центрального офиса. На производительность могут влиять как общие параметры (применимые ко всем платформам/процессорам), так и частные, которые необходимо настраивать в зависимости от модели АП/процессора и типа обрабатываемого трафика.

Общая упрощенная логика обработки трафика следующая:

1. Пакеты, попавшие на интерфейс сетевой карты криптошлюза, раскладываются по очередям сетевой карты в режиме per-flow (пакеты, которые находятся в рамках одной сетевой сессии, отправляются в определённую очередь и в разных очередях находиться не могут).
2. Irq-ядра Продукта, отвечающие за первичную обработку пакетов (далее – «обработка прерываний» или «предварительная обработка»), ставят эти пакеты во внутренние очереди (sendq и workq).
3. Пакеты из данных очередей берутся на обработку work-ядрами, на которых могут производиться следующие процедуры: фильтрация, шифрование, маркирование и др. (далее – «основная обработка»).
4. Пакеты отправляются на маршрутизацию.

Параметры оптимизации производительности позволяют указать необходимое количество очередей на сетевой карте, настроить нужное количество ядер обработки прерываний и ядер основной обработки и т.д.

Проверка настроек аппаратных средств ПАК

Необходимо убедиться в том, что Hyper-Threading и NUMA включены (на всех ПАК, поддерживающих данные технологии, они должны быть включены по умолчанию). Подробное описание проверки настроек аппаратных средств ПАК можно найти в главе «Проверка настроек аппаратных средств ПАК».

Настройка параметра распределения ядер драйвера Продукта

Параметр `cpu_distribution` в файле `/etc/modprobe.d/vpndrvr.conf` отвечает за распределение ядер драйвера (`vpndrv`) Продукта, то есть за то, какое количество ядер драйвера будет отведено на обработку прерываний от пакетов, а какое – на основную обработку пакетов (межсетевое экранирование, шифрование, маркирование и т.д.). Для того, чтобы посмотреть содержимое файла `/etc/modprobe.d/vpndrvr.conf`, необходимо воспользоваться командой:

```
root@Gate7000HE:~# cat /etc/modprobe.d/vpndrvr.conf | grep cpu
options vpndrvr cpu_distribution=*:2/*
```

Параметр `cpu_distribution` драйвера `vpndrv` (является модулем ядра Linux) имеет следующий формат:

***:<irq cores>/<working cores>**

- **<irq cores>** позволяет указать количество (или же конкретные номера) процессорных ядер, выделенных **на обработку прерываний** от пакетов с сетевых интерфейсов;

В качестве ядер, отвечающих за прерывания, должны быть выбраны только **физические ядра** (использовать hyper-threading ядра с целью обработки прерываний крайне не рекомендуется, так как из-за этого может произойти существенная просадка производительности). Посмотреть тип ядра можно в выводе утилиты `lscpu`.

- **<working cores>** позволяет указать количество (или же конкретные номера) процессорных ядер, выделенных **на основную обработку пакетов (фильтрация, шифрование, маркирование и др.)**.

Вывод утилиты выше свидетельствует о том, что на обработку прерываний выделено два ядра, а на обработку пакетов драйвером Продукта выделены все остальные ядра. Параметр `cpu_distribution` всегда выставляется при прохождении инициализации Продукта, но может быть изменен вручную в конфигурационном файле `/etc/modprobe.d/vpndrvr.conf`.

Изменять параметр `cpu_distribution` без согласования с вендором – запрещено! Если параметр `cpu_distribution` в файле `/etc/modprobe.d/vpndrvr.conf` был изменён, то для применения настроек необходимо перезагрузить АП.

Также параметр `cpu_distribution` имеет возможность гибкой настройки, при которой вручную указываются номера конкретных ядер, выделенных на обработку прерываний и на основную обработку пакетов, а также, взаимосвязь этих ядер друг с другом. Предположим, на ПАК «С-Терра Шлюз» 7000HE с двумя процессорами «Intel Xeon 2643v4 3.40GHz» выставлено следующее распределение:

```
root@Gate7000HE:~# vim.tiny /etc/modprobe.d/vpndrvr.conf
```

```
options vpndrvr cpu_distribution=*:#0,1,2,3/#8~0,9~1,10~2,11~3,12~0,13~1,14~2,15~3
```

Данное распределение означает, что на обработку прерываний будет выделено 4 ядра (0, 1, 2, 3), а на основную обработку пакетов будет выделено 8 ядер (8, 9, 10, 11, 12, 13, 14, 15). Причём, пакеты, прерывания от которых, произошли на ядре под номером 0 будут отправляться на обработку на «рабочие» ядра 8 и 12, с 1 на 9 и 13, со 2 на 10 и 14, с 3 на 11 и 15.

Необходимо следить за тем, чтобы ядра, на которых обрабатываются прерывания, и привязанные к ним ядра, на которых производится основная обработка, находились на одной NUMA-ноде (определить, на какой NUMA-ноде находятся ядра, можно при помощи команды `lscpu | grep NUMA`).

Для просмотра текущего распределения ядер можно воспользоваться следующими командами:

- для просмотра ядер, которые не участвуют в обработке пакетов драйвером Продукта (далее – «свободные ядра»):

```
root@Gate7000HE:~# cat /proc/sys/agent/cpu/free
```

```
4 5 6 7 16 17 18 19 20 21 22 23
```

- для просмотра ядер, отвечающих за обработку прерываний от пакетов на сетевых интерфейсах:

```
root@Gate7000HE:~# cat /proc/sys/agent/cpu/irq
```

```
0 1 2 3
```

- для просмотра ядер, отвечающих за обработку пакетов драйвером `vpndrv`:

```
root@Gate7000HE:~# cat /proc/sys/agent/cpu/work
```

```
8 12|9 13|10 14|11 15
```

Оптимальное распределение ядер для режимов работы, в которых используется шифрование, – 1 к 3. То есть, на 1 ядро, отвечающее за предварительную обработку пакетов, должно приходиться 3 ядра, отвечающих за основную обработку пакетов. Также стоит учитывать, что при таком распределении производительность обработки одной сетевой сессии будет ограничена 3 ядрами для основной обработки пакетов.
Рекомендуемые настройки параметра `cpu_distribution` для каждой из АП (в зависимости от режима работы) можно найти в главе «Приложение».

Конфигурирование количества очередей на сетевых картах

Конфигурирование количества очередей на сетевых картах должно выполняться независимо от модели АП и типа передаваемого трафика на **всех физических** сетевых интерфейсах ПАК, которые поддерживают возможность выставления очередей (узнать, поддерживает ли сетевая карта выставление очередей, можно в инструкции к сетевой карте, либо же фактически попытаться выставить произвольное количество очередей. Если заданное значение применится, значит сетевая карта поддерживает возможность выставления очередей).

Данная настройка является необходимым условием для правильной работы механизма распределения прерываний (вызванных пакетами) от очередей сетевой карты по ядрам процессора(ов) и оптимальной производительности аппаратных платформ.

Количество очередей, на которые сетевая карта будет раскладывать пакеты, должно быть равно количеству ядер, отведённых на обработку прерываний (irq) драйвером Продукта.

Для того, чтобы определить, какое количество очередей установлено на сетевом интерфейсе, необходимо воспользоваться командой (вывод команды приведён на АП «С-Терра Шлюз» 8000HE с двумя процессорами «Intel Xeon Gold 6226R 2.90GHz»):

```
root@Gate8000HE:~# ethtool -l eth2
```

```
Channel parameters for eth2:
Pre-set maximums:
RX:             n/a
TX:             n/a
Other:          1
Combined:       64
Current hardware settings:
RX:             n/a
TX:             n/a
Other:          1
Combined:       64
```

Параметр «Combined» в поле «Current hardware settings» свидетельствует о том, что на сетевом интерфейсе `eth2` выставлено 64 очереди (количество очередей по умолчанию зависит от модели сетевой карты и количества ядер процессоров).

Конфигурирование количества очередей на сетевых картах может осуществляться двумя способами: при помощи опции «combined_queue» (данная опция доступна не на всех версиях ПО) или же при помощи конфигурационного файла `/etc/rc.local.inc` (универсальный скрипт, который можно настроить вручную на всех версиях ПО).

Для того, чтобы определить, имеется ли на криптошлюзе возможность конфигурирования количества очередей на сетевых интерфейсах при помощи опции «combined_queue», необходимо выполнить команду:

```
root@Gate8000HE:~# cat /etc/image_version | grep FEATURE
```

```
FEATURE="enable_mgmt combined_queue"
```

Наличие записи «combined_queue» свидетельствует о том, что криптошлюз поддерживает данную опцию, значит конфигурирование количества очередей на сетевых интерфейсах нужно производить при помощи конфигурационного файла `/etc/combined_queue.conf`:

```
root@Gate8000HE:~# vim.tiny /etc/combined_queue.conf
```

```
# Number of queues to be set up
QUEUES="2"
# Space-separated blacklist of network interface names.
# IFNAMES_BLACKLIST="eth1"
# Space-separated blacklist of network driver names.
# DRIVERS_BLACKLIST="vmxnet3"
```

- QUEUES – количество очередей, которое необходимо выставить;
- IFNAMES_BLACKLIST – имена сетевых интерфейсов, на которых заданное количество очередей **не будет выставлено**;
- DRIVERS_BLACKLIST – имена драйверов сетевых карт, для интерфейсов которых заданное количество очередей **не будет выставлено**.

Если на криптошлюзе не поддерживает функционал «combined_queue», то конфигурирование очередей на сетевых интерфейсах необходимо производить при помощи конфигурационного файла `/etc/rc.local.inc`.

Для того, чтобы количество очередей выставлялось автоматически при помощи конфигурационного файла `/etc/rc.local.inc`, необходимо:

1. Создать файл (если отсутствует) `/etc/rc.local.inc`:

   ```
   root@Gate8000HE:~# vim.tiny /etc/rc.local.inc
   ```

2. Поместить в файл `/etc/rc.local.inc` скрипт, указанный ниже, записав в переменную **QNUM** количество **<irq cores>**, которое указано в параметре `cpu_distribution` (в примере ниже оно равно 16):

   ```
   QNUM=16
   for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
   do
       echo Setting combined queue to $QNUM on $phys_iface ...
       ethtool -L $phys_iface combined $QNUM
   done
   ```

3. Перезапустить сервис `rc-local.service`:

   ```
   root@Gate8000HE:~# systemctl restart rc-local.service
   ```

   Скрипт, выставляющий количество очередей, будет выполняться автоматически после загрузки ОС или же после перезагрузки сервиса `rc-local.service`.

4. Проверить текущее количество очередей на всех физических интерфейсах:

   ```
   root@Gate8000HE:~# ethtool -l eth2
   ```

   ```
   Channel parameters for eth2:
   Pre-set maximums:
   RX:             n/a
   TX:             n/a
   Other:          1
   Combined:       64
   Current hardware settings:
   RX:             n/a
   TX:             n/a
   Other:          1
   Combined:       16
   ```

Вывод утилиты свидетельствует о том, что теперь количество очередей на сетевом интерфейсе равно 16.

Будьте внимательны! Если криптошлюз поддерживает возможность конфигурирования количества очередей на сетевых интерфейсах при помощи опции «combined_queue», то пользоваться выставлением очередей при помощи конфигурационного файла `/etc/rc.local.inc` строго запрещено!

Проверка результатов оптимизации производительности АП

Помимо выставления количества очередей на сетевой карте, пропускная способность также зависит от того, каким образом сетевая карта разбалансировала пакеты по очередям.

Сетевые карты Intel не могут корректно балансировать фрагментированный трафик по очередям, что приводит к снижению производительности, поэтому старайтесь избегать фрагментированного трафика. В версии 5.0 «С-Терра Шлюз» поддерживает префрагментацию (фрагментирование пакетов до их зашифрования).

При большом количестве IPsec туннелей пакеты должны распределиться по очередям и, соответственно, по ядрам-обработчикам равномерно. Для определения загруженности ядер процессора необходимо воспользоваться командой `cpupower monitor`. В выводе данной команды раздел «Mperf» содержит столбец «C0», который показывает степень загруженности ядра/потока в данный момент времени. При равномерной балансировке трафика вывод команды `watch cpupower monitor` должен выглядеть подобным образом (вывод команды приведён на АП «С-Терра Шлюз» 8000HE с двумя процессорами «Intel Xeon Gold 6226R 2.90GHz»):

```
root@Gate8000HE:~# watch cpupower monitor
```

			Nehalem				Mperf			Idle_Stats			
PKG	CORE	CPU	C3	C6	PC3	PC6	C0	Cx	Freq	POLL	C1	C1E	C6
0	0	0	0.00	0.00	0.00	0.00	42.82	57.18	3081	0.06	59.79	20.23	3.13
0	0	32	0.00	0.00	0.00	0.00	96.77	3.23	3082	0.04	3.17	0.01	0.00
0	1	1	0.00	0.00	0.00	0.00	52.65	47.35	3081	0.03	29.59	12.94	21.76
0	1	33	0.00	0.00	0.00	0.00	96.54	3.46	3082	0.01	3.31	0.05	0.00
0	2	2	0.00	0.00	0.00	0.00	52.91	47.09	3081	0.03	23.69	8.85	28.73
0	2	34	0.00	0.00	0.00	0.00	95.70	4.30	3082	0.01	4.22	0.05	0.00
0	3	3	0.00	0.00	0.00	0.00	48.58	51.42	3081	0.02	31.47	10.35	27.07
0	3	35	0.00	0.00	0.00	0.00	96.50	3.50	3082	0.06	3.51	0.00	0.00
0	4	4	0.00	0.00	0.00	0.00	51.46	48.54	3081	0.01	21.98	10.64	28.90
0	4	36	0.00	0.00	0.00	0.00	95.34	4.66	3082	0.02	4.48	0.16	0.00
0	5	5	0.00	0.00	0.00	0.00	41.51	58.49	3081	0.02	42.99	15.11	19.87
0	5	37	0.00	0.00	0.00	0.00	95.79	4.21	3082	0.03	4.22	0.02	0.00
0	6	6	0.00	0.00	0.00	0.00	49.20	50.80	3081	0.03	42.25	13.74	17.07
0	6	38	0.00	0.00	0.00	0.00	96.75	3.25	3082	0.04	3.22	0.01	0.00
0	7	7	0.00	0.00	0.00	0.00	50.02	49.98	3081	0.02	38.20	12.17	20.46
0	7	39	0.00	0.00	0.00	0.00	96.97	3.03	3082	0.03	2.98	0.00	0.00
0	8	8	0.00	0.00	0.00	0.00	97.15	2.85	3081	0.03	2.73	0.02	0.00
0	8	40	0.00	0.00	0.00	0.00	97.25	2.75	3082	0.01	2.61	0.01	0.00
0	9	9	0.00	0.00	0.00	0.00	96.62	3.38	3081	0.00	3.13	0.13	0.00
0	9	41	0.00	0.00	0.00	0.00	96.78	3.22	3082	0.01	2.96	0.13	0.00
0	10	10	0.00	0.00	0.00	0.00	95.84	4.16	3081	0.00	3.95	0.16	0.00
0	10	42	0.00	0.00	0.00	0.00	96.06	3.94	3082	0.01	3.73	0.13	0.00
0	11	11	0.00	0.00	0.00	0.00	96.83	3.17	3081	0.01	3.11	0.01	0.00
0	11	43	0.00	0.00	0.00	0.00	96.95	3.05	3082	0.02	2.94	0.04	0.00
0	12	12	0.00	0.00	0.00	0.00	95.60	4.40	3081	0.00	4.12	0.24	0.00
0	12	44	0.00	0.00	0.00	0.00	95.77	4.23	3082	0.00	3.91	0.27	0.00
0	13	13	0.00	0.00	0.00	0.00	96.30	3.70	3081	0.01	3.64	0.05	0.00
0	13	45	0.00	0.00	0.00	0.00	96.42	3.58	3082	0.03	3.47	0.05	0.00
0	14	14	0.00	0.00	0.00	0.00	97.14	2.86	3081	0.03	2.79	0.00	0.00
0	14	46	0.00	0.00	0.00	0.00	97.26	2.74	3082	0.02	2.62	0.03	0.00
0	15	15	0.00	0.00	0.00	0.00	97.23	2.77	3081	0.01	2.68	0.00	0.00
0	15	47	0.00	0.00	0.00	0.00	97.38	2.62	3082	0.02	2.49	0.00	0.00
1	0	16	0.00	0.00	0.00	0.00	42.15	57.85	3082	0.03	44.42	14.91	18.83
1	0	48	0.00	0.00	0.00	0.00	97.94	2.06	3082	0.02	1.83	0.00	0.00
1	1	17	0.00	0.00	0.00	0.00	40.91	59.09	3081	0.04	35.66	14.81	25.28
1	1	49	0.00	0.00	0.00	0.00	95.40	4.60	3082	0.03	4.59	0.03	0.00
1	2	18	0.00	0.00	0.00	0.00	47.59	52.41	3081	0.07	49.47	16.39	10.88
1	2	50	0.00	0.00	0.00	0.00	95.65	4.35	3082	0.03	4.32	0.01	0.00
1	3	19	0.00	0.00	0.00	0.00	46.23	53.77	3081	0.02	30.19	10.09	29.76
1	3	51	0.00	0.00	0.00	0.00	95.38	4.62	3082	0.05	4.60	0.05	0.00
1	4	20	0.00	0.00	0.00	0.00	46.10	53.90	3081	0.03	35.05	11.41	25.70
1	4	52	0.00	0.00	0.00	0.00	95.61	4.39	3082	0.05	4.32	0.06	0.01
1	5	21	0.00	0.00	0.00	0.00	47.56	52.44	3081	0.05	48.77	15.89	12.14
1	5	53	0.00	0.00	0.00	0.00	96.34	3.66	3082	0.04	3.60	0.01	0.00
1	6	22	0.00	0.00	0.00	0.00	49.93	50.07	3081	0.01	18.91	7.95	34.79
1	6	54	0.00	0.00	0.00	0.00	95.10	4.90	3082	0.01	4.73	0.11	0.00
1	7	23	0.00	0.00	0.00	0.00	47.14	52.86	3081	0.02	39.41	13.10	20.83
1	7	55	0.00	0.00	0.00	0.00	96.03	3.97	3082	0.03	3.89	0.02	0.00
1	8	24	0.00	0.00	0.00	0.00	98.13	1.87	3081	0.01	1.61	0.00	0.00
1	8	56	0.00	0.00	0.00	0.00	98.13	1.87	3082	0.01	1.62	0.00	0.00
1	9	25	0.00	0.00	0.00	0.00	96.01	3.99	3081	0.01	3.89	0.05	0.00
1	9	57	0.00	0.00	0.00	0.00	96.01	3.99	3082	0.00	3.88	0.05	0.00
1	10	26	0.00	0.00	0.00	0.00	96.17	3.83	3081	0.02	3.72	0.03	0.00
1	10	58	0.00	0.00	0.00	0.00	96.17	3.83	3082	0.01	3.74	0.02	0.00
1	11	27	0.00	0.00	0.00	0.00	96.07	3.93	3081	0.01	3.81	0.05	0.00
1	11	59	0.00	0.00	0.00	0.00	96.06	3.94	3082	0.02	3.81	0.07	0.00
1	12	28	0.00	0.00	0.00	0.00	96.21	3.79	3081	0.02	3.56	0.13	0.00
1	12	60	0.00	0.00	0.00	0.00	96.10	3.90	3082	0.01	3.72	0.10	0.00
1	13	29	0.00	0.00	0.00	0.00	96.76	3.24	3081	0.02	3.07	0.05	0.00
1	13	61	0.00	0.00	0.00	0.00	96.78	3.22	3082	0.01	3.07	0.02	0.00
1	14	30	0.00	0.00	0.00	0.00	95.54	4.46	3081	0.00	4.12	0.24	0.00
1	14	62	0.00	0.00	0.00	0.00	95.48	4.52	3082	0.00	4.14	0.27	0.00
1	15	31	0.00	0.00	0.00	0.00	96.46	3.54	3081	0.01	3.39	0.03	0.00
1	15	63	0.00	0.00	0.00	0.00	96.45	3.55	3082	0.03	3.38	0.03	0.00

Рисунок 1. Загрузка ядер процессоров при равномерной балансировке трафика на АП «С-Терра Шлюз» 8000НЕ (2 x Intel Xeon Gold 6226R 2.90GHz)

Столбец «CPU» содержит порядковые номера ядер в контексте операционной системы. Два логических ядра, входящих в одно физическое ядро, располагаются друг под другом (нулевому ядру соответствуют логические ядра 0 и 32, первому ядру соответствуют логические ядра 1 и 33 и так далее). Столбец «C0» свидетельствует о том, что нагрузка по ядрам распределена достаточно равномерно.

Распределение по очередям входящих пакетов можно посмотреть при помощи команды (регулярное выражение после утилиты `grep` может зависеть от версии драйвера сетевой карты и количества выставленных очередей):

```
root@Gate8000HE:~# ethtool -S eth2 | grep -E rx.*[0-9]+.*packets: | head -n 16
     rx-0.packets: 7920480
     rx-1.packets: 7941043
     rx-2.packets: 7922586
     rx-3.packets: 7921941
     rx-4.packets: 7918976
     rx-5.packets: 7920041
     rx-6.packets: 7921949
     rx-7.packets: 7923393
     rx-8.packets: 7921948
     rx-9.packets: 7923390
     rx-10.packets: 7918978
     rx-11.packets: 7920041
     rx-12.packets: 7922584
     rx-13.packets: 7921938
     rx-14.packets: 7920478
     rx-15.packets: 7919044
```

В данном случае используется команда для просмотра распределения пакетов по 16 очередям (с 0 до 15). Пакеты распределены равномерно, но так бывает не всегда и, как следствие, один из процессоров перегружается (пропускная способность из-за этого падает).

Чем больше сетевых сессий, тем лучше распределяются пакеты по очередям.

Помимо распределения пакетов по очередям, необходимо проверить, на каком ядре процессора обрабатываются пакеты из очереди. Например, на интерфейсе `eth2` распределение будет следующим:

```
root@Gate8000HE:~# cat /proc/interrupts | grep eth2
```

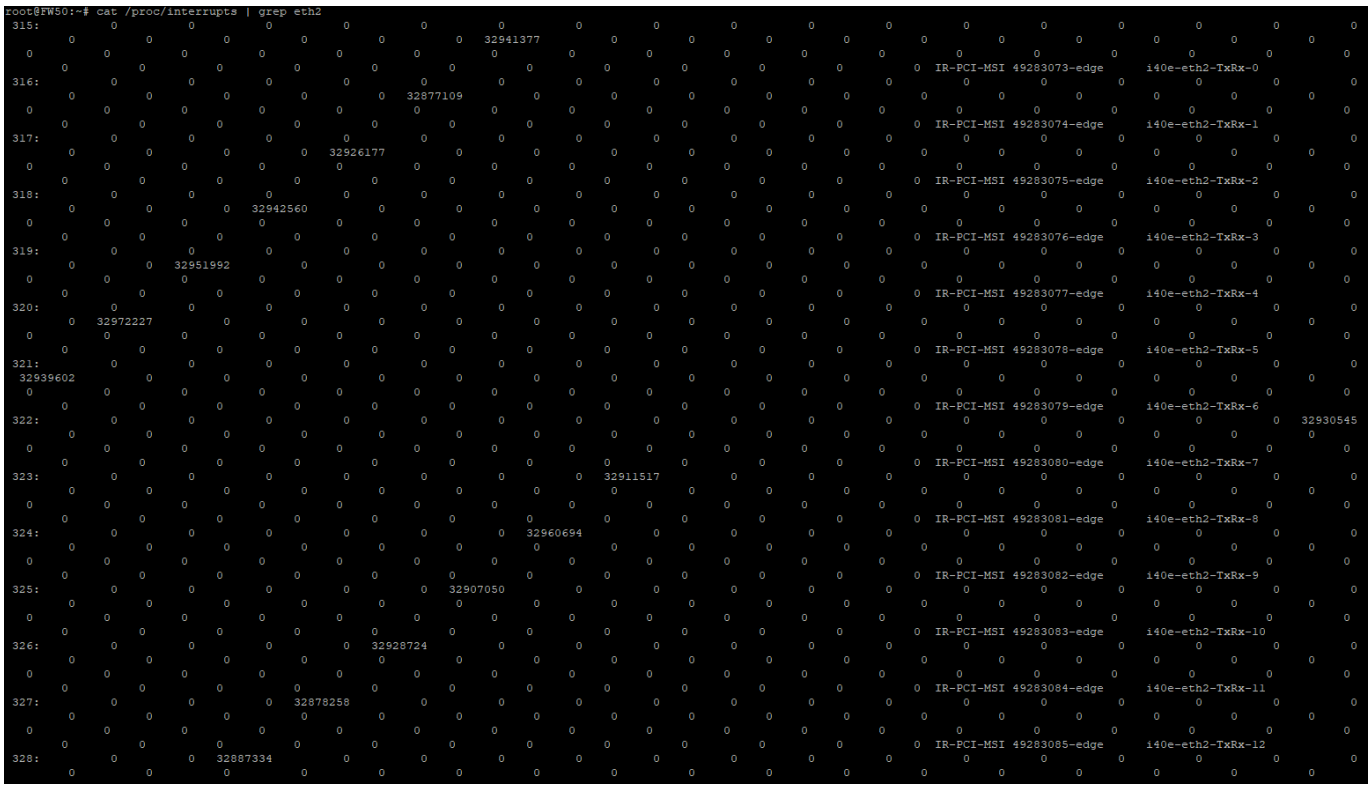


Рисунок 2. Вывод информации о прерываниях сетевого интерфейса eth2 после выставления двух очередей на АП «С-Терра Шлюз» 8000HE (2 x Intel Xeon Gold 6226R 2.90GHz)

Количество чисел в первом столбце (без учёта последней строки) равно количеству очередей, в которые сетевая карта будет раскладывать пакеты, попавшие на интерфейс eth2. В первом столбце располагаются номера прерываний (315, 316, 317, …). В строке, расположенной справа от номера прерывания, находятся числа, обозначающие количество пакетов, обработанных на конкретном ядре (нумерация ядер начинается с нуля). Рассмотрим прерывание 315. На 23 ядре обработано 32941377 прерываний. Это значит, что прерывание 315 привязано к 23 ядру и обрабатываться будет на нём. Аналогично и для остальных ядер.

Если у пользователя небольшое количество сессий и IPsec туннелей, то сетевая карта не сможет автоматически разбалансировать трафик по очередям. Одним из решений является ручная балансировка трафика. С правилами настройки ручной балансировки можно ознакомиться в главе «Ручная балансировка трафика по нескольким IPsec туннелям**Ошибка! Источник ссылки не найден.**».

Также необходимо следить за количеством ошибок и потерь средствами утилиты `kstat_show` (ключ `-e` позволяет просмотреть расширенную статистику, которая включена по умолчанию):

```
root@GW1-8000HE:~# kstat_show  -e
ipsec in pkt:           0
ipsec in oct:           0
ipsec in drop:          0
ipsec no_sa:            0
ipsec replay drops:     0
ipsec auth fails:       0
ipsec decrypt fails:    0
ipsec out pkt:          0
ipsec out oct:          0
ipsec out drop:         0
frag ok:                0
frag fail:              0
link send:              0
link send err:          0
link recv:              0
prefrag ok:             0
```

```
prefrag fail:           0
fw drop:                0
fw pass:                0
fw assemble:            0
fw not found:           0
fw error:               0

rx errors:              0
tx errors:              0
route errors:           0
skb allocation errors:  0
high priority packets dropped:                  0
low priority packets dropped:                   0
high priority packets passed while overloaded:  0

                                        Low priority      High priority             Total
incoming packets came to hook:                 0                 0                 0
outgoing packets came to hook:                 0                 0                 0
incoming packets taken by hook:                0                 0                 0
outgoing packets taken by hook:                0                 0                 0
incoming send queue length:                    0                 0                 0
outgoing send queue length:                    0                 0                 0
incoming send queue overflows:                 0                 0                 0
outgoing send queue overflows:                 0                 0                 0

packets came to hook:                          0                 0                 0
packets taken by hook:                         0                 0                 0
send queue length:                             0                 0                 0
work queue length:                             0                 0                 0
send queue overflows:                          0                 0                 0
work queue overflows:                          0                 0                 0
work queue wait events:                        0                 0                 0
packets allocated:                             0                 0                 0
packets freed:                                 0                 0                 0
packets not yet freed:                         0                 0                 0
```

В Продукте «С-Терра Шлюз» версии 5.0 появилась расширенная статистика, которая позволяет увидеть ошибки и потери при обработке трафика, особое внимание нужно обращать на счетчики `high priority packets dropped`, `low priority packets dropped`, `send queue overflows`, `work queue overflows`, если они растут, то Шлюза перегружен.

Для того, чтобы сбросить накопившуюся статистику в `kstat_show`, необходимо воспользоваться следующей командой:

```
root@Gate8000HE:~# kstat_show -c
```

Если после выполнения всех рекомендаций выше производительность не выросла, то необходимо искать причину, по которой это происходит. Для этого можно обратиться к главам, представленным ниже.

Настройка программных средств ПАК

Режим «шифрование на уровне L3»

Для повышения производительности необходимо, чтобы **количество ядер, отвечающих за прерывания (irq), было равно количеству очередей (QNUM), выставленных на сетевой карте**. При этом, нужно следить за тем, чтобы пакеты ровно раскладывались по очередям. Равномерное распределение пакетов по очередям, как правило, достигается за счёт большого количества сессий. Так как высокопроизводительные платформы практически всегда используются в качестве центральных шлюзов, то под их обработку, как правило, попадает большое количество сессий. Если пакеты по очередям не будут распределены равномерно, то это приведёт к существенному снижению производительности.

Настройка параметра драйвера DefaultCryptoContextsPerIPSecSA

При использовании АП в режиме «шифрование» необходимо проверить параметр `DefaultCryptoContextsPerIPSecSA` в файле `/opt/VPNagent/etc/agent.ini`. Данный параметр определяет количество криптографических контекстов, создаваемых для одного IPsec SA. Наличие нескольких контекстов позволяет параллельно обрабатывать пакеты с использованием одного IPsec SA. Однако создание дополнительных криптографических контекстов требует ресурсов и может привести к превышению ограничений в криптографических библиотеках.

Значение `DefaultCryptoContextsPerIPSecSA` должно превышать общее количество процессорных ядер на 1. Например, для АП «С-Терра Шлюз» 7000HE с количеством ядер (поле «CPU(s)» в выводе команды lscpu) равным 24, вывод команды будет следующим:

```
root@Gate7000HE:~# cat /opt/VPNagent/etc/agent.ini | grep DefaultCryptoContextsPerIPSecSA
```

```
! of the DefaultCryptoContextsPerIPSecSA field. Value must be from 1
DefaultCryptoContextsPerIPSecSA = 25
```

Если в файл `/opt/VPNagent/etc/agent.ini` записано другое значение, необходимо изменить его вручную.

Для того, чтобы выставить корректное значение `DefaultCryptoContextsPerIPSecSA`, необходимо:

1. С помощью команды `lscpu | grep CPU` (поле «CPU(s)») узнать, сколько ядер доступно в операционной системе.
2. В файл `/opt/VPNagent/etc/agent.ini` записать в переменную `DefaultCryptoContextsPerIPSecSA` полученное количество ядер **+ 1** (в данном случае 25):

   ```
   root@Gate7000HE:~# vim.tiny /opt/VPNagent/etc/agent.ini
   ```

   ```
   [Local Policy]
   ! Local Security Policy IPsecRule structure has optional field named
   ! CryptoContextsPerIPSecSA defining number of cryptographic contexts
   ! opened for each IPSec SA built with the IPsecRule. If the field is
   ! not defined there, number of cryptographic contexts equals the value
   ! of the DefaultCryptoContextsPerIPSecSA field. Value must be from 1
   ! to 128.
   DefaultCryptoContextsPerIPSecSA = 25
   ```

3. Пересчитать контрольную сумму файла /opt/VPNagent/etc/agent.ini с помощью утилиты `integr_mgr calc`:

   ```
   root@Gate7000HE:~# integr_mgr calc -f /opt/VPNagent/etc/agent.ini
   ```

   ```
   SUCCESS:  Operation was successful.
   ```

4. Перезагрузить АП с помощью команды `reboot`.

Пример настройки распределения ядер

Предположим, ПАК «С-Терра Шлюз» 8000HE работает в режиме «шифрование на уровне L3». Наиболее высокопроизводительным вариантом будет следующее распределение ядер: 16 ядер драйвера Продукта на обработку прерываний (irq) и 48 ядер на основную обработку пакетов (work), в данном случае – шифрование:

```
root@Gate8000HE:~# cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options vpndrvr
cpu_distribution=*:#0,1,2,3,4,5,6,7,16,17,18,19,20,21,22,23/#8~0,9~1,10~2,11~3,12~4,13~5,14~6,15~7,32~0,33~1,34~2,35~3,36~4,37~5,38~6,39~7,40~0,41~1,42~2,43~3,44~4,45~5,46~6,47~7,24~16,25~17,26~18,27~19,28~20,29~21,30~22,31~23,48~16,49~17,50~18,51~19,52~20,53~21,54~22,55~23,56~16,57~17,58~18,59~19,60~20,61~21,62~22,63~23
```

Наглядная схема распределения ядер выглядит следующим образом (слева указаны номера ядер, на которых происходит обработка прерываний, а справа – ядра, на которых происходит основная обработка;

пометка «phys» означает, что данное ядро является физическим, а пометка «thread» означает, что данное ядро является hyper-threading):

```
NUMA0:
      |---------8 (work)-phys
0(irq)|---------32(work)-thread 0
 phys |---------40(work)-thread 8

      |---------9 (work)-phys
1(irq)|---------33(work)-thread 1
 phys |---------41(work)-thread 9

      |---------10(work)-phys
2(irq)|---------34(work)-thread 2
 phys |---------42(work)-thread 10

      |---------11(work)-phys
3(irq)|---------35(work)-thread 3
 phys |---------43(work)-thread 11

      |---------12(work)-phys
4(irq)|---------36(work)-thread 4
 phys |---------44(work)-thread 12

      |---------13(work)-phys
5(irq)|---------37(work)-thread 5
 phys |---------45(work)-thread 13

      |---------14(work)-phys
6(irq)|---------38(work)-thread 6
 phys |---------46(work)-thread 14

      |---------15(work)-phys
7(irq)|---------39(work)-thread 7
 phys |---------47(work)-thread 15
```

```
NUMA1:
       |---------24(work)-phys
16(irq)|---------48(work)-thread 16
 phys  |---------56(work)-thread 24

       |---------25(work)-phys
17(irq)|---------49(work)-thread 17
 phys  |---------57(work)-thread 25

       |---------26(work)-phys
18(irq)|---------50(work)-thread 18
 phys  |---------58(work)-thread 26

       |---------27(work)-phys
19(irq)|---------51(work)-thread 19
 phys  |---------59(work)-thread 27

       |---------28(work)-phys
20(irq)|---------52(work)-thread 20
 phys  |---------60(work)-thread 28

       |---------29(work)-phys
21(irq)|---------53(work)-thread 21
 phys  |---------61(work)-thread 29

       |---------30(work)-phys
22(irq)|---------54(work)-thread 22
 phys  |---------62(work)-thread 30

       |---------31(work)-phys
23(irq)|---------55(work)-thread 23
 phys  |---------63(work)-thread 31
```

Таким образом, можно сделать вывод о том, что в режиме «шифрование на уровне L3» задействуются две NUMA-ноды. На каждой из них на 1 ядро, обрабатывающее прерывания, приходится 3 ядра, задействованных в основной обработке пакетов.

Соответствующие настройки для ПАК «С-Терра Шлюз» 7000HE и «С-Терра Шлюз» 8000HE можно найти в главе «Приложение».

Режим «межсетевое экранирование»

При использовании «С-Терра Шлюз» 8000HE в режиме «межсетевой экран» необходимо задействовать только один процессор, так как при использовании двух процессоров производительность снизится в силу ограничений межпроцессорного взаимодействия. Для «С-Терра Шлюз» 7000HE такой особенности нет.

В режиме «межсетевое экранирование» также необходимо, чтобы **количество ядер, отвечающих за прерывания (irq), было равно количеству очередей (QNUM), выставленных на сетевой карте**.

Настройка параметров драйвера для stateless и stateful фильтраций

Необходимо выставить максимальные значения в параметрах для stateless-фильтрации при помощи файла `vpndrvr.conf`:

```
root@Gate8000HE:~# vim.tiny /etc/modprobe.d/vpndrvr.conf
```

```
options vpndrvr fw_filter_cache_bucket_size=32
options vpndrvr fw_filter_cache_hash_bits=17
options vpndrvr fw_filter_cache_ttl=3600
```

- «fw_filter_cache_bucket_size» (число, 8-128, по умолчанию 8) – среднее количество одинаковых значений хэша в кэше правил.
- «fw_filter_cache_hash_bits» (число, 10-17, по умолчанию 10) – количество бит хэш-таблицы кэша правил.
- «fw_filter_cache_ttl» (секунды, 1-3600, по умолчанию 5 секунд) – время жизни одной записи в кэше правил.

Параметр «fw_filter_cache_bucket_size» используется для вычисления максимального количества записей в кэше: `2 ^ fw_filter_cache_hash_bits * fw_filter_bucket_size` (далее по тексту «размер кэша»). Для увеличения производительности, при изменении размера кэша, рекомендуется сначала увеличивать параметр «fw_filter_cache_hash_bits» и только потом – «fw_filter_cache_bucket_size». Если размер кэша таков, что он вмещает все реально возможные комбинации сигнатур пакетов, рекомендуется выставить параметр «fw_filter_cache_ttl» в значение, которое будет больше максимального времени между прохождениями пакетов с одинаковыми сигнатурами.

Если в политике безопасности есть цепочки с очень большим количеством правил (>10000), при наличии ОЗУ не менее 16 Гб, рекомендуется делать размер кэша правил не меньше, чем количество возможных комбинаций сигнатур пакетов (адреса и порты источника и приемника, протокол).

Также необходимо выставить максимальный размер хэш-таблицы для stateful-фильтрации (фильтрация с контролем состоянии сессии) при помощи утилиты `drv_mgr`:

```
root@Gate8000HE:~# drv_mgr set fw_storage_hash_bits 22
root@Gate8000HE:~# drv_mgr show | grep fw_storage_hash_bits
```

```
fw_storage_hash_bits     22
```

Соответствующие настройки для ПАК «С-Терра Шлюз» 7000HE и «С-Терра Шлюз» 8000HE можно найти в главе «Приложение».

Также стоит учитывать, что инспектирование, например, 40 миллионов сессий, может занимать около 30 гигабайт оперативной памяти, поэтому необходимо следить за количеством используемой оперативной памяти при помощи команды `free -h` (графа `used`).

Пример настройки распределения ядер

Предположим, ПАК «С-Терра Шлюз» 8000НЕ работает в режиме «межсетевое экранирование». Наиболее высокопроизводительным вариантом для ПАК «С-Терра Шлюз» 8000НЕ будет следующее распределение ядер: 7 ядер на обработку прерываний (irq) и 25 ядер на основную обработку пакетов.

Наглядная схема распределения ядер выглядит следующим образом:

```
NUMA0:
      |---------8 (work)-phys
0(irq)|---------32(work)-thread 0
 phys |---------40(work)-thread 8

      |---------9 (work)-phys
1(irq)|---------33(work)-thread 1
 phys |---------41(work)-thread 9

      |---------10(work)-phys
2(irq)|---------34(work)-thread 2
 phys |---------42(work)-thread 10

      |---------11(work)-phys
3(irq)|---------35(work)-thread 3
 phys |---------43(work)-thread 11

      |---------12(work)-phys
4(irq)|---------36(work)-thread
 phys |---------44(work)-thread 12

      |---------13(work)-phys
5(irq)|---------37(work)-thread
 phys |---------45(work)-thread 13

      |---------14(work)-phys
6(irq)|---------38(work)-thread 6
 phys |---------46(work)-thread 14

      |---------15(work)-phys
7(irq)|---------39(work)-thread 39
 phys |---------47(work)-thread 15
```

```
NUMA1:
16(free)
17(free)
18(free)
19(free)
20(free)
21(free)
22(free)
23(free)
24(free)
25(free)
26(free)
27(free)
28(free)
29(free)
30(free)
31(free)
48(free)
49(free)
50(free)
51(free)
52(free)
53(free)
54(free)
55(free)
56(free)
57(free)
58(free)
59(free)
60(free)
61(free)
62(free)
63(free)
```

При формировании списков доступа рекомендуется придерживаться следующей последовательности:
Начало списка: запрещающие правила.
Середина списка: разрешающие правила, под которые попадает наибольшее количество сессий.
Конец списка: разрешающие правила, под которые попадают конкретные сессии.
Данная последовательность является наиболее оптимальной с точки зрения обработки сетевых сессий драйвером Продукта.

Режим «шифрование на уровне L2»

В режиме «шифрование на уровне L2» также необходимо, чтобы **количество ядер, отвечающих за прерывания (irq), было равно количеству очередей (QNUM), выставленных на сетевой карте**. Для работы с трафиком уровня L2 предусмотрен сервис `l2svc`. Логика работы следующая: на сетевой интерфейс платформы поступают L2-фреймы, далее, они инкапсулируются в UDP и IP при помощи сервиса `l2svc`, маршрутизируются, а затем, шифруются средствами драйвера Продукта (`vpndrv`).

Настройка туннельных интерфейсов

Сервис `l2svc` позволяет привязать работу определенного туннельного L2 интерфейса (далее – «туннельный интерфейс») к определенному ядру при помощи строки `bind_to_core_id 0` в конфигурационном файле туннельного интерфейса:

```
root@Gate8000HE:~# cat /opt/l2svc/etc/tap1.conf
vif tap1
local 1.1.1.1
remote 1.1.1.2
port 50000
bridge br0
capture eth2
mssfix 1400
passtos
bind_to_core_id 0
bonding bond0
rcvbuf 1048576
sndbuf 1048576
```

- bind_to_core_id 0 – туннельный интерфейс tap1 будет работать на ядре под номером 0;
- rcvbuf 1048576 – размер буфера отправки пакетов на данном tap-интерфейсе;
- sndbuf 1048576 – размер буфера приёма пакетов на данном tap-интерфейсе;
- bonding bond0 – туннельный интерфейс tap1 будет добавлен в интерфейс-балансировщик bond0.

Настройка файла global.cfg

Файл `global.cfg` позволяет выставить количество **последних** ядер, которое будет отведено под сервис `l2svc`. Для этого необходимо привести его к следующему виду:

```
root@Gate8000HE:~# vim.tiny /opt/l2svc/etc/global.cfg | grep CORE
L2VPN_CORE_COUNT="4"
root@Gate8000HE:~# vim.tiny /opt/l2svc/etc/global.cfg | grep BOND
BOND_MODE="balance-xor"
```

- «L2_VPN_CORE_COUNT» – задает количество ядер для работы l2svc процессов;
- «BOND_MODE» – задает режим балансировки трафика между туннельными интерфейсами, объединёнными в bond.

Параметр `L2_VPN_CORE_COUNT` задает число последних ядер в системе, на которых будут выполняться процессы l2svc. По умолчанию (`L2VPN_CORE_COUNT=1`) все процессы l2svc выполняются на ядре с максимальным номером в системе (см. вывод команды `lscpu`). Например, если в системе 64 ядра, то при `L2VPN_CORE_COUNT=4`, процессам l2svc будут доступны ядра с номерами 60, 61, 62, 63. Выбор ядра для выполнения процесса l2svc происходит по алгоритму кругового цикла (round-robin), соответственно, количество конфигурационных файлов может превышать количество отведенных ядер для выполнения l2svc.

Если настройка распределения ядер для работы туннельных интерфейсов производится через конфигурационные файлы `/opt/l2svc/etc/tap.conf` и через глобальный файл конфигурации `/opt/l2svc/etc/global.cfg`, то наибольший приоритет будут иметь настройки, произведенные в файлах `/opt/l2svc/etc/tap.conf`. То есть, если в файле `/opt/l2svc/etc/global.cfg` параметр `L2_VPN_CORE_COUNT` будет равен 4, а в настройках четырёх туннельных интерфейсов будут привязки, например, к 0, 1, 2 и 3 ядрам, то сервис `l2svc` будет использовать для работы туннельных интерфейсов именно ядра 0, 1, 2 и 3 (по умолчанию сервис `l2svc` считывает значение параметра `L2_VPN_CORE_COUNT` и занимает заданное в нём количество ядер, начиная с последнего ядра второй NUMA-ноды в обратом порядке – последнее, предпоследнее и т.д.).

Рекомендуемые настройки сервиса l2svc при использовании ПАК в топологии «точка-точка»

При использовании криптошлюзов в топологии «точка-точка» в режиме «шифрование на уровне L2» рекомендуется объединять туннельные интерфейсы в bonding-интерфейс. Bonding-интерфейс является балансировщиком, который позволяет равномерно распределять сетевые пакеты по туннельным интерфейсам. Рекомендуется использовать bonding-интерфейс в режиме `balance-xor`.

Будьте внимательны, в режиме `balance-xor` при вычислении хеша от MAC-адреса учитывается только последний (младший) байт MAC-адреса. Если он в двух пакетах от разных устройств будет одинаковым, то токие пакеты попадут на один туннельный интерфейс.

Также рекомендуется использовать такое количество туннельных интерфейсов, которое будет кратно значению параметра «L2_VPN_CORE_COUNT» в том случае, если привязки к ядрам не настроены через туннельные интерфейсы (для АП «С-Терра Шлюз» 8000HE значение «L2_VPN_CORE_COUNT» рекомендуется выставить равным 16). При этом, рекомендуется использовать такое количество оконечных устройств, которое будет в 2 раза больше количества туннельных интерфейсов. Чем больше количество оконечных устройств относительно количества туннельных интерфейсов, тем более равномерно сетевые пакеты будут распределяться по этим туннельным интерфейсам. В режиме «шифрование на уровне L2» распределение ядер зависит от модели платформы.

Пример настройки распределения ядер

Предположим, ПАК «С-Терра Шлюз» 8000HE работает в режиме «шифрование на уровне L2». Наиболее высокопроизводительным вариантом будет следующее распределение ядер: 16 на обработку прерываний (irq), 32 ядра на основную обработку пакетов (work, 16 из которых отведены на работу туннельных интерфейсов сервиса l2svc) и 16 свободных ядер (free). Таким образом, распределение ядер будет следующим:

```
root@Gate8000HE:~# cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options vpndrvr cpu_distribution=*:#0,1,2,3,4,5,6,7,16,17,18,19,20,21,22,23/#8~0,9~1,10~2,11~3,12~4,13~5,14~6,15~7,32~0,33~1,34~2,35~3,36~4,37~5,38~6,39~7,24~16,25~17,26~18,27~19,28~20,29~21,30~22,31~23,48~16,49~17,50~18,51~19,52~20,53~21,54~22,55~23
```

Наглядная схема распределения ядер выглядит следующим образом:

```
NUMA0:
      |---------8 (l2svc+work)-phys
0(irq)|---------32(work)-thread 0
 phys |---------40(free)-thread 8

      |---------9 (l2svc+work)-phys
1(irq)|---------33(work)-thread 1
 phys |---------41(free)-thread 9

      |---------10(l2svc+work)-phys
2(irq)|---------34(work)-thread 2
 phys |---------42(free)-thread 10

      |---------11(l2svc+work)-phys
3(irq)|---------35(work)-thread 3
 phys |---------43(free)-thread 11

      |---------12(l2svc+work)-phys
4(irq)|---------36(work)-thread
 phys |---------44(free)-thread 12

      |---------13(l2svc+work)-phys
5(irq)|---------37(work)-thread 5
 phys |---------45(free)-thread 13

      |---------14(l2svc+work)-phys
6(irq)|---------38(work)-thread 6
 phys |---------46(free)-thread 14

      |---------15(l2svc+work)-phys
7(irq)|---------39(work)-thread 7
 phys |---------47(free)-thread 15
```

```
NUMA1:
      |---------24(l2svc+work)-phys
16(irq)|---------48(work)-thread 16
 phys |---------56(free)-thread 24

      |---------25(l2svc+work)-phys
17(irq)|---------49(work)-thread 17
 phys |---------57(free)-thread 25

      |---------26(l2svc+work)-phys
18(irq)|---------50(work)-thread 18
 phys |---------58(free)-thread 26

      |---------27(l2svc+work)-phys
19(irq)|---------51(work)-thread 19
 phys |---------59(free)-thread 27

      |---------28(l2svc+work)-phys
20(irq)|---------52(work)-thread 20
 phys |---------60(free)-thread 28

      |---------29(l2svc+work)-phys
21(irq)|---------53(work)-thread 21
 phys |---------61(free)-thread 29

      |---------30(l2svc+work)-phys
22(irq)|---------54(work)-thread 22
 phys |---------62(free)-thread 30

      |---------31(l2svc+work)-phys
23(irq)|---------55(work)-thread 23
 phys |---------63(free)-thread 31
```

Привязки туннельных интерфейсов сервиса `l2svc` к ядрам будут следующими:

```
root@Gate8000HE:~# cat /opt/l2svc/etc/tap* | grep bind
bind_to_core_id 24
bind_to_core_id 9
bind_to_core_id 10
bind_to_core_id 11
bind_to_core_id 12
bind_to_core_id 13
bind_to_core_id 14
bind_to_core_id 15
bind_to_core_id 25
bind_to_core_id 26
bind_to_core_id 27
bind_to_core_id 28
bind_to_core_id 29
bind_to_core_id 30
bind_to_core_id 31
bind_to_core_id 8
```

Проверка результатов оптимизации производительности АП в режиме «шифрование на уровне L2»

Вывод команды `watch cpupower monitor`:

		Nehalem				Mperf			Idle_Stats			
PKG	CORE	CPU	C3	C6	PC3	PC6	C0	Cx	Freq	POLL	C1	C1E	C6
0	0	0	0.00	0.00	0.00	0.00	47.32	52.68	3078	0.10	94.31	0.61	0.00
0	0	32	0.00	0.00	0.00	0.00	97.48	2.52	3080	0.04	2.91	0.00	0.00
0	1	1	0.00	0.00	0.00	0.00	52.50	47.50	3078	0.04	94.45	0.82	0.00
0	1	33	0.00	0.00	0.00	0.00	98.52	1.48	3080	0.06	1.77	0.00	0.00
0	2	2	0.00	0.00	0.00	0.00	52.08	47.92	3078	0.07	94.17	1.00	0.00
0	2	34	0.00	0.00	0.00	0.00	98.45	1.55	3080	0.04	1.84	0.00	0.00
0	3	3	0.00	0.00	0.00	0.00	50.85	49.15	3078	0.05	94.60	0.75	0.00
0	3	35	0.00	0.00	0.00	0.00	98.34	1.66	3080	0.09	2.00	0.00	0.00
0	4	4	0.00	0.00	0.00	0.00	46.72	53.28	3078	0.06	94.72	0.76	0.00
0	4	36	0.00	0.00	0.00	0.00	97.85	2.15	3080	0.04	2.52	0.00	0.00
0	5	5	0.00	0.00	0.00	0.00	47.65	52.35	3078	0.04	58.41	10.04	12.88
0	5	37	0.00	0.00	0.00	0.00	92.25	7.75	3080	0.00	8.55	0.01	0.00
0	6	6	0.00	0.00	0.00	0.00	46.57	53.43	3078	0.04	73.04	7.61	7.24
0	6	38	0.00	0.00	0.00	0.00	94.76	5.24	3080	0.04	5.96	0.00	0.00
0	7	7	0.00	0.00	0.00	0.00	50.94	49.06	3078	0.05	94.44	0.90	0.00
0	7	39	0.00	0.00	0.00	0.00	96.96	3.04	3080	0.02	3.47	0.00	0.00
0	8	8	0.00	0.00	0.00	0.00	99.78	0.22	3078	0.06	0.29	0.00	0.00
0	8	40	0.00	0.00	0.00	0.00	0.02	99.98	3119	0.00	0.00	0.00	100.0
0	9	9	0.00	0.00	0.00	0.00	99.86	0.14	3078	0.01	0.18	0.00	0.00
0	9	41	0.00	0.00	0.00	0.00	0.02	99.98	3237	0.00	0.00	0.00	100.0
0	10	10	0.00	0.00	0.00	0.00	99.74	0.26	3078	0.04	0.32	0.00	0.00
0	10	42	0.00	0.00	0.00	0.00	0.02	99.98	3433	0.00	0.00	0.00	100.0
0	11	11	0.00	0.00	0.00	0.00	99.93	0.07	3078	0.02	0.08	0.00	0.00
0	11	43	0.00	0.00	0.00	0.00	0.08	99.92	3146	0.00	0.00	0.00	99.96
0	12	12	0.00	0.00	0.00	0.00	99.89	0.11	3078	0.03	0.15	0.00	0.00
0	12	44	0.00	0.00	0.00	0.00	0.03	99.97	3392	0.00	0.00	0.00	100.0
0	13	13	0.00	0.00	0.00	0.00	99.39	0.61	3078	0.07	0.76	0.00	0.00
0	13	45	0.00	0.00	0.00	0.00	0.03	99.97	3097	0.00	0.00	0.00	99.99
0	14	14	0.00	0.00	0.00	0.00	99.86	0.14	3078	0.04	0.19	0.00	0.00
0	14	46	0.00	0.00	0.00	0.00	0.12	99.88	3117	0.00	0.00	0.00	100.1
0	15	15	0.00	0.00	0.00	0.00	99.55	0.45	3079	0.05	0.55	0.00	0.00
0	15	47	0.00	0.00	0.00	0.00	0.02	99.98	3851	0.00	0.00	0.00	100.00
1	0	16	0.00	0.00	0.00	0.00	48.30	51.70	3079	0.05	94.68	0.85	0.00
1	0	48	0.00	0.00	0.00	0.00	98.33	1.67	3079	0.05	1.99	0.00	0.00
1	1	17	0.00	0.00	0.00	0.00	40.40	59.60	3079	0.06	91.71	3.11	0.00
1	1	49	0.00	0.00	0.00	0.00	97.99	2.01	3079	0.04	2.37	0.00	0.00
1	2	18	0.00	0.00	0.00	0.00	41.50	58.50	3079	0.05	60.22	22.76	0.00
1	2	50	0.00	0.00	0.00	0.00	91.45	8.55	3079	0.01	9.46	0.01	0.00
1	3	19	0.00	0.00	0.00	0.00	38.92	61.08	3079	0.06	94.99	0.63	0.00
1	3	51	0.00	0.00	0.00	0.00	98.09	1.91	3079	0.03	2.21	0.00	0.00
1	4	20	0.00	0.00	0.00	0.00	41.46	58.54	3079	0.04	95.11	0.75	0.00
1	4	52	0.00	0.00	0.00	0.00	97.55	2.45	3079	0.05	2.87	0.00	0.00
1	5	21	0.00	0.00	0.00	0.00	36.44	63.56	3079	0.06	93.74	2.03	0.00
1	5	53	0.00	0.00	0.00	0.00	98.68	1.32	3079	0.07	1.60	0.00	0.00
1	6	22	0.00	0.00	0.00	0.00	42.61	57.39	3079	0.05	93.02	2.21	0.00
1	6	54	0.00	0.00	0.00	0.00	97.68	2.32	3079	0.05	2.74	0.00	0.00
1	7	23	0.00	0.00	0.00	0.00	50.03	49.97	3081	0.00	48.93	18.18	6.58
1	7	55	0.00	0.00	0.00	0.00	97.03	2.97	3079	0.05	3.44	0.00	0.00
1	8	24	0.00	0.00	0.00	0.00	99.90	0.10	3080	0.02	0.13	0.00	0.00
1	8	56	0.00	0.00	0.00	0.00	0.08	99.92	2972	0.00	0.00	0.00	99.93
1	9	25	0.00	0.00	0.00	0.00	99.92	0.08	3080	0.01	0.09	0.00	0.00
1	9	57	0.00	0.00	0.00	0.00	0.03	99.97	2718	0.00	0.00	0.00	99.98
1	10	26	0.00	0.00	0.00	0.00	99.74	0.26	3080	0.07	0.35	0.00	0.00
1	10	58	0.00	0.00	0.00	0.00	0.04	99.96	3144	0.00	0.00	0.00	99.98
1	11	27	0.00	0.00	0.00	0.00	99.87	0.13	3080	0.05	0.16	0.00	0.00
1	11	59	0.00	0.00	0.00	0.00	0.03	99.97	3137	0.00	0.00	0.00	99.98
1	12	28	0.00	0.00	0.00	0.00	99.89	0.11	3080	0.04	0.14	0.00	0.00
1	12	60	0.00	0.00	0.00	0.00	0.08	99.92	3189	0.00	0.00	0.00	99.93
1	13	29	0.00	0.00	0.00	0.00	99.90	0.10	3080	0.01	0.12	0.00	0.00
1	13	61	0.00	0.00	0.00	0.00	0.03	99.97	3128	0.00	0.00	0.00	99.97
1	14	30	0.00	0.00	0.00	0.00	99.93	0.07	3080	0.02	0.09	0.00	0.00
1	14	62	0.00	0.00	0.00	0.00	0.03	99.97	3109	0.00	0.00	0.00	99.97
1	15	31	0.00	0.00	0.00	0.00	99.74	0.26	3080	0.09	0.35	0.00	0.00
1	15	63	0.00	0.00	0.00	0.00	0.04	99.96	3492	0.00	0.00	0.00	99.96

Рисунок 3. Загрузка ядер процессоров в режиме работы «шифрование на уровне L2» на АП «С-Терра Шлюз» 8000НЕ (2 x Intel Xeon Gold 6226R 2.90GHz)

Ядра, на которых работают шифрование и туннельные интерфейсы, загружены практически по максимуму. Ядра, которые не заняты вообще, являются hyper-threading ядрами тех, на которых работают одновременно и шифрование, и туннельные интерфейсы. Если занять данные ядра, то будет наблюдаться существенное снижение производительности. Соответствующие настройки для ПАК «С-Терра Шлюз» 7000HE и «С-Терра Шлюз» 8000HE можно найти в главе «Приложение».

Ручная балансировка трафика по нескольким IPsec туннелям

В некоторых случаях сетевая карта может неравномерно распределить поступающие пакеты по очередям, что может привести к неравномерной загрузке процессоров и, как следствие, к снижению производительности платформы. Посмотреть балансировку пакетов, например, на интерфейсе eth1, можно при помощи команды (регулярное выражение после grep может зависеть от версии драйвера сетевой карты):

```
root@GW1:~# ethtool -S eth1 | grep -E rx.*[0-9]+.*packets: | head -n 16
```

Для равномерного распределения пакетов по очередям необходимо применить ручную балансировку трафика.

Основными условиями для получения максимальной производительности являются следующие:
1. Количество очередей на сетевой карте должно быть равно количеству ядер, отвечающих за прерывания (irq), и количеству IPsec-туннелей.
2. Трафик должен быть равномерно распределён по очередям сетевой карты.

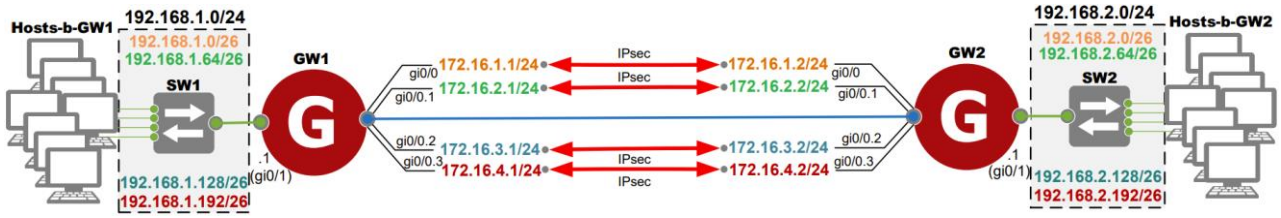


Рисунок 4. Схема взаимодействия при балансировке трафика по нескольким IPsec туннелям на примере АП «С-Терра Шлюз» 7000HE (2 x Intel Xeon 2643v4 3.40GHz)

Рассмотрим детальнее схему взаимодействия при балансировке трафика по нескольким IPsec туннелям на примере криптошлюза GW1 (предполагается, что настроено такое распределение ядер, при котором **на предварительную обработку пакетов выделено 4 ядра** и **на каждом сетевом интерфейсе выставлено 4 очереди**): пакеты из подсети 192.168.1.0/26 направляются в очередь 0, из подсети 192.168.1.64/26 в очередь 1, из подсети 192.168.1.128/26 в очередь 2, из подсети 192.168.1.192/26 в очередь 3 (данное распределение достигается либо большим количеством сетевых сессий, либо ручной балансировкой трафика средствами утилиты `ethtool`, описание представлено ниже), благодаря этому, нагрузка будет равномерно распределена по ядрам предварительной обработки, при этом, пакеты из данных подсетей, средствами настроек Продукта, будут направлены в соответствующие им IPsec туннели (на схеме отмечены одинаковыми цветами), что поспособствует равномерному распределению нагрузки по ядрам основной обработки (так как процесс шифрования происходит на ядрах основной обработки). Таким образом, ядра предварительной обработки и ядра основной обработки будут загружены равномерно, что позволит достичь максимальной производительности криптошлюза.

Для того, чтобы настроить балансировку трафика по четырём IPsec туннелям на криптошлюзе GW1 на примере АП «С-Терра Шлюз» 7000HE (2 x Intel Xeon 2643v4 3.40GHz), выполните следующие шаги:

Настройки ниже предполагают, что устройства «С-Терра Шлюз» инициализированы, на них установлены корневой сертификат и сертификат устройства, созданы isakmp-профили в соответствии со сценарием «Организация защищенного взаимодействия между подсетями центрального офиса и филиалов посредством IPsec VPN туннеля (протокол IPv4)».

1. Установите следующее распределение ядер путём приведения файла `/etc/modprobe.d/vpndrvr.conf` к следующему виду:

```
root@GW1:~# vim.tiny /etc/modprobe.d/vpndrvr.conf
```

```
options                                                                    vpndrvr
cpu_distribution=*:#0,1,6,7/#2~0,3~1,4~0,5~1,12~0,13~1,14~0,15~1,16~0,17~1,8~6,9~7,10
~6,11~7,18~6,19~7,20~6,21~7,22~6,23~7
```

2. Установите адреса на LAN и WAN интерфейсы согласно Рисунок 4:

```
GW1(config)#interface TenGigabitEthernet 0/1
```

```
GW1(config-if)#ip address 192.168.1.1 255.255.255.0
GW1(config-if)# exit
GW1(config)#interface TenGigabitEthernet 0/0
GW1(config-if)#ip address 172.16.1.1 255.255.255.0
GW1(config-if)# ip address 172.16.2.1 255.255.255.0
GW1(config-if)# ip address 172.16.3.1 255.255.255.0
GW1(config-if)# ip address 172.16.4.1 255.255.255.0
GW1(config-if)# exit
```

3. Создайте списки доступа (ACL) для трафика, который нужно защищать между криптошлюзами. Так как трафик будет передаваться между подсетями 192.168.1.0/24 и 192.168.2.0/24, то необходимо на криптошлюзе GW1 разделить подсеть 192.168.1.0/24 на 4 подсети с маской 255.255.255.192 (/26) для того, чтобы разбалансировать трафик по 4 IPsec-туннелям (аналогичным образом можно создать списки доступа и для другого количества туннелей):

```
GW1(config)#ip access-list extended IPSEC_ACL_FROM_GW11
GW1(config-ext-nacl)#permit ip 192.168.1.0 0.0.0.63 192.168.2.0 0.0.0.63
GW1(config-ext-nacl)#exit
GW1(config)#ip access-list extended IPSEC_ACL_GW12
GW1(config-ext-nacl)#permit ip 192.168.1.64 0.0.0.63 192.168.2.64 0.0.0.63
GW1(config-ext-nacl)#exit
GW1(config)#ip access-list extended IPSEC_ACL_GW13
GW1(config-ext-nacl)#permit ip 192.168.1.128 0.0.0.63 192.168.2.128 0.0.0.63
GW1(config-ext-nacl)#exit
GW1(config)#ip access-list extended IPSEC_ACL_GW14
GW1(config-ext-nacl)#permit ip 192.168.1.192 0.0.0.63 192.168.2.192 0.0.0.63
GW1(config-ext-nacl)#exit
```

4. Настройте 4 IPsec-туннеля и к каждому из них привяжите свой список доступа:

   4.1. Трафик из подсети **192.168.1.0/26** будет попадать в IPsec-туннель **172.16.1.1-172.16.1.2**

   4.2. Трафик из подсети **192.168.1.64/26** будет попадать в IPsec-туннель **172.16.2.1-172.16.2.2**

   4.3. Трафик из подсети **192.168.1.128/26** будет попадать в IPsec-туннель **172.16.3.1-172.16.3.2**

   4.4. Трафик из подсети **192.168.1.192/26** будет попадать в IPsec-туннель **172.16.4.1-172.16.4.2**

```
GW1(config)#crypto map VPN 1 ipsec-isakmp
GW1(config-crypto-map)#match address IPSEC_ACL_GW11
GW1(config-crypto-map)#set transform-set MAGMA_GOST_ENCRYPT_AND_INTEGRITY
GW1(config-crypto-map)#set isakmp profile ALGS
GW1(config-crypto-map)#set pre-fragmentation
GW1(config-crypto-map)#set local-address 172.16.1.1
GW1(config-crypto-map)#set peer 172.16.1.2
GW1(config-crypto-map)#set dead-connection history off
GW1(config-crypto-map)#exit
GW1(config)#crypto map VPN 2 ipsec-isakmp
GW1(config-crypto-map)#match address IPSEC_ACL_GW12
GW1(config-crypto-map)#set transform-set MAGMA_GOST_ENCRYPT_AND_INTEGRITY
GW1(config-crypto-map)#set isakmp profile ALGS
GW1(config-crypto-map)#set pre-fragmentation
GW1(config-crypto-map)#set local-address 172.16.2.1
GW1(config-crypto-map)#set peer 172.16.2.2
GW1(config-crypto-map)#set dead-connection history off
GW1(config-crypto-map)#exit
GW1(config)#crypto map VPN 3 ipsec-isakmp
GW1(config-crypto-map)#match address IPSEC_ACL_GW13
GW1(config-crypto-map)#set transform-set MAGMA_GOST_ENCRYPT_AND_INTEGRITY
GW1(config-crypto-map)#set isakmp profile ALGS
GW1(config-crypto-map)#set pre-fragmentation
GW1(config-crypto-map)#set local-address 172.16.3.1
GW1(config-crypto-map)#set peer 172.16.3.2
GW1(config-crypto-map)#set dead-connection history off
GW1(config-crypto-map)#exit
GW1(config)#crypto map VPN 4 ipsec-isakmp
GW1(config-crypto-map)#match address IPSEC_ACL_GW14
GW1(config-crypto-map)#set transform-set MAGMA_GOST_ENCRYPT_AND_INTEGRITY
```

```
GW1(config-crypto-map)#set isakmp profile ALGS
GW1(config-crypto-map)#set pre-fragmentation
GW1(config-crypto-map)#set local-address 172.16.4.1
GW1(config-crypto-map)#set peer 172.16.4.2
GW1(config-crypto-map)#set dead-connection history off
GW1(config-crypto-map)#exit
```

5. Прикрепите крипто-карту к WAN-интерфейсу:

```
GW1(config)#interface TenGigabitEthernet 0/0
GW1(config-if)#crypto map VPN
GW1(config-if)#exit
```

6. Установите маршруты до защищаемых подсетей следующим образом:

```
GW1(config)#ip route 192.168.2.0 255.255.255.192 172.16.1.2
GW1(config)#ip route 192.168.2.64 255.255.255.192 172.16.2.2
GW1(config)#ip route 192.168.2.128 255.255.255.192 172.16.3.2
GW1(config)#ip route 192.168.2.192 255.255.255.192 172.16.4.2
GW1(config)#end
```

В большинстве случаев, за счёт большого количества сессий, пакеты должны быть равномерно распределены по нескольким IPsec-туннелям, и настроек выше, как правило, достаточно. Если же сессий немного, то стоит обратиться к пункту ниже.

7. Предположим, ПАК «С-Терра Шлюз» 7000HE, устройство GW1, обрабатывает четыре сетевые сессии: LAN-интерфейс – это eth0 и поступающие на него пакеты имеют destination-адреса 192.168.2.1/24, 192.168.2.65/24, 192.168.2.129/24, 192.168.2.193/24, WAN-интерфейс – это eth1 и поступающие на него пакеты имеют destination-адреса 172.16.1.1/24, 172.16.2.1/24, 172.16.3.1/24, 172.16.4.1/24. Тогда установить количество очередей на сетевых интерфейсах и осуществить балансировку пакетов по данным очередям необходимо следующим образом (скрипт ниже необходимо добавить в файл `/etc/rc.local.inc`):

```
QNUM=4
LAN_IF=eth0
WAN_IF=eth1

for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done

ethtool -K $WAN_IF ntuple on
ethtool -K $LAN_IF ntuple on

ethtool --config-ntuple $WAN_IF flow-type ip4 dst-ip 172.16.1.1 action 0 loc 100
ethtool --config-ntuple $WAN_IF flow-type ip4 dst-ip 172.16.2.1 action 1 loc 101
ethtool --config-ntuple $WAN_IF flow-type ip4 dst-ip 172.16.3.1 action 2 loc 102
ethtool --config-ntuple $WAN_IF flow-type ip4 dst-ip 172.16.4.1 action 3 loc 103

ethtool --config-ntuple $LAN_IF flow-type ip4 dst-ip 192.168.2.1 action 0 loc 110
ethtool --config-ntuple $LAN_IF flow-type ip4 dst-ip 192.168.2.65 action 1 loc 111
ethtool --config-ntuple $LAN_IF f low-type ip4 dst-ip 192.168.2.129 action 2 loc 112
ethtool --config-ntuple $LAN_IF flow-type ip4 dst-ip 192.168.2.193 action 3 loc 113
```

- «action 0» – очередь под номером 0.
- «ip4» – название протокола.
- «loc 100» – номер правила на интерфейсе.

Пакеты, поступающие на интерфейс eth0 (LAN-интерфейс) и имеющие destination-адрес 192.168.2.1/24, будут отправлены в нулевую очередь, имеющие destination-адрес 192.168.2.65/24 – в первую очередь и так далее.

Подробнее ознакомиться с правилами распределения пакетов при помощи утилиты ethtool можно в инструкции: `man ethtool`.

Настройка криптошлюза GW1 завершена. Аналогичные настройки необходимо произвести и на криптошлюзе GW2.

После балансировки трафика необходимо отследить загруженность ядер и распределение пакетов по очередям и ядрам. Загрузка должна быть равномерной (если количество трафика по сессиям также рас переделено равномерно).

Параметр драйвера Продукта pq_wait_cycles

Параметр `pq_wait_cycles` задает количество холостых попыток рабочих потоков драйвера Продукта для ожидания получения сетевого пакета перед тем, как этот поток «уснёт». Поскольку холостые попытки потребляют процессорное время в случае, если через драйвер проходит небольшой объем сетевого трафика, полезная производительность ядра, на котором выполняется рабочий поток, будет расходоваться на эти холостые попытки.

Если значение `pq_wait_cycles` выставить в 1, то количество холостых попыток будет равно 1, и, следовательно, загруженность ядер процессора будет отображаться верно. Но, если трафик, проходящий через драйвер, имеет неравномерный характер, то производительность драйвера может снизиться. Это происходит в силу того, что в один момент времени может прийти большое количество сетевых пакетов, а часть ядер основной обработки «спит» и требуется некоторое время на их «пробуждение». Из-за эффекта «пробуждения» ядер основной обработки увеличивается и время обработки пакетов и, возможно, будут происходить потери пакетов.

Если необходимо получать более точную информацию по загрузке ядер, **то временно** воспользуйтесь командой:

```
root@Gate8000HE:~# drv_mgr set pq_wait_cycles 1
```

Для того, чтобы посмотреть текущую настройку данного параметра, необходимо воспользоваться командой:

```
root@Gate8000HE:~# drv_mgr show | grep pq_wait_cycles
pq_wait_cycles            1
```

Без согласования с вендором менять значение параметра `pq_wait_cycles` не рекомендуется.

Приложение

Проверка настроек аппаратных средств ПАК

Hyper-Threading

Hyper-Threading – это аппаратная технология, позволяющая обрабатывать на каждом ядре процессора несколько программных потоков (в случае отсутствия Hyper-Threading одно ядро процессора обрабатывает один программный поток). Чем больше программных потоков обрабатывается на одном ядре, тем больше задач может выполняться на нём параллельно. После включения технологии Hyper-Threading в рамках каждого физического ядра, будут созданы два логических ядра, которые могут обрабатывать разные программные потоки одновременно. Для того, чтобы посмотреть, включена ли технология Hyper-Threading на устройстве, необходимо воспользоваться командой `lscpu` и обратить внимание на поле «Thread(s) per core»:

```
root@Gate8000HE:~# lscpu
```

```
…
CPU(s):                  63
On-line CPU(s) list:     0-63
…
Thread(s) per core:      2
Core(s) per socket:      16
…
```

Вывод утилиты приведён на АП «С-Терра Шлюз» 8000HE. Поле «Thread(s) per core» показывает, какое количество потоков может обрабатывать одно ядро. Если включена технология Hyper-Threading, то каждое ядро будет обрабатывать два потока параллельно. Поле «CPU(s)» показывает, сколько ядер доступно в операционной системе.

Если технология Hyper-Threading отключена, необходимо обратиться в техническую поддержку для получения инструкции по её включению.

NUMA

NUMA (Non-Uniform Memory Access) – это аппаратная технология взаимодействия одного процессора с блоками памяти другого процессора. Технология NUMA способствует распределению процессов в системе таким образом, чтобы они получали области оперативной памяти, расположенные максимально близко к процессорам, на которых работают. Для АП «С-Терра Шлюз» 7000HE и «С-Терра Шлюз» 8000HE доступно разделение процессоров и памяти на два NUMA узла. Для того, чтобы посмотреть, включена ли технология NUMA на устройстве, необходимо воспользоваться командой `lscpu`:

```
root@Gate8000HE:~# lscpu | grep NUMA
```

```
NUMA:
  NUMA node(s):          2
  NUMA node0 CPU(s):     0-15,32-47
  NUMA node1 CPU(s):     16-31,48-63
```

Строка «NUMA node(s)» содержит количество NUMA узлов, доступных операционной системе. Вывод утилиты свидетельствует о том, что в системе доступны 2 NUMA узла (данная технология на устройстве включена). В первом NUMA узле находятся ядра с номерами: 0-15, 32-47 (ядра процессора cpu0), во втором NUMA узле находятся ядра с номерами: 16-31, 48-63 (ядра процессора cpu1).

Если технология NUMA отключена, необходимо обратиться в техническую поддержку для получения инструкции по её включению.

ПАК «С-Терра Шлюз» 7000HE

lscpu

```
Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         46 bits physical, 48 bits virtual
```

```
  Byte Order:            Little Endian
CPU(s):                  24
  On-line CPU(s) list:   0-23
Vendor ID:               GenuineIntel
  BIOS Vendor ID:        Intel
  Model name:            Intel(R) Xeon(R) CPU E5-2643 v4 @ 3.40GHz
    BIOS Model name:     Intel(R) Xeon(R) CPU E5-2643 v4 @ 3.40GHz  CPU @ 3.4GHz
    BIOS CPU family:     179
    CPU family:          6
    Model:               79
    Thread(s) per core:  2
    Core(s) per socket:  6
    Socket(s):           2
    Stepping:            1
    CPU(s) scaling MHz:  100%
    CPU max MHz:         3700.0000
    CPU min MHz:         1200.0000
    BogoMIPS:            6784.89
    Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm
constant_tsc arc
                         h_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid
aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr
pdcm pcid dca ss
                         e4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave
avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb cat_l3 cdp_l3 invpcid_single
pti intel_ppin ss
                         bd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid
ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm cqm rdt_a rdseed
adx smap intel_pt xsave
                         opt cqm_llc cqm_occup_llc cqm_mbm_total cqm_mbm_local dtherm
ida arat pln pts md_clear flush_l1d
Virtualization features:
  Virtualization:        VT-x
Caches (sum of all):
  L1d:                   384 KiB (12 instances)
  L1i:                   384 KiB (12 instances)
  L2:                    3 MiB (12 instances)
  L3:                    40 MiB (2 instances)
NUMA:
  NUMA node(s):          2
  NUMA node0 CPU(s):     0-5,12-17
  NUMA node1 CPU(s):     6-11,18-23
Vulnerabilities:
  Gather data sampling:  Not affected
  Itlb multihit:         KVM: Mitigation: VMX disabled
  L1tf:                  Mitigation; PTE Inversion; VMX conditional cache flushes, SMT
vulnerable
  Mds:                   Mitigation; Clear CPU buffers; SMT vulnerable
  Meltdown:              Mitigation; PTI
  Mmio stale data:       Mitigation; Clear CPU buffers; SMT vulnerable
  Retbleed:              Not affected
  Spec rstack overflow:  Not affected
  Spec store bypass:     Mitigation; Speculative Store Bypass disabled via prctl
  Spectre v1:              Mitigation; usercopy/swapgs barriers and __user pointer
sanitization
  Spectre v2:             Mitigation; Retpolines, IBPB conditional, IBRS_FW, STIBP
conditional, RSB filling, PBRSB-eIBRS Not affected
  Srbds:                 Not affected
  Tsx async abort:       Mitigation; Clear CPU buffers; SMT vulnerable
```

Режим «шифрование на уровне L3»

Настройка распределения ядер

```
cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options                                                                                 vpndrvr
cpu_distribution=*:#0,1,6,7/#2~0,3~1,4~0,5~1,12~0,13~1,14~0,15~1,16~0,17~1,8~6,9~7,10
~6,11~7,18~6,19~7,20~6,21~7,22~6,23~7
```

Настройка количества очередей на сетевой карте

```
cat /etc/rc.local.inc
```

```
QNUM=4
for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done
```

Режим «межсетевое экранирование»

Настройка распределения ядер

```
cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options                                                                                 vpndrvr
cpu_distribution=*:#0,1,2,3,4,5,6,7,8,9,10,11/#12~0,13~1,14~2,15~3,16~4,17~5,18~6,19~
7,20~8,21~9,22~10,23~11
```

Настройка количества очередей на сетевой карте

```
cat /etc/rc.local.inc
```

```
QNUM=12
for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done
```

Настройка параметров драйвера

```
cat /etc/modprobe.d/vpndrvr.conf | grep fw
```

```
options vpndrvr fw_filter_cache_bucket_size=32
options vpndrvr fw_filter_cache_hash_bits=17
options vpndrvr fw_filter_cache_ttl=3600
```

```
drv_mgr show | grep fw_storage_hash_bits
```

```
fw_storage_hash_bits     22
```

Режим «шифрование на уровне L2»

Настройка распределения ядер

```
cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options                                                                                 vpndrvr
cpu_distribution=*:#0,1,6,7/#2~0,3~1,12~0,13~1,14~0,15~1,8~6,9~7,18~6,19~7,20~6,21~7
```

Настройка количества очередей на сетевой карте

```
cat /etc/rc.local.inc
```

```
QNUM=4
for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
```

```
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done
```

Настройка туннельных интерфейсов (привязки к ядрам)

```
cat /opt/l2svc/etc/tap* | grep bind
```

```
bind_to_core_id 4
bind_to_core_id 5
bind_to_core_id 10
bind_to_core_id 11
```

Глобальные настройки сервиса l2svc

```
cat /opt/l2svc/etc/global.cfg | grep BOND
```

```
BOND_MODE="balance-xor"
BOND_MIIMON="100"
BOND_XMIT_HASH_POLICY="layer2+3"
```

ПАК «С-Терра Шлюз» 8000HE

lscpu

```
Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         46 bits physical, 48 bits virtual
  Byte Order:            Little Endian
CPU(s):                  64
  On-line CPU(s) list:   0-63
Vendor ID:               GenuineIntel
  BIOS Vendor ID:        Intel(R) Corporation
  Model name:            Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
    BIOS Model name:     Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz  CPU @ 2.9GHz
    BIOS CPU family:     179
    CPU family:          6
    Model:               85
    Thread(s) per core:  2
    Core(s) per socket:  16
    Socket(s):           2
    Stepping:            7
    CPU(s) scaling MHz:  33%
    CPU max MHz:         3900.0000
    CPU min MHz:         1200.0000
    BogoMIPS:            5800.00
    Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm
constant_tsc art
                          arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc
cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma
cx16 xtpr pdcm pcid dc
                         a sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes
xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb cat_l3 cdp_l3
invpcid_single intel_ppin ss
                         bd mba ibrs ibpb stibp ibrs_enhanced tpr_shadow vnmi
flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid cqm
mpx rdt_a avx512f avx512dq
                          rdseed adx smap clflushopt clwb intel_pt avx512cd avx512bw
avx512vl xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mbm_total
cqm_mbm_local dtherm ida arat
                          pln pts hwp hwp_act_window hwp_epp hwp_pkg_req pku ospke
avx512_vnni md_clear flush_l1d arch_capabilities
```

```
Virtualization features:
  Virtualization:        VT-x
Caches (sum of all):
  L1d:                   1 MiB (32 instances)
  L1i:                   1 MiB (32 instances)
  L2:                    32 MiB (32 instances)
  L3:                    44 MiB (2 instances)
NUMA:
  NUMA node(s):          2
  NUMA node0 CPU(s):     0-15,32-47
  NUMA node1 CPU(s):     16-31,48-63
Vulnerabilities:
  Gather data sampling:  Mitigation; Microcode
  Itlb multihit:         KVM: Mitigation: VMX disabled
  L1tf:                  Not affected
  Mds:                   Not affected
  Meltdown:              Not affected
  Mmio stale data:       Mitigation; Clear CPU buffers; SMT vulnerable
  Retbleed:              Mitigation; Enhanced IBRS
  Spec rstack overflow:  Not affected
  Spec store bypass:     Mitigation; Speculative Store Bypass disabled via prctl
  Spectre v1:              Mitigation; usercopy/swapgs barriers and __user pointer
sanitization
  Spectre v2:            Mitigation; Enhanced IBRS, IBPB conditional, RSB filling,
PBRSB-eIBRS SW sequence
  Srbds:                 Not affected
  Tsx async abort:       Mitigation; TSX disabled
```

Режим «шифрование на уровне L3»

Настройка распределения ядер

```
cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options                                                                    vpndrvr
cpu_distribution=*:#0,1,2,3,4,5,6,7,16,17,18,19,20,21,22,23/#8~0,9~1,10~2,11~3,12~4,1
3~5,14~6,15~7,32~0,33~1,34~2,35~3,36~4,37~5,38~6,39~7,40~0,41~1,42~2,43~3,44~4,45~5,4
6~6,47~7,24~16,25~17,26~18,27~19,28~20,29~21,30~22,31~23,48~16,49~17,50~18,51~19,52~2
0,53~21,54~22,55~23,56~16,57~17,58~18,59~19,60~20,61~21,62~22,63~23
```

Настройка количества очередей на сетевой карте

```
cat /etc/rc.local.inc
```

```
QNUM=16
for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done
```

Режим «межсетевое экранирование»

Настройка распределения ядер

```
cat /etc/modprobe.d/vpndrvr.conf | grep cpu_distribution
```

```
options                                                                    vpndrvr
cpu_distribution=*:#0,1,2,3,4,5,6,7/#8~0,9~1,10~2,11~3,12~4,13~5,14~6,15~7,32~0,33~1,
34~2,35~3,36~4,37~5,38~6,39~7,40~0,41~1,42~2,43~3,44~4,45~5,46~6,47~7
```

Настройка количества очередей на сетевой карте

```
cat /etc/rc.local.inc
```

```
QNUM=8
```

```
for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done
```

Настройка параметров драйвера

```
cat /etc/modprobe.d/vpndrvr.conf | grep fw
```

```
options vpndrvr fw_filter_cache_bucket_size=32
options vpndrvr fw_filter_cache_hash_bits=17
options vpndrvr fw_filter_cache_ttl=3600
```

```
drv_mgr show | grep fw_storage_hash_bits
```

```
fw_storage_hash_bits     22
```

Режим «шифрование на уровне L2»

Настройка распределения ядер

```
options                                                                                vpndrvr
cpu_distribution=*:#0,1,2,3,4,5,6,7,16,17,18,19,20,21,22,23/#8~0,9~1,10~2,11~3,12~4,1
3~5,14~6,15~7,32~0,33~1,34~2,35~3,36~4,37~5,38~6,39~7,24~16,25~17,26~18,27~19,28~20,2
9~21,30~22,31~23,48~16,49~17,50~18,51~19,52~20,53~21,54~22,55~23
```

Настройка количества очередей на сетевой карте

```
cat /etc/rc.local.inc
```

```
QNUM=16
for phys_iface in $(ls /sys/class/net | grep -P "^eth\d+$")
do
    echo Setting combined queue to $QNUM on $phys_iface ...
    ethtool -L $phys_iface combined $QNUM
done
```

Настройка туннельных интерфейсов (привязки к ядрам)

```
cat /opt/l2svc/etc/tap* | grep bind
```

```
bind_to_core_id 24
bind_to_core_id 9
bind_to_core_id 10
bind_to_core_id 11
bind_to_core_id 12
bind_to_core_id 13
bind_to_core_id 14
bind_to_core_id 15
bind_to_core_id 25
bind_to_core_id 26
bind_to_core_id 27
bind_to_core_id 28
bind_to_core_id 29
bind_to_core_id 30
bind_to_core_id 31
bind_to_core_id 8
```

Глобальные настройки сервиса l2svc

```
cat /opt/l2svc/etc/global.cfg | grep BOND
```

```
BOND_MODE="balance-xor"
BOND_MIIMON="100"
BOND_XMIT_HASH_POLICY="layer2+3"
```