Российская сервисная платформа виртуализации РУСТЭК

PCI passthrough

Релизы 2.5 и 2.6

2023

Оглавление

1 Проброс PCI-устройств в гостевые системы

Прямой проброс PCI-устройств в РУСТЭК позволяет получить полный доступ и прямое управление физическим устройством в виртуальной машине. Этот механизм универсален для любого PCI-устройства и может работать с сетевой картой (NIC), графическим процессором (GPU) или любым другим устройством, которое можно подключить к PCI шине ВМ. Единственное требование для правильного использования устройств гостевой системой — это правильная установка драйвера.

Некоторые устройства PCI поддерживают технологию виртуализации устройств, позволяющую предоставить виртуальным машинам прямой доступ к части аппаратных возможностей устройства (SR-IOV). При использовании SR-IOV физическое устройство виртуализируется и отображается как несколько PCI-устройств. Виртуальные PCI-устройства можно назначить одной виртуальной машине или нескольким. В случае PCI passthrough физическое устройство целиком назначается только одной виртуальной машине и не может использоваться совместно двумя или более ВМ. Дополнительная информация приведена в **Руководстве администратора**.

PCI-устройства запрашиваются через дополнительные конфигурации, в частности через pci_passthrough:alias. В этом руководстве показано, как включить PCI passthrough для PCI-устройства (сетевого адаптера Intel X520) с vendor ID 8086 и product ID 154d путем их сопоставления с alias a1. Вам следует скорректировать инструкции для других устройств с потенциально другими возможностями.

Ограничения SR-IOV

- Невозможна живая миграция;
- Не поддерживаются группы безопасности;
- При использовании Quality of Service (QoS), опция `max_burst_kbps` не поддерживается, а `max_kbps` округляется до Mbps;
- При запуске переконфигурации РУСТЭК параметры в `/etc/nova/nova.conf` и `/etc/neutron/plugin.ini` придётся добавлять заново.

Ограничения Nova

Nova поддерживает PCI-адреса, поля которых ограничены следующим максимальным значением:

- domain: 0xFFFF
- bus: 0xFF
- slot: 0x1F
- function: 0x7

Nova будет игнорировать PCI-устройства, если адрес находится вне этих диапазонов.

2 Включение PCI passthrough

2.1 Настройка физического узла

Для включения PCI passthrough на физическом узле выполните следующее:

- Убедитесь, что в BIOS включен VT-d;
- Убедитесь, что IOMMU включен в ОС РУСТЭК, проверить это можно в файле `/proc/cmdline`: опция `intel_iommu` должна иметь значение `on;`
- Убедитесь, что PCI-устройства, которые будут пробрасываться, отображаются.
- Убедитесь, что у пользователя **nova** есть права на PCI-устройства, которые будут пробрасываться.

При отсутствии прав на устройство необходимо создать файл `/etc/udev/rules.d/vfio.rules` с содержимым:

```
SUBSYSTEM=="vfio", OWNER="root", GROUP="nova", MODE="0666"
```

и перезагрузить сервер.

2.2 Настройка nova-compute

После настройки PCI passthrough для узла, необходимо настроить nova-compute для проброса PCI-устройства в виртуальную машину. Это делается с помощью опции `pci.passthrough_whitelist`. Используемый в примере образец PCI-устройства имеет PCI-адрес 41:00.0 на каждом узле:

```
[pci]
passthrough_whitelist = { "address": "0000:41:00.0" }
```

В качестве альтернативы, можно указать следующее, чтобы включить проброс всех устройств с одинаковым product и vendor ID:

```
[pci]
passthrough_whitelist = { "vendor_id": "8086", "product_id": "154d" }
```

При использовании vendor ID и product ID все PCI-устройства, совпадающие с `vendor_id` и `product_id`, добавляются в пул PCI-устройств, доступных для проброса в ВМ.

Кроме того, необходимо указать `pci.alias`, который представляет собой параметр конфигурации в JSON. Он позволяет сопоставлять заданный тип устройства, идентифицированный стандартными полями PCI `vendor_id` и `product_id`, с произвольным именем или `alias`. Этот `alias` можно использовать для запроса PCI-устройства с помощью дополнительной конфигурации `pci_passthrough:alias`, как упоминалось ранее. В нашем примере PCI-устройство с vendor ID `0x8086` и product ID `0x154d` будет выглядеть следующим образом:

```
[pci]
alias = { "vendor_id":"8086", "product_id":"154d", "device_type":"type-PF",
"name":"a1" }
```

Важно отметить добавление поля `device_type`. Это необходимо, поскольку данное PCI-устройство поддерживает SR-IOV. Служба **nova-compute** классифицирует устройства по одному из трёх типов, в зависимости от возможностей, о которых сообщают устройства:

- **type-PF** — устройство поддерживает SR-IOV и является родительским или корневым устройством;
- **type-VF** — устройство является дочерним устройством устройства, поддерживающего SR-IOV;
- **type-PCI** — устройство не поддерживает SR-IOV.

По умолчанию можно подключать устройства `type-PCI` только с помощью PCI passthrough. Если вы хотите подключить устройства `type-PF` или `type-VF`, вы должны указать поле `device_type` в параметре конфигурации. Если устройство не поддерживает SR-IOV, поле `device_type` можно пропустить.

Обратите внимание
Этот параметр также должен быть настроен на узлах контроллера.

После настройки перезапустите службу **nova-compute**.

2.3 Настройка nova-scheduler

Служба **nova-scheduler** должна быть настроена для включения фильтра `PciPassthroughFilter`. Для этого добавьте этот фильтр в список фильтров, указанных в `filter_scheduler.enabled_filters`, и для `filter_scheduler.available_filters` установите `nova.scheduler.filters.all_filters` как значение по умолчанию. Например:

```
[filter_scheduler]
enabled_filters = ...,PciPassthroughFilter
available_filters = nova.scheduler.filters.all_filters
```

После настройки перезапустите службу **nova-sheduler**.

2.4 Настройка nova-api

Необходимо настроить параметр конфигурации `pci.alias` на контроллере. Эта конфигурация должна соответствовать конфигурации на физических узлах. Например:

```
[pci]
alias = { "vendor_id":"8086", "product_id":"154d", "device_type":"type-PF",
"name":"a1", "numa_policy":"preferred" }
```

Информацию о синтаксисе см. в `pci.alias`. Информацию о `numa_policy` см. в разделе Политики PCI-NUMA affinity.

После настройки перезапустите службу **nova-api**.

2.5 Настройка конфигурации или образа

После настройки `alias` его можно использовать для дополнительной конфигурации. Например, чтобы запросить два устройства PCI, на которые ссылается alias `a1`, выполните:

```
$ openstack flavor set m1.large --property "pci_passthrough:alias"="a1:2"
```

2.5.1 Параметр pci_passthrough

Данный параметр используется для настройки прямого проброса PCI-устройства физического узла в виртуальную машину. Для этого необходимо предварительно настроить физический узел.

> Поддерживается только драйвером libvirt.

`pci_passthrough:alias`

Type

`str`

Укажите количество `$alias` PCI-устройств для подключения к виртуальной машине. Должен использоваться формат `$alias:$number`. Используйте запятые для указания нескольких значений.

2.6 Политики PCI-NUMA affinity

По умолчанию драйвер libvirt устанавливает строгое NUMA affinity для PCI-устройств, будь то устройства PCI passthrough или интерфейсы neutron SR-IOV. Это означает, что по умолчанию PCI-устройство должно быть выделено из той же NUMA-ноды узла, что и хотя бы один из CPU виртуальной машины. Однако это не всегда необходимо, вы можете настроить эту политику с помощью дополнительных параметров конфигурации `hw:pci_numa_affinity_policy` или эквивалентного свойства метаданных образа. Допускается три возможных значения:

required

Эта политика означает, что nova будет загружать виртуальные машины с PCI-устройствами только в том случае, если хотя бы одна из NUMA-нод виртуальной машины связана с этими PCI-устройствами. Следовательно, если информация о NUMA-ноде для некоторых PCI-устройств не может быть определена, эти PCI-устройства не будут использоваться виртуальной машиной. Это обеспечивает максимальную производительность.

socket

Эта политика означает, что PCI-устройство должно быть привязано к тому же сокету вычислительного узла, что и хотя бы одна из гостевых NUMA-нод. Например, рассмотрим ВМ с двумя сокетами, каждый из которых имеет две NUMA-ноды, пронумерованные как нода 0 и нода 1 на сокете 0, нода 2 и нода 3 на сокете 1. К ноде 0 подключено PCI-устройство. Экземпляр PCI с двумя гостевыми NUMA-нодами и политикой сокетов может быть привязан к любому из:

- нода 0 или нода 1
- нода 0 или нода 2
- нода 0 или нода 3
- нода 1 или нода 2
- нода 1 или нода 3

Экземпляр не может быть привязан к ноде 2 и ноде 3, поскольку ни один из них не находится в одном сокете с PCI-устройством. Если другие ноды используются другими экземплярами и доступны только ноды 2 и 3, экземпляр не загрузится.

preferred

Эта политика означает, что nova-scheduler будет выбирать узел, минимально учитывая NUMA-affinity PCI-устройств. nova-compute попытается выбрать PCI-устройства исходя из NUMA-affinity, но если это невозможно — будет использовать NUMA-ноду, которая не ассоциирована с устройством.

legacy

Это политика по умолчанию, и она описывает текущее поведение nova. Обычно мы можем получить информацию о связи PCI-устройств с NUMA-нодами. Однако некоторые PCI-устройства не предоставляют такой информации. Значение legacy означает, что nova будет загружать виртуальные машины с PCI-устройством, если:

- Устройство PCI связано как минимум с одной NUMA-нодой, на которой будет загружаться виртуальная машина.
- Информация о привязке PCI-NUMA отсутствует.

Например, чтобы настроить конфигурацию на использование `preferred` политики PCI NUMA affinity для всех интерфейсов neutron SR-IOV, подключенных пользователем:

```
$ openstack flavor set $FLAVOR \
    --property hw:pci_numa_affinity_policy=preferred
```

3 Синтаксис pci.passthrough_whitelist

Type
multi-valued

Default
''

Белый список PCI-устройств, доступных для виртуальных машин.

Возможные значения:

- Словарь JSON, описывающий устройство PCI из белого списка. Он должен иметь следующий формат:

```
["vendor_id": "<id>",] ["product_id": "<id>",]
["address": "[[[[<domain>]:]<bus>]:][<slot>][.[<function>]]" |
 "devname": "<name>",]
{"<tag>": "<tag_value>",}
```

  - [— указывает на нулевое или одно вхождение;
  - { — указывает на нулевое или несколько вхождений;
  - | — взаимоисключающие варианты.

Допустимые значения ключей:

vendor_id

Идентификатор вендора устройства в шестнадцатеричном формате.

product_id

Идентификатор продукта устройства в шестнадцатеричном формате.

address

PCI-адрес устройства. Поддерживается как традиционный glob style, так и синтаксис регулярных выражений. Обратите внимание, что поля адреса ограничены следующими максимальными значениями:

- domain: 0xFFFF
- bus: 0xFF
- slot: 0x1F
- function: 0x7

devname

Имя устройства (например, имя интерфейса). Не все PCI-устройства имеют имя.

<tag>

Дополнительные <tag> и <tag_value> используются для сопоставления PCI-устройств. Поддерживаемые значения <tag>:

- physical_network
- trusted

Ниже приведены некоторые примеры:

```
passthrough_whitelist = {"devname":"eth0",
                         "physical_network":"physnet"}
passthrough_whitelist = {"address":"*:0a:00.*"}
passthrough_whitelist = {"address":":0a:00.",
                         "physical_network":"physnet1"}
passthrough_whitelist = {"vendor_id":"1137",
                         "product_id":"0071"}
passthrough_whitelist = {"vendor_id":"1137",
                         "product_id":"0071",
                         "address": "0000:0a:00.1",
                         "physical_network":"physnet1"}
passthrough_whitelist = {"address":{"domain": ".*",
                                    "bus": "02", "slot": "01",
                                    "function": "[2-7]"},
                         "physical_network":"physnet1"}
passthrough_whitelist = {"address":{"domain": ".*",
                                    "bus": "02", "slot": "0[1-2]",
                                    "function": ".*"},
                         "physical_network":"physnet1"}
passthrough_whitelist = {"devname": "eth0", "physical_network":"physnet1",
                         "trusted": "true"}
```

JSON-список, соответствующий приведённому выше формату. Например:

```
passthrough_whitelist = [{"product_id":"0001", "vendor_id":"8086"},
                         {"product_id":"0002", "vendor_id":"8086"}]
```

Следующий вариант недопустим, так как в нём указаны взаимоисключающие параметры:

```
passthrough_whitelist = {"devname":"eth0",
                         "physical_network":"physnet",
                         "address":"*:0a:00.*"}
```