Инструкция по настройке NAT и межсетевого экрана на «С-Терра Шлюз» при помощи утилиты «iptables»

Назначение документа

Данный документ описывает особенности использования утилиты «iptables» в Продукте «С-Терра Шлюз», а также кратко поясняет важные моменты подсистемы «netfilter» ядра Linux.

Краткое описание утилиты «iptables»

Утилита «iptables» – это инструмент для настройки фильтрации и трансляции сетевых адресов (NAT – network address translation) применительно к пакетам IPv4. Утилита «iptables» является пользовательским интерфейсом к подсистеме «netfilter» ядра Linux.

Краткое описание архитектуры подсистемы «netfilter» ядра Linux

Подсистема «netfilter» состоит из набора таблиц (filter, nat, mangle, raw, security) и цепочек (PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING, также могут использоваться пользовательские цепочки). Таблицы определяют действия над пакетами (фильтрация, трансляция адресов и др.), а цепочки – место совершения действия относительно пути прохождения пакетов и политику обработки пакетов. Каждая таблица состоит из набора цепочек, например, в таблице filter находятся три цепочки INPUT, FORWARD, OUTPUT.

Сетевой пакет последовательно проходит через ряд цепочек.

1. Входящий пакет, предназначенный для локального хоста: [драйвер ПО «С-Терра Шлюз»] -> [PREROUTING] -> [INPUT].
2. Входящий пакет, предназначенный для другого хоста (транзитный): [драйвер ПО «С-Терра Шлюз»] -> [PREROUTING] -> [FORWARD] -> [POSTROUTING] -> [драйвер ПО «С-Терра Шлюз»].
3. Локально сгенерированный пакет: [OUTPUT] -> [POSTROUTING] -> [драйвер ПО «С-Терра Шлюз»].

> Драйвер ПО «С-Терра Шлюз» обрабатывает пакеты только на тех интерфейсах, на которых включены функции контроля. Например, если к интерфейсу прикреплены списки доступа или крипто-карты.

В каждой из этих цепочек пакет проверяется на соответствие набору условий, определенных в правилах таблицы (например, пакет должен иметь определенные адрес отправителя и адрес получателя, либо же определённый набор портов и т.д.). Если пакет удовлетворяет критериям какого-либо из этих правил, то над ним выполняется соответствующее действие (как правило, это либо разрешающая политика – ACCEPT, либо запрещающая политика – DROP).

Каждая цепочка имеет предопределенную политику (политику по умолчанию), которая определяет действие, выполняемое над пакетом, если он не соответствует ни одному из правил в таблице. Эта политика служит стандартным поведением для пакетов, которые не соответствуют указанным критериям. На «С-Терра Шлюз» политика по умолчанию – ACCEPT, соответственно, подсистема «netfilter» разрешает прохождение любого трафика. Посмотреть политику во всех цепочках таблицы filter можно при помощи команды:

```
root@sterragate:~# iptables -L -n -t filter
```

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

В подсистему «netfilter» также встроен драйвер (обработчик пакетов) ПО «С-Терра Шлюз», а именно – входящие на интерфейс пакеты обрабатываются до цепочки PREROUTING, а исходящие – после POSTROUTING. Утилита «iptables» не может напрямую управлять правилами обработки трафика ПО «С-Терра Шлюз», но может повлиять на процесс обработки, поэтому нужно очень внимательно относиться к конфигурированию подсистемы «netfilter» при помощи утилиты «iptables».

Назначение таблиц и рекомендации к их использованию

1. Filter – применяется для фильтрации пакетов (используется по умолчанию, если в «iptables» не задан параметр «-t»). Рекомендуется использовать данную таблицу для разрешения/блокировки пакетов. Например, правило, разрешающее входящие SSH пакеты на хост, будет выглядеть следующим образом:

   ```
   root@sterragate:~# iptables –A INPUT -p tcp --dport 22 -j ACCEPT
   ```

2. Nat – применяется для выполнения трансляции адресов (source NAT и destination NAT). При выполнении source NAT нужно указывать цепочку POSTROUTING, а при выполнении destination NAT нужно указывать цепочку PREROUTING:

   ```
   root@sterragate:~# -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j SNAT --to-source
   172.16.1.2
   root@sterragate:~# iptables -t nat -A PREROUTING --dst 172.16.1.2 -p tcp --dport 2200
   -j DNAT --to-destination 192.168.1.100:22
   ```

3. Mangle – используется для специализированного изменения пакетов (например, действия TCPMSS, TTL, TOS, MARK). Используйте данную таблицу только в том случае, если Вы отчетливо понимаете то, что делаете. Например, правило, изменяющее MSS в TCP-пакетах, будет выглядеть следующим образом:

   ```
   root@sterragate:~# iptables -t mangle -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j
   TCPMSS --set-mss 1400
   ```

4. Raw – используется для фильтрации и изменения пакетов до их регистрации в подсистеме отслеживания соединений (connection tracking), также данная таблица используется, если необходимо совершать действия над еще несобранными пакетам (фрагментами). Используйте данную таблицу только в том случае, если Вы отчетливо понимаете то, что делаете. Например, правила, выключающие отслеживание всех соединений, будут выглядеть следующим образом:

   ```
   root@sterragate:~# iptables -t raw –I PREROUTING -j NOTRACK
   root@sterragate:~# iptables -t raw –I OUTPUT -j NOTRACK
   ```

5. Security – используется подсистемой «SELinux». Так как сетевую безопасность устройства обеспечивает ПО «С-Терра Шлюз», использование таблицы Security крайне не рекомендуется.

Основные положения по использованию утилиты «iptables»

1. Входящие ESP/NAT-T пакеты, направляющиеся для расшифрования в подсистему IPsec ПО «С-Терра Шлюз», не могут быть обработаны в подсистеме «netfilter» (совершать операции при помощи утилиты «iptables» можно только над расшифрованными пакетами).

2. Исходящие ESP/NAT-T пакеты, которые уже зашифрованы, могут быть перенаправлены снова в цепочки «netfilter» при помощи опции `re-route` в соответствующей крипто-карте. Такие пакеты пройдут следующий путь: [OUTPUT] -> [POSTROUTING] -> [драйвер ПО «С-Терра Шлюз»].

3. Подсистема «netfilter» не может разрешить прохождение трафика, который заблокирован в подсистеме Firewall ПО «С-Терра Шлюз», но может его изменить так, чтобы не сработали функции перехвата ПО «С-Терра Шлюз» (только для исходящего трафика).

4. Если в пользовательских правилах применяется маркирование пакетов (действие MARK), то на криптошлюзе запрещено изменять старшие 16 бит значения метки, так как они используются для внутренних нужд ПО «С-Терра Шлюз». Для работы с метками используйте маску.

   4.1. Выборка пакетов при помощи утилиты «iptables»:

   ```
   root@sterragate:~# iptables -t mangle -A PREROUTING -m mark --mark 0x00000001/0x0000FFFF -j ACCEPT
   ```

   В данном примере будут выбраны пакеты, у которых в младших 16 битах значение 1.

   Обратите внимание на то, что в тех местах, где в маске стоят нули, значения этих битов не учитываются.

   4.2. Установка меток на пакеты при помощи утилиты «iptables»:

   ```
   root@sterragate:~# iptables -t mangle -A POSTROUTING -p icmp -j MARK --set-mark 0x00000001/0x0000FFFF
   ```

   Обратите внимание на то, что в тех местах, где у маски нули, значения этих битов не изменяются - они переносятся в результирующую метку.

   В Приложении есть более детальный пример, объясняющий работу с --set-mark value[/mask].

5. Если используется политика DROP для цепочек INPUT/OUTPUT, то необходимо разрешить прием входящих/исходящих пакетов протокола IKE (порты 500 и 4500 протокола UDP) в подсистеме «netfilter» (пакеты протокола ESP нужно разрешать только в том случае, если используется опция `re-route`).

6. Если криптошлюз используется для предоставления доступа в Интернет и необходимо осуществлять source NAT, то нужно обеспечить следующие условия:

   6.1. Source NAT **должен быть отключен** для перечисленных ниже пакетов:

   - локальные IKE/ESP/NAT-T пакеты;
   - локальные VRRP пакеты (при настройке на нодах VRRP кластера);
   - все пакеты, которые предназначены для подсети 127.0.0.0/8.
   - все пакеты, которые должны шифроваться.

   6.2. При наличии транзитного IPsec-трафика через криптошлюз нужно либо изменить source порты пакетов IKE/NAT-T в настройках третьих устройств с 500 на любой из диапазона 40500-40540 и с 4500 на любой из диапазона 44500-44540 (так как порты 500 и 4500 используются по умолчанию на «С-Терра Шлюз»), либо сделать это при помощи утилиты «iptables».

Настройка NAT при помощи утилиты «iptables»

Настройка NAT осуществляется при помощи утилит «iptables» и «netfilter-persistent», входящих в состав Продукта «С-Терра Шлюз».

Пример настройки Source NAT

Перед настройкой NAT необходимо остановить целевой трафик и очистить таблицу соединений/трансляций при помощи команды `conntrack --delete`.

Предположим, криптошлюз защищает взаимодействие между подсетями 192.168.100.0/24 (локальная подсеть) и 192.168.1.0/24 (удаленная подсеть) и имеет публичный IP-адрес 172.16.100.100/24 на внешнем интерфейсе «eth0», а также, пропускает транзитный IPsec-трафик от устройств из локальный подсети.

В таком случае source NAT настраивается следующим образом:

Для source NAT без явной необходимости не используйте маскарадинг (`-j MASQUERADE`). Настоятельно рекомендуется явно указывать IP-адрес, в который осуществляется трансляция.

Примечание: соответствие между именем интерфейса в нотации cisco-like и linux можно посмотреть в файле `/etc/ifaliases.cf`.

1. Войдите в linux bash.

```
administrator@sterragate] system

Entering system shell...
```

2. Очистите настройки утилиты «iptables»:

```
root@sterragate:~# netfilter-persistent flush

run-parts: executing /usr/share/netfilter-persistent/plugins.d/10-ipset flush
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables flush
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables flush
run-parts: executing /usr/share/netfilter-persistent/plugins.d/35-ebtables flush
run-parts: executing /usr/share/netfilter-persistent/plugins.d/40-ipset flush
```

3. Добавьте две цепочки в таблицу NAT:

```
root@sterragate:~# iptables -N NO_SOURCE_NAT -t nat
root@sterragate:~# iptables -N SOURCE_NAT -t nat
```

В цепочку NO_SOURCE_NAT будет направляться трафик, для которого **не нужно** делать source NAT. В цепочку SOURCE_NAT будет направляться трафик, для которого **нужно** делать source NAT.

4. Добавьте правила в цепочку POSTROUTING для перенаправления пакетов из POSTROUTING в пользовательские цепочки NO_SOURCE_NAT и SOURCE_NAT:

```
root@sterragate:~# iptables -I POSTROUTING -t nat -j NO_SOURCE_NAT -m comment --comment
'pass traffic to chain to bypass source NAT'
root@sterragate:~# iptables -A POSTROUTING -t nat -j SOURCE_NAT -m comment --comment
'pass traffic to chain to do source NAT'
```

Логика работы следующая: все пакеты из цепочки POSTROUTING сначала направляются в цепочку NO_SOURCE_NAT, в которой определяется трафик, для которого **не нужно** делать source NAT. После того как трафик был определен, он прекращает свою обработку (действие ACCEPT) и, соответственно, не попадает в цепочку SOURCE_NAT. То есть в цепочку SOURCE_NAT попадает все, что не «отсеялось» в цепочке NO_SOURCE_NAT.

5. Добавьте правила в цепочку NO_SOURCE_NAT для трафика, который не подлежит source NAT:

5.1. Выключите source NAT для локальных IKE/ESP/NAT-T пакетов:

```
root@sterragate:~# iptables -t nat -A NO_SOURCE_NAT -m mark --mark
0x8000000/0x8000000 -j ACCEPT -m comment --comment 'local IKE packets'
root@sterragate:~# iptables -t nat -A NO_SOURCE_NAT -m mark --mark
0x40000000/0x40000000 -j ACCEPT -m comment --comment 'local ESP/NAT-T packets'
```

Метки 0x8000000 и 0x40000000 являются внутренними метками Продукта для локальных IKE и ESP/NAT-T пакетов соответственно.

5.2. Если настройки делаются на кластере, то выключите source NAT для пакетов протокола VRPP:

```
root@sterragate:~# iptables -t nat -A NO_SOURCE_NAT -p 112 -j ACCEPT -m comment --
comment 'local VRRP packets'
```

5.3. Выключите source NAT для трафика в подсеть 127.0.0.0/8:

```
root@sterragate:~# iptables -t nat -A NO_SOURCE_NAT -d 127.0.0.0/8 -j ACCEPT -m comment
--comment 'traffic to loopback iface'
```

Это правило необходимо для корректной работы локальных сетевых сервисов, например, Syslog.

5.4. Выключите source NAT для трафика, который должен шифроваться:

```
root@sterragate:~# iptables -t nat -A NO_SOURCE_NAT -s 192.168.100.0/24 -d
192.168.1.0/24 -j ACCEPT -m comment --comment 'traffic which must be protected by
IPsec'
```

6. Добавьте при помощи утилиты «iptables» правила в цепочку SOURCE_NAT для трафика, к которому должен быть применён source NAT:

   6.1. Включите source NAT для транзитного IPsec трафика, использующего порты 500/4500 протокола UDP (используется для подмены портов):

```
root@sterragate:~# iptables -t nat -A SOURCE_NAT -p udp --sport 500 -o eth0 -j SNAT -
-to-source 172.16.100.100:40500-40540 -m comment --comment 'transit IKE packets'
root@sterragate:~# iptables -t nat -A SOURCE_NAT -p udp --sport 4500 -o eth0 -j SNAT
--to-source 172.16.100.100:44500-44540 -m comment --comment 'transit ESP/NAT-T
packets'
```

   Нельзя, чтобы транзитный IPsec трафик использовал те же source UDP порты (500/4500), что и Продукт, так как в этом случае Продукт будет считать входящие транзитные IPsec пакеты за свои.

   6.2. Включите source NAT для трафика из подсети 192.168.100.0/24:

```
root@sterragate:~# iptables -t nat -A SOURCE_NAT -s 192.168.100.0/24 -o eth0 -j SNAT
--to-source 172.16.100.100
```

   Для Продукта версии 5.0.26871: если требуется включить source NAT для расшифрованных пакетов на LAN интерфейсе, то такое правило следует добавлять в таблицу nat в цепочку POSTROUTING **перед цепочкой** NO_SOURCE_NAT. Это связано с тем, что в Продукте расшифрованные пакеты имеют такую же метку, что и IKE пакеты, и соответственно попадут под действие ACCEPT в цепочке NO_SOURCE_NAT. При добавлении такого правила внимательно следите за тем, чтобы под него не попадали IKE пакеты.

7. Для просмотра правил в таблице NAT воспользуйтесь командой:

```
root@sterragate:~# iptables -L -n -v -t nat
```

Сохранение правил NAT при помощи утилиты «netfilter-persistent»

Для того, чтобы сохранить настройки утилиты «iptables», необходимо воспользоваться утилитой «netfilter-persistent»: сначала сохранить правила безопасности, а потом добавить сервис в автозапуск:

```
root@sterragate:~# netfilter-persistent save
```

```
run-parts: executing /usr/share/netfilter-persistent/plugins.d/10-ipset save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/35-ebtables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/40-ipset save
```

```
root@sterragate:~# systemctl enable netfilter-persistent.service
```

```
Synchronizing state of netfilter-persistent.service with SysV service script with
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable netfilter-persistent
```

Особенности настройки при наличии VRF

При настройке NAT на криптошлюзах при наличии VRF, где может присутствовать трафик из пересекающихся адресных пространств, требуется выполнить дополнительную настройку для обеспечения корректной работы conntrack и NAT. Без этой настройки подсистема conntrack может ошибочно принимать сетевые соединения в разных VRF с полностью совпадающими параметрами (IP адреса, номера портов и др.) за одно соединение.

Настройка заключается в назначении уникального идентификатора зоны conntrack для трафика каждого VRF инстанса. Эти действия выполняются в таблице `raw`, до момента обработки трафика подсистемой conntrack.

Допустим, на криптошлюзе настроено два VRF инстанса – `vrf_blue` и `vrf_red`. В `vrf_blue` входят интерфейсы `eth1` и `st_vti1`, в `vrf_red` – `eth2` и `st_vti2`. Трафик VRF инстанса `vrf_blue` будет ассоциирован с зоной под номером 10, `vrf_red` – под номером 20. В таком случае настройка будет выглядеть следующим образом:

1. Для `vrf_blue`:

   1.1. Для входящего и транзитного трафика:

   ```
   root@sterragate:~# iptables -t raw -A PREROUTING -i eth1 -j CT --zone 10
   root@sterragate:~# iptables -t raw -A PREROUTING -i st_vti1 -j CT --zone 10
   ```

   1.2. Для локально сгенерированного трафика:

   ```
   root@sterragate:~# iptables -t raw -A OUTPUT -o eth1 -j CT --zone 10
   root@sterragate:~# iptables -t raw -A OUTPUT -o st_vti1 -j CT --zone 10
   ```

2. Для `vrf_red`:

   2.1. Для входящего и транзитного трафика:

   ```
   root@sterragate:~# iptables -t raw -A PREROUTING -i eth2 -j CT --zone 20
   root@sterragate:~# iptables -t raw -A PREROUTING -i st_vti2 -j CT --zone 20
   ```

   2.2. Для локально сгенерированного трафика:

   ```
   root@sterragate:~# iptables -t raw -A OUTPUT -o eth2 -j CT --zone 20
   root@sterragate:~# iptables -t raw -A OUTPUT -o st_vti2 -j CT --zone 20
   ```

3. Сохраните настройки:

   ```
   root@sterragate:~# netfilter-persistent save
   ```

Принцип работы межсетевого экрана на основе зон безопасности

Stateful Zone Based Firewall – это межсетевой экран с возможностью контроля состояния сессий, использующий в качестве модели конфигурации не сетевые интерфейсы, а зоны. В зоны безопасности можно добавлять один или несколько сетевых интерфейсов. Другими словами, вместо использования списков доступа, относящихся к определенным интерфейсам, можно использовать правила, применяемые к зонам безопасности. При этом, данный межсетевой экран позволяет осуществлять контроль сетевого трафика при помощи сохранения информации о текущих состояниях сессий.

Дальнейшие настройки будут осуществляться согласно следующей схеме (за основу взят сценарий «Построение кластера на базе протокола VRRP с настройкой при помощи CLI»):

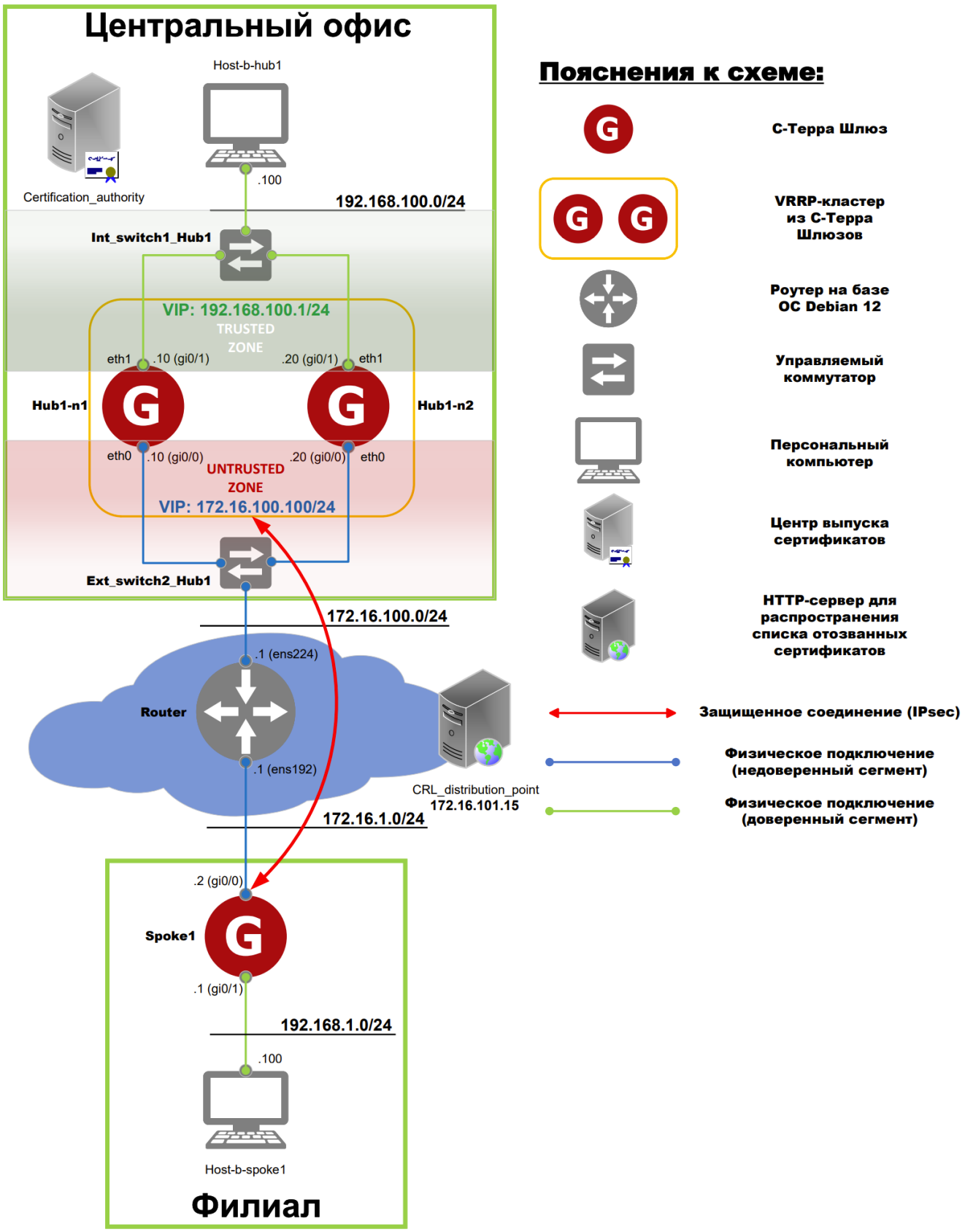


Рисунок 1. Схема разделения сети на зоны безопасности

- `trusted zone` – доверенная (локальная) зона безопасности
- `untrusted zone` – недоверенная (публичная) зона безопасности

Утилита «iptables» управляет правилами межсетевого экранирования подсистемы «netfilter» ядра Linux. По умолчанию в подсистеме «netfilter» разрешён весь входящий, исходящий и транзитный сетевой

трафик. Безопасность сети, представленной на схеме (см. Рисунок 1), должна обеспечиваться определёнными политиками. По умолчанию все входящие и транзитные пакеты должны быть отброшены. Далее, должны быть установлены необходимые разрешающие правила безопасности, обеспечивающие корректную работу межсетевого экрана.

При настройке межсетевого экрана с возможностью сохранения информации о состояниях сетевых сессий необходимо уметь оперировать понятиями ESTABLISHED, RELATED и NEW:

- ESTABLISHED – это установленное, уже существующее соединение;
- RELATED – это соединение, установленное с другим соединением, имеющим статус ESTABLISHED;
- NEW – это новое соединение. Под пометкой NEW понимается первый пакет, отправленный при инициировании нового соединения.

Все настройки должны производиться согласно следующей логике: всегда должны быть разрешены RELATED и ESTABLISHED соединения, но, если нужно добавить какое-то разрешающее или запрещающее правило, то при добавлении данного правила нужно указывать состояние NEW.

Настройку межсетевого экрана на основе зон безопасности можно разделить на следующие этапы:

1. Создание зон безопасности;
2. Добавление сетевых интерфейсов в зоны безопасности;
3. Создание политик безопасности и привязка их к зонам безопасности при помощи утилиты «iptables».

Настройки ниже будут производиться на примере криптошлюза Hub1-n1.

Создание зон безопасности

Создание зон безопасности осуществляется путём изменения содержимого файла `/etc/iproute2/group` (по умолчанию в данном файле находится только одна группа - `default`):

```
root@Hub1-n1:~# cat /etc/iproute2/group
```

```
# device group names
0       default
```

Группы – это и есть так называемые зоны безопасности. Для того, чтобы добавить зону безопасности, необходимо в отдельной строке указать индекс и название.

Нужно создать две зоны безопасности: `untrusted` и `trusted`. Для этого необходимо привести файл `/etc/iproute2/group` к следующему виду:

```
root@Hub1-n1:~# vim.tiny /etc/iproute2/group
```

```
# device group names
0       default
1       untrusted
2       trusted
```

Зоны безопасности успешно созданы.

Добавление сетевых интерфейсов в зоны безопасности

Для того, чтобы добавить сетевой интерфейс в созданную зону безопасности, необходимо воспользоваться утилитой «ip». Согласно схеме (см. Рисунок 1), сетевой интерфейс eth1 находится в зоне `trusted`, а сетевой интерфейс eth0 – в зоне `untrusted`:

```
root@Hub1-n1:~# ip link set eth1 group trusted
root@Hub1-n1:~# ip link set eth0 group untrusted
```

Для того, чтобы принадлежность интерфейса зоне безопасности сохранялась после перезагрузки, необходимо:

1. Создать файл с названием `zifs_zones` в директории `/etc/network/if-pre-up.d/` :

```
root@Hub1-n1:~# vim.tiny /etc/network/if-pre-up.d/zifs_zones
```

Скрипты из директории `/etc/network/if-pre-up.d/` выполняются в лексикографическом порядке во время загрузки криптошлюза, поэтому, в случае создания пользовательских скриптов, рекомендуется давать им такое название, которое обеспечивало бы их выполнение в последнюю очередь.

2. Скопировать содержимое ниже в этот файл (массивы INTERFACE_IN_TRUSTED и INTERFACE_IN_UNTRUSTED должны быть заданы согласно логике принадлежности интерфейсов зонам безопасности):

```
#!/bin/bash

# In the array INTERFACES_IN_<ZONE_NAME> you can specify multiple network interfaces.
# Example: INTERFACES_IN_TRUSTED=(eth1 eth2 eth3)

INTERFACES_IN_TRUSTED=(eth1)
INTERFACES_IN_UNTRUSTED=(eth0)

for i in ${INTERFACES_IN_TRUSTED[@]}
do
    if [[ "$IFACE" == $i ]]; then
        ip link set $i group trusted
    fi
done

for i in ${INTERFACES_IN_UNTRUSTED[@]}
do
    if [[ "$IFACE" == $i ]]; then
        ip link set $i group untrusted
    fi
done
```

3. Дать файлу `/etc/network/if-pre-up.d/zifs_zones` права на исполнение:

```
root@Hub1-n1:~# chmod u+x /etc/network/if-pre-up.d/zifs_zones
```

4. Перезапустить сетевые интерфейсы:

```
root@Hub1-n1:~# ifdown eth1; ifup eth1
root@Hub1-n1:~# ifdown eth0; ifup eth0
```

5. Убедиться в том, что сетевые интерфейсы были добавлены в нужные зоны безопасности:

```
root@Hub1-n1:~# ip link show eth1
```

```
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT
group trusted qlen 10000
    link/ether 1c:57:d8:14:26:ff brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    altname GigabitEthernet0_1
```

```
root@Hub1-n1:~# ip link show eth0
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT
group untrusted qlen 10000
    link/ether 1c:57:d8:14:26:fe brd ff:ff:ff:ff:ff:ff
    altname enp1s0
    altname GigabitEthernet0_0
```

Сетевые интерфейсы успешно добавлены в зоны безопасности.

Будьте внимательны! Сетевые интерфейсы, не добавленные ни в одну из зон безопасности, будут находиться в зоне default. Это значит, что к ним будут применяться политики по умолчанию всех цепочек утилиты «iptables».

Создание политик безопасности на межсетевом экране при помощи утилиты «iptables»

В данной главе будет производиться подробное описание настройки политик безопасности межсетевого экрана на базе Продукта «С-Терра Шлюз». Правила, указанные ниже, предполагают, что на межсетевом экране используется шифрование.

1. Установка правил безопасности по умолчанию:

```
root@Hub1-n1:~# iptables -P INPUT DROP
root@Hub1-n1:~# iptables -P FORWARD DROP
root@Hub1-n1:~# iptables -P OUTPUT ACCEPT
```

2. Создание новых цепочек.

   2.1. Цепочка для трафика, который попадает из доверенной зоны на межсетевой экран:

```
root@Hub1-n1:~# iptables -N from_TRUSTED_to_SELF
```

   2.2. Цепочка для трафика, который попадает из недоверенной зоны на межсетевой экран:

```
root@Hub1-n1:~# iptables -N from_UNTRUSTED_to_SELF
```

   2.3. Цепочка для трафика, который транзитом проходит через межсетевой экран из доверенной зоны в недоверенную:

```
root@Hub1-n1:~# iptables -N from_TRUSTED_to_UNTRUSTED
```

   2.4. Цепочка для трафика, который транзитом проходит через межсетевой экран из недоверенной зоны в доверенную:

```
root@Hub1-n1:~# iptables -N from_UNTRUSTED_to_TRUSTED
```

3. Установка разрешающих правил безопасности на входящий на межсетевой экран (SELF) сетевой трафик.

   3.1. Новые соединения на петлевой интерфейс (для обмена данными между сервисами, работающими на межсетевом экране):

```
root@Hub1-n1:~# iptables -A INPUT -m conntrack --ctstate NEW -i lo -j ACCEPT
```

   3.2. Перенаправление трафика, попавшего из доверенной зоны на межсетевой экран, в цепочку from_TRUSTED_to_SELF:

```
root@Hub1-n1:~# iptables -A INPUT -m devgroup --src-group trusted -j
from_TRUSTED_to_SELF
```

   3.2.1 Новые соединения из доверенной зоны с сервисами, функционирующими на межсетевом экране (правила в цепочку from_TRUSTED_to_SELF), а также установленные соединения:

- Установленные соединения:

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate
ESTABLISHED,RELATED -j ACCEPT
```

- DNS

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
udp --dport 53 -m comment --comment "DNS" -j ACCEPT
```

- NTP

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
udp --dport 123 -m comment --comment "NTP" -j ACCEPT
```

- DHCP (как сервер)

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
udp --dport 67 -m comment --comment "DHCP_server" -j ACCEPT
```

- DHCP (как клиент)

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
udp --dport 68 -m comment --comment "DHCP_client" -j ACCEPT
```

- SSH

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
tcp --dport 22 -m comment --comment "SSH" -j ACCEPT
```

- BGP

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
udp --dport 179 -m comment --comment "BGP" -j ACCEPT
```

- CONNTRACKD

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
udp --dport 3780 -m comment --comment "CONNTRACKD" -j ACCEPT
```

- VRRP

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
vrrp -m comment --comment "VRRP" -j ACCEPT
```

- ICMP

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p
icmp -m comment --comment "ICMP" -j ACCEPT
```

3.3. Перенаправление трафика, попавшего из недоверенной зоны на межсетевой экран, в цепочку from_UNTRUSTED_to_SELF:

```
root@Hub1-n1:~# iptables -A INPUT -m devgroup --src-group untrusted -j
from_UNTRUSTED_to_SELF
```

3.3.1 Новые соединения из недоверенной зоны с сервисами, функционирующими на межсетевом экране (правила в цепочку from_UNTRUSTED_to_SELF), а также установленные соединения:

- Установленные соединения:

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate
ESTABLISHED,RELATED -j ACCEPT
```

- ICMP

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -
p icmp -m comment --comment "ICMP" -j ACCEPT
```

- IKE

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -
p udp --dport 500 -m comment --comment "IKE" -j ACCEPT
```

- NAT-T

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -
p udp --dport 4500 -m comment --comment "IKE/NAT-T" -j ACCEPT
```

- VRRP

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -
p vrrp -m comment --comment "VRRP" -j ACCEPT
```

- Расшифрованный трафик

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -
m mark --mark 0xc0000000/0xc0000000 -m comment --comment "Decrypted traffic" -j
ACCEPT
```

Важно! В версии 5.0.26871 для расшифрованного трафика используется метка 0x8000000, которая совпадает с меткой для локальных IKE пакетов.

4. Установка разрешающих правил безопасности на транзитный сетевой трафик.

4.1. Перенаправление трафика, проходящего из доверенной зоны в недоверенную, в цепочку from_TRUSTED_to_UNTRUSTED:

```
root@Hub1-n1:~# iptables -A FORWARD -m devgroup --src-group trusted --dst-group untrusted -j from_TRUSTED_to_UNTRUSTED
```

4.1.1 Все соединения из доверенной зоны в недоверенную (правило в цепочку from_TRUSTED_to_UNTRUSTED):

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_UNTRUSTED -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT
```

4.1.2 Трафик, который подвергся DNAT из доверенной зоны в недоверенную (правило в цепочку from_TRUSTED_to_UNTRUSTED):

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_UNTRUSTED -m conntrack --ctstate DNAT -j ACCEPT
```

4.2. Перенаправление трафика, проходящего из недоверенной зоны в доверенную, в цепочку from_UNTRUSTED_to_TRUSTED:

```
root@Hub1-n1:~# iptables -A FORWARD -m devgroup --src-group untrusted --dst-group trusted -j from_UNTRUSTED_to_TRUSTED
```

4.2.1 Установленные соединения:

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_TRUSTED -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

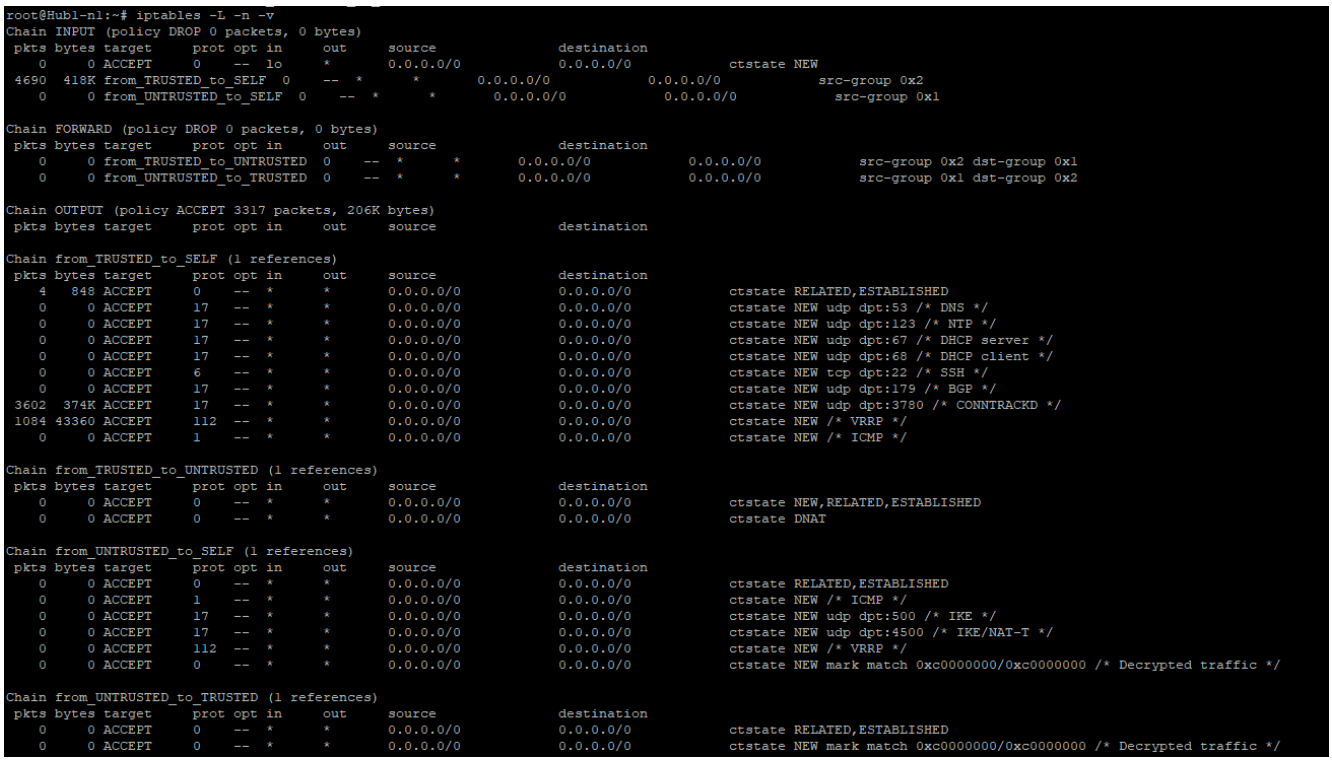
4.2.2 Новые соединения из недоверенной зоны в доверенную расшифрованного трафика (правило в цепочку from_UNTRUSTED_to_TRUSTED):

```
root@Hub1-n1:~# iptables -A from_UNTRUSTED_to_TRUSTED -m conntrack --ctstate NEW -m mark --mark 0xc0000000/0xc0000000 -m comment --comment "Decrypted traffic" -j ACCEPT
```

Важно! В версии 5.0.26871 для расшифрованного трафика используется метка 0x8000000, которая совпадает с меткой для локальных IKE пакетов.

Необходимо убедиться в том, что правила применились. Для этого нужно воспользоваться командой:

```
root@Hub1-n1:~# iptables -L -n -v
```



```
root@Hub1-n1:~# iptables -L -n -v
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  lo     *       0.0.0.0/0            0.0.0.0/0           ctstate NEW
 4690  418K from_TRUSTED_to_SELF  0    --  *      *       0.0.0.0/0            0.0.0.0/0           src-group 0x2
    0     0 from_UNTRUSTED_to_SELF  0    --  *      *       0.0.0.0/0            0.0.0.0/0           src-group 0x1

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 from_TRUSTED_to_UNTRUSTED  0    --  *      *       0.0.0.0/0            0.0.0.0/0           src-group 0x2 dst-group 0x1
    0     0 from_UNTRUSTED_to_TRUSTED  0    --  *      *       0.0.0.0/0            0.0.0.0/0           src-group 0x1 dst-group 0x2

Chain OUTPUT (policy ACCEPT 3317 packets, 206K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain from_TRUSTED_to_SELF (1 references)
 pkts bytes target     prot opt in     out     source               destination
    4   848 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:53 /* DNS */
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:123 /* NTP */
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:67 /* DHCP server */
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:68 /* DHCP client */
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW tcp dpt:22 /* SSH */
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:179 /* BGP */
 3602  374K ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:3780 /* CONNTRACKD */
 1084 43360 ACCEPT     112  --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW /* VRRP */
    0     0 ACCEPT     1    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW /* ICMP */

Chain from_TRUSTED_to_UNTRUSTED (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW,RELATED,ESTABLISHED
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate DNAT

Chain from_UNTRUSTED_to_SELF (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     1    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW /* ICMP */
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:500 /* IKE */
    0     0 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW udp dpt:4500 /* IKE/NAT-T */
    0     0 ACCEPT     112  --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW /* VRRP */
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW mark match 0xc0000000/0xc0000000 /* Decrypted traffic */

Chain from_UNTRUSTED_to_TRUSTED (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate NEW mark match 0xc0000000/0xc0000000 /* Decrypted traffic */
```

Рисунок 2. Вывод созданных политик безопасности при помощи утилиты «iptables»

Создание политик безопасности завершено.

Для того, чтобы сохранить настройки утилиты «iptables», необходимо воспользоваться утилитой «netfilter-persistent»: сначала сохранить правила безопасности, а потом добавить сервис в автозапуск:

```
root@Hub1-n1:~# netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/10-ipset save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/35-ebtables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/40-ipset save
root@Hub1-n1:~# systemctl enable netfilter-persistent.service
Synchronizing state of netfilter-persistent.service with SysV service script with
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable netfilter-persistent
```

Таким образом, была произведена настройка межсетевого экрана в режиме stateful с добавлением сетевых интерфейсов в зоны безопасности. Все правила, добавленные в ходе настройки политик безопасности, можно найти в главе «Приложение».

Настройка Source NAT на межсетевом экране при помощи утилиты «iptables»

Для того, чтобы корректно настроить Source NAT, необходимо:

1. Создать две новые цепочки в таблице NAT:

   1.1. Цепочка NO_SOURCE_NAT, в которую будет направляться трафик, для которого не нужно делать source NAT:

   ```
   root@Hub1-n1:~# iptables -N NO_SOURCE_NAT -t nat
   ```

   1.2. Цепочка SOURCE_NAT, в которую будет направляться трафик, для которого нужно делать source NAT:

   ```
   root@Hub1-n1:~# iptables -N SOURCE_NAT -t nat
   ```

2. Перенаправить весь трафик, который был отправлен в недоверенную зону и попал в цепочку POSTROUTING, в цепочку NO_SOURCE_NAT:

   ```
   root@Hub1-n1:~# iptables -I POSTROUTING -m devgroup --dst-group untrusted -t nat -j
   NO_SOURCE_NAT -m comment --comment 'pass traffic to chain to bypass source NAT'
   ```

3. Перенаправить трафик, который был отправлен в недоверенную зону, попал в цепочку POSTROUTING и не «отсеялся» в цепочке NO_SOURCE_NAT, в цепочку SOURCE_NAT:

   ```
   root@Hub1-n1:~# iptables -A POSTROUTING -m devgroup --dst-group untrusted -t nat -j
   SOURCE_NAT -m comment --comment 'pass traffic to chain to do source NAT'
   ```

4. Выключить source NAT для следующего трафика (правила в цепочку NO_SOURCE_NAT):

   4.1. Локальные IKE/ESP/NAT-T пакеты:

   ```
   root@Hub1-n1:~# iptables -t nat -A NO_SOURCE_NAT -m mark --mark 0x8000000/0x8000000
   -j ACCEPT -m comment --comment 'local IKE packets'
   root@Hub1-n1:~#  iptables  -t  nat  -A  NO_SOURCE_NAT  -m  mark  --mark
   0x40000000/0x40000000 -j ACCEPT -m comment --comment 'local ESP/NAT-T packets'
   ```

   4.2. Протокол VRRP (если настройки делаются на кластере):

   ```
   root@Hub1-n1:~# iptables -t nat -A NO_SOURCE_NAT -p 112 -j ACCEPT -m comment --
   comment 'local VRRP packets'
   ```

   4.3. В подсеть 127.0.0.0/8:

   ```
   root@Hub1-n1:~# iptables -t nat -A NO_SOURCE_NAT -d 127.0.0.0/8 -j ACCEPT -m comment
   --comment 'traffic to loopback iface'
   ```

Это правило необходимо для корректной работы локальных сетевых сервисов, например Syslog.

4.4. Который должен шифроваться:

```
root@Hub1-n1:~# iptables -t nat -A NO_SOURCE_NAT -s 192.168.100.0/24 -d 192.168.1.0/24
-j ACCEPT -m comment --comment 'traffic which must be protected by IPsec'
```

5. Включить source NAT для следующего трафика (правила в цепочку SOURCE_NAT):

5.1. Транзитный IPsec, использующий порты 500/4500 протокола UDP (используется для подмены портов):

```
root@Hub1-n1:~# iptables -t nat -A SOURCE_NAT -p udp --sport 500 -j SNAT --to-source
172.16.100.100:40500-40540 -m comment --comment 'transit IKE packets'
root@Hub1-n1:~# iptables -t nat -A SOURCE_NAT -p udp --sport 4500 -j SNAT --to-
source 172.16.100.100:44500-44540 -m comment --comment 'transit ESP/NAT-T packets'
```

Нельзя допускать того, чтобы транзитный IPsec трафик использовал те же source UDP порты (500/4500), что и Продукт, так как в этом случае Продукт будет считать входящие транзитные IPsec пакеты за свои.

5.2. Из локальной подсети:

```
root@Hub1-n1:~# iptables -t nat -A SOURCE_NAT -s 192.168.100.0/24 -j SNAT --to-
source 172.16.100.100
```

6. Посмотреть текущие NAT правила при помощи команды:

```
root@Hub1-n1:~# iptables -L -n -v -t nat
```

7. Сохранить добавленные правила при помощи команды:

```
root@Hub1-n1:~# netfilter-persistent save
```

```
run-parts: executing /usr/share/netfilter-persistent/plugins.d/10-ipset save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/35-ebtables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/40-ipset save
```

Настройка глубокого инспектирования пакетов при помощи модуля nDPI

В данной главе приведены примеры использования глубокого инспектирования пакетов при помощи модуля nDPI. DPI анализирует не только заголовки пакетов, но и полезную нагрузку, позволяя определить приложение или протокол. При этом, для идентификации приложения или протокола достаточно указать название.

Правила глубокого инспектирования настраиваются при помощи утилиты «iptables». Например, для того, чтобы заблокировать транзитный трафик youtube, telegram, instagram, ssh из доверенной зоны в недоверенную зону сети (например, нужно ограничить пользователям доступ к некоторым ресурсам в контексте уже настроенных выше stateful-правил), необходимо воспользоваться командой:

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_UNTRUSTED -m ndpi --clevel -port --proto
youtube,telegram,instagram,ssh -j DROP
```

Для того, чтобы глубокое инспектирование на межсетевом экране типа stateful работало корректно, правило, использующее модуль nDPI, должно быть самым первым в цепочке from_TRUSTED_to_UNTRUSTED.

Ключ `--proto` позволяет указать название приложения или протокола.

Если рассматривать использование модуля nDPI без зонирования, но в контексте межсетевого экрана типа stateful, то настройку глубокого инспектирования пакетов можно осуществить двумя способами:

1. Создать «белый» список доступа (запретить по умолчанию прохождение всех пакетов и при помощи модуля nDPI разрешить прохождение, например, следующих пакетов: google, microsoft и HTTP). Список правил будет выглядеть следующим образом:

```
root@Hub1-n1:~# iptables -A FORWARD -m conntrack --ctstate NEW,ESTABLISHED,RELATED -m
ndpi --inprogress google,microsoft,http -j ACCEPT
root@Hub1-n1:~# iptables -A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -m ndpi
--proto google,microsoft,http -j ACCEPT
root@Hub1-n1:~# <все остальные разрешающие правила, в которых не используется модуль
ndpi>
root@Hub1-n1:~# iptables -A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -j
ACCEPT
```

Ключ --inprogress позволяет корректно использовать процесс обнаружения необходимых пакетов, если модуль nDPI используется совместно со stateful-фильтрацией. Например, необходимо разрешить доступ к google, microsoft и подключения по протоколу HTTP. Если удалить правило с ключом --inprogress и оставить только лишь правило с ключом --proto (см. правила выше), то может произойти следующая ситуация:

1.1 Предположим, инициируется входящее или транзитное HTTP-соединение из недоверенной зоны. Прежде чем осуществлять соединение по протоколу HTTP, нужно построить TCP-сессию. Модуль nDPI определит, что осуществляется новое TCP-соединение и так как оно инициируется на порт 80, то предварительно будет обнаружен трафик HTTP (при передаче первых трёх пакетов SYN, SYN-ACK, ACK модуль nDPI пытается заранее определить, какой протокол будет использоваться далее). Подключение будет разрешено, так как протокол (HTTP) и состояние (NEW) соответствуют второму правилу.

1.2 Далее, после обмена пакетами SYN, SYN-ACK, ACK, будет установлена TCP-сессия (ESTABLISHED), в рамках которой уже можно будет передавать HTTP-трафик.

1.3 Затем HTTP-трафик будет передаваться в рамках установленной ESTABLISHED-сессии.

1.4 Однако, модуль nDPI в п.1.1 может ошибочно определить, что доступ осуществляется именно по протоколу HTTP (так как подключение может инициироваться, например, по протоколу SHH, но на 80 порт). Так как соединение уже будет находиться в состоянии ESTABLISHED, то фактически получится передавать пакеты протокола SSH на 80 порт под видом протокола HTTP, что представляет угрозу для безопасности системы (так как фактически пользователь в «белом» списке доступа разрешает только google, microsoft и HTTP, но никак не SSH).

При использовании протокола TCP ключ --inprogress позволяет начать процесс обнаружения необходимых сервисов или протоколов не в самом начале по первому пакету, а только после того, как TCP-сессия уже будет установлена (после обмена сообщениями SYN, SYN-ACK, ACK, то есть, фактически начиная с четвёртого пакета - с того, что будет передаваться уже в рамках созданной TCP-сессии).

2. Создать «чёрный» список доступа (разрешить по умолчанию прохождение всех пакетов и при помощи модуля nDPI запретить прохождение пакетов от необходимых пользователю сервисов и протоколов, например, от ssh):

```
root@Hub1-n1:~# iptables -A FORWARD -m ndpi --clevel -port --proto ssh -j DROP
root@Hub1-n1:~# iptables -A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -j
ACCEPT
root@Hub1-n1:~# <все остальные запрещающие правила, в которых не используется модуль
ndpi>
```

В данном случае используется ключ --clevel, который позволяет начать процесс обнаружения необходимых сервисов или протоколов с выбранного уровня обработки. Список уровней следующий: unknown, port, ip, nbpf, dpart, dcpart, dcache, dpi, user. В правиле, указанном выше, процесс обнаружения будет начинаться с уровня «ip», то есть с уровня, который находится после уровня «port». Использование --clevel -port для протокола ssh позволяет начать процесс обнаружения, не затрагивая при этом порт. То есть, если существует необходимость использовать 22 порт для каких-нибудь других протоколов, но, при этом, нужно заблокировать протокол ssh, то данное правило позволит это сделать.

Также модуль nDPI позволяет осуществлять работу с пользовательскими доменными именами. Предположим, необходимо заблокировать ресурс «defense.gov». Для этого нужно создать файл `/etc/modprobe.d/xt_ndpi.conf` и добавить в него следующую строку:

```
install xt_ndpi modprobe --ignore-install xt_ndpi; echo 'add_custom defense' > /proc/net/xt_ndpi/proto; echo 'defense:defense.gov' > /proc/net/xt_ndpi/host_proto
```

Будьте внимательны! Часть строки, содержащую команды по загрузке модуля xt_ndpi (а конкретно - `xt_ndpi modprobe --ignore-install xt_ndpi`) видоизменять запрещено!

- Команда `echo 'add_custom defense' > /proc/net/xt_ndpi/proto` позволяет записать в список протоколов новый (в данном примере - «defense»).
- Команда `echo 'defense:defense.gov' > /proc/net/xt_ndpi/host_proto` позволяет добавить в созданный или уже существующий протокол нужное DNS-имя (в данном примере – «defense.gov»).
- Также есть возможность добавить несколько DNS-имён в один протокол, например, `echo 'defense:defense.gov,foxnews.com,nytimes.com' > /proc/net/xt_ndpi/host_proto`, таким образом, включив в протокол «defense» несколько веб-ресурсов – «defense.gov», «foxnews.com», «nytimes.com».
- Для того, чтобы добавить ещё один протокол, необходимо через символ «;» добавить ещё одну команду, например, `echo 'add_custom ibm' > /proc/net/xt_ndpi/proto` добавит в список протоколов ещё один (помимо нового пользовательского протокола «defense», будет добавлен протокол «ibm»). Доменные имена добавляются согласно примеру выше, аналогичным образом.

Далее, необходимо добавить правило nDPI с новым протоколом, указав ключ --clevel:

```
root@Hub1-n1:~# iptables -A FORWARD -m conntrack --ctstate NEW,ESTABLISHED,RELATED -m ndpi --clevel -port --proto defense -j DROP
```

В этом случае доступ к ресурсу «defense.gov» для пользователей, находящихся за межсетевым экраном, будет заблокирован.

Для того, чтобы сохранить созданные правила, необходимо воспользоваться командой:

```
root@Hub1-n1:~# netfilter-persistent save
```

```
run-parts: executing /usr/share/netfilter-persistent/plugins.d/10-ipset save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/35-ebtables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/40-ipset save
```

Настройка подсистемы conntrack для увеличения лимита отслеживаемых сессий

Подсистема conntrack реализует в Linux отслеживание состояний сетевых соединений. Основой подсистемы является таблица, поддерживаемая модулем `nf_conntrack`, которая содержит записи о сетевых потоках. Эти данные используются для stateful-фильтрации, работы NAT и идентификации пакетов.

По умолчанию инспектирование средствами Linux отключено. Для его включения требуется явно добавить правило (для сохранения правила можно воспользоваться утилитой `netfilter-persistent`, работа которой описана в конце главы выше):

```
root@Hub1-n1:~# iptables -A INPUT -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT
```

Подсистема conntrack имеет ограничения на максимальное количество отслеживаемых соединений (по умолчанию – 524288):

```
root@Hub1-n1:~# sysctl net.netfilter.nf_conntrack_max
```

```
net.netfilter.nf_conntrack_max=524288
```

При превышении данного лимита новые соединения перестают регистрироваться, что может приводить к потерям пакетов и обрывам сессий, в `dmesg` появляются сообщения вида:

```
root@Hub1-n1:~# dmesg | grep conntrack
```

```
[  884.624923] nf_conntrack: nf_conntrack: table full, dropping packet
[  884.624933] nf_conntrack: nf_conntrack: table full, dropping packet
[  884.624947] nf_conntrack: nf_conntrack: table full, dropping packet
[  884.625109] nf_conntrack: nf_conntrack: table full, dropping packet
```

При работе с большим количеством одновременных сетевых соединений необходимо увеличивать как сам лимит на максимальное количество регистрируемых соединений, так и размеры хэш-таблицы и лимита ожидаемых соединений, чтобы избежать переполнения таблицы потоков и потерь пакетов соответственно. Предположим, необходимо инспектировать до 1 миллиона сетевых соединений. Для этого:

1. Создайте файл `/etc/sysctl.d/03-nf_conntrack.conf` и добавьте туда:

```
-net.netfilter.nf_conntrack_max=1048576
-net.netfilter.nf_conntrack_buckets=262144
-net.netfilter.nf_conntrack_expect_max=4096
```

Примечание:

- `nf_conntrack_max` – максимальное количество инспектируемых соединений. Увеличение значения данного параметра приводит к росту потребления оперативной памяти (например, инспектирование 1 миллиона сетевых соединений может потреблять до 700 Мегабайт оперативной памяти). На системах с ограниченным объёмом оперативной памяти увеличивать значение данного параметра не рекомендуется.
- `nf_conntrack_buckets` – количество «ячеек» в хеш-таблице для быстрого поиска по инспектируемым сетевым соединениям. Увеличивать значение данного параметра рекомендуется только при увеличении значения параметра `net.netfilter.nf_conntrack_max` (когда инспектировать необходимо более 1 миллиона сетевых соединений). При изменении данного параметра обязательно обращайте внимание на то, что значение должно быть степенью двойки. Относительно максимального количества инспектируемых соединений данный параметр рекомендуется выставлять в соотношении **nf_conntrack_max / 4**, чтобы на одну хеш-ячейку приходилось не более 4 записей о сессиях, так как это позволяет быстрее производить поиск записей по хэш-таблице (на умеренных нагрузках может быть достаточно соотношения **nf_conntrack_max / 8**).
- `nf_conntrack_expect_max` – лимит на ожидаемые соединения (например, для FTP). Увеличение значения данного параметра (с 1024 по умолчанию до 4096) необходимо для корректной работы протоколов, согласующих динамические порты (когда управление производится по одному порту, а передача – по другому порту). Данный параметр рекомендуется выставлять в соотношении соотношения **nf_conntrack_max / 256**.

2. Примените настройки:

```
root@Hub1-n1:~# sysctl -p /etc/sysctl.d/03-nf_conntrack.conf
```

```
net.netfilter.nf_conntrack_max = 1048576
net.netfilter.nf_conntrack_buckets = 262144
net.netfilter.nf_conntrack_expect_max = 4096
```

3. Перезагрузите криптошлюз:

```
root@Hub1-n1:~# reboot
```

4. Проверьте, что параметры применились:

```
root@Hub1-n1:~# sysctl net.netfilter.nf_conntrack_max
```

```
net.netfilter.nf_conntrack_max = 1048576
```

```
root@Hub1-n1:~# sysctl net.netfilter.nf_conntrack_buckets
```

```
net.netfilter.nf_conntrack_buckets = 262144
```

```
root@Hub1-n1:~# sysctl net.netfilter.nf_conntrack_expect_max
```

```
net.netfilter.nf_conntrack_expect_max = 4096
```

Настройка синхронизации сетевых сессий при помощи сервиса conntrackd

Будьте внимательны! В силу программных ограничений настройка синхронизации сетевых сессий может осуществляться только в том случае, когда модуль nDPI не загружен. Совместное использование модуля nDPI и сервиса conntrackd может привести к падению ядра операционной системы. Модуль nDPI не будет загружен в том случае, если соответствующие правила, настроенные при помощи утилиты «iptables», не содержат настройки `-m ndpi`.

При помощи сервиса conntrackd можно осуществить синхронизацию сетевых сессий между несколькими межсетевыми экранами. Это поможет избежать разрыва сетевых сессий при отказе основной ноды VRRP-кластера.

По умолчанию сервис conntrackd выключен и удалён из автозапуска. Перед запуском сервиса необходимо внести изменения в конфигурационный файл `/etc/conntrackd/conntrackd.conf` (содержимое по умолчанию конфигурационного файла `/etc/conntrackd/conntrackd.conf` можно найти в главе «Приложение»).

При использовании сервиса conntrackd необходимо различать такие понятия, как *internal cache* (внутренний кэш) и *external cache* (внешний кэш).

Internal cache – это кэш, содержащий информацию о состояниях локальных сетевых сессий.

External cache – это кэш, содержащий информацию о состояниях сетевых сессий, полученных от других межсетевых экранов, находящихся в VRRP-кластере.

При правильной настройке сервиса conntrackd, внутренний кэш ноды, находящейся в состоянии MASTER, должен совпадать с внешним кэшем ноды, находящейся в состоянии BACKUP.

Будьте внимательны! Перед настройкой сервиса conntrackd необходимо убедиться в том, что на обеих нодах VRRP-кластера сетевые интерфейсы, по которым будет осуществляться передача информации о состояниях сетевых сессий, добавлены в доверенную зону сети.

Будем считать, что на схеме (см. Рисунок 1) сетевые интерфейсы, по которым будет осуществляться передача информации о состояниях сетевых сессий между нодами Hub1-n1 и Hub1-n2, имеют адреса 192.168.100.10/24 и 192.168.100.20/24 соответственно.

Для того, чтобы настроить синхронизацию сетевых сессий на устройствах, которые находятся в одном VRRP-кластере, необходимо осуществить следующие настройки:

1. Привести файл `/etc/conntrackd/conntrackd.conf` к следующему виду:

```
root@Hub1-n1:~# vim.tiny /etc/conntrackd/conntrackd.conf
```

```
Sync {
    Mode FTFW {
        DisableExternalCache off
        CommitTimeout 5
        PurgeTimeout 5
    }

    UDP {
        # Dedicated link for connection replication
        IPv4_address 192.168.100.10
        IPv4_Destination_Address 192.168.100.20
        Port 3780
        Interface eth1
        SndSocketBuffer 1249280
        RcvSocketBuffer 1249280
```

```
        Checksum on
    }
}

General {
    Systemd on
    Nice -20
    HashSize 32768
    HashLimit 131072
    LogFile on
    Syslog on
    NetlinkOverrunResync 5
    PollSecs 5
    EventIterationLimit 200
    LockFile /var/lock/conntrack.lock
    UNIX {
        Path /var/run/conntrackd.ctl
        Backlog 20
    }
    NetlinkBufferSize 2097152
    NetlinkBufferSizeMaxGrowth 8388608
    Filter From Userspace {
        Protocol Accept {
            TCP
            UDP
            ICMP # This requires a Linux kernel >= 2.6.31
        }
        Address Ignore {
            IPv4_address 127.0.0.1 # loopback
        }
    }
}
```

Где:

- `IPv4_address <Address>` - адрес локального устройства;
- `IPv4_Destination_Address <Address>` - адрес устройства, с которым будет осуществляться синхронизация сетевых сессий;
- `Port <Number>` - номер порта;
- `Interface <Name>` - имя интерфейса, по которому будет осуществляться синхронизация.

Это основные параметры, значения которых следует менять при настройке конфигурационного файла `/etc/conntrackd/conntrackd.conf`. Остальные параметры можно оставить без изменений (настройки файлов `/etc/conntrackd/conntrackd.conf` на обеих нодах VRRP-кластера можно найти в главе «Приложение»).

2. Разрешить приём пакетов синхронизации по протоколу UDP на порт 3780 из доверенной зоны на межсетевой экран:

```
root@Hub1-n1:~# iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --
dport 3780 -m comment --comment "conntrackd.service" -j ACCEPT
```

По умолчанию устройства могут передавать информацию о состояниях сетевых сессий из внутреннего кэша во внешний, но не наоборот. Для того, чтобы информация из внешнего кэша переходила во внутренний, необходимо выполнить команду `conntrackd -c`. Однако, данная команда должна выполняться лишь в том случае, когда нода VRRP кластера переходит из состояния Backup в состояние Master. На нодах присутствует скрипт `/etc/keepalived/scripts/notify_common`, который автоматически выполняется при изменении состояния ноды (включить автоматическое выполнение скрипта `/etc/keepalived/scripts/notify_common` при изменении состояния ноды необходимо

через cisco-like консоль) – при помощи данного скрипта и будет выполняться команда `conntrackd -c`.

3. Если запуск скрипта `/etc/keepalived/scripts/notify_common` при переходах между состояниями кластера не включён, то необходимо его включить через cisco-like консоль:

   3.1. Зайти в cisco-like консоль криптошлюза:

   ```
   root@Hub1-n1:~# su cscons
   ```

   3.2. Перейти в привилегированный режим настройки:

   ```
   Hub1-n1# configure terminal
   ```

   3.3. Включить запуск скрипта`/etc/keepalived/scripts/notify_common`:

   ```
   Hub1-n1(config)# vrrp notify common
   ```

4. В файле `/etc/keepalived/notify_common.conf` значение параметра `FLAG_SYNC_NETFILTER_SESSIONS` изменить с «false» на «true»:

```
root@Hub1-n1:~# vim.tiny /etc/keepalived/notify_common.conf
```

```
FLAG_SYNC_NETFILTER_SESSIONS="true"
```

5. Добавить сервис conntrackd в автозапуск:

```
root@Hub1-n1:~# systemctl enable conntrackd.service
```

```
Synchronizing state of conntrackd.service with SysV service script with
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable conntrackd
```

6. Запустить сервис:

```
root@Hub1-n1:~# systemctl start conntrackd.service
```

7. Просмотреть информацию о синхронизации сетевых сессий:

```
root@Hub1-n1:~# conntrackd -s
```

```
cache internal:
current active connections:              22
connections created:                  1068499    failed:            0
connections updated:                  6092740    failed:            0
connections destroyed:                1068477    failed:            0

cache external:
current active connections:              17
connections created:                  1068020    failed:            0
connections updated:                  6071951    failed:            0
connections destroyed:                1068003    failed:            0

traffic processed:
                  0 Bytes                          0 Pckts

UDP traffic (active device=eth1):
           448801428 Bytes sent            447626432 Bytes recv
             1906738 Pckts sent              1928210 Pckts recv
                   0 Error send                    0 Error recv

message tracking:
                  0 Malformed msgs                 126 Lost msgs
```

Настройка синхронизации сетевых сессий при помощи сервиса `conntrackd` завершена.

Описание параметров конфигурационного файла /etc/conntrackd/conntrackd.conf

Содержимое конфигурационного файла `/etc/conntrackd/conntrackd.conf` разделяется на несколько секций:

- Sync – секция синхронизации, в ней указывается режим работы сервиса и протокол, по которому будет передаваться информация о состояниях сетевых сессий;
- General – основная секция, в ней указываются основные параметры, касающиеся функциональности сервиса (логирование, размеры буферов приёма и отправки, адреса и т.д.).

Режимы работы, которые могут быть указаны в секции Sync:

- **FTFW** – данный режим является самым надёжным, так как набор его параметров синхронизации позволяет восстанавливать во внешнем кэше таблицы conntrackd потерянную, повреждённую или не до конца подтвержденную информацию о состояниях сетевых сессий. При работе с сервисом conntrackd рекомендуется использовать именно этот режим. Используемые параметры:
  - **ResendQueueSize** (размер очереди повторной отправки в сессиях) – максимальное количество сессий, которое будет сохранено до подтверждения того, что информация о них получена устройством-соседом (значение по умолчанию 131072);
  - **CommitTimeout** (время подтверждения состояний в секундах) – если некоторые сессии не были восстановлены надлежащим образом, то после указанного тайм-аута во внешнем кэше таблицы conntrackd сохранятся только корректно восстановленные сессии, - остальные будут удалены (значение по умолчанию 180);
  - **PurgeTimeout** (время очистки таблицы состояний в секундах) – время очистки внешнего кэша таблицы conntrackd (вызов команды `conntrackd -t`), когда устройство переходит из состояния основной ноды в состояние резервной (значение по умолчанию 60);
  - **ACKWindowSize** (размер ACK-окна в байтах) – количество ACK-байтов, которое принимающая сторона готова принять в настоящий момент без подтверждения. Чем меньше данное значение, тем больше пакетов подтверждения будет отправлено (значение по умолчанию 300);
  - **DisableExternalCache** (отключение внешнего кэша) – отключение внешнего кэша таблицы conntrackd для того, чтобы записи не сохранялись в нём, а переносились сразу во внутреннюю таблицу conntrackd. Данный параметр рекомендуется использовать только с флагом off, так как запись сразу во внутреннюю таблицу состояний требует больших аппаратных затрат, что может значительно снизить производительность (значение по умолчанию off);
  - **StartupResync** (повторная синхронизация при запуске) – повторная синхронизация внешнего кэша таблицы conntrackd при запуске резервной ноды (значение по умолчанию off).
- **ALARM** – данный режим является так называемой «спам-рассылкой», так как его набор параметров синхронизации позволяет быстро осуществить синхронизацию устройств. Важно понимать, что, несмотря на скорость синхронизации, данный режим требует больших аппаратных затрат и отдельный канал для передачи данных. Используемые параметры:
  - **RefreshTime** (время обновления в секундах) – если таблица contrackd не была изменена в течение данного тайм-аута, то на broadcast-адрес передаётся соответствующее сообщение всем устройствам в данной подсети (значение по умолчанию 15);
  - **CacheTimeout** (тайм-айут внешнего кэша) – если не будет получено уведомление о состоянии сессии во внешнем кэше таблицы conntrackd через данный тайм-аут, то данная сессия будет удалена;
  - **CommitTimeout**;
  - **PurgeTimeout**.
- **NOTRACK** – самый простой режим, так как набор его параметров синхронизации позволяет получать информацию во внешний кэш таблицы conntrackd без каких-либо проверок:
  - **DisableInternalCache** – отключение внутреннего кэша;
  - **DisableExternalCache**;
  - **CommitTimeout**;
  - **PurgeTimeout**;

- **StartupResync**.

При использовании сервиса conntrackd рекомендуется выбирать режим FTFW, так как он позволяет наиболее корректно и гибко обрабатывать и передавать информацию о состояниях сетевых сессий.

Протоколы синхронизации, которые могут быть указаны в секции Sync:

• **Multicast** – информация о состояниях сетевых сессий будет передаваться при помощи multicast-пакетов. Используемые параметры:
- **IPv4_address** – multicast-адрес группы получателей;
- **Group** – номер multicast-группы получателей;
- **IPv4_interface** – адрес интерфейса, который будет использоваться в качестве отправителя;
- **Interface –** имя интерфейса, который будет использоваться в качестве отправителя;
- **SndSocketBuffer** – буфер постановки в очередь пакетов, которые будут переданы. Размер буфера по умолчанию доступен в файле `/proc/sys/net/core/wmem_default`. Чем больше размер данного буфера, тем меньше вероятность того, что пакеты с информацией о состояниях сетевых сессий будут потеряны.
- **RcvSocketBuffer** – буфер постановки в очередь пакетов, которые будут приняты (пакетов, ожидающих обработки сокетом). Размер буфера по умолчанию доступен в файле `/proc/sys/net/core/rmem_default`. Чем больше размер данного буфера, тем меньше вероятность того, что пакеты с информацией о состояниях сетевых сессий будут потеряны.
- **Checksum** – подсчёт контрольной суммы пакетов с информацией о состояниях сетевых сессий.

• **UDP** – информация о состояниях сетевых сессий будет передаваться при помощи UDP-пакетов. При работе с сервисом conntrackd рекомендуется использовать именно этот протокол синхронизации. Используемые параметры:
- **IPv4_address** – IPv4-адрес для получения сообщений о состояниях сетевых сессий;
- **IPv6_address** – IPv6-адрес для получения сообщений о состояниях сетевых сессий;
- **IPv4_Destination_Address** – IPv4-адрес получателя сообщений о состояниях сетевых сессий;
- **IPv6_Destination_Address** – IPv6-адрес получателя сообщений о состояниях сетевых сессий;
- **Port** – номер порта, который будет использовать протокол UDP;
- **Interface;**
- **SndSocketBuffer;**
- **RcvSocketBuffer;**
- **Checksum.**

• **TCP** – информация о состояниях сетевых сессий будет передаваться при помощи TCP-пакетов. Используемые параметры аналогичны тем, которые используются при выборе UDP.

Пример секции Sync:

```
Sync
{
    Mode FTFW
    {
        DisableExternalCache off
        CommitTimeout 5
        PurgeTimeout 5
    }

#    Mode ALARM
#    {
#        RefreshTime 5
#        CommitTimeout 5
#        PurgeTimeout 5
#    }

#     Mode NOTRACK
```

```
#     {
#         DisableInternalCache off
#         DisableExternalCache off
#     }

    UDP
    {
        IPv4_address 192.168.100.10
        IPv4_Destination_Address 192.168.100.20
        Port 3780
        Interface eth0
        SndSocketBuffer 1249280
        RcvSocketBuffer 1249280
        Checksum on

    }

#    Multicast
#    {
#       IPv4_address 225.0.0.51
#       Group 3782
#       IPv4_interface 192.168.100.102
#       Interface eth4
#       SndSocketBuffer 1249280
#       RcvSocketBuffer 1249280
#       Checksum on
#    }
}
```

Параметры, которые могут быть указаны в секции General:

- **Systemd <on|off>** – поддержка systemd;
- **Nice <value>** – в текущей версии conntrackd данный параметр присутствует, но игнорируется, в дальнейшем, он будет удалён;
- **HashSize <value>** – количество сегментов в хеш-таблице кэша;
- **HashLimit <value>** – максимальное количество подключений (оно должно быть вдвое больше, чем в файле `/proc/sys/net/netfilter/nf_conntrack_max`);
- **Logfile <on|off|filename>** – логирование (при включении этой опции логирование будет производиться в файл `/var/log/conntrackd.log`);
- **Syslog <on|off|facility>** – логирование через syslog;
- **Lockfile <filename>** – Lockfile, который будет использоваться сервисом conntrackd (при включении этой опции по умолчанию этим файлом будет `/var/lock/conntrack.lock`);
- NetlinkBufferSize <value> – размер буфера событий Netlink. Размер буфера по умолчанию доступен в файле `/proc/sys/net/core/rmem_default`. Чем больше размер данного буфера, тем меньше вероятность того, что сообщения о событиях будут потеряны;
- **NetlinkBufferSizeMaxGrowth <value>** – размер буфера, который может быть максимально достигнут. Размер буфера NetlinkBufferSize будет удваиваться при обнаружении того, что сообщения о событиях Netlink были потеряны;
- **NetlinkOverrunResync <on|off|value>** – повторная синхронизация с ядром через 30 секунд (по умолчанию), если обнаружен факт того, что Netlink отбрасывает события изменения состояния;
- **NetlinkEventsReliable <on|off>** – опция может работать при условии установленного пакета NetlinkOverrunResync;
- **PollSecs <seconds>** – сброс состояний в ядре каждые N-секунд;
- **EventIterationLimit <value>** – максимальное количество событий изменения состояний сетевых сессий (поступающих из ядра), которые будет обрабатывать демон, после чего он будет обрабатывать другие события, поступающие из сети или пользовательского окружения. Чем меньше значение, тем выше интерактивность;
- **UNIX** – секция, определяющая конфигурацию сокета Unix. Данный сокет используется сервисом conntrackd для прослушивания внешних команд. В данной секции указывается путь к сокету;

• **Filter From Userspace** – секция, при помощи которой осуществляется фильтрация событий. Доступно три набора фильтров: по протоколам, по адресам, по состояниям. Фильтр может быть привязан к действию: можно принять пакет, можно проигнорировать его. Таким образом, можно определить политику фильтрации событий наборов фильтров в зависимости от потребностей:

- **Protocol Accept** – секция, в которой указывается список протоколов, которые будут разрешены;
- **Address Ignore** – секция, в которой указывается список адресов, которые будут игнорироваться (если в пакете в качестве адреса отправителя будет указан какой-то из данных адресов, то сервис conntrackd будет игнорировать содержимое таких пакетов);
- **State Accept** – секция, в которой указывается список состояний, которые будут разрешены.

Пример секции General:

```
General
{
    Systemd on
    Nice -20
    HashSize 32768
    HashLimit 131072
    LogFile on
    Syslog on
    NetlinkOverrunResync 5
    PollSecs 5
    EventIterationLimit 200
    LockFile /var/lock/conntrack.lock
    UNIX
    {
        Path /var/run/conntrackd.ctl
        Backlog 20
    }
    NetlinkBufferSize 2097152
    NetlinkBufferSizeMaxGrowth 8388608
    Filter From Userspace
    {
        Protocol Accept
        {
            TCP
            UDP
            ICMP # This requires a Linux kernel >= 2.6.31
        }
        Address Ignore
        {
            IPv4_address 127.0.0.1 # loopback
            IPv4_address 10.0.0.1
            IPv4_address 10.0.0.2
            IPv4_address 10.0.0.3
            IPv4_address 192.168.255.2
            IPv4_address 192.168.255.52
            IPv4_address 192.168.255.250
        }
        State Accept
        {
         ESTABLISHED CLOSED TIME_WAIT CLOSE_WAIT for TCP
        }
}
```

Приложение

Полный список команд для создания политик безопасности при помощи утилиты «iptables»

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

iptables -N from_TRUSTED_to_SELF
iptables -N from_UNTRUSTED_to_SELF
iptables -N from_TRUSTED_to_UNTRUSTED
iptables -N from_UNTRUSTED_to_TRUSTED
iptables -N NO_SOURCE_NAT -t nat
iptables -N SOURCE_NAT -t nat

iptables -A INPUT -m conntrack --ctstate NEW -i lo -j ACCEPT
iptables -A INPUT -m devgroup --src-group trusted -j from_TRUSTED_to_SELF
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 53 -m comment
--comment "DNS" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 123 -m
comment --comment "NTP" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 67 -m comment
--comment "DHCP server" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 68 -m comment
--comment "DHCP client" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p tcp --dport 22 -m comment
--comment "SSH" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 179 -m
comment --comment "BGP" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 3780 -m
comment --comment "CONNTRACKD" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p vrrp -m comment --
comment "VRRP" -j ACCEPT
iptables -A from_TRUSTED_to_SELF -m conntrack --ctstate NEW -p icmp -m comment --
comment "ICMP" -j ACCEPT

iptables -A INPUT -m devgroup --src-group untrusted -j from_UNTRUSTED_to_SELF
iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -p icmp -m comment --
comment "ICMP" -j ACCEPT
iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 500 -m
comment --comment "IKE" -j ACCEPT
iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -p udp --dport 4500 -m
comment --comment "IKE/NAT-T" -j ACCEPT
iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -p vrrp -m comment --
comment "VRRP" -j ACCEPT
iptables -A from_UNTRUSTED_to_SELF -m conntrack --ctstate NEW -m mark --mark
0xc0000000/0xc0000000 -m comment --comment "Decrypted traffic" -j ACCEPT

iptables -A FORWARD -m devgroup --src-group trusted --dst-group untrusted -j
from_TRUSTED_to_UNTRUSTED
iptables -A from_TRUSTED_to_UNTRUSTED -m ndpi --clevel -port --proto
youtube,telegram,Instagram,ssh -j DROP
iptables -A from_TRUSTED_to_UNTRUSTED -m conntrack --ctstate NEW,ESTABLISHED,RELATED -
j ACCEPT
iptables -A from_TRUSTED_to_UNTRUSTED -m conntrack --ctstate DNAT -j ACCEPT
```

```
iptables -A FORWARD -m devgroup --src-group untrusted --dst-group trusted -j
from_UNTRUSTED_to_TRUSTED
iptables -A from_UNTRUSTED_to_TRUSTED -m conntrack --ctstate ESTABLISHED,RELATED -j
ACCEPT
iptables -A from_UNTRUSTED_to_TRUSTED -m conntrack --ctstate NEW -m mark --mark
0xc0000000/0xc0000000 -m comment --comment "Decrypted traffic" -j ACCEPT

iptables -I POSTROUTING -m devgroup --dst-group untrusted -t nat -j NO_SOURCE_NAT -m
comment --comment 'pass traffic to chain to bypass source NAT'
iptables -A POSTROUTING -m devgroup --dst-group untrusted -t nat -j SOURCE_NAT -m
comment --comment 'pass traffic to chain to do source NAT'
iptables -t nat -A NO_SOURCE_NAT -m mark --mark 0x8000000/0x8000000 -j ACCEPT -m comment
--comment 'local IKE packets'
iptables -t nat -A NO_SOURCE_NAT -m mark --mark 0x40000000/0x40000000 -j ACCEPT -m
comment --comment 'local ESP/NAT-T packets'
iptables -t nat -A NO_SOURCE_NAT -p 112 -j ACCEPT -m comment --comment 'local VRRP
packets'
iptables -t nat -A NO_SOURCE_NAT -d 127.0.0.0/8 -j ACCEPT -m comment --comment 'traffic
to loopback iface'
iptables -t nat -A NO_SOURCE_NAT -s 192.168.100.0/24 -d 192.168.1.0/24 -j ACCEPT -m
comment --comment 'traffic which must be protected by IPsec'
iptables -t nat -A SOURCE_NAT -p udp --sport 500 -j SNAT --to-source
172.16.100.100:40500-40540 -m comment --comment 'transit IKE packets'
iptables -t nat -A SOURCE_NAT -p udp --sport 4500 -j SNAT --to-source
172.16.100.100:44500-44540 -m comment --comment 'transit ESP/NAT-T packets'
iptables -t nat -A SOURCE_NAT -s 192.168.100.0/24 -j SNAT --to-source 172.16.100.100
```

Протоколы приложений, которые поддерживаются модулем nDPI

1kxun	accuweather	activision	afp
aimini	ajp	alibaba	alicloud
amazon	amazonalexa	amazonaws	amazonvideo
amongus	amqp	anydesk	apple
appleicloud	appleitunes	applejuice	applepush
applesiri	applestore	appletvplus	armagetron
avast	avastsecuredns	ayiya	azure
badoo	bgp	bittorrent	bjnp
bloomberg	cachefly	capwap	cassandra
checkmk	ciscoskinny	ciscovpn	citrix
cloudflare	cloudflarewarp	cnn	coap
collectd	corba	cpha	crashlytics
crossfire	crynetwork	csgo	cybersec
dailymotion	datasaver	dazn	deezer
dhcp	dhcpv6	diameter	direct_download_link
directconnect	directv	discord	disneyplus
dnp3	dns	dnscrypt	dofus
doh_dot	drda	dropbox	dtls
eaq	ebay	edgecast	edonkey
egp	elasticsearch	ethernetip	facebook
facebookvoip	fastcgi	fasttrack	fiesta
fix	florensia	forticlient	ftp_control
ftp_data	ftps	fuze	genshinimpact
git	github	gitlab	gmail
gnutella	google	googleclassroom	googlecloud
googledocs	googledrive	googlehangoutduo	googlemaps
googleplus	googleservices	goto	gre
gtp	gtp_c	gtp_prime	gtp_u
guildwars	h323	halflife2	hbo
hotspotshield	hp_virtgrp	hsrp	http
http_connect	http_proxy	hulu	i3d
iax	icecast	icloudprivaterelay	icmp
icmpv6	iec60870	iflix	igmp

```
  iheartradio          imap                 imaps                imo
  instagram            ip_in_ip             ip_pim               ipp
  ipsec                irc                  jabber               kakaotalk
  kakaotalk_voice      kerberos             kismet               kontiki
  lastfm               ldap                 likee                line
  linecall             linkedin             lisp                 livestream
  llmnr                lotusnotes           maplestory           mdns
  megaco               memcached            messenger            mgcp
  microsoft            microsoft365         mining               modbus
  mongodb              mpeg_ts              mpegdash             mqtt
  ms_onedrive          mssql-tds            munin                mysql
  nat-pmp              nats                 nestlogsink          netbios
  netflix              netflow              nfs                  nintendo
  noe                  ntop                 ntp                  ocs
  ocsp                 ookla                opendns              openft
  openvpn              oracle               ospf                 outlook
  pandora              pastebin             pgm                  pinterest
  playstation          playstore            pluralsight          pop3
  pops                 postgresql           ppstream             pptp
  psiphon              qq                   quic                 radius
  raknet               rdp                  reddit               redis
  riotgames            rpc                  rsh                  rsync
  rtcp                 rtmp                 rtp                  rtsp
  rx                   s7comm               salesforce           sap
  sctp                 sd-rtn               sflow                shoutcast
  showtime             signal               signalvoip           sina(weibo)
  sip                  siriusxmradio        skype_teams          skype_teamscall
  slack                smbv1                smbv23               smpp
  smtp                 smtps                snapchat             snapchatcall
  snmp                 soap                 socks                softether
  someip               sopcast              soundcloud           spotify
  ssdp                 ssh                  starcraft            stealthnet
  steam                stun                 syncthing            syslog
  targusdataspeed      teams                teamspeak            teamviewer
  telegram             telnet               tencent              tencentvideo
  teredo               tftp                 threema              thunder
  tidal                tiktok               tinc                 tivoconnect
  tls                  tocaboca             tor                  truphone
  tuenti               tumblr               tunein               tunnelbear
  tvuplayer            twitch               twitter              ubntac2
  ubuntuone            ultrasurf            usenet               vevo
  vhua                 viber                vimeo                vmware
  vnc                  vrrp                 vudu                 vxlan
  warcraft3            waze                 webex                websocket
  wechat               whatsapp             whatsappcall         whatsappfiles
  whois-das            wikipedia            windowsupdate        wireguard
  worldofkungfu        worldofwarcraft      wsd                  xbox
  xdmcp                xiaomi               yahoo                youtube
  youtubeupload        z3950                zabbix               zattoo
  zeromq               zoom
```

Настройки conntrackd

Конфигурационный файл /etc/conntrackd/conntrackd.conf (содержимое по умолчанию)

```
# Default debian config. Please, take a look at conntrackd.conf(5)

General {
        HashSize 8192
        HashLimit 65535
```

```
        Syslog on

        LockFile /var/lock/conntrackd.lock

        UNIX {
                Path /var/run/conntrackd.sock
                Backlog 20
        }

        SocketBufferSize 262142
        SocketBufferSizeMaxGrown 655355
        # default debian service unit file is of Type=notify
        Systemd on
}

Stats {
        LogFile on
}
```

Конфигурационный файл /etc/conntrackd/conntrackd.conf устройства Hub1-n1

```
Sync {
    Mode FTFW {
        DisableExternalCache off
        CommitTimeout 5
        PurgeTimeout 5
    }

    UDP {
        # Dedicated link for connection replication
        IPv4_address 192.168.100.10
        IPv4_Destination_Address 192.168.100.20
        Port 3780
        Interface eth1
        SndSocketBuffer 1249280
        RcvSocketBuffer 1249280
        Checksum on
    }
}

General {
    Systemd on
    Nice -20
    HashSize 32768
    HashLimit 131072
    #LogFile on
    #Syslog on
    NetlinkOverrunResync 5
    PollSecs 5
    EventIterationLimit 200
    LockFile /var/lock/conntrack.lock
    UNIX {
        Path /var/run/conntrackd.ctl
        Backlog 20
    }
    NetlinkBufferSize 2097152
    NetlinkBufferSizeMaxGrowth 8388608
    Filter From Userspace {
        Protocol Accept {
```

```
            TCP
            UDP
            ICMP # This requires a Linux kernel >= 2.6.31
        }
        Address Ignore {
            IPv4_address 127.0.0.1 # loopback
        }
    }
}
```

Конфигурационный файл /etc/conntrackd/conntrackd.conf устройства Hub1-n2

```
Sync {
    Mode FTFW {
        DisableExternalCache off
        CommitTimeout 5
        PurgeTimeout 5
    }

    UDP {
        # Dedicated link for connection replication
        IPv4_address 192.168.100.20
        IPv4_Destination_Address 192.168.100.10
        Port 3780
        Interface eth1
        SndSocketBuffer 1249280
        RcvSocketBuffer 1249280
        Checksum on
    }
}

General {
    Systemd on
    Nice -20
    HashSize 32768
    HashLimit 131072
    LogFile on
    Syslog on
    NetlinkOverrunResync 5
    PollSecs 5
    EventIterationLimit 200
    LockFile /var/lock/conntrack.lock
    UNIX {
        Path /var/run/conntrackd.ctl
        Backlog 20
    }
    NetlinkBufferSize 2097152
    NetlinkBufferSizeMaxGrowth 8388608
    Filter From Userspace {
        Protocol Accept {
            TCP
            UDP
            ICMP # This requires a Linux kernel >= 2.6.31
        }
        Address Ignore {
            IPv4_address 127.0.0.1 # loopback
        }
    }
}
```

Пояснение работы --set-mark value[/mask] в iptables

Параметр `--set-mark value[/mask]` используется для установки метки (mark) на пакет (чаще всего метки ставятся в цепочках PREROUTING – для входящих пакетов или OUTPUT – для исходящих локально сгенерированных пакетов). Сама по себе эта метка ничего не меняет в поведении пакета (не меняет IP, порт и т.д.). Её главная цель — пометить пакет для последующей обработки другими механизмами в операционной системе.

Сначала на значение метки в пакете накладывается маска, которая обнуляет биты, указанные в маске (mask), после чего, выполняется побитовая операция OR между получившимся значением и value.

Если маска (mask) не указана, то исполняется значение 0xFFFFFFFF.

Пример 1. Установка **значения** 0x00000001

```
--set-mark 0x00000001
```

Примечание: в примерах ниже старшие биты опущены.

```
1010 1010 (значение в пакете)
1111 1111 (маска по умолчанию)
0000 0000 (получилось после наложения маски)

0000 0001 (заданное значение value)
0000 0001 (получилось после выполнения операции OR)
```

Пример 2. Установка **бита** 0x00000001

```
--set-mark 0x00000001/0x00000001
```

Примечание: в примерах ниже старшие биты опущены.

```
1010 1010 (значение в пакете)
0000 0001 (заданная маска)
1010 1010 (получилось после наложения маски)

0000 0001 (заданное значение value)
1010 1011 (получилось после выполнения операции OR)
```