GeekBrains

Компьютерные сети

Транспортный уровень. Часть 2. NAT

Технология NAT

Введение

OSI/ISO	TCP/IP (DOD)
7. Прикладной уровень	4. Уровень приложений
6. Уровень представления	
5. Сеансовый уровень	
4. Транспортный уровень	3. Транспортный уровень
3. Сетевой уровень	2. Сетевой уровень
2. Канальный уровень	1. Уровень сетевых интерфейсов
1. Физический уровень	

Проблема нехватки IPv4-адресов и NAT

NAT

Под термином NAT (Network Address Translation — трансляция сетевых адресов) понимается ряд родственных технологий, относящихся к сетевому, транспортному и даже сеансовому уровню модели OSI.

Традиционно выделяют:

- Static NAT (статический NAT);
- Dynamic NAT (динамический NAT);
- Overload NAT (перегруженный NAT, он же PAT — Port Address Translation)

В каждом из видов NAT вы можете менять разные адреса в IP-пакете:

- Source NAT — трансляция IP адреса отправителя;
- Destination NAT — трансляция IP адреса получателя;
- BiNAT — одновременная трансляция адресов и отправителя и получателя.

Статический NAT

Предположим, что у нас имеется компьютерная сеть с белыми, маршрутизируемыми в Интернете адресами. Например, это может быть компьютерный класс. Любой компьютер, если он в сети, может

быть доступен извне, с любого компьютера, подключенного к сети Интернет (такая ситуация наблюдалась в начале 2000-х годов, когда университеты еще обладали сетями класса A и имели возможность щедро назначать их любым собственным ЭВМ). Но такой подход небезопасен. Один из вариантов решения — использовать вместо «белых» «серые» адреса (из диапазонов 10.0.0.0, 192.168.0.0–192.168.255.0 и т. д.). Но тогда возникает вопрос, как обеспечить этим машинам доступ в Интернет. (Обратную задачу решать не надо: доступ извне к обычным рабочим станциям, как правило, не нужен и даже противопоказан.) Самое простое решение — выделить одну из машин, которая будет подключена и к внутренней сети, с «серыми» адресами, и к внешней сети, имея «белый» IP-адрес, и настроить на этой машине прокси-сервер. Получая запросы от клиентов во внутренней сети, прокси-сервер будет передавать их во внешнюю сеть уже от своего имени, таким образом скрывая инфраструктуру локальной сети от внешних наблюдателей. Но это неудобно, так как прокси-сервер (для HTTP-сервера) — технология прикладного уровня: она позволяет получать доступ к WWW, но остальные услуги будут неработоспособны (ни ping, ни SMTP, кроме веб-мейл, и т. д.).

Технология статического NAT позволяет организовать сокрытие внутренних адресов и доступ к Интернет не на прикладном уровне (HTTP-proxy), а на сетевом. Статический NAT подразумевает, что имеется набор из равного количества «серых» и «белых» адресов. При прохождении пакета из внутренней сети во внешнюю сеть «серый» адрес меняется на соответствующий «белый», при прохождении пакета в обратном направлении, наоборот, соответствующий «белый» адрес будет меняться на «серый».

Предположим, имеется таблица:

«Серый» (частный) IPv4-адрес	«Белый» IPv4-адрес
192.168.32.1	213.18.123.101
192.168.32.2	213.18.123.102
……	
192.168.32.10	213.18.123.110
192.168.32.11	213.18.123.111
192.168.32.12	213.18.123.112

Компьютер с IPv4-адресом 192.168.32.11 будет виден из внешней сети как 213.18.123.111, когда бы он ни подключился к какому-либо ресурсу. Сам узел 192.168.32.11, в общем-то, ничего не знает об

адресе 213.18.123.111, так как шлюз для каждого пакета, адресованного на 213.18.123.111, также будет обратно заменять адрес на 192.168.32.11.

Динамический NAT

Если же количество машин больше, чем «белых» IP-адресов, и используются они не слишком часто, может использоваться динамический NAT. В этом случае имеется диапазон «белых» IP-адресов, которые теперь не привязаны к «серым» IP-адресам. «Белые» IP-адреса выдаются по мере надобности, и одна машина в разное время может иметь разные IP-адреса.

NAT-шлюз должен иметь помимо таблицы IPv4 адресов, которые могут быть выделены, также таблицу текущих сессий, связывая «серые» и выделенные им в соответствие в данный момент «белые» адреса. После того, как машина перестает пользоваться сетевыми услугами, «белый» IPv4-адрес освобождается и теперь может быть выделен, например, другой машине.

Итак, у нас есть доступный диапазон IPv4-адресов: 213.18.123.110 — 213.18.123.115. Предположим, что три компьютера решили подключиться к каким-либо внешним ресурсам (обратите внимание на порядок подключения).

Динамическая таблица соответствий будет иметь вид:

«Серый» (частный) IPv4-адрес	«Белый» IPv4-адрес
192.168.32.10	213.18.123.110
192.168.32.12	213.18.123.111
192.168.32.11	213.18.123.112

Предположим, что узел 192.168.32.11 завершил сессию.

Теперь таблица выглядит вот так.

«Серый» (частный) IPv4-адрес	«Белый» IPv4-адрес
192.168.32.10	213.18.123.110
192.168.32.12	213.18.123.111

Предположим, что узел 192.168.32.10 тоже завершил сессию, но к сети подключается узел 192.168.32.15.

«Серый» (частный) IPv4-адрес	«Белый» IPv4-адрес
192.168.32.10	213.18.123.110
192.168.32.12	213.18.123.111
192.168.32.15	213.18.123.112

Если узел 192.168.32.11 снова обратится к сети, он уже получит другой адрес, так как предыдущий адрес, который он использовал, занят. Предположим, что 192.168.32.10 завершил сессию. Вполне возможно, что теперь 192.168.32.11 получит адрес 213.18.123.110.

«Серый» (частный) IPv4-адрес	«Белый» IPv4-адрес
192.168.32.12	213.18.123.111
192.168.32.15	213.18.123.112
192.168.32.11	213.18.123.110

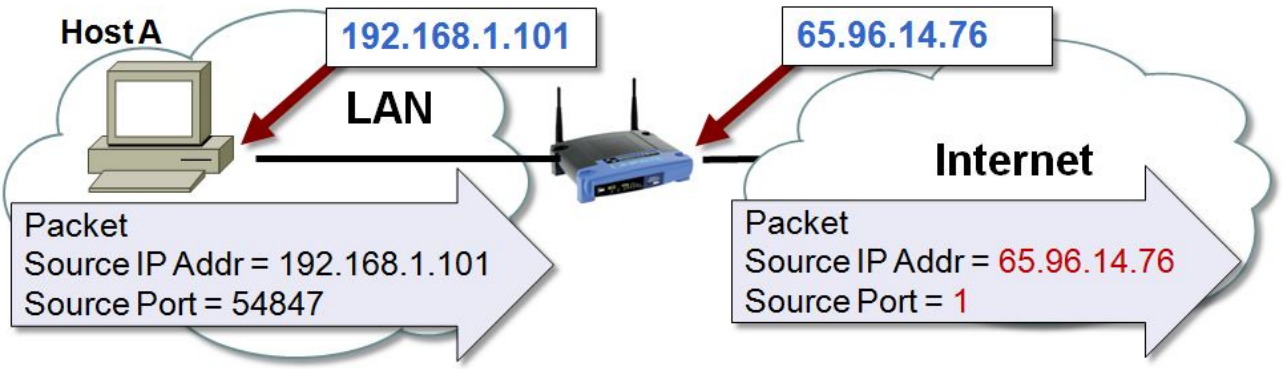
Проблема нехватки IPv4-адресов

Пространство IPv4-адресов оказалось слишком маленьким. В настоящее время оно почти полностью исчерпано. Из-за нехватки адресов была разработана бесклассовая адресация, блоки адресов стали выделяться меньшими порциями, провайдеры перестали выделять белые IP-адреса (даже динамические). То, что адресов надолго не хватит, стало понятно еще в 1992 году, когда была образована рабочая группа IPng, которая к 1996 году подготовила первую спецификацию IPv6, обладавшую гораздо большим пространством адресов. При этом сама технология NAT появилась в 1994 году. Технология статического NAT позволяет обеспечить безопасность, но не сэкономить

пространство адресов. Динамический NAT позволяет частично сэкономить пространство адресов, но все равно требует определенного диапазона белых адресов. Дальнейшее развитие идеи: перегруженный NAT, позволяющий обеспечить одновременный доступ в Интернет для нескольких машин с «серыми» адресами, если в наличии есть всего лишь один «белый» IP-адрес. Сейчас идет внедрение IPv6-адресов, и можно сказать, что для серверной инфраструктуры оно состоялось. (Купите VDS у какого-нибудь провайдера, удостоверьтесь, что IPv6-адрес подключен, и с помощью **tcpdump** отследите любой трафик, например, при попытке с помощью **apt-get install** установить какое-нибудь новое ПО. Вы будете наблюдать IPv6-трафик.) Но провайдеры не спешат переходить на NAT, скрывая всё больше и больше «серых» адресов за NAT-шлюзом. Для этих целей даже был выделен еще один блок «серых» адресов — 100.64.0.0/10. Провайдеры все чаще используют IP-адреса из этого диапазона вместо выделения динамических, но «белых» IPv4-адресов. Стоит отметить, что NAT — тупиковая, «костыльная» технология. Ряд протоколов не может нормально работать с NAT, и для них приходится изобретать «заплатки» (такие, как «пассивный режим» FTP).

Перегруженный NAT

Перегруженный NAT (NAT Overload) имеет много синонимов: PAT (Port Address Translation), NAPT (Network Address and Port Translation), Masquerading (маскарадинг). Если статический NAT может осуществляться устройством «на лету» и заменяются только L3-заголовки, а динамический NAT также осуществляет замену IP-адресов в L3-заголовках, но устройство уже вынуждено отслеживать активные сессии, то для реализации перегруженного NAT устройство должно анализировать и менять не только L3-, но и L4-заголовки. Теперь вместо соответствия «„серый” IPv4-адрес — „белый” IPv4-адрес» используется соответствие «„серый” IPv4-адрес: исходный порт — новый порт на шлюзе с „белым” IPv4-адресом». Таким образом, шлюз должен отслеживать сессии и очищать таблицы соответствий при завершении клиентских сессий. В частности, в Linux механизм NAT встроен в сетевой фильтр **netfilter** (и, соответственно, управляется правилами **iptables**), то есть фактически это сеансовый уровень модели OSI/ISO — при том, что фильтр позволяет анализировать заголовки и канального, и сетевого, и транспортного уровней, и отслеживать прикладные протоколы. Суть в том, что для ушедшего пакета/дейтаграммы необходимо запомнить подстановку и при обратном ответе суметь сделать подстановку обратно. Так сетевые фильтры отслеживают сессии — такая технология называется stateful filtering. Есть также stateless filtering — без отслеживания состояний, но NAT — это как раз stateful filtering.



NAT Translation Table

	Local IP Address	Source Port #		Internet IP Address	Source Port #
process X, Host A	192.168.1.101	54,847	=	65.96.14.76	1
Host B	192.168.1.103	24,123	=	65.96.14.76	2
process Y, Host A	192.168.1.101	42,156	=	65.96.14.76	3
Host C	192.168.1.102	33,543	=	65.96.14.76	4

Предположим, компьютер с «серым» адресом 192.168.1.101 обращается с TCP-порта 54 847 к некоторой машине 65.96.14.76 (на какой порт — неважно, пусть на 80). Шлюз, получив сообщение с 192.168.1.101:54847, добавляет эти данные в таблицу, дополнив еще одним полем — сведениями о выделенном для этого соединения порте, например 54. После этого в заголовках сетевого и транспортного уровня будет произведена замена: 192.168.1.101:54847 на 65.96.14.76:54 (65.96.14.76 — «белый» адрес шлюза).

Сервер, в свою очередь, будет отвечать на 65.96.14.76:54, и шлюз произведет обратную трансляцию, таким образом, клиент получит ответ на 192.168.1.101:54847 и в общем случае не заметит подмены.

Рассмотрим интересную ситуацию. Если следующий клиент (например, машина 192.168.1.104) при этом по стечению обстоятельств попытается отправить сообщение с такого же порта (54 847/TCP), у него, как вы уже догадываетесь, получится. Только в таблице появится еще одна запись: 192.168.1.104:54847, 55. Порт на шлюзе будет выделен другой, следующий по порядку.

Таким образом, в данной модели уже не IP-адрес идентифицирует хост, а пара «IP-адрес — порт», хотя следует упомянуть, что порт на шлюзе уже идентифицирует не отдельное приложение на данном хосте, а отдельное приложение и хост в локальной сети, на котором оно запущено.

Destination NAT

Как вы уже поняли, при использовании NAT любая машина в локальной сети может получить доступ к услугам в Интернете, но внешние узлы не смогут инициализировать соединения во внутреннюю сеть.

Предположим обратную ситуацию: у нас имеется «белый» IPv4-адрес на шлюзе (например, домашнем роутере), но нам необходимо дать доступ извне на один или два компьютера в локальной сети. В этом помогает Destination NAT. Заходим в настройки роутера (в примере — настройки для домашнего Wi-Fi-роутера TP-LINK TL-MR3420) и прописываем соответствия внешних портов на роутере и внутренних IP-адресов и портов на наших серверах. Трансляция напоминает вышеописанный алгоритм, но таблица статична и работает в обратную сторону.

Виртуальные серверы

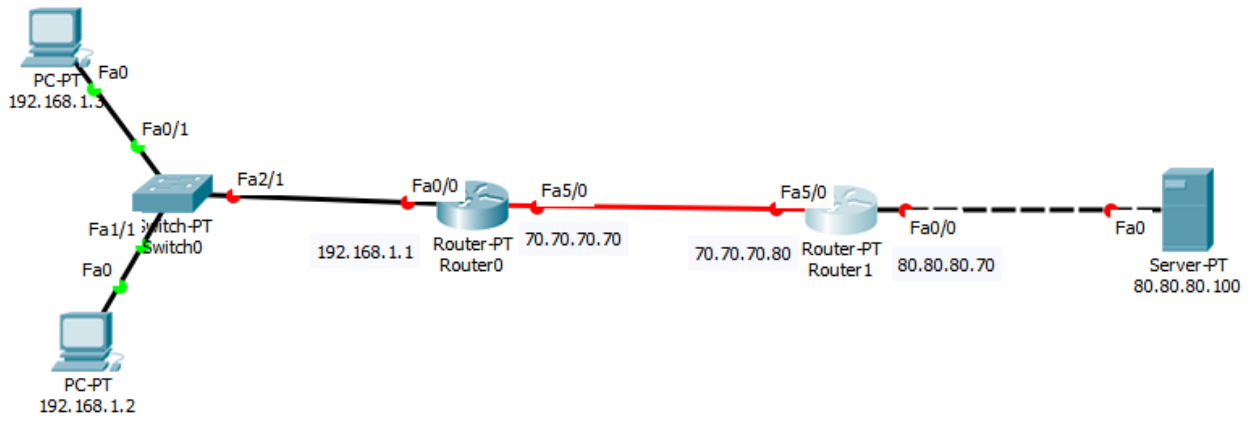
ID	Порт сервиса	Внутренний порт	IP-Адрес	Протокол	Состояние	Изменить
1	8080	80	192.168.0.200	Все	Включено	Редактировать Удалить
2	8088	80	192.168.0.201	Все	Включено	Редактировать Удалить
3	8000	80	192.168.0.202	Все	Включено	Редактировать Удалить

Добавить новую... | Включить все | Отключить все | Удалить все

Практика

Настройка NAT Overload в Cisco Packet Tracer

Соберем простую схему:



Предположим, нам необходимо настроить сеть таким образом, чтобы сеть 192.168.1.0/24 была скрыта за NAT и все сообщения в остальные сети шли от IPv4-адреса шлюза 70.70.70.70.

Сначала настроим Router-1.

Настроим сетевые интерфейсы и включим их:

```
Router>
Router>ena
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int fa0/0
Router(config-if)#ip addr 80.80.80.70 255.255.255.0
Router(config-if)#no shut
Router(config-if)#int fa5/0
Router(config-if)#ip addr 70.70.70.80 255.255.255.0
Router(config-if)#no shut
```

Настроим маршрутизацию по протоколу RIP2. Мы анонсируем обе сети, так как нужно предоставить доступ к сети 80.80.80.0/24, в том числе к серверу 80.80.80.100, а сервер должен иметь возможность отправлять сообщения в сеть 70.70.70.0/24:

```
Router(config-if)#route rip
Router(config-router)#version 2
Router(config-router)#network 80.80.80.0
Router(config-router)#network 70.70.70.0
Router(config-router)#end
```

Можем проверить таблицу маршрутизации:

```
Router#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     70.0.0.0/24 is subnetted, 1 subnets
C       70.70.70.0 is directly connected, FastEthernet5/0
     80.0.0.0/24 is subnetted, 1 subnets
C       80.80.80.0 is directly connected, FastEthernet0/0
```

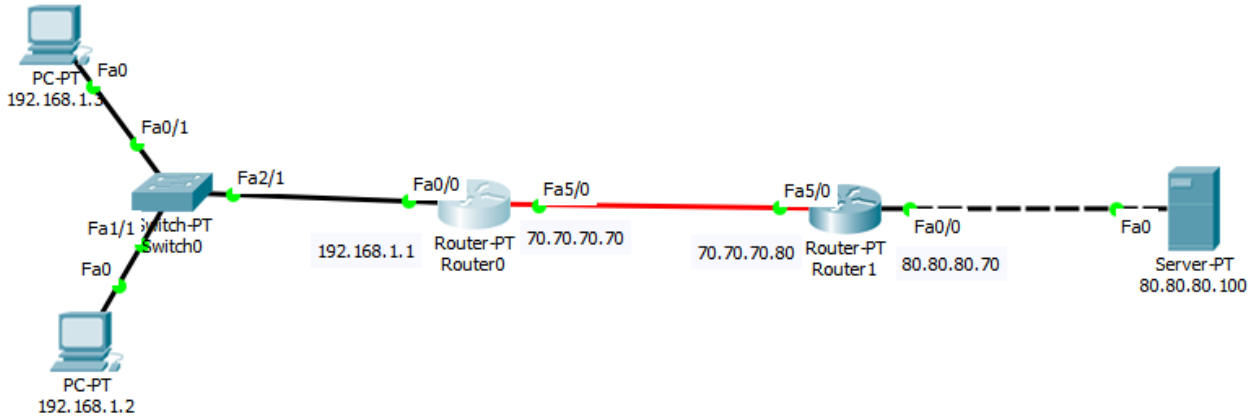
Мы видим только непосредственно доступные сети. Протокол RIP2 настроен, но нужно настроить и другой маршрутизатор.

Теперь настроим Router0.

Настроим сетевые интерфейсы и включим:

```
Router>ena
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
Router(config)#int fa0/0
Router(config-if)#ip addr 192.168.1.1 255.255.255.0
Router(config-if)#no shut

%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state
to up

Router(config-if)#int fa5/0
Router(config-if)#ip addr 70.70.70.70 255.255.255.0
Router(config-if)#no shut

%LINK-5-CHANGED: Interface FastEthernet5/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet5/0, changed state
to up
```



Настроим динамическую маршрутизацию:

```
Router(config-if)#route rip
Router(config-router)#version 2
Router(config-router)#network 70.70.70.0
Router(config-router)#exit
Router(config)#exit
```

Обратите внимание: мы не будем анонсировать сеть 192.168.1.0 (она скрыта будет за NAT), но нам необходимо анонсировать сеть 70.70.70.0, иначе мы не получим данные о сетях от маршрутизатора Router1.

Проверяем:

```
Router#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
```

```
       P - periodic downloaded static route

Gateway of last resort is not set

     70.0.0.0/24 is subnetted, 1 subnets
C       70.70.70.0 is directly connected, FastEthernet5/0
R    80.0.0.0/8 [120/1] via 70.70.70.80, 00:00:12, FastEthernet5/0
C    192.168.1.0/24 is directly connected, FastEthernet0/0
```

Сети 70.70.70.0 и 192.168.1.0 доступны напрямую через сетевые интерфейсы, а вот маршрут в сеть 80.0.0.0/8 доступен через шлюз 70.70.70.80 — эта информация получена нами по протоколу RIP2 от второго маршрутизатора.

Теперь настало время настроить NAT. Сначала создаем список доступа, указывая, какие адреса могут использовать NAT:

```
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#access-list 1 permit 192.168.1.0 0.0.0.255
```

Укажем, что пакеты клиентов с IP-адресов из списка 1 будут подвергаться перегруженной NAT-трансляции при следовании через интерфейс fa5/0:

```
Router(config)#ip nat inside source list 1 int fa5/0 overload
```

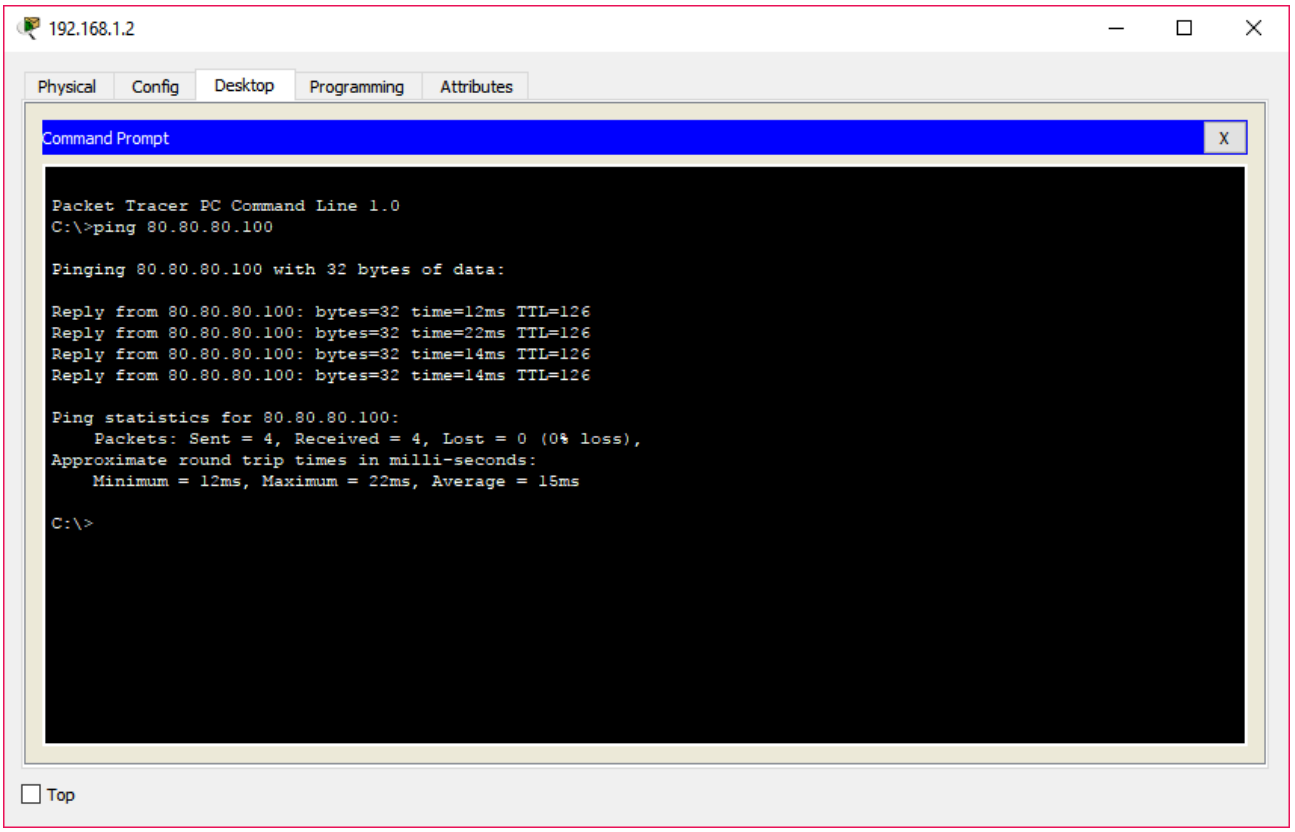
Укажем, что int fa0/0 — внутренний сетевой интерфейс для NAT:

```
Router(config)#int fa0/0
Router(config-if)#ip nat inside
```

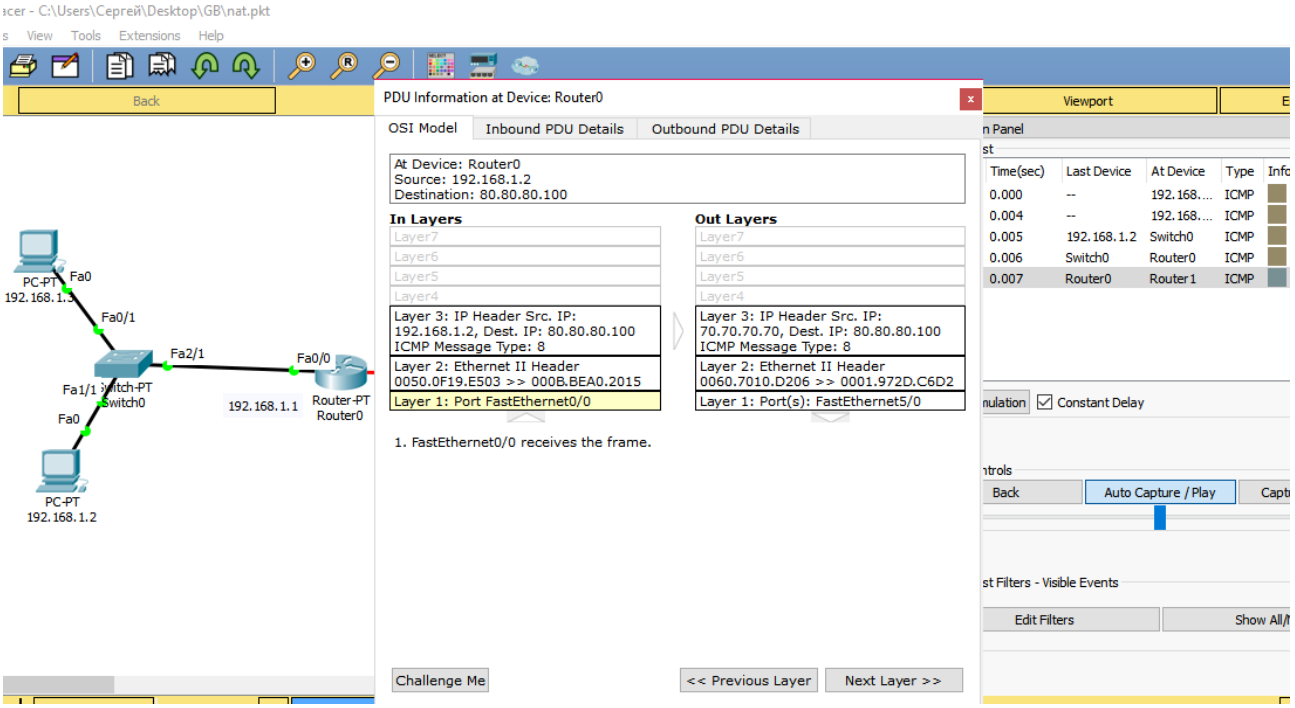
Укажем что int fa5/0 — внешний сетевой интерфейс для NAT:

```
Router(config-if)#int fa5/0
Router(config-if)#ip nat outside
Router(config-if)#exit
```

Выполним **ping** с машины 192.168.1.2:

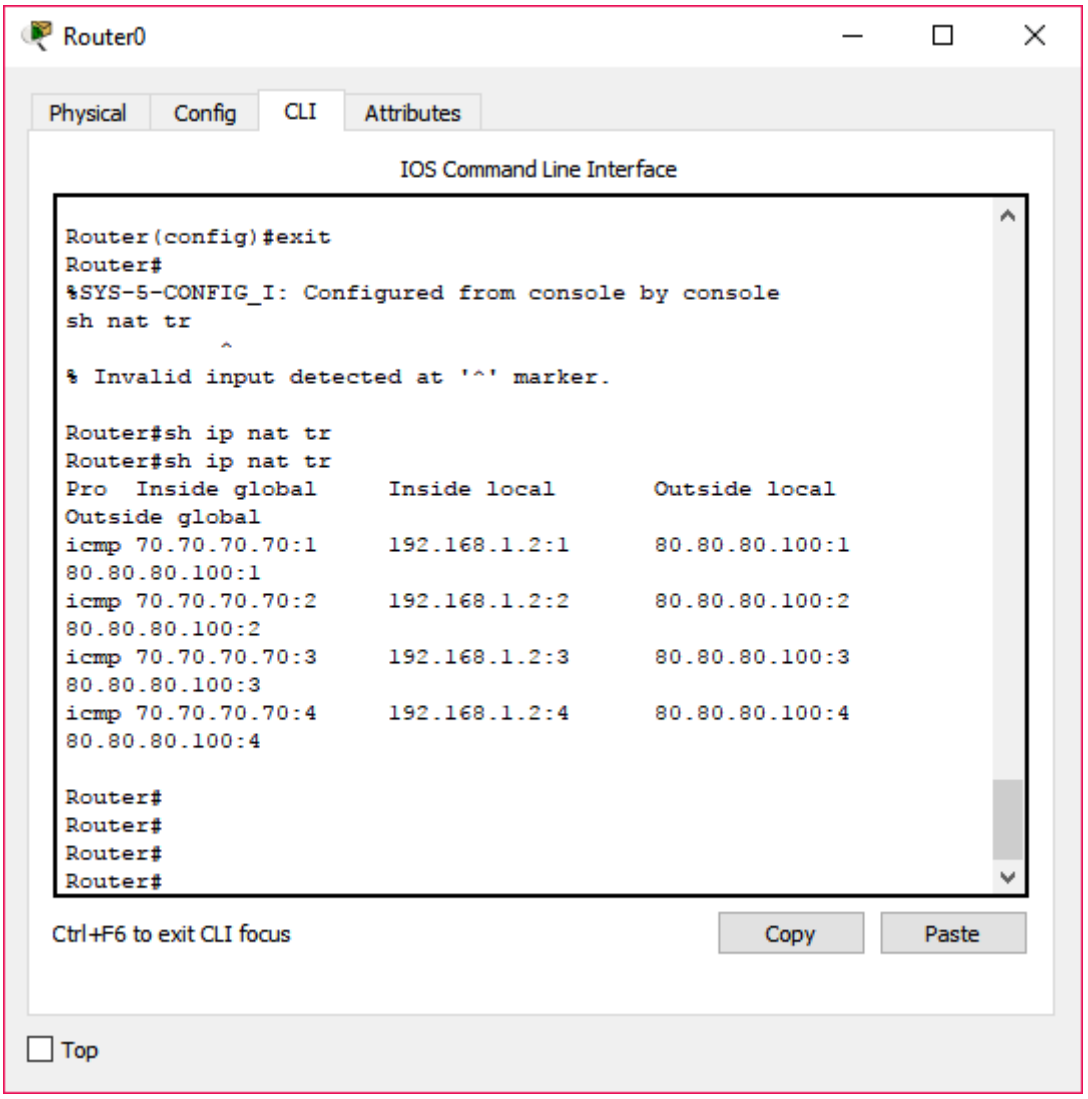


В режиме симуляции удостоверимся, что IP-адрес действительно подменяется:



Может возникнуть интересный вопрос: каким образом происходит трансляция ICMP, если в ICMP нет портов? Но в ICMP имеется номер последовательности, в трансляции он заменяется аналогично портам. Мы можем в этом убедиться, если посмотрим на маршрутизаторе таблицу трансляций:

```
Router# sh ip nat tr
```

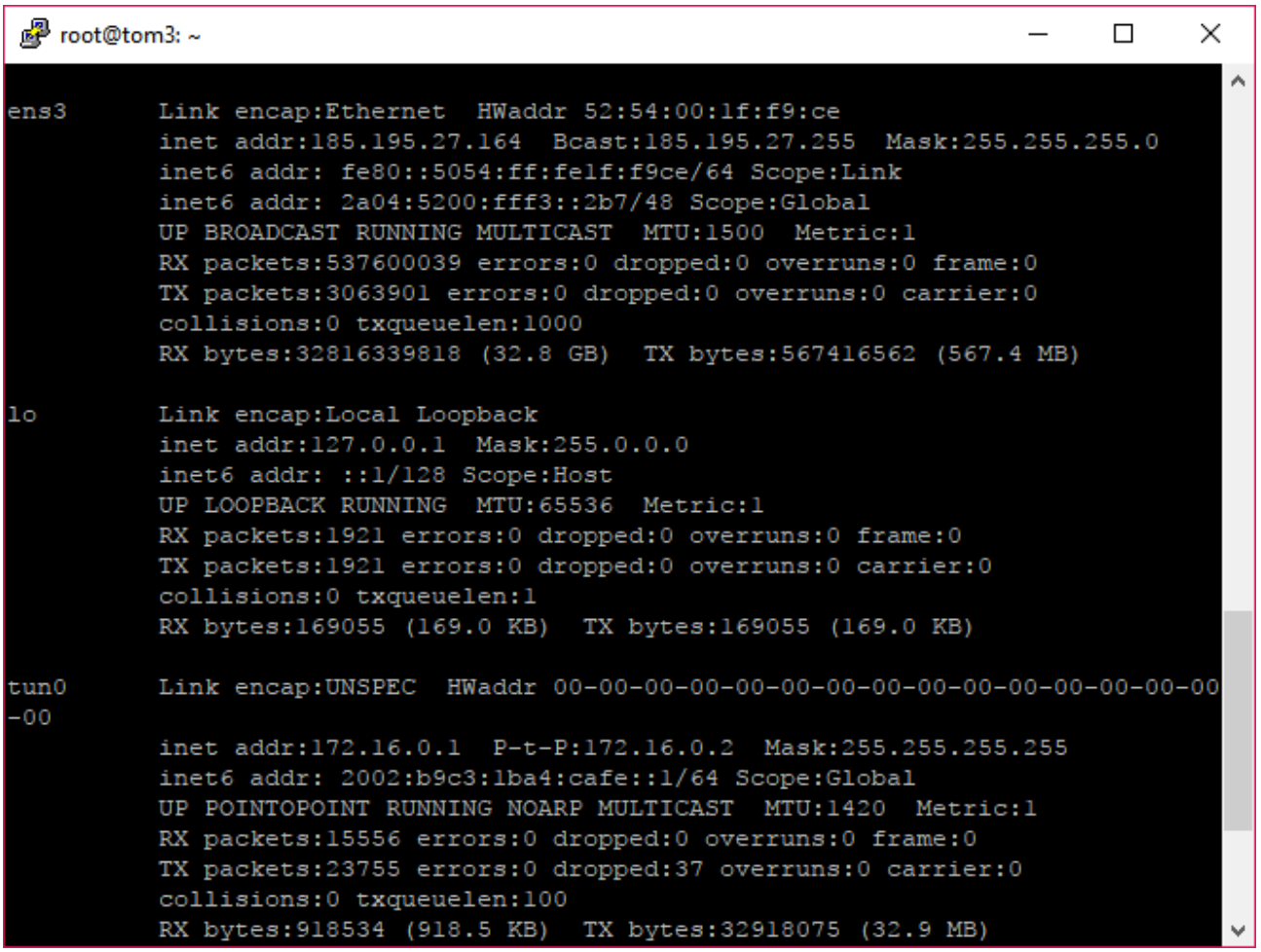


Самостоятельно настройте NAT-трансляцию (перегруженный NAT) и убедитесь, что подменяются не только IP-адреса, но и порты (зайдите на сервер с помощью браузера по протоколу HTTP, посмотрите пакеты в режиме симуляции, посмотрите на маршрутизаторе таблицу трансляции).

Настройка перегруженного NAT в Linux

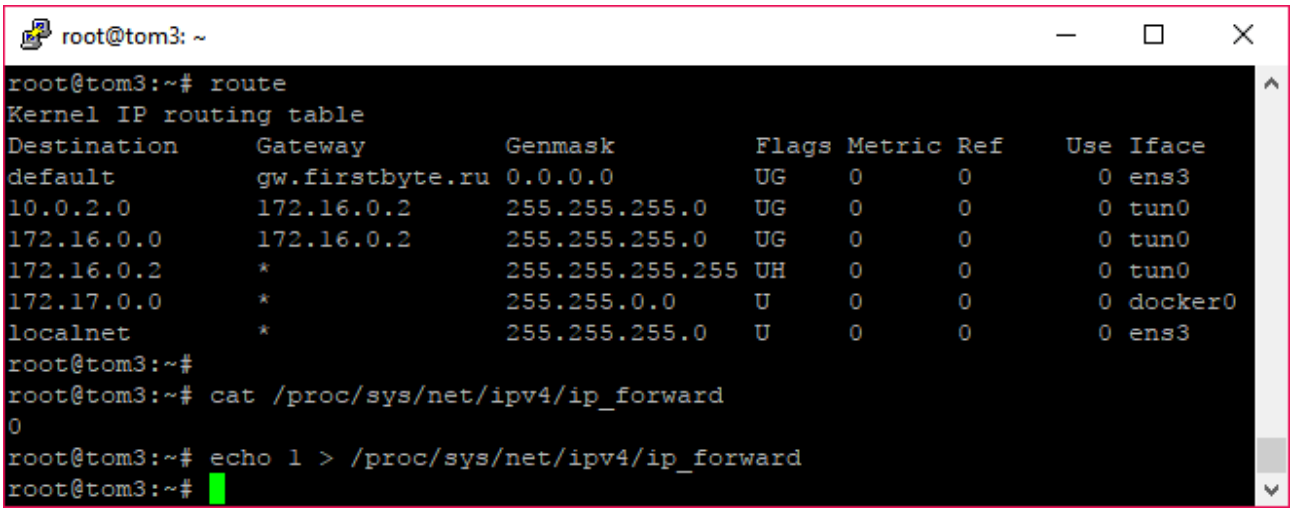
В Linux NAT является частью механизма сетевого фильтра. Доступ к настройкам NAT осуществляется с помощью утилиты **iptables**, для которой надо указать, что мы работаем с таблицей NAT (по умолчанию **iptables** работает с таблицей filter).

Предположим, что у нас имеется некая машина с «белым» адресом 185.195.27.164 и «серым» адресом 172.16.0.1, смотрящим в локальную сеть (на самом деле это туннельный адрес, но за ним действительно скрывается сеть с «серыми» IP-адресами).



Необходимо настроить машину так, чтобы пакеты, приходящие из сетевого интерфейса **tun0**, отправлялись в Интернет, но от имени 185.195.27.164.

Во-первых, нам придется включить IP-forwarding, чтобы можно было не только осуществлять маршрутизацию (отправку сообщений и в **ens33**, и в **tun0** в зависимости от масок сетей), но и пересылку сообщений из одних сетей в другие. Посмотрим таблицу маршрутизации и включена ли пересылка в ядре операционной системы (выведем в консоль содержимое файла /proc/sys/net/ipv4/ip_forward из **procfs**, отображающего содержимое соответствующих данных ядра операционной системы).

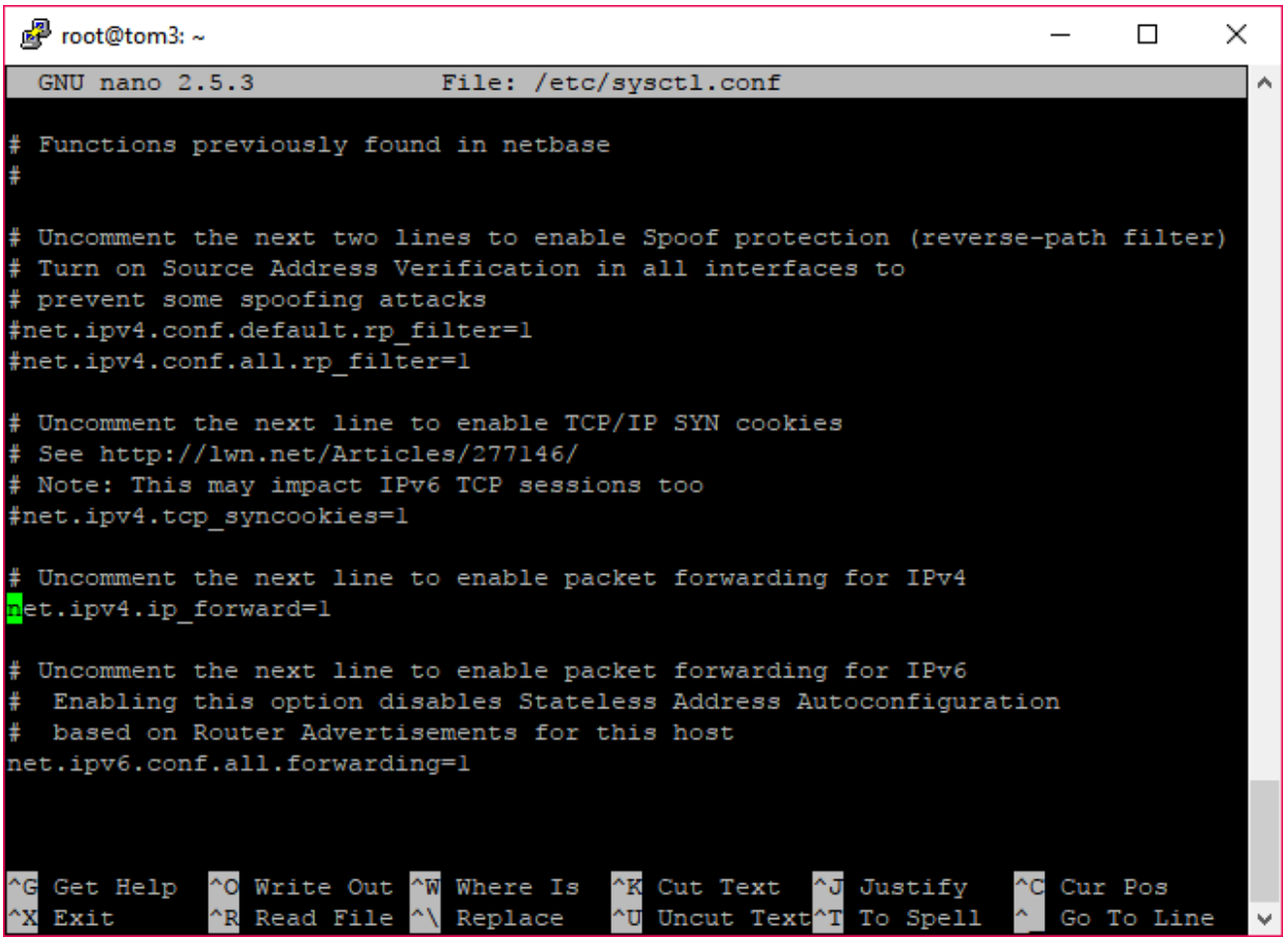


Видим, что пересылка пакетов выключена. Включим ее:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Чтобы пересылка была включена постоянно, нужно отредактировать другой файл, который хранится уже на диске, — /etc/sysctl.conf.

```
# nano /etc/sysctl.conf
```



Найдем и заменим значение **net.ipv4.ip_forward=0** на **net.ipv4.ip_forward=1**.

Нажмем **Ctrl-O**, чтобы сохранить изменения, и **Ctrl-X**, чтобы выйти.

Выполним команду:

```
# iptables -t nat -A POSTROUTING -o ens3 -j MASQUERADE
```

Мы указываем, что в таблице NAT после выполнения маршрутизации (т. е. в цепочке POSTROUTING) для сообщений, проходящих в сетевой интерфейс **ens3** (с «белым» адресом) мы будем выполнять маскарад. При получении ответов трансляция будет выполняться обратно, так как Netfilter работает на сеансовом уровне модели OSI/ISO, отслеживая сессии и выполняя замену в обе стороны.

После этого можно проверить работу NAT, прописав в качестве шлюза 172.16.0.1 на других машинах и проверив, что Интернет действительно работает, а также воспользовавшись сервисом 2ip.ru, убедившись, что виден только внешний, «белый» IP-адрес.

Мы выполняем трансляцию для всех сетей. С помощью ключа **-s** можно указать сеть, для которой мы хотим выполнять маскарад.

Чтобы трансляция работала постоянно, необходимо выполнить команду:

```
# iptables-save >/etc/iptables.rules
```

(Убедитесь, что файла не существует, иначе предыдущее содержимое будет утеряно.)

В файле будет строчка:

```
-A POSTROUTING -o ens3 -j MASQUERADE
```

Для Ubuntu 16, чтобы это работало при рестартах системы, в файл /etc/network/interfaces для соответствующего сетевого интерфейса нужно задать правило:

```
iface ens3 inet static
   # пропущено несколько строк
     pre-up iptables-restore < /etc/iptables.rules
```

Практическое задание

В файле **Lesson4Homework.pkt** (с ним вы уже работали, когда выполняли задания по статической, динамической маршрутизации и DHCP в 4 и 6 уроках) необходимо добавить новый шлюз между сетью 192.168.10.0/24 и Router0 (либо настройте Router0 как такой шлюз, но добавьте взамен новый роутер, входящий в сети 172.16.0.0/16 и 172.17.0.0/16).

Необходимо выполнить такие настройки, чтобы на данном шлюзе выполнялась трансляция адресов и портов — перегруженный NAT.

Обратите внимание на правильные настройки протоколов маршрутизации.

Прежде чем приступить к выполнению практического задания, обязательно тщательно продумайте, какой адрес вы выделите для образовавшейся новой сети, какие адреса вы назначите сетевым интерфейсам, где нужен DHCP-сервер, а где нет. Не остались ли на сетевых интерфейсах IP-адреса, которые не используются? Внимательно подпишите все настройки и проверьте.

В качестве домашнего задания пришлите сам файл .pkt и в комментарии к работе — перечень настроек (вводимых команд или вывод **sh run**) на Router 0 и новом роутере.

Если домашнее задание кажется слишком сложным, можно выполнить простое домашнее задание: воспроизвести пример из урока с двумя маршрутизаторами и двумя компьютерами.

Дополнительные материалы

1. http://simple4ip.com/blog/rus/2012/04/новая-серая-сеть-100-64-0-010/.
2. http://xgu.ru/wiki/NAT.

Используемая литература

Для подготовки данного методического пособия были использованы следующие ресурсы:

1. https://webhamster.ru/mytetrashare/index/mtb0/28.
2. http://xgu.ru/wiki/NAT.