

# Сеть и сетевые протоколы: NAT



Андрей  
Вахутинский



# Андрей Вахутинский

Заместитель начальника IT-отдела  
АО “ИНТЕКО”

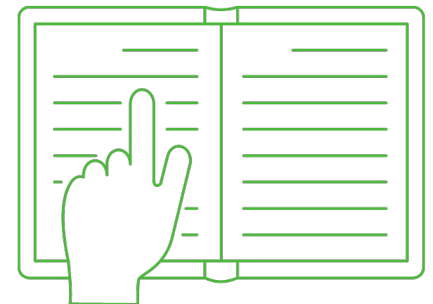


[Андрей Вахутинский](#)

# Предисловие

Эта лекция содержит основные понятия, связанные с технологией NAT (Network Address Translation).

Также вы познакомитесь с основными командами, которые позволяют настраивать NAT в ОС Linux.



---

# План занятия

1. [Приватные и публичные сети](#)
2. [NAT и его виды](#)
3. [Static NAT / Dynamic NAT](#)
4. [Source NAT](#)
5. [Destination NAT](#)
6. [Преимущества и недостатки NAT](#)
7. [Примеры настройки](#)
8. [Итоги](#)
9. [Домашнее задание](#)



# Приватные и публичные сети

---

# Виды сетей

IPv4 адреса бывают двух типов:

- публичные (public);
- приватные (private).

Первые так же часто называют белыми, или внешними.

Вторые – серыми, частными или внутренними.

→ Почему так сделано и для чего существует такое разделение?

---

# Предпосылки к созданию частных сетей

1. Ограниченное количество IP-адресов (4,3 млрд).

Как думаете, откуда такая цифра?

2. Существуют сети, которым не нужно ни с кем взаимодействовать.

Для чего им задействовать адреса, через которые к ним смогут подключиться?



---

# Уникальность адресов в сети интернет

В интернете идентификация устройств осуществляется **уникальными IPv4-адресами**, которые не могут повторяться в глобальной сети.

За этим следят специальные организации, выделяющие адреса (RIPE).

Однако число таких уникальных адресов ограничено, поэтому было определено так называемое **частное пространство IP-адресов**.

Частные IPv4-адреса предназначены для использования в локальных компьютерных сетях и **не маршрутизируются** в глобальную сеть интернет.



# Частные адреса

Частные IPv4-адреса не являются уникальными и могут использоваться во внутренней сети.

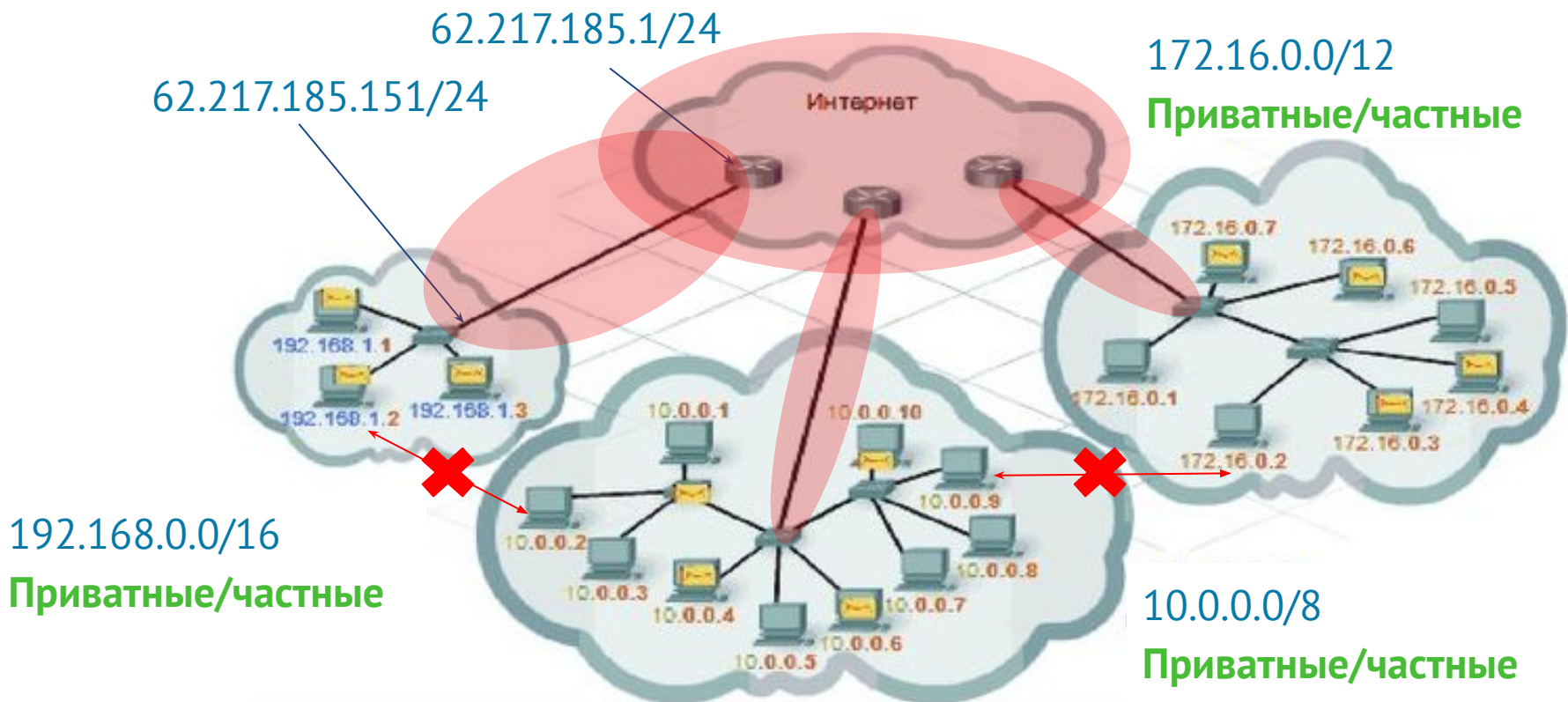
В частности, блоками частных адресов являются:

- 10.0.0.0/8 или от 10.0.0.0 до 10.255.255.255;
- 172.16.0.0/12 или от 172.16.0.0 до 172.31.255.255;
- 192.168.0.0/16 или от 192.168.0.0 до 192.168.255.255.

➡ Адреса в этих блоках адресов не допустимы для использования в Интернете и должны отклоняться интернет-маршрутизаторами.

# Виды сетей

**Публичные** (почти все остальные)



---

## Выход в интернет с частных адресов

Следующий важный вопрос – допустим, мы выдали сотрудникам в офисе много частных адресов.

Как они будут сидеть в интернете?

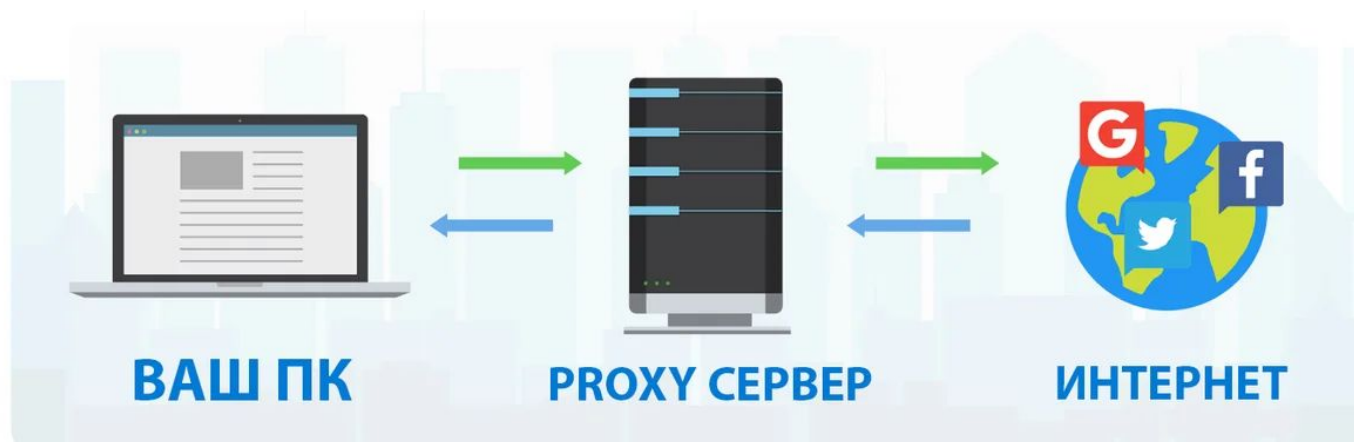
Они будут отправлять запросы в сеть, а ответы на эти запросы должны будут возвращаться на обратные адреса, которые в данном случае **частные**.

Так как в интернете никто не знает маршрутов к частным адресам, то это **невозможно**.

# Выход в интернет с частных адресов

## 1. Использование прокси.

Компьютер подключается, например, к серверу по протоколу HTTP и всё взаимодействие идёт между ПК и сервером. Дальше сервер уже сам делает нужные запросы, либо отдаёт информацию из кэша.



# Выход в интернет с частных адресов

## 2. Подмена адресов (NAT, Network Address Translation).

Компьютер подключается к конечному серверу напрямую по любому протоколу, при этом промежуточный роутер преобразовывает частные «серые» адреса в публичные (например, в свой).





# **NAT и его виды**

# Что такое NAT

**NAT** (Network Address Translation) — это специальный механизм, реализованный в сетях TCP/IP, который позволяет изменять IP-адреса и/или номера портов TCP/UDP в пересылаемых пакетах.

→ Реализуется через маршрутизатор, на программном уровне.



---

## Виды NAT

- Static NAT / One to one NAT;
- Dynamic NAT;
- Source NAT / NAT Overload / Masquerade / Many to one;
- Destination NAT (PAT).







# Static NAT / Dynamic NAT

## Принцип действия Static NAT

Берется один IP-адрес из внутренней сети и преобразовывается в один внешний/публичный IP-адрес маршрутизатора.

→ Таким образом, все запросы, которые будут на внешний IP-адрес посылаться извне, будут пересылаться только на этот внутренний IP.

Т.е. для всемирной паутины внутренний адрес будет выглядеть один-в-один, как белый/публичный IP.



---

# Static NAT

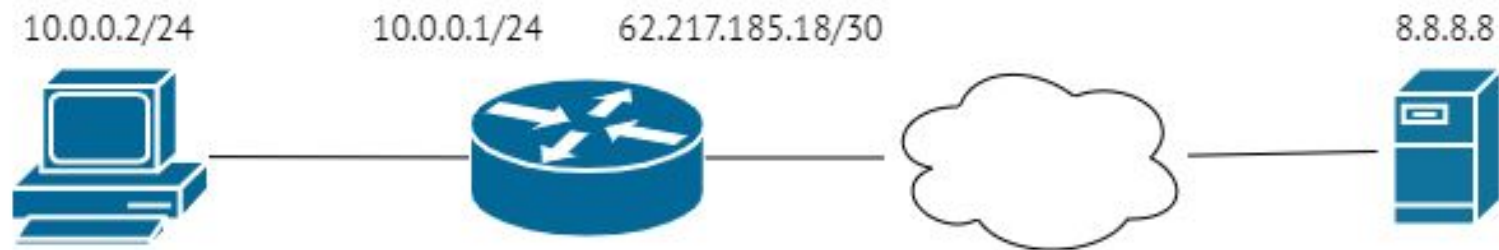
Для чего используется:

Обычно такая схема используется в случае если есть необходимость «опубликовать» внутренний сервер компании в интернет, причём не просто один какой-то сервис (HTTP или HTTPS), а все сервисы сразу.

Важно:

Несмотря на то, что у сервера «серый» адрес, он полноценно отвечает на запросы извне (по другому, «белому» адресу).

# Схема Static NAT



## Преобразование

src IP	dst IP		src IP	dst IP		src IP	dst IP
10.0.0.2	8.8.8.8	→	62.217.185.18	8.8.8.8	→	62.217.185.18	8.8.8.8

## Преобразование

src IP	dst IP		src IP	dst IP		src IP	dst IP
8.8.8.8	10.0.0.2	←	8.8.8.8	10.0.0.2	←	8.8.8.8	62.217.185.18

## Из локальной сети в интернет

Наш сервер **10.0.0.2** обращается ко внешнему серверу на **8.8.8.8**, отправляет пакет данных на шлюз.

Шлюз **10.0.0.1**, на котором настроен NAT переписывает в пакете IP-адрес отправителя (поле *source ip* IP-пакета) в белый адрес **62.217.185.15** (выдан провайдером).

Всё взаимодействие этого компьютера в интернете идёт через «белый» IP-адрес **62.217.185.15**, ответ от сервера **8.8.8.8** отправляется также на этот адрес, т.к. **в интернете никто ничего не знает про 10.0.0.2** благодаря трансляции.

## Из интернета в локальную сеть

Из интернета происходит попытка подключения к нашему серверу по адресу **62.217.185.15** (т.к. *про 10.0.0.2 никто ничего не знает, т.к. сеть немаршрутизируемая* и пакеты до неё ни от кого не дойдут).

На нашем шлюзе есть трансляция, преобразующая **все** пакеты из **10.0.0.2** в **62.217.185.15** и из **62.217.185.15** в **10.0.0.2**

На основании этого правила шлюз преобразовывает в IP-пакете поле **destination IP** из **62.217.185.15** в **10.0.0.2** и отправляет его уже в локальной сети.

## Принцип действия Dynamic NAT

Берется несколько IP-адресов из внутренней сети и преобразовываются в несколько внешних/публичных IP-адресов, выделенных провайдером.

Всё аналогично **static NAT**, с той лишь разницей, что реальные адреса теперь будут выдаваться **динамически** каждому нуждающемуся пользователю во внутренней сети, а не одному определенному узлу.

# Схема Dynamic NAT

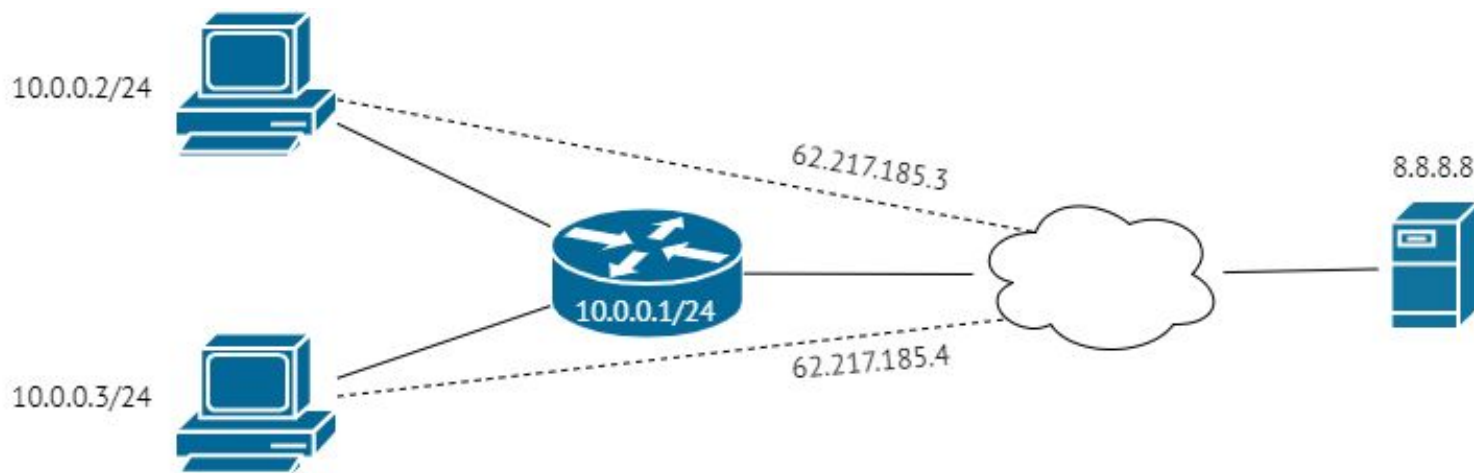


Таблица преобразований

src IP	dst IP
10.0.0.2	8.8.8.8
10.0.0.3	8.8.8.8

src IP local	src IP global	dst IP
10.0.0.2	62.217.185.3	8.8.8.8
10.0.0.3	62.217.185.4	8.8.8.8

src IP	dst IP
62.217.185.3	8.8.8.8
62.217.185.4	8.8.8.8

src IP	dst IP
8.8.8.8	10.0.0.2
8.8.8.8	10.0.0.3

src IP	dst IP local	dst IP global
8.8.8.8	10.0.0.2	62.217.185.3
8.8.8.8	10.0.0.3	62.217.185.4

src IP	dst IP
8.8.8.8	62.217.185.3
8.8.8.8	62.217.185.4



## Ограничения Dynamic NAT

Если внешних адресов, например, 10, а пользователей 300?

А мы помним, что свободные «белые» IP-адреса на вес золота – мы не сможем для таких целей попросить у провайдера 100 адресов.

→ Те хосты, кто успел первым, те и смогут их использовать.

Используется редко.





# Source NAT

---

# Source NAT / Masquerading / Nat overload

## Принцип действия

Подменяем IP адреса источников ([source](#)), например, для организации выхода в Интернет нескольких компьютеров через один публичный IP адрес.

*Адрес не обязательно прописан на интерфейсе.*

При [source NAT](#) для серверов внешней сети, запросы от наших компьютеров из внутренней сети будут выглядеть так, как будто с ними общается напрямую шлюз.

---

# Source NAT / Masquerading / Nat overload

## Для чего используется

Обычно такая схема используется для выхода в интернет группы компьютеров с внутренними адресами через 1 внешний адрес (в случае с маскарadingом – адрес интерфейса).

## Важно:

Снаружи на этот адрес пропускаются только пакеты, содержащиеся в таблице трансляций.



# Схема Source NAT

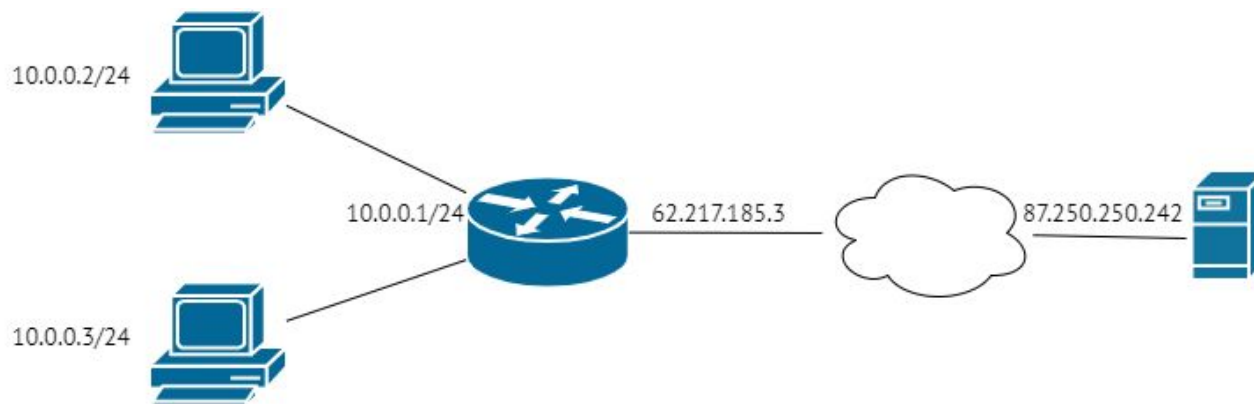


таблица преобразований (динамическая)

src IP:port	dst IP:port	src IP local:port	src IP global:port	dst IP:port
10.0.0.2:2001	87.250.250.242:80	10.0.0.2:2001	62.217.185.3:3500	87.250.250.242:80
10.0.0.3:4999	87.250.250.242:80	10.0.0.3:4999	62.217.185.3:2999	87.250.250.242:80

## Из локальной сети в интернет

Наши компьютеры `10.0.0.0/24` обращаются ко внешним серверам, например на `87.250.250.242:80`, отправляют пакеты на шлюз, подставляя рандомный порт в качестве `src port`.

Шлюз `10.0.0.1`, на котором настроен NAT, переписывает в пакете IP-адрес и порт отправителя (поля *source ip*, *source port* IP-пакета) в «белый» адрес `62.217.185.3` и рандомный `src port`.

Дальше он записывает в таблицу NAT трансляций запись:

```
10.0.0.2:2001 - 62.217.185.3:3500 - 87.250.250.242:80
```

для того, чтобы правильно обработать ответный пакет.

## Из интернета в локальную сеть

Шлюз пропустит только пакеты, соответствующие существующим NAT трансляциям, остальные будут отброшены.

```
10.0.0.2:2001 - 62.217.185.3:3500 - 87.250.250.242:80  
10.0.0.3:4999 - 62.217.185.3:2999 - 87.250.250.242:80
```

Разрешены будут пакеты:

src IP:port	dst IP:port
87.250.250.242:80	62.217.185.3:3500
87.250.250.242:80	62.217.185.3:2999

\* Через некоторый таймаут строка из таблицы трансляций удалится.

---

# NAT Masquerading

## Принцип действия

Аналогичен Source NAT, однако IP-адрес, в который происходит подмена, **должен быть прописан на интерфейсе**.

Подменяем IP адреса источников (**source**), например, для организации выхода в Интернет нескольких компьютеров через один публичный IP адрес.

Адрес обязательно прописан на интерфейсе.

Термин чаще используется в Linux.





# Destination NAT

---

# Destination NAT / PAT

## Принцип действия

Подменяем IP адреса получателя (**destination**), например, для подключения снаружи, с публичного адреса, на «серый» адрес внутри сети.

При **destination NAT** для серверов внешней сети, ответы от наших серверов из внутренней сети будут выглядеть так, как будто с ними общается напрямую шлюз.

---

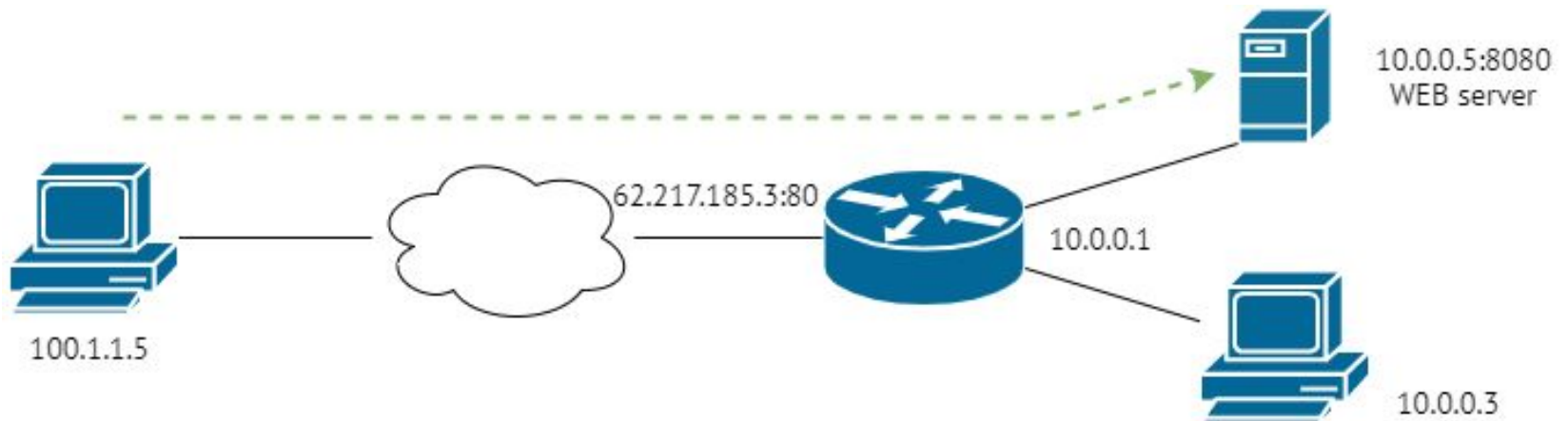
# Destination NAT / PAT

## Для чего используется

Обычно такая схема используется для публикации сервиса (port), находящегося внутри сети, для внешних пользователей.



## Схема Destination NAT / PAT



### Таблица преобразований (статическая)

src IP:port	dst IP:port
100.1.1.5:2222	<b>62.217.185.3:80</b>


src IP	dst IP global:port	dst IP local:port
100.1.1.5:2222	<b>62.217.185.3:80</b>	<b>10.0.0.5:8080</b>

## Снаружи в локальную сеть

Существует одна статическая трансляция, которая преобразовывает трафик снаружи на конкретный хост внутри сети.

```
62.217.185.3:80 - 10.0.0.5:8080
```

Порты могут быть разными, они не обязаны совпадать.



# Преимущества и недостатки NAT

---

## Преимущества NAT

1. Под одним внешним IPv4 адресом может сидеть в глобальной паутине множество пользователей одновременно.
2. Скрывает ваш настоящий внутренний IP в частной сети и показывает лишь внешний. Так, все устройства из вне видят только ваш общедоступный IP.
3. В определенной степени выполняет функции фаервола – если на устройство с NAT извне приходит пакет, который не ожидался – то он категорически не будет допущен.

---

## Недостатки NAT

1. Невысокая скорость передачи данных для протоколов реального времени, например, для VoIP. Когда NAT переделывает заголовки пакета — происходят задержки.
2. Проблемы с идентификацией – под одним IP может находиться сразу несколько человек.
3. Сервис может заблокировать внешний IP-адрес из-за злоумышленника (который, например, подбирает пароли), а работать перестанет у всех, кто закрывается этим адресом.





# Примеры настройки

# NAT в Linux

NAT в Linux реализуется с помощью пакета [iptables](#).

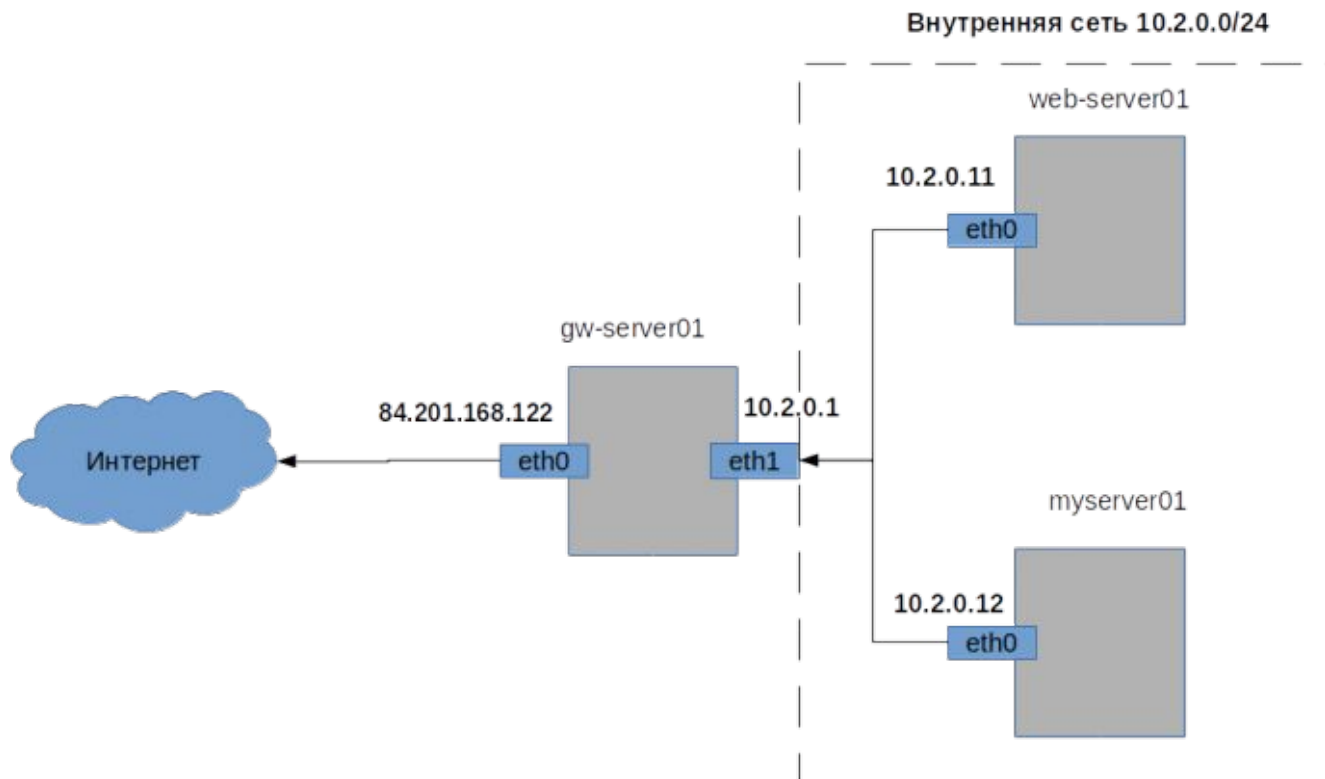
В курсе есть целая лекция, посвященная этому продукту, здесь мы вкратце затронем базовый функционал, с помощью которого в Linux реализуется NAT.

[iptables](#) оперирует такими сущностями как:

- «таблицы» – [tables](#);
- «цепочки» – [chains](#).

 В рамках данной лекции нас интересует таблица «nat».

# Настройка SNAT: доступ из LAN в интернет



# SNAT



---

# SNAT

1. IP пакет поступил на внутренний интерфейс **eth1** сервера **gw-server01**.
2. После для IP пакета определяется исходящий сетевой интерфейс, с которого он должен быть отправлен, это отмечено желтым цветом.
3. В конце IP пакет проходит цепочку **POSTROUTING**, в которой происходит подмена IP адреса **источника**, на IP адрес **внешнего** интерфейса **eth0** сервера **gw-server01**.

# Команды routing в Linux

Для начала необходимо разрешить пересылку пакетов с одного интерфейса на другой (по умолчанию в Linux это отключено).

```
# до перезагрузки
sudo sysctl -w net.ipv4.ip_forward=1

# на постоянной основе
/etc/sysctl.conf =>
net.ipv4.ip_forward = 1

$ sudo sysctl -p /etc/sysctl.conf
```

# Команды SNAT в Linux

Создадим правило в `iptables`, разрешающее передачу пакетов между внутренним (`eth1`) и внешним (`eth0`) интерфейсом и разрешим передавать между интерфейсами пакеты, относящиеся к уже установленным соединениям.

```
iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
```

- `-A` – add;
- `-i / -o` – input/output;
- `-m` – использовать доп.модуль;
- `-j` – действие.

# Команды SNAT в Linux

## Включим SNAT:

```
iptables -t nat -A POSTROUTING -s 10.2.0.0/24 -o eth1 -j SNAT --to-source 84.201.168.122
```

- `-A` – add;
- `-t` – таблица;
- `-s` – source (необязательно);
- `-j` – действие;
- `--to-source` должен быть адресом на интерфейсе, с которого планируется выпускать во внешнюю сеть IP пакеты.



# Команды NAT, iptables в Linux

Посмотрим получившуюся конфигурацию для таблицы **filter** и цепочки **FORWARD** (вывод обрезан):

```
iptables -L -n -v
```

```
Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target      prot opt in     out     source           destination
 136 8863 ACCEPT      all  --  eth1    eth0    0.0.0.0/0        0.0.0.0/0
  12 1234 ACCEPT      all  --  *       *       0.0.0.0/0        0.0.0.0/0          state RELATED,ESTABLISHED

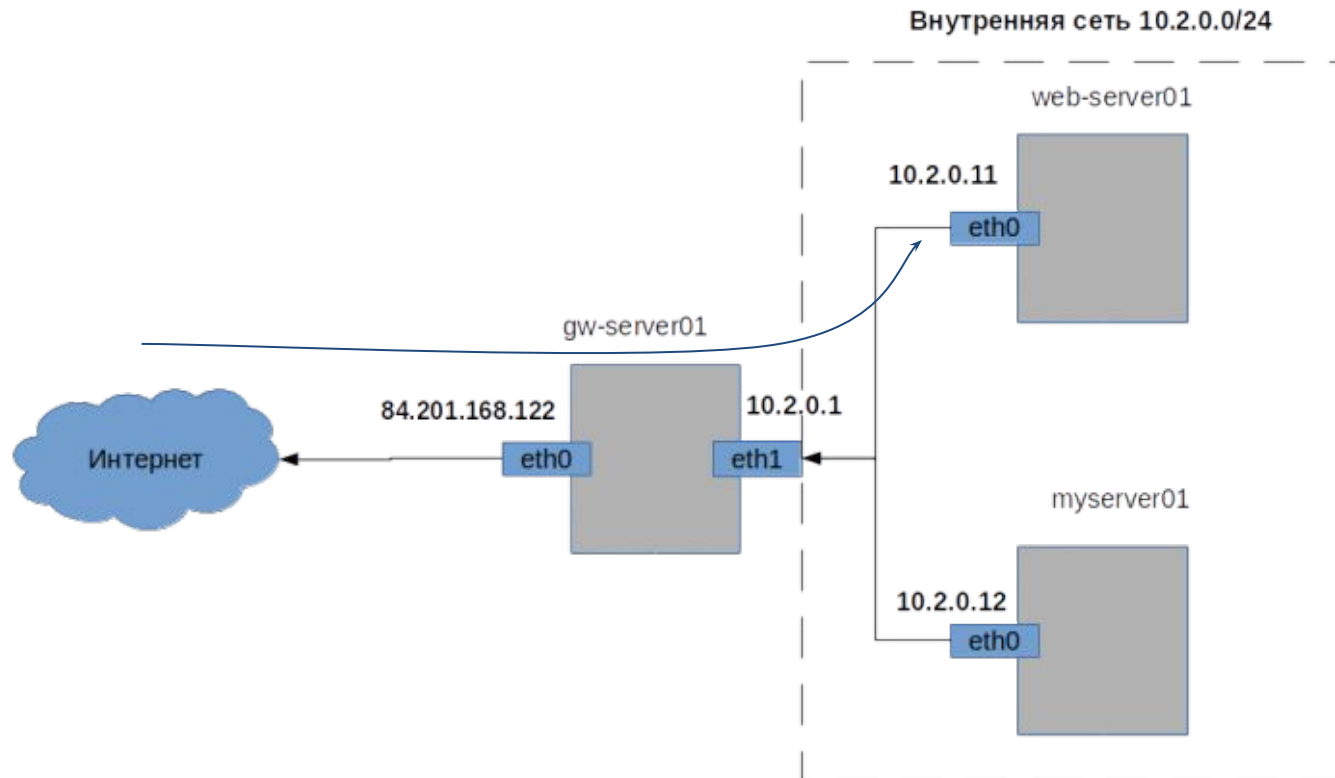
Chain OUTPUT (policy ACCEPT 96 packets, 10160 bytes)
pkts bytes target      prot opt in     out     source           destination
```

и конфигурацию для таблицы **nat** (вывод обрезан):

```
iptables -t nat -L -n -v
```

```
Chain POSTROUTING (policy ACCEPT 8 packets, 556 bytes)
pkts bytes target      prot opt in     out     source           destination
  0    0 SNAT        all  --  *       eth0    10.2.0.0/24      0.0.0.0/0          to:84.201.168.122
```

# Настройка DNAT: доступ из интернет в LAN



---

# DNAT

1. IP пакет поступил на внешний интерфейс **eth0** сервера **gw-server01**.
2. После в IP пакете меняется **destination IP** и при необходимости **destination port**.
3. Дальше происходит маршрутизация в соответствии с правилами на сервере (таблица маршрутизации).

## Команды DNAT в Linux

Сначала разрешим передачу пакетов с внешнего интерфейса (eth0) на внутренний (eth1) интерфейс:

При необходимости можно отдельным правилом запретить подключение через NAT для отдельных IP адресов или подсетей.

```
iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT
```

```
iptables -I FORWARD 1 -o eth1 -s 167.71.67.136 -j DROP
```

## Команды DNAT в Linux

Теперь перенаправим все соединения на порт 80 интерфейса внешней сети (eth0) на IP адрес веб сервера внутренней сети `web-server01`.

И все соединения на порт `13389` перенаправлять на порт `3389` сервера внутренней сети (в целях безопасности).

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 -j DNAT --to-destination 10.2.0.11
```

```
iptables -t nat -A PREROUTING -p tcp --dport 13389 -i eth0 -j DNAT --to-destination 10.2.0.12:3389
```



# Итоги

---

# Итоги

Сегодня мы:

- познакомились с базовыми представлениями о трансляции сетевых адресов;
- узнали для чего они используются;
- познакомились с базовыми командами для настройки NAT в Linux.





# Домашнее задание



---

# Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

**Задавайте вопросы и  
пишите отзыв о лекции!**

**Андрей Вахутинский**