

# Firewall

Артур Сагутдинов  
DevOps - инженер



# Артур Сагутдинов

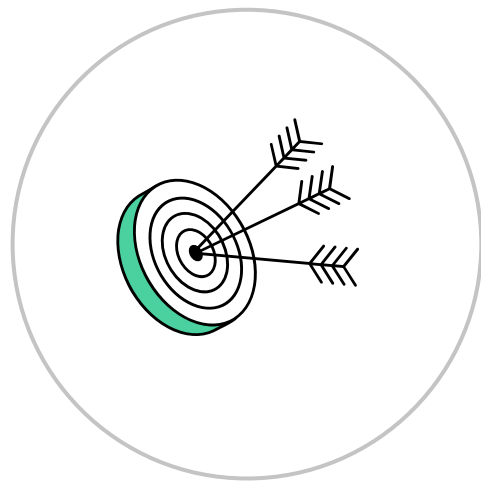
О спикере:

- Опыт в ИТ с 2007 года
- Эксперт в разработке и внедрении Linux - инфраструктуры
- Индивидуальный предприниматель



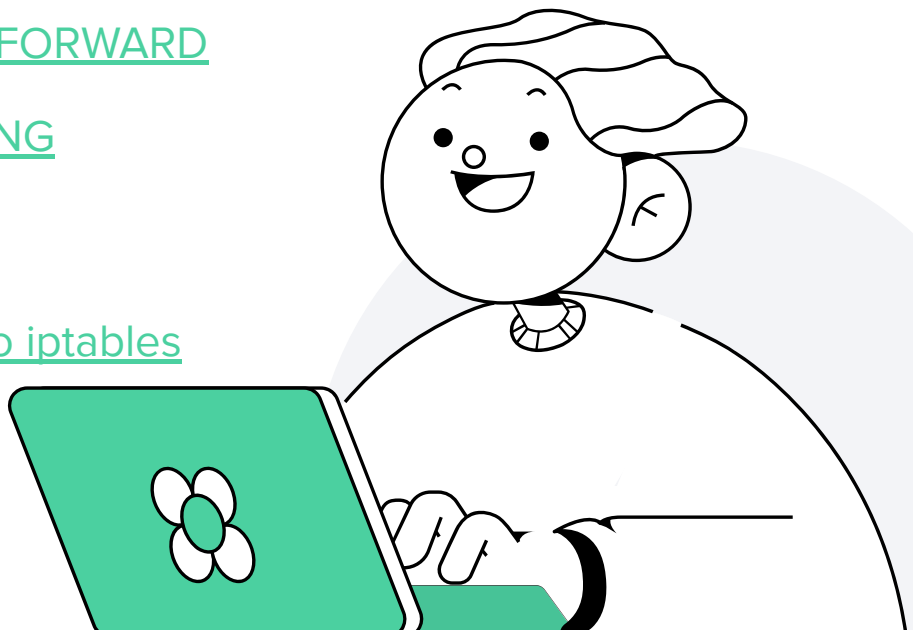
# Цели занятия

- Познакомиться с протоколом Firewall
- Изучить возможности межсетевого экрана Netfilter и его утилиты iptables
- Научиться управлять трафиком с помощью собственного небольшого роутера
- Научиться создавать базовые правила в iptables

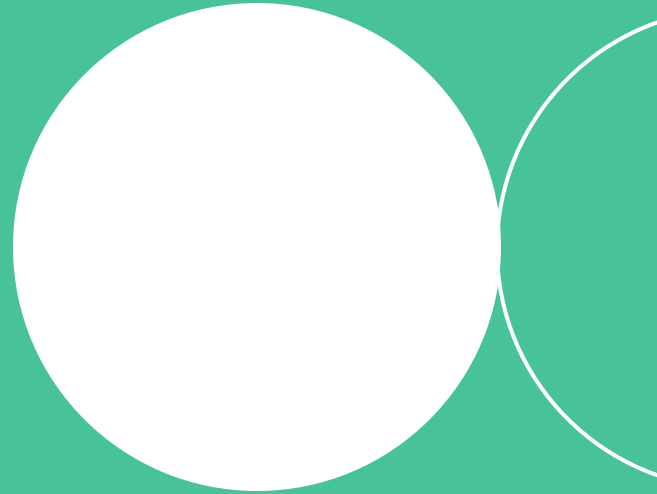


# План занятия

- 1 [Firewall](#)
- 2 [Историческая справка о Firewall](#)
- 3 [Netfilter и iptables](#)
- 4 [Цепочки iptables: PREROUTING, INPUT, FORWARD](#)
- 5 [Цепочки iptables: OUTPUT, POSTROUTING](#)
- 6 [Синтаксис iptables](#)
- 7 [Настройка доступа по порту с помощью iptables](#)
- 8 [Итоги](#)
- 9 [Домашнее задание](#)

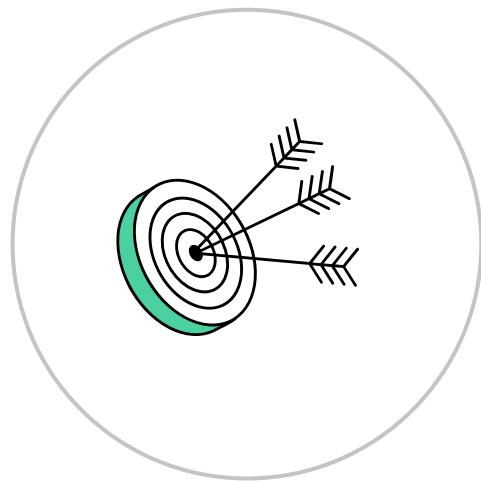


# Firewall



# Цели темы

- Познакомиться с Firewall, видами и особенностями его реализации
- Разобраться с видами угроз, которые может устранить Firewall
- Понять, против чего Firewall бессилен



# Дословный перевод Firewall

Firewall

Огненная стена



## Firewall

**дополнительный слой защиты между  
вами и проблемами**





# Реализации Firewall

1

Программное  
решение

2

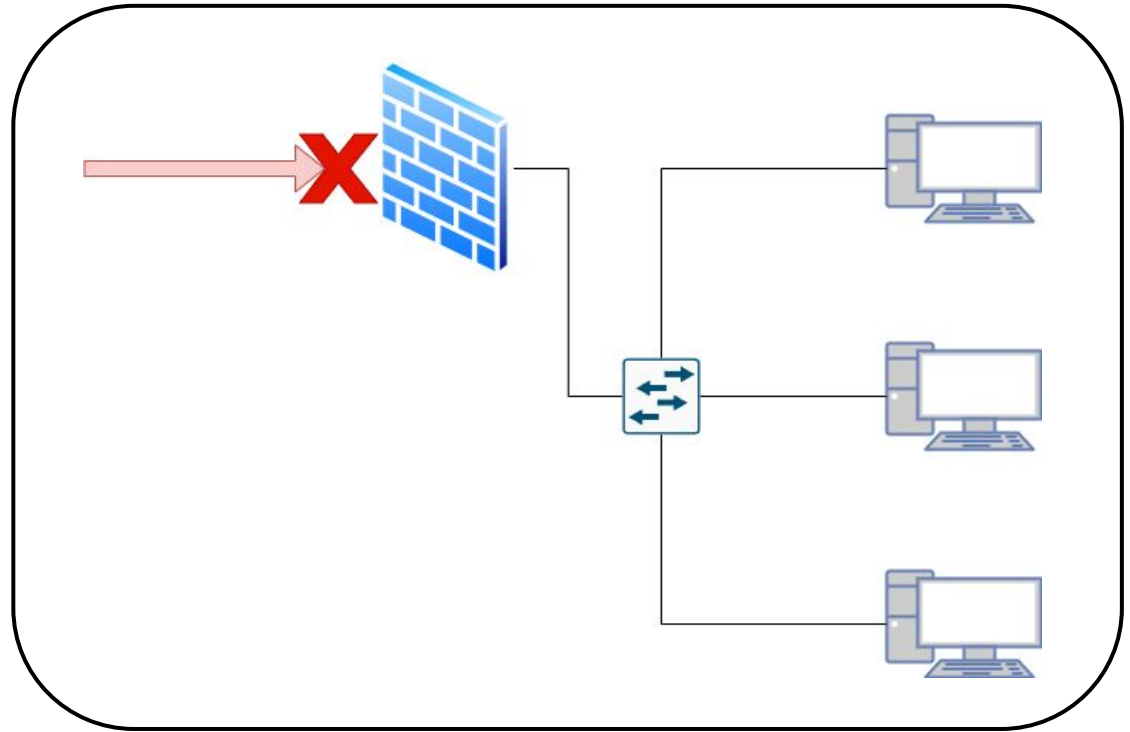
Программно-аппаратная  
реализация



**Задача Firewall -  
фильтрация проходящего  
через него трафика на основе  
определенных ранее правил**

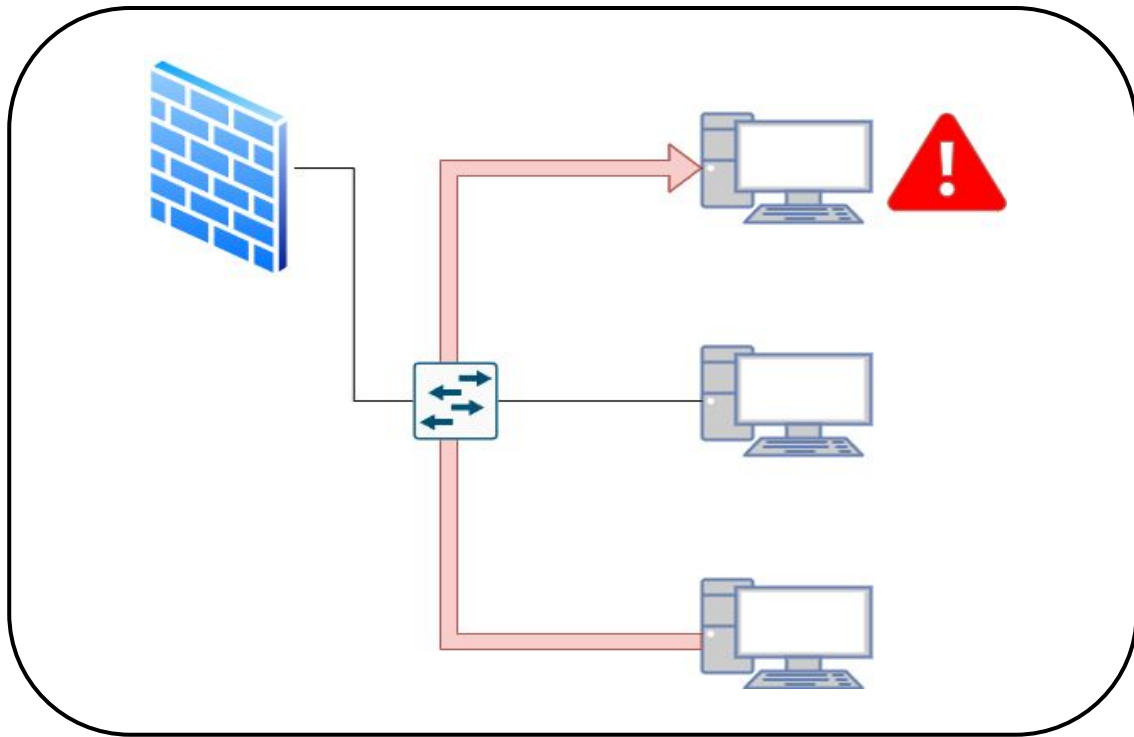
# Когда Firewall нужен?

Защита сетей или отдельных хостов от атак, направленных извне внутрь защищаемой сети



# Когда Firewall бесполезен?

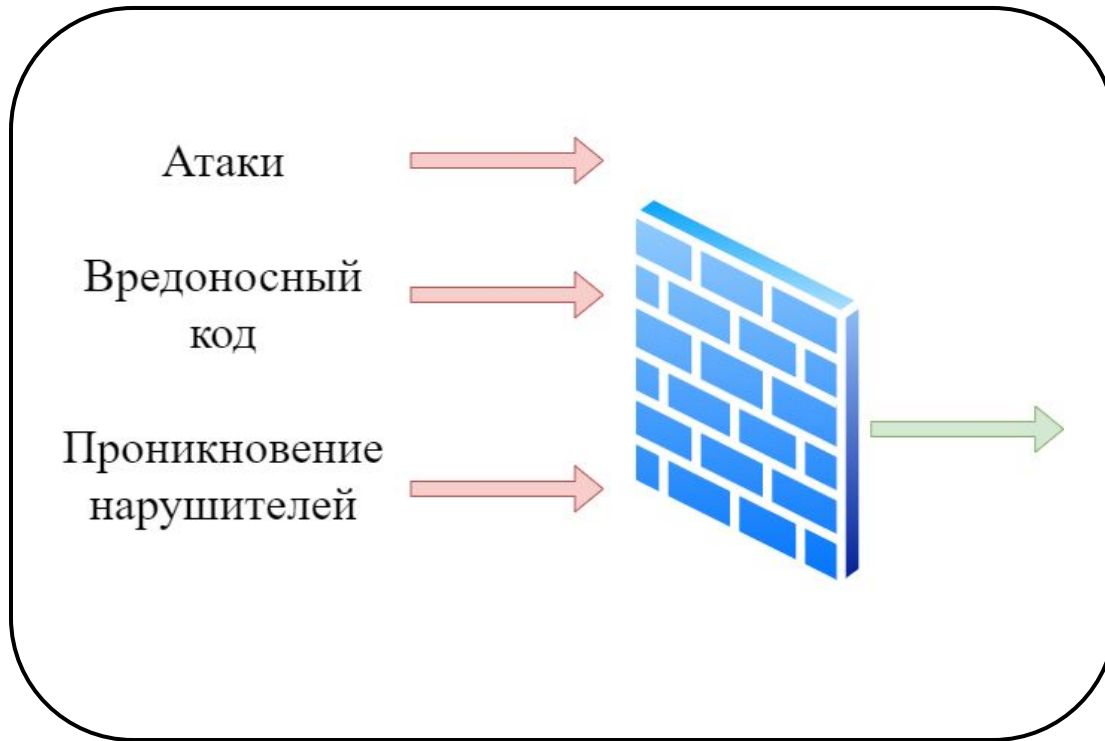
В борьбе с атаками,  
проводимыми внутри  
периметра, чей  
трафик не проходит  
сквозь его  
интерфейсы



## Отсутствие Firewall

- **высокие риски  
возникновения проблем**

# Какие есть риски без Firewall?



# На что направлены атаки?

- Вызов всевозможных неполадок
- Работа по принципу крипто-вымогателей
- Кража информации

**Наличие Firewall**  
**ОБЯЗАТЕЛЬНО**  
**при работе с публичным**  
**трафиком (общедоступными**  
**сетями), чтобы избежать**  
**утечки данных**

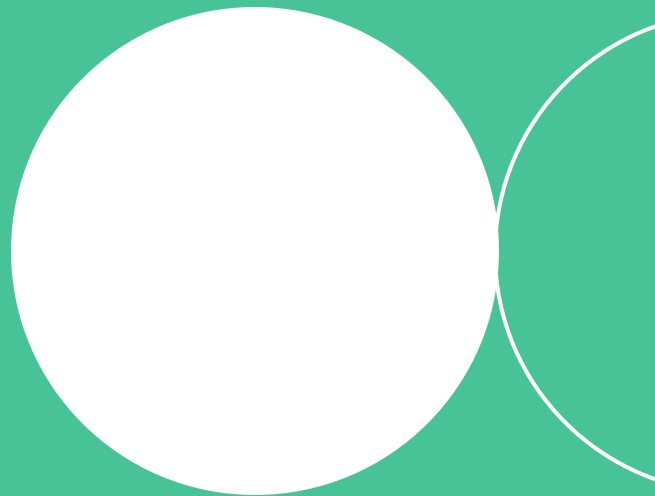


# Итоги темы

- 1 Firewall представляет собой программный или программно-аппаратный фильтр, анализирующий и управляющий проходящим через него трафиком
- 2 Если у вас нет Firewall , а всё работает, то велика вероятность, что ваши данные постоянно крадут
- 3 Firewall может справиться только с теми угрозами, потоком которых он может управлять

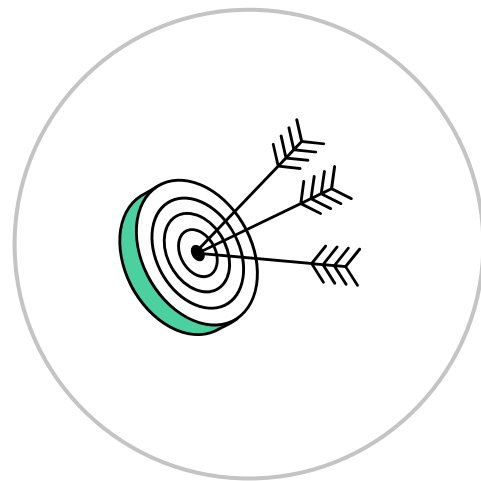


# Историческая справка о Firewall

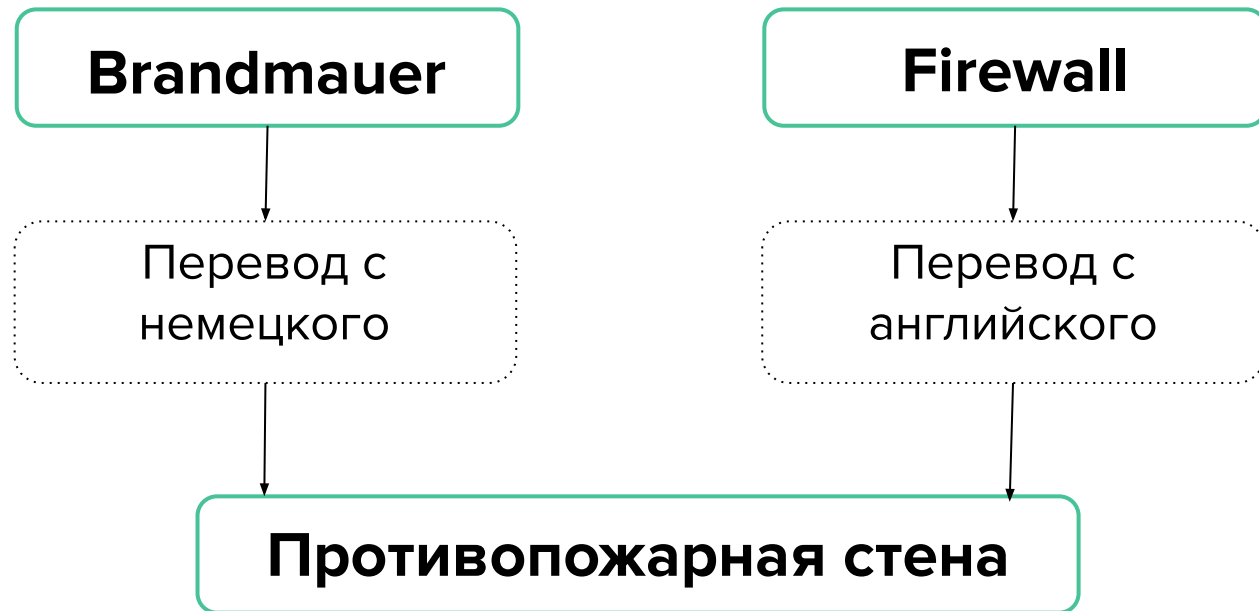


# Цели темы

- Узнать историю появления firewall
- Познакомится с ранними версиями firewall и понять их принцип работы



# Происхождение слова Firewall



# Предшественник Firewall



# Расти Рассел

Основатель ipchains

В 1998 г. создал проект  
**Netfilter/iptables**



# Функции Firewall выполняли:



The diagram consists of three circles arranged horizontally. Each circle contains text representing a Linux version and its associated firewall function. The first circle on the left is for Linux 2.0 with the function 'ipfwadm'. The middle circle is for Linux 2.2 with the function 'ipchains'. The third circle on the right is for Linux 2.4 with the function 'Netfilter'. The function names are in green, and the version numbers are in black.

**ipfwadm**

**Linux 2.0**

**ipchains**

**Linux 2.2**

**Netfilter**

**Linux 2.4**

# Мануалы по работе с firewall

- Руководство по [ipfwadm](#) на английском языке
- Руководство по [ipchains](#) на английском и [Перевод](#) на русском.
- [Руководство по iptables](#) (Iptables Tutorial 1.1.19).
- Оригинал(1.2.2) на английском на [github](#), на [одной странице](#).
- [man iptables](#) на русском языке.
- Статья на [Викиучебнике](#).



# Различные мануалы по работе



ipfwadm

ipchains

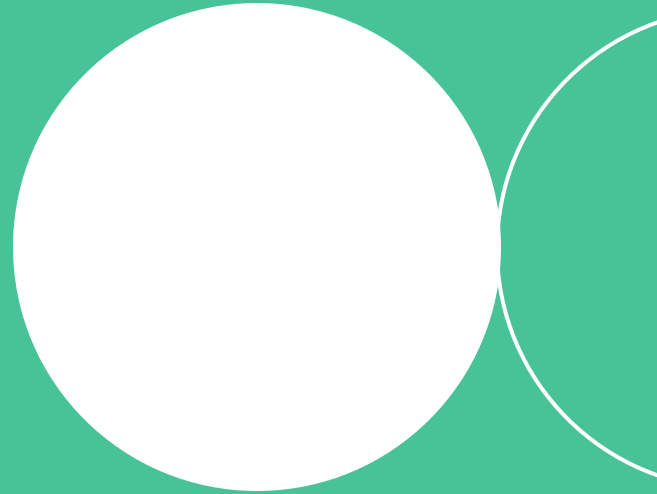
iptables

# Итоги темы

- 1 Первоначально Firewall был представлен в виде устройства, программные реализации появились гораздо позже
- 2 За 25 лет развития Firewall в Linux сменил множество реализаций (ipfwadm, ipchains, Netfilter), но основной принцип у всех похож
- 3 Хорошей практикой является брать с собой оффлайн документацию по всем реализациям Firewall

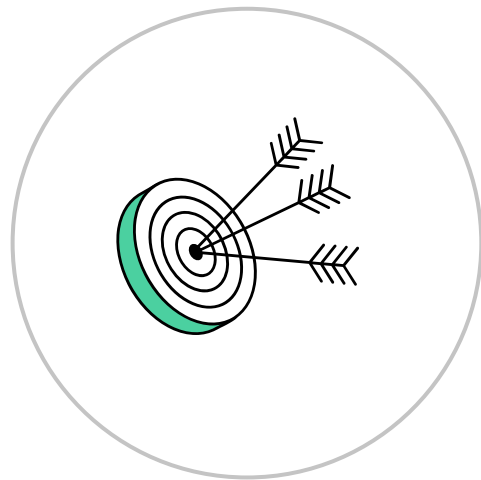


# Netfilter



# Цели темы

- Познакомится с современной реализацией файрвола в Linux
- Рассказать об особенностях реализации Netfilter
- Понять взаимодействие таблиц и цепочек в Netfilter



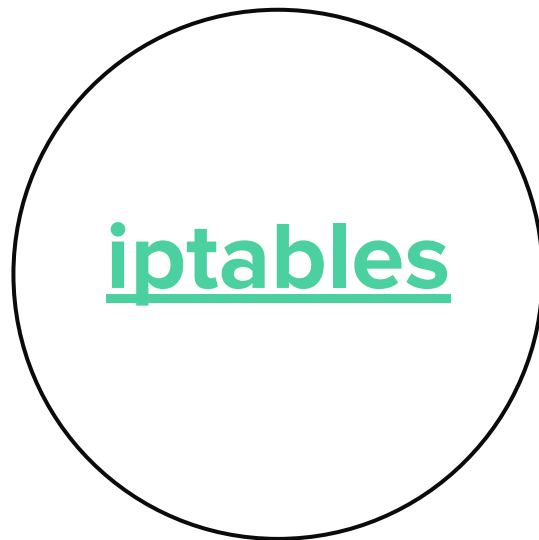


## Netfilter

межсетевой экран, встроенный в ядро Linux, начиная с версии 2.4.

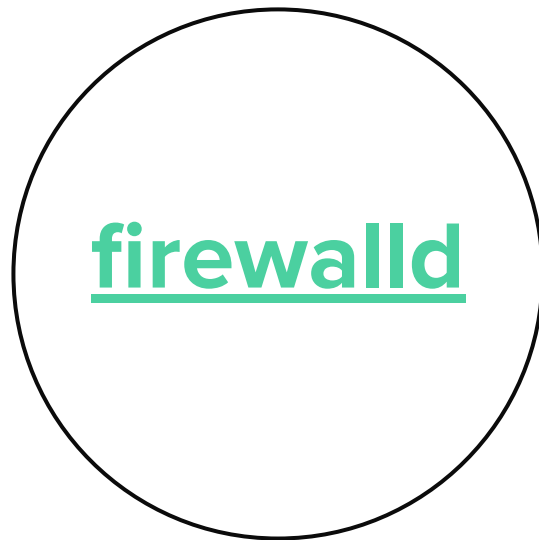


# Netfilter имеет свою утилиту



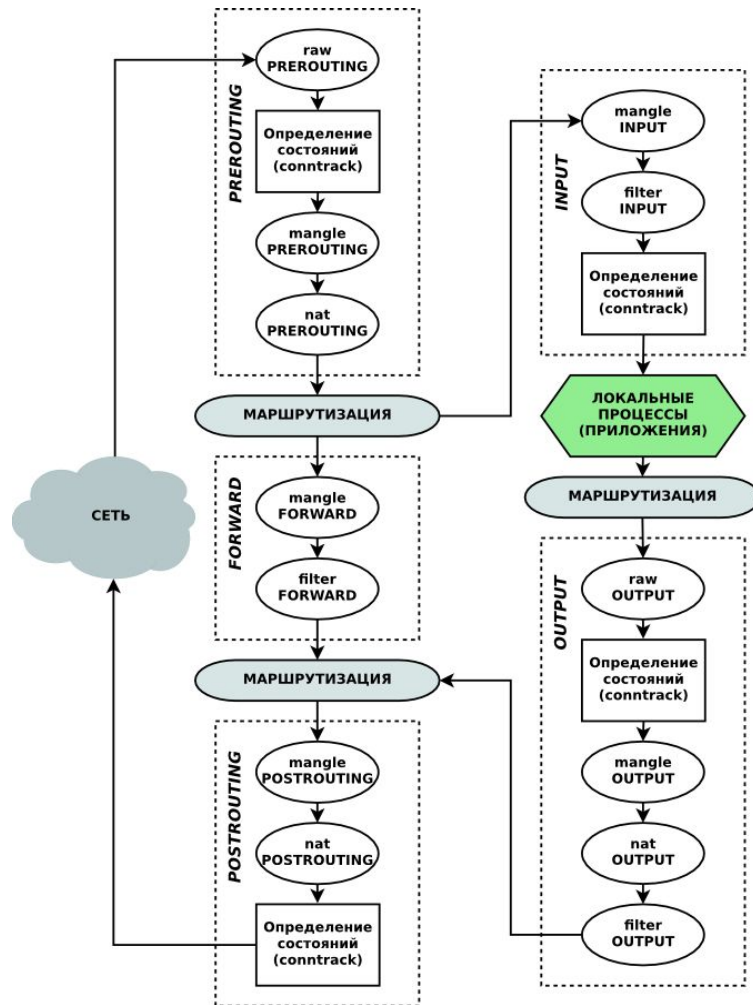
Можно создавать  
и изменять правила, которые  
фильтруют трафик

# Аналог iptables



В операционных системах  
CentOS, Fedora, OpenSUSE,  
Red Hat Enterprise Linux, SUSE  
Linux Enterprise

# Архитектура Netfilter





# Архитектура Netfilter

1

Подразумевает  
прохождение  
пакетов через  
цепочки правил

2

Каждое правило  
содержит различные  
критерии и действие  
или переход,  
выполняющиеся в  
случае полного  
соответствия пакета  
критериям

3

Отсутствие  
критериев применяет  
правило ко всем  
проходящим через  
него пакетам

# Архитектура Netfilter

Цепочка / Таблица	PREROUTING	INPUT	FORWARD	OUTPUT	POSTROUTING
filter		+	+	+	
nat	+	+		+	+
mangle	+	+	+	+	+
raw				+	
security		+	+	+	



**iptables**

**интерфейс управления netfilter**



# iptables оперирует

**Правилами**

**Цепочками**

**Таблицами**

The background features three large, light gray circles that overlap each other. The central circle is the most prominent, and the text is centered within it.

**iptables**

**является полноценным  
инструментом позволяющим  
настроить фаерволл**

# iptables

**ip6tables**

Управляет сегментом firewall, отвечающим за  
фильтрацию IPv6-пакетов (ip6\_tables)

**iptables + ip6tables = iptables**

Похожий синтаксис и  
выполняют схожие задачи

# В состав правила iptables входят



Условие

Логическое выражение на основании которого происходит анализ свойств пакета / соединения и которое определяет попадание пакета / соединения под текущее правило

Действие

Выполняется в случае соответствия пакета / соединения текущему правилу

Счетчик

Учитывает количество пакетов попавших под условие текущего правила



## Цепочки iptables

**упорядоченная последовательность правил**





# Цепочки iptables

1

## Пользовательская цепочка

Создаётся пользователем и используется только в пределах своей таблицы

2

## Базовая цепочка

Создаётся по умолчанию при создании таблицы и в отличие от пользовательской обладает действием по умолчанию

# Базовые цепочки iptables



The diagram consists of five circles arranged in two rows. The top row contains three circles labeled PREROUTING, INPUT, and FORWARD. The bottom row contains two circles labeled OUTPUT and POSTROUTING. All circles are white with black outlines and green text.

**PREROUTING**

**INPUT**

**FORWARD**

**OUTPUT**

**POSTROUTING**



## Таблица iptables

**совокупность базовых и пользовательских цепочек,  
имеющих общее назначение**



**iptables имеет 4 типа таблиц:**



The diagram consists of four identical circles arranged horizontally. Each circle has a black outline and contains a label in green, bold, sans-serif font. The labels are 'Filter', 'NUT', 'Mangle', and 'RAW' from left to right.

**Filter**

**NUT**

**Mangle**

**RAW**

# Дословный перевод SELinux

Security Enhanced  
Linux



```
graph LR; A[Security Enhanced Linux] --> B[Безопасность улучшенной Linux]
```

The diagram consists of two rounded rectangular boxes connected by a horizontal arrow pointing from left to right. The left box has a solid black border and contains the text 'Security Enhanced Linux' in a green font. The right box has a dotted black border and contains the Russian text 'Безопасность улучшенной Linux' in a black font.

Безопасность  
улучшенной Linux



## SELinux

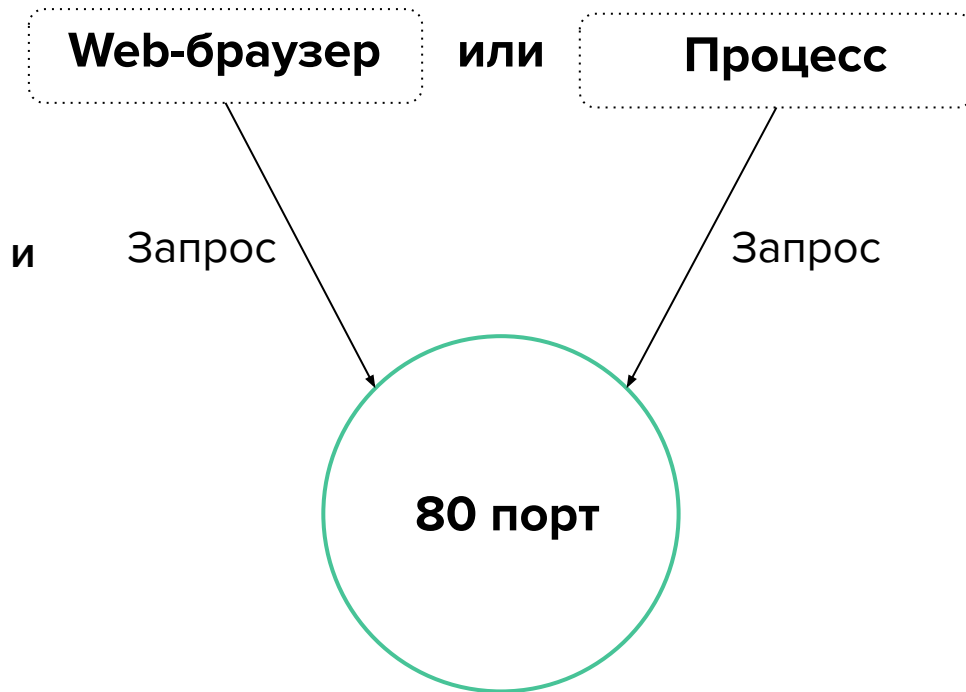
**улучшенный механизм управления доступом,  
разработанный Агентством национальной  
безопасности США для предотвращения  
злонамеренных вторжений**



**SELinux добавляет в  
Netfilter дополнительную  
таблицу security, где  
проходящим пакетам могут  
назначаться особые метки для  
предотвращения доступа  
сторонних процессов, не  
находящихся под  
контролем SELinux**

# Пример, SELinux

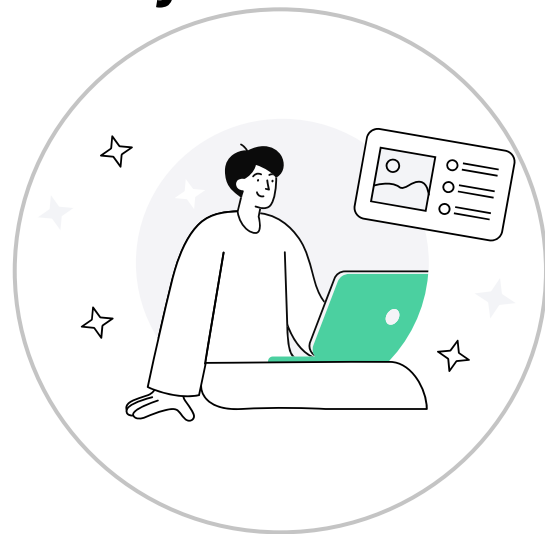
Можно указать, что запросы на 80 порт может отправлять только определенный web-браузер или процесс никто иной







**Conntrack** (англ. отслеживание соединения)  
специальная подсистема, отслеживающая состояния  
соединений и позволяющая использовать эту  
информацию при принятии решений о  
судьбе отдельных пакетов



# Состояния соединений

## NEW

пакет является  
первым в соединении

## ESTABLISHED

пакет относится к уже  
установленному  
соединению

## INVALID

установить  
принадлежность  
пакета не удалось

## RELATED

пакет открывает новое  
соединение, логически  
связанное с уже  
установленными

## UNTRACKED

отслеживание состояния  
соединения для данного  
пакета было отключено

# Дословный перевод host

Host



Хозяин



## Host

**любое устройство подключенное к сети TCP/IP,  
принимающее или создающее подключения**



# Дословный перевод localhost

**Localhost**



**Локальный хост**



## Localhost

официально зарезервированное доменное имя для IP-адресов 127.0.0.1/8



## Localhost

**с помощью специального  
сетевого интерфейса  
«внутренней петли»  
(loopback) позволяет создавать  
сети, состоящие из одного  
компьютера**

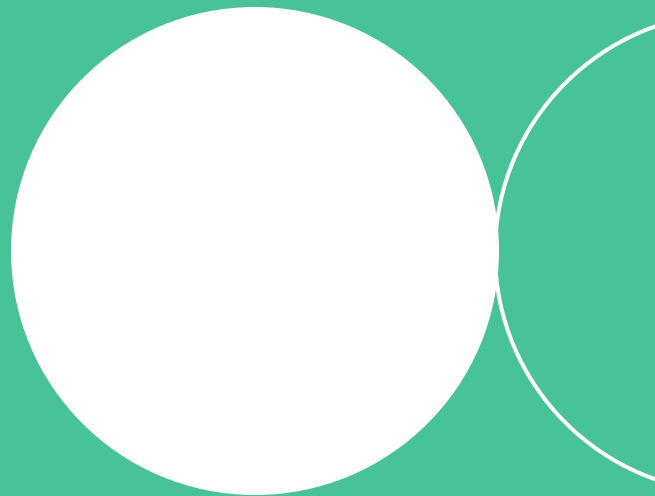
# Итоги темы

- 1 Netfilter состоит из пяти цепочек и пяти таблиц
- 2 Для операций с метаданными используют таблицу mangle, для операции с адресами - nat, для фильтрации - таблица filter
- 3 Деление правил по цепочкам условно и жестко не контролируется, однако размещение правила в не соответствующей таблице может привести к сбоям и ошибкам обработки



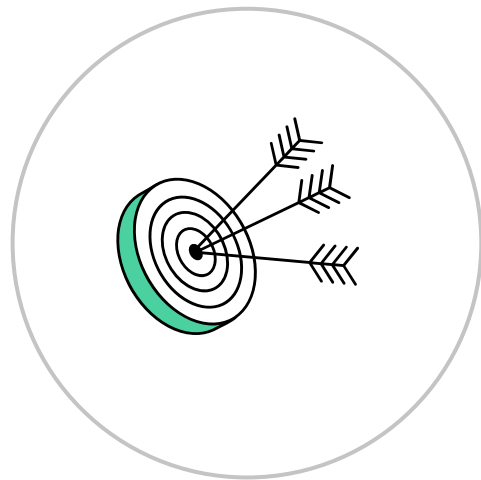


# Цепочки iptables: PREROUTING, INPUT, FORWARD

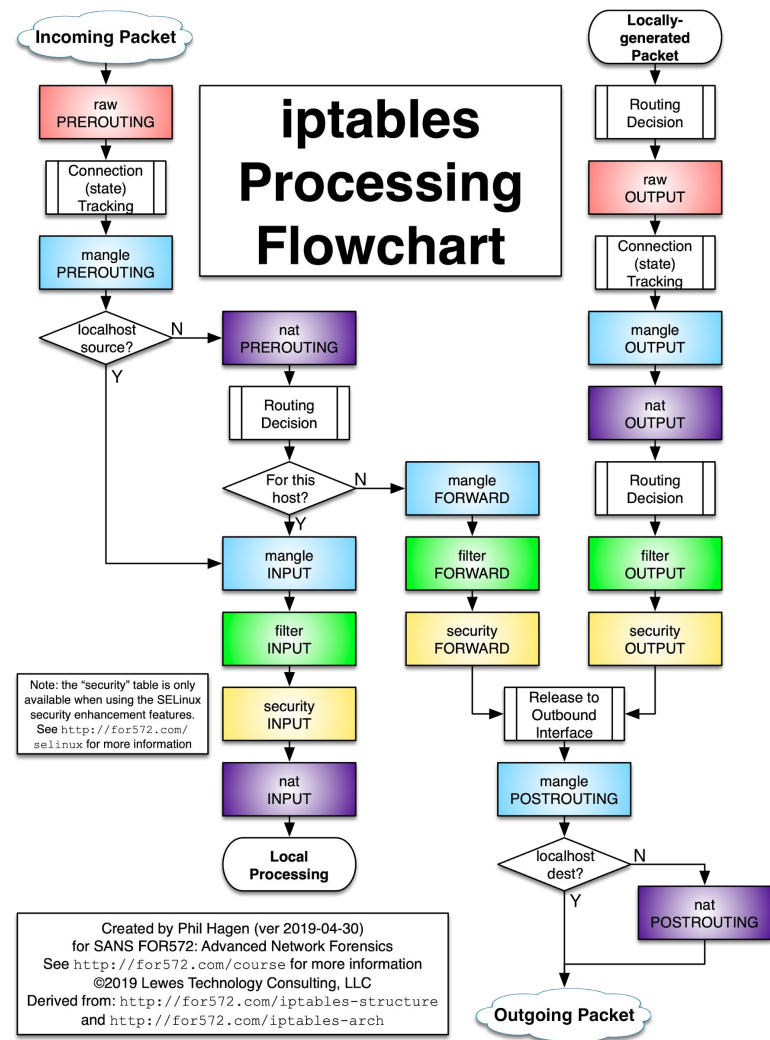


# Цели темы

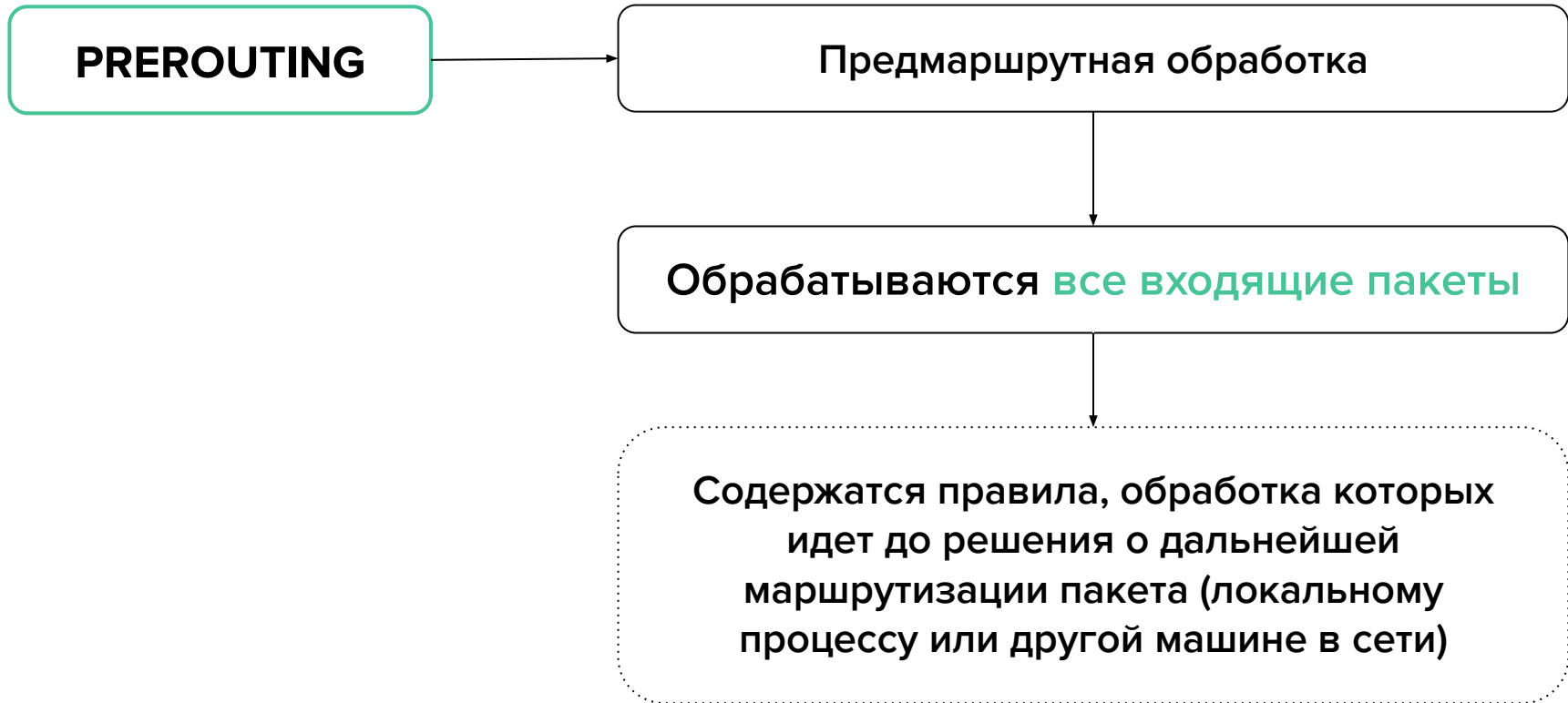
- Узнать о цепочках PREROUTING, INPUT, FORWARD
- Выяснить, каким образом принимается решение о прохождении пакетом этих цепочек
- Понять, какие операции с трафиком может выполнять firewall в этих цепочках



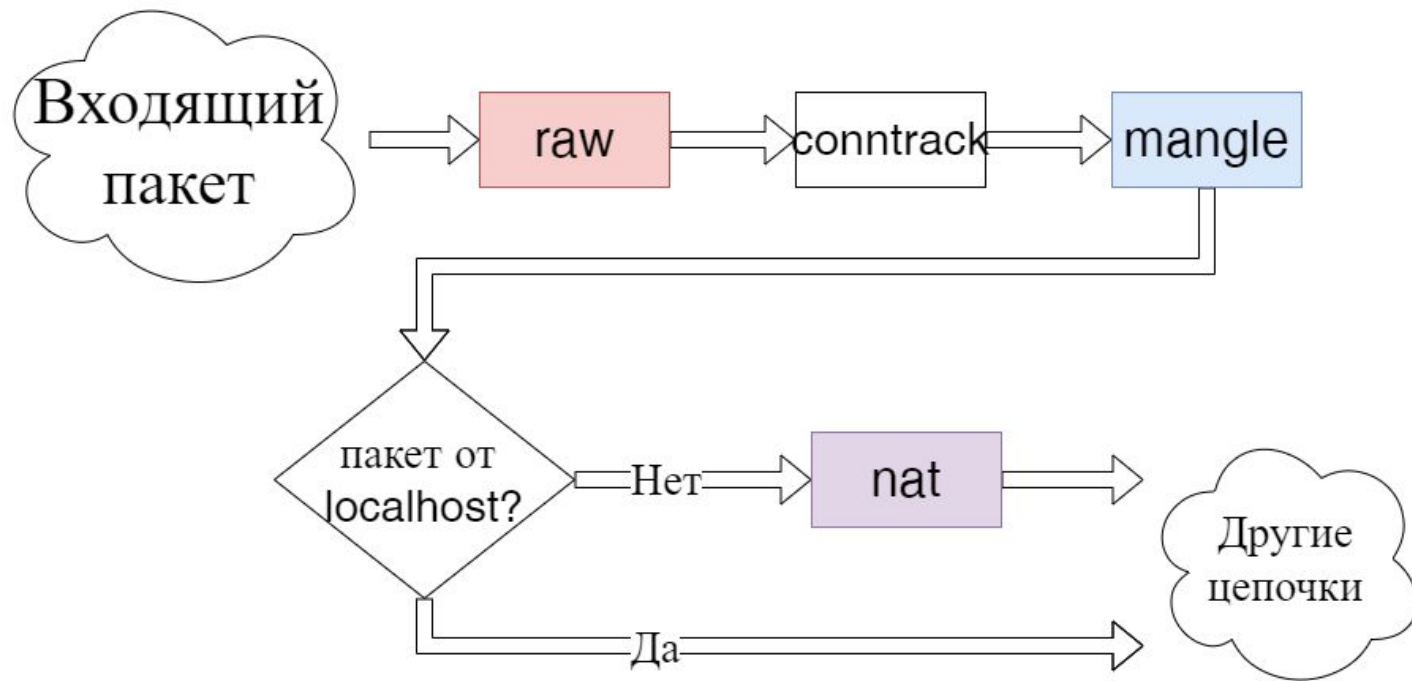
# Цепочка PREROUTING



# Цепочка PREROUTING



# Цепочка PREROUTING



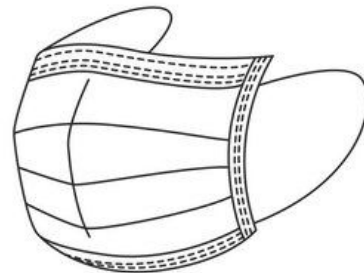
# Цепочка PREROUTING: аналогия

В больнице предписание:

нужно надеть маску и  
бахилы, прежде чем  
подойти к регистратуре



**БАХИЛЫ**



**МАСКА**

# В цепочке PREROUTING помещаются правила:

- Для управления отслеживанием (таблица raw): отменить, настроить, ограничить отслеживание и т.д
- Если необходимо модифицировать пакет до маршрутизации (mangle), например изменить поле TOS (IPv4), DSCP, TTL. Также можно сделать маркировку пакета или соединения
- Для изменения адреса получателя в таблице nat: как IP-адреса через (Destination Network Address Translation) так и порта (с помощью действия REDIRECT)

# Пример PREROUTING

Скрытие порта приложения с помощью таблицы nat (REDIRECT с внешнего порта 12345 на 22 в локальной сети)





# Пример PREROUTING

Использование двух провайдеров с разделением по спискам пользователей, кто каким провайдером пользуется (маркировка с помощью таблицы mangle)

Провайдер

A



Пользователь:

Катя  
Сергей  
Настя  
Коля

Провайдер

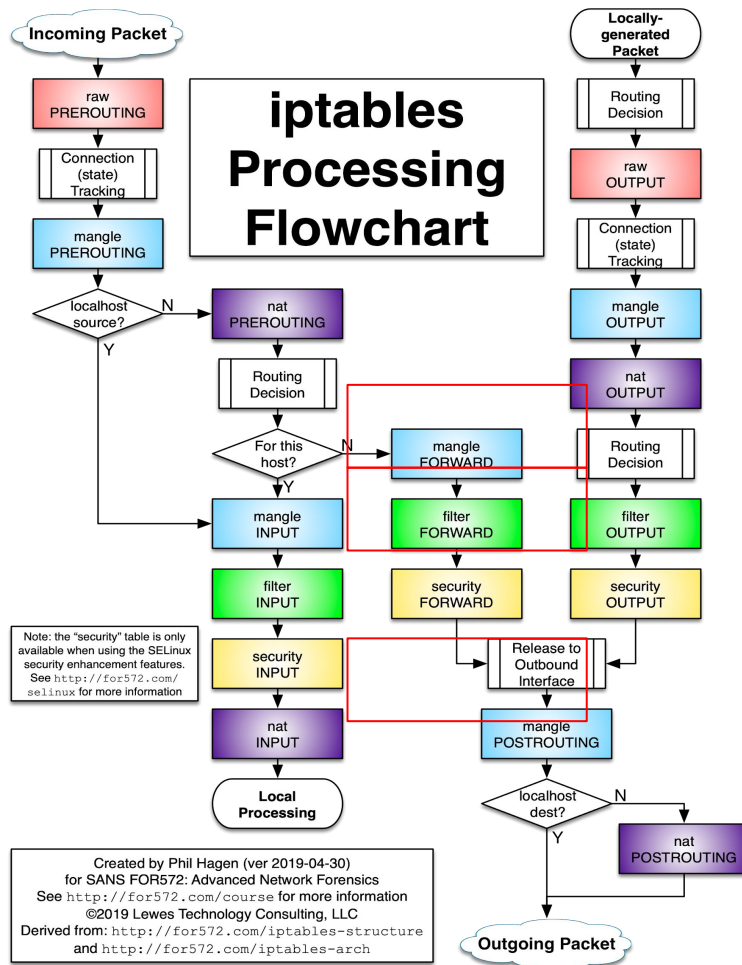
B



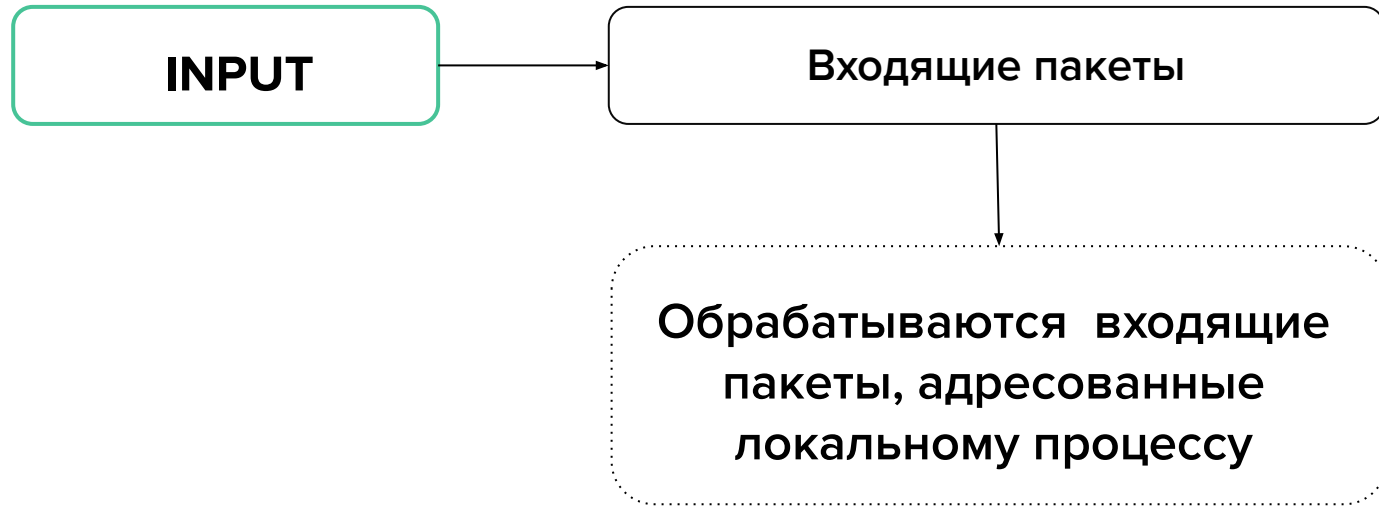
Пользователь:

Маша  
Витя  
Даша  
Паша

# Цепочка INPUT



# Цепочка INPUT

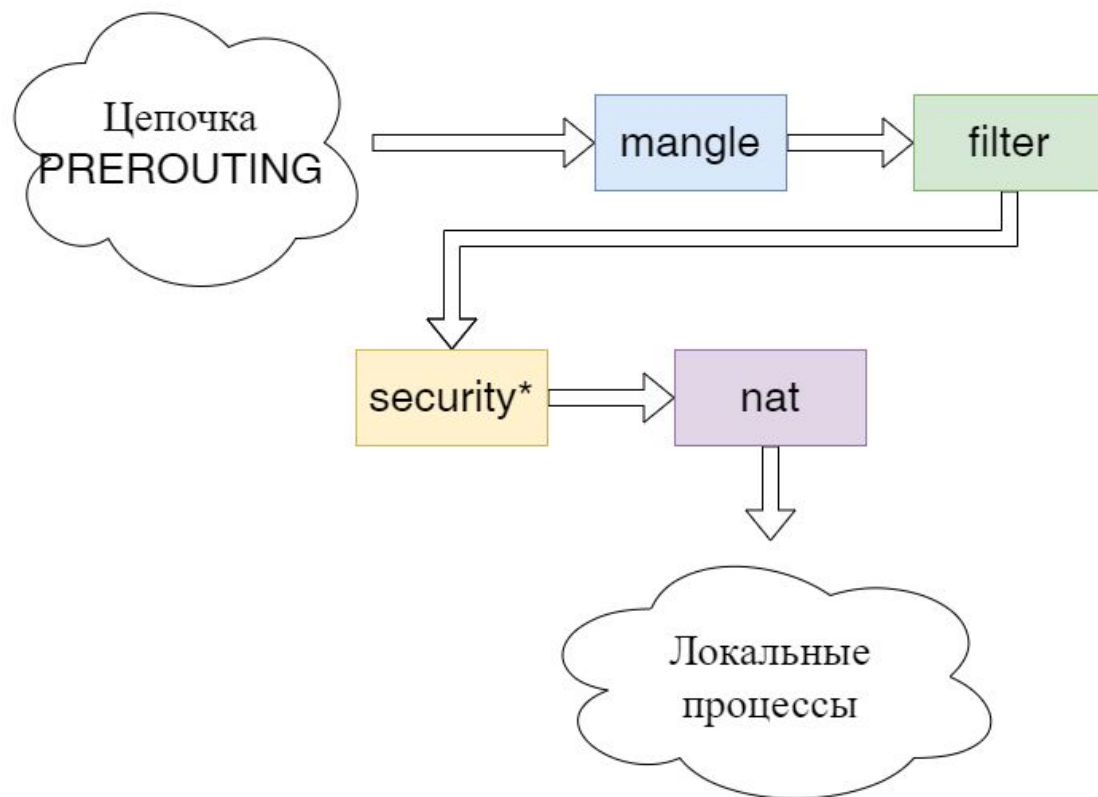


# Цепочка INPUT: аналогия

Правила поведения в  
поликлинике при  
прохождении  
диспансеризации



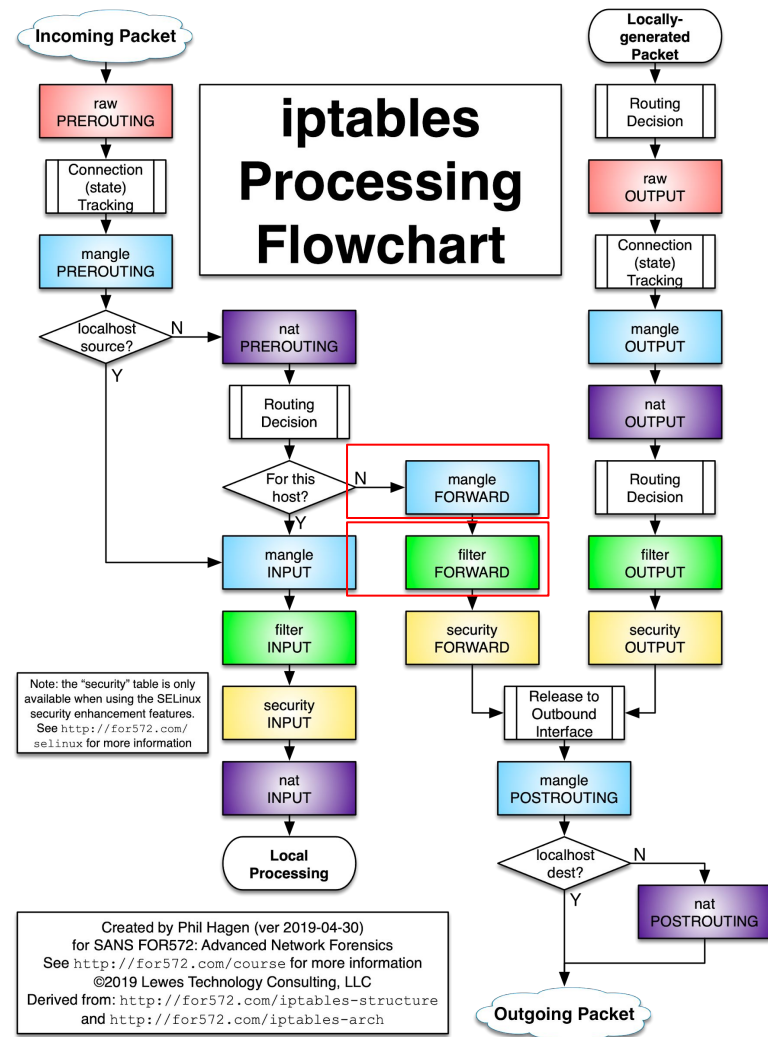
# Цепочка INPUT



# Что происходит при прохождении пакета через цепочку правил?

- Изменение заголовка пакета, прежде чем он попадет к локальному процессу (таблица mangle)
- Фильтрация входящего трафика (таблица filter)
- Передача специфичным системам принудительного контроля доступа (security) Данная таблица появляется только с использованием возможностей SELinux
- Иногда необходимо обработать два идентичных потока из разных зон, когда получателем выступает машина с фаерволом. В таких случаях в цепочке INPUT используется таблица nat

# Цепочка FORWARD



# Цепочка FORWARD



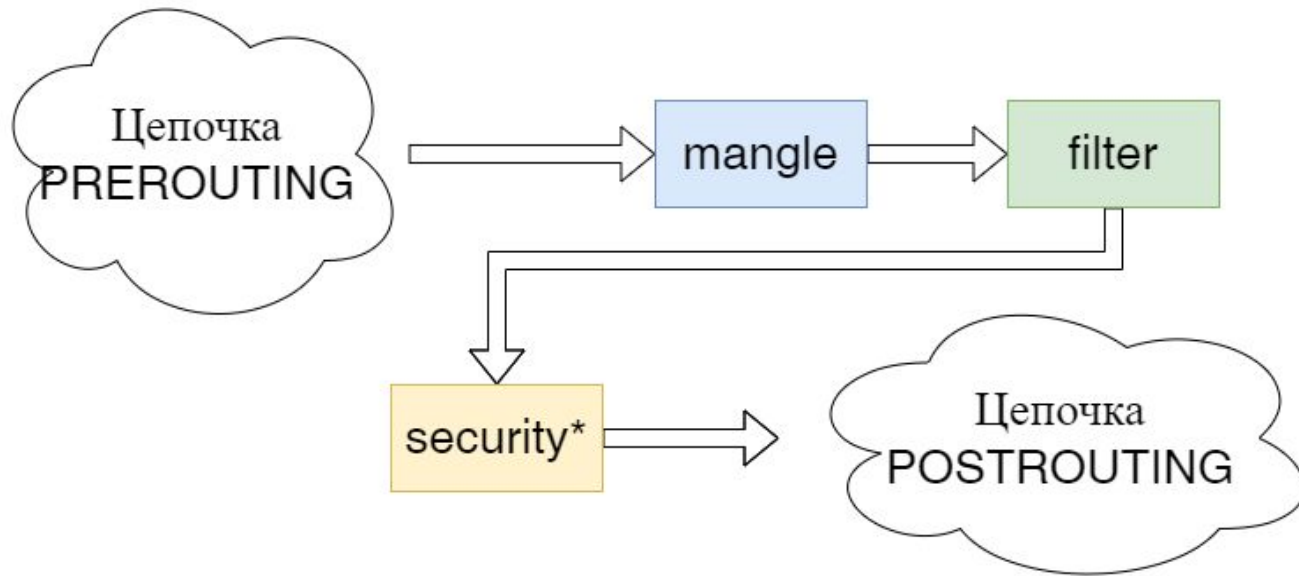


# Цепочка FORWARD: аналогия

Проходная на режимную  
территорию



# Цепочка FORWARD



# Правила в цепочке FORWARD

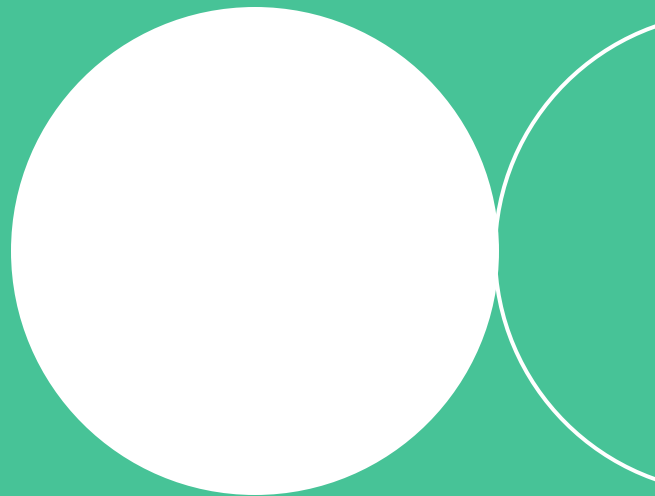
- В исключительных случаях вносить изменение в заголовок транзитного пакета
- Фильтрация трафика, идущего в обоих направлениях (в локальную и внешнюю сеть)

# Итоги темы

- 1 Все входящие пакеты, независимо от их источника, обязательно проходят через цепочку PREROUTING. В этой цепочке можно отключать отслеживание conntrack через таблицу raw
- 2 В цепочке INPUT обрабатываются пакеты, предназначенные для локальной машины. В большинстве случаев достаточно правил в таблицах mangle и filter
- 3 Обработку промежуточного трафика обеспечивает цепочка FORWARD. Необходимо тщательно проверять правила этой цепочки, потому что они применяются как к исходящему клиентскому, так и входящему внешнему потоку

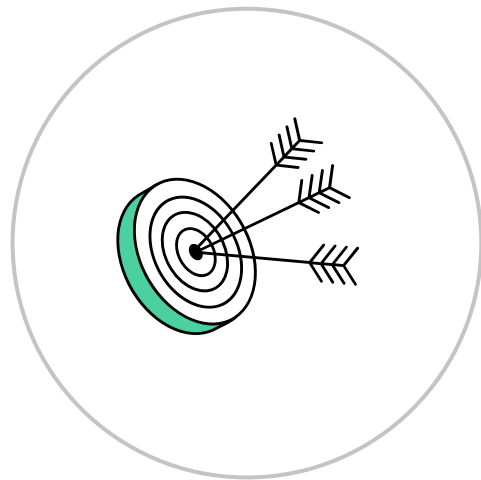


# Цепочки iptables: OUTPUT, POSTROUTING

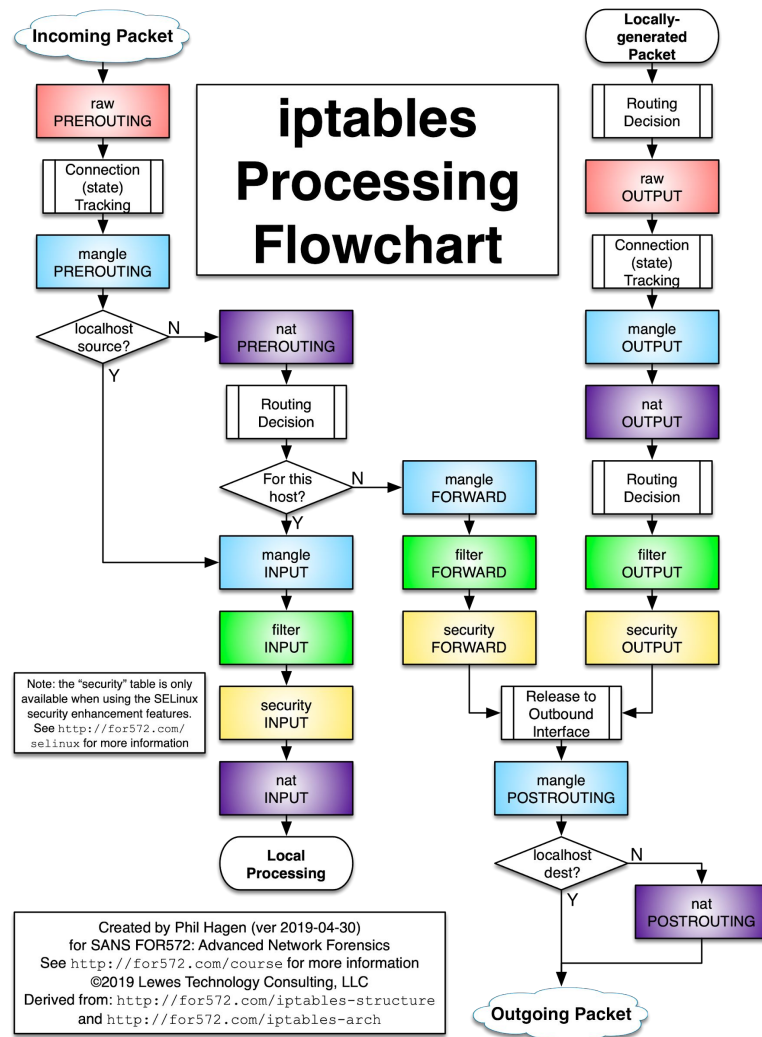


# Цели видео

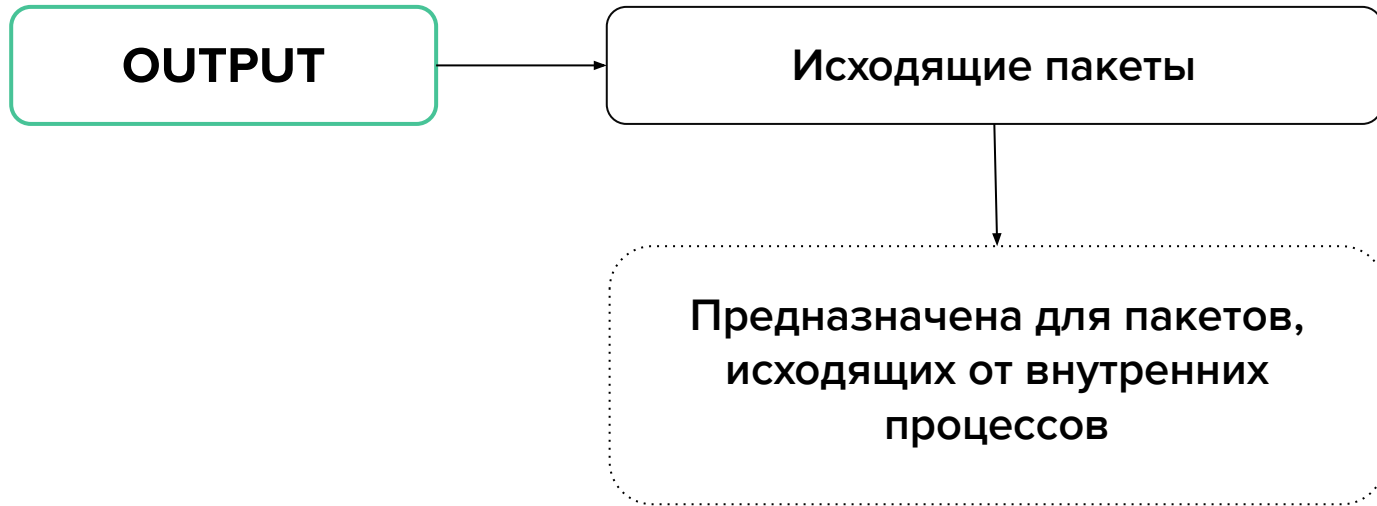
- Узнать о прохождении пакетами цепочек OUTPUT, POSTROUTING
- Разобраться с особенностями управления трафиком, применяемыми на данных этапах



# Цепочка OUTPUT

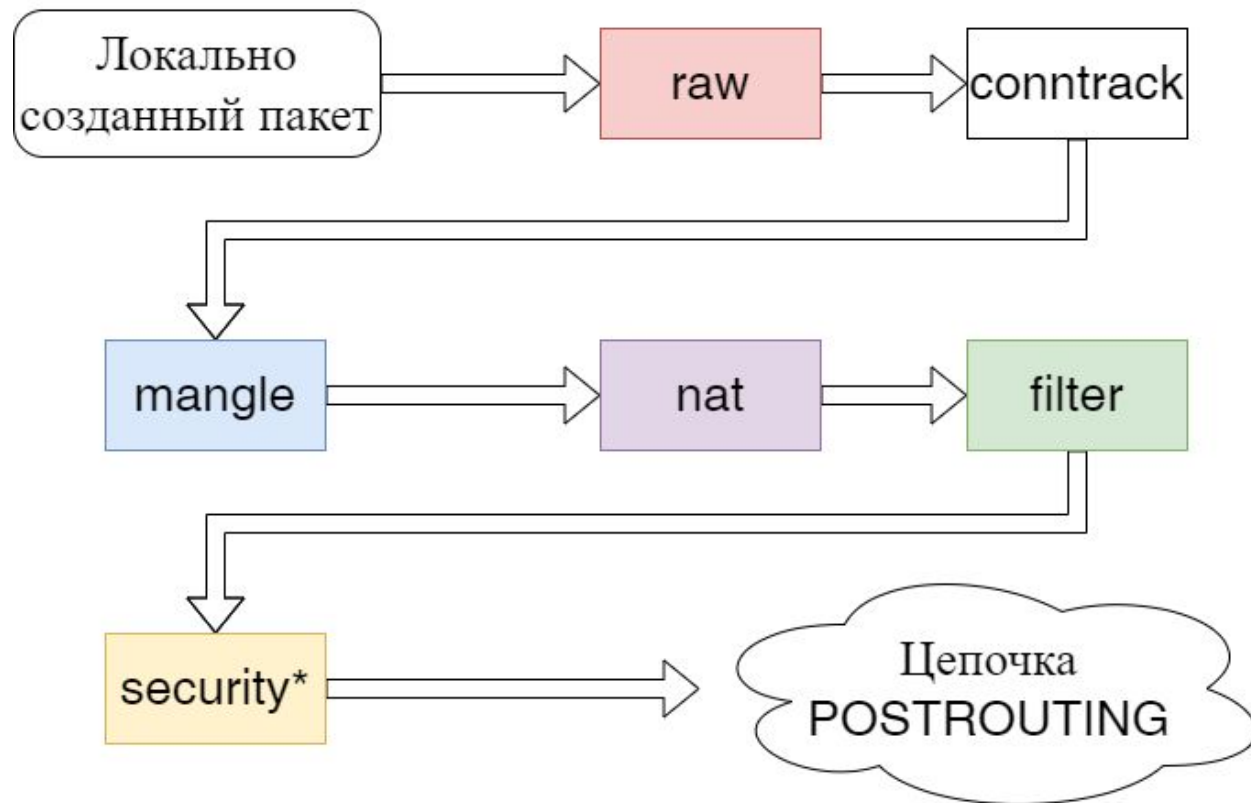


# Цепочка OUTPUT





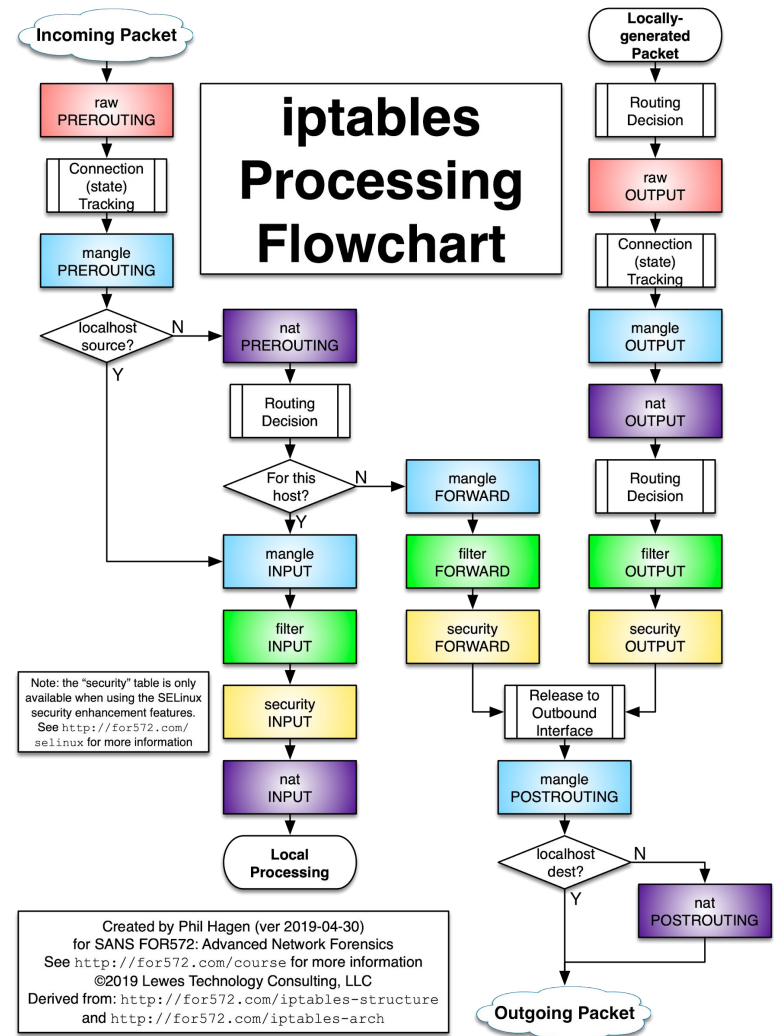
# Цепочка OUTPUT



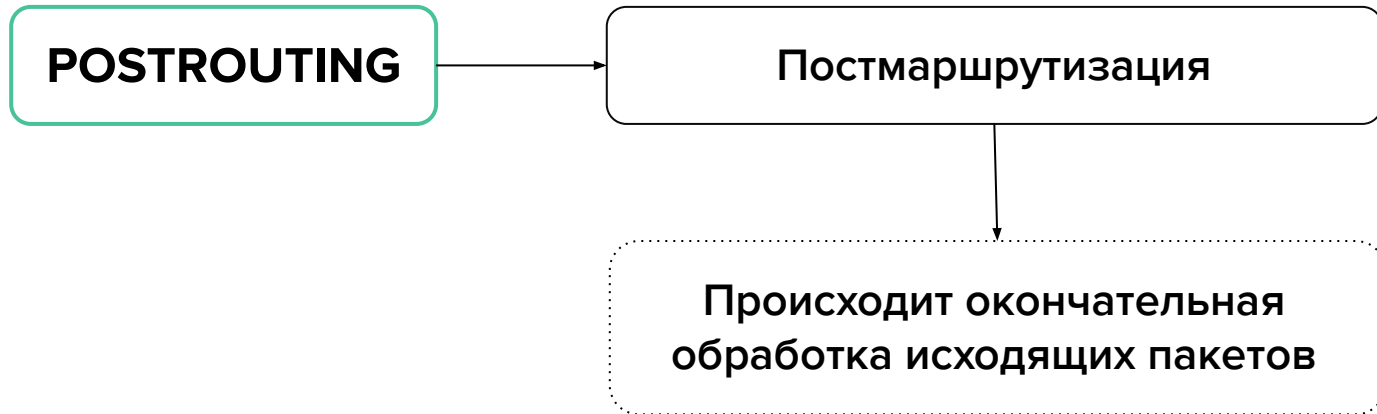
# Правила цепочки OUTPUT предназначены для:

- Управления отслеживанием (таблица raw), как то: задать зону conntrack для пакета, отменить, настроить, ограничить отслеживание и т.д.
- Внесение изменений в заголовок исходящего пакета (таблица mangle)
- Повторения в случае необходимости подмены адресов (IP, порт TCP) для локально созданных пакетов (таблица nat)
- Обычной (таблица filter) и усиленной (таблица security) фильтрации исходящих пакетов

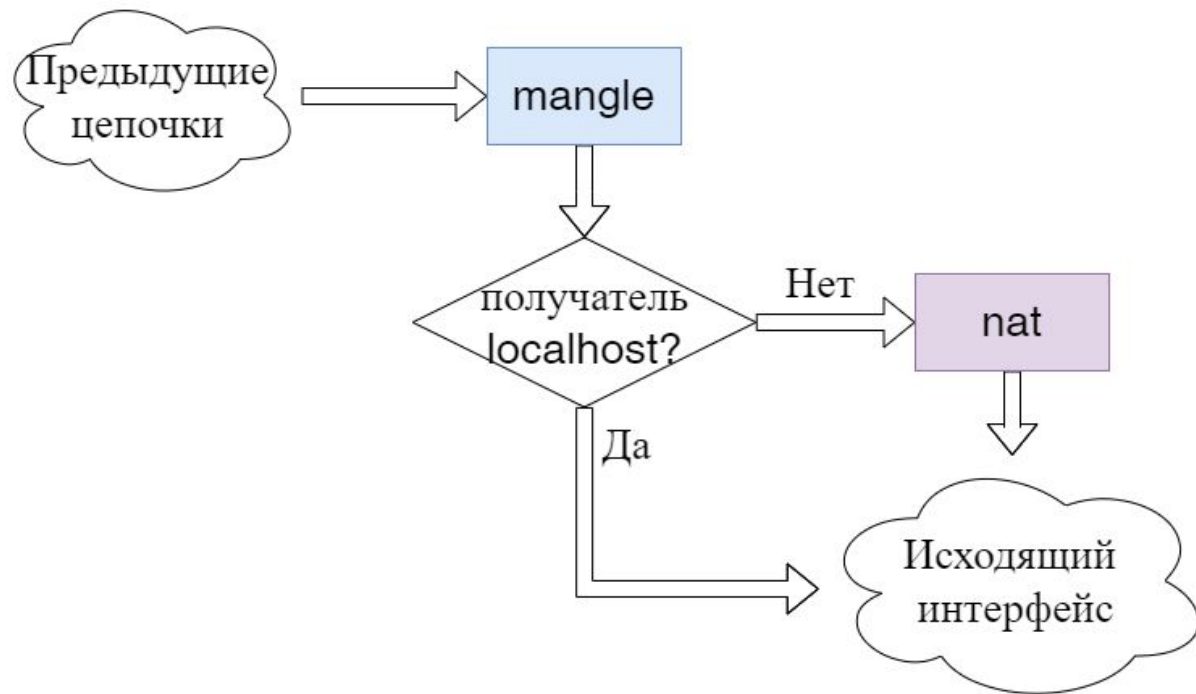
# Цепочка POSTROUTING



# Цепочка POSTROUTING



# Цепочка POSTROUTING



# Правила цепочки POSTROUTING предназначены для:

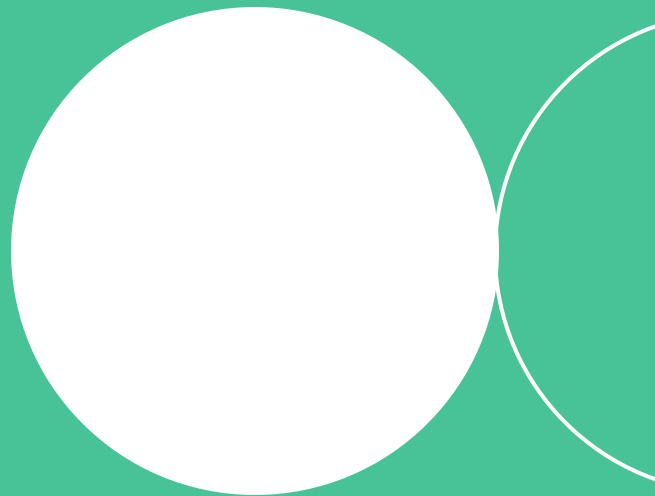
- Внесения изменения в заголовок исходящего или транзитного пакета/сегмента уже после того, как принято последнее решение о маршрутизации (таблица mangle)
- Замены адреса отправителя (Source Network Address Translation), проводить операции маскарадинга (таблица nat)

# Итоги

- 1 Правила цепочки OUTPUT обрабатывают все пакеты, созданные на локальной машине.
- 2 Через правила цепочки POSTROUTING проходят весь исходящий и транзитный трафик для получателей во внутренней, внешней сетях, а также сгенерированный локальными процессами и предназначенный другим локальным процессам



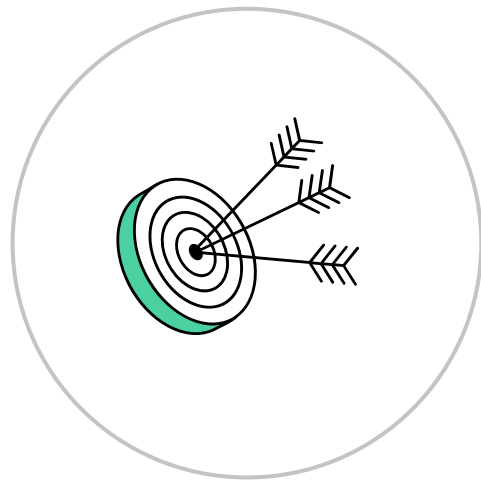
# Синтаксис iptables





# Цели видео

- Ознакомиться с синтаксисом iptables
- Узнать об особенностях добавлений правил
- Обзорно поговорить об аналоге iptables для L2-сетей



## Утилита netstat

**позволяет смотреть  
состояния соединений, таблиц  
маршрутизации, чисто сетевых  
интерфейсов и статистику  
по протоколам**

# Утилита netstat позволяет

Посмотреть слушает ли  
сервер порт 22

```
netstat -an | grep “:22”
```

# Утилита netstat позволяет

Посмотреть слушает ли  
сервер порт 22

```
netstat -an | grep ":22"
```

Посмотреть все сокеты  
с состоянием LISTEN

```
netstat -l
```

# Утилита netstat позволяет

Посмотреть слушает ли  
сервер порт 22

```
netstat -an | grep ":22"
```

Посмотреть все сокеты  
с состоянием LISTEN

```
netstat -l
```

Узнать статистику для  
каждого протокола

```
netstat -s
```

# Утилита netstat позволяет

Посмотреть слушает ли  
сервер порт 22

`netstat -an | grep ":22"`

Посмотреть все сокеты  
с состоянием LISTEN


`netstat -l`

Узнать статистику для  
каждого протокола

`netstat -s`

Посмотреть руководство  
по netstat

`man netstat`



**Для работы с iptables  
всегда необходимы  
повышенные привилегии**

# Шаблон работы с iptables

```
iptables [-t table] command [match] [target/jump]
```




# Шаблон работы с iptables

`iptables [-t table] command [match] [target/jump]`

**-t** – указывает на таблицу (raw, mangle, nat, security), по умолчанию без указания параметра выбирается таблица filter

**[match]** – задает критерии проверки, по которым определяется подпадает ли пакет под действие этого правила или нет

**[target]** – указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле



**При прохождении пакетом  
цепочек, в которых Netfilter  
ищет совпадение с правилом,  
необходимо придерживаться  
принципов**

# Принципы в работе с iptables

1

Чем выше правило (меньше порядковый номер), тем раньше оно будет обработано, поэтому порядок правил имеет огромное значение

2

Если ни одно правило не подошло, будет выполнено действие по умолчанию

# Важно

При начальной настройке всегда нужно задавать политику обработки пакетов по умолчанию для каждой цепочки

Например:

```
sudo iptables -P INPUT DROP
```

```
sudo iptables -P FORWARD DROP
```

**При работе с iptables  
всегда приходится обращаться  
к просмотру содержимого  
таблиц, для получения  
информации о текущих  
настройках**

# Команда для просмотра таблиц iptables

1

```
sudo iptables -nvL -t raw
```

3

```
sudo iptables -nvL -t nat
```

2


```
sudo iptables -nvL -t mangle
```

4


```
sudo iptables -nvL -t filter
```

Если не указать имя таблицы, команда выдаст содержимое таблицы **filter**


# Способы управления порядком правил

`iptables -A {-t таблица} ЦЕПОЧКА[...]`  `append`, добавляет правило в конец цепочки

`iptables -I {-t таблица} ЦЕПОЧКА {номер правила}[...]`  `insert`, добавляет правило перед правилом с указанным номером

`iptables -R {-t таблица} ЦЕПОЧКА {номер правила}[...]`  `replace`, заменяет правило с указанным номером

`iptables -D {-t таблица} ЦЕПОЧКА {номер правила}[...]`  `delete`, удаляет правило с указанным номером из цепочки

`iptables -F {-t таблица} ЦЕПОЧКА`  `flush`, удаляет все правила из таблицы и цепочки

# Пример добавления правила iptables

Необходимо разрешить подключение к локальной машине на порт 22 из локальной сети 192.168.0.0/24

Используем команду:

```
sudo iptables -A INPUT -p tcp --dport 22 -m state \  
--state NEW,ESTABLISHED -s 192.168.0.0/24 -j ACCEPT
```



# Пример добавления правила iptables

Используем команду:

```
sudo iptables -A INPUT -p tcp --dport 22 -m state \  
--state NEW,ESTABLISHED -s 192.168.0.0/24 -j ACCEPT
```

**-A INPUT** – (append, добавить) указывает цепочку (например, INPUT ) для добавления правила

**-p tcp** – указываем сетевой протокол (например, tcp или udp )

**--dport 22** – порт назначения пакетов

**-m state** – критерий, свойство пакета, которое мы хотим сопоставить (например, state )

**--state NEW, ESTABLISHED** – состояние(я) пакета для соответствия

**-s 192.168.0.0/24** – (source, источник) IP-адрес и маска источника, из которого исходят пакеты

**-j ACCEPT** – цель или что делать с пакетами (например, ACCEPT, DROP, REJECT и т. д.)

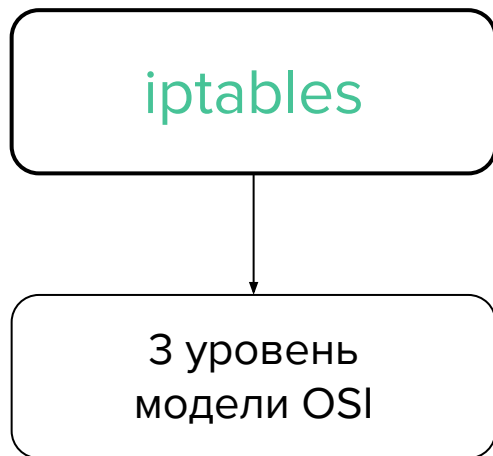
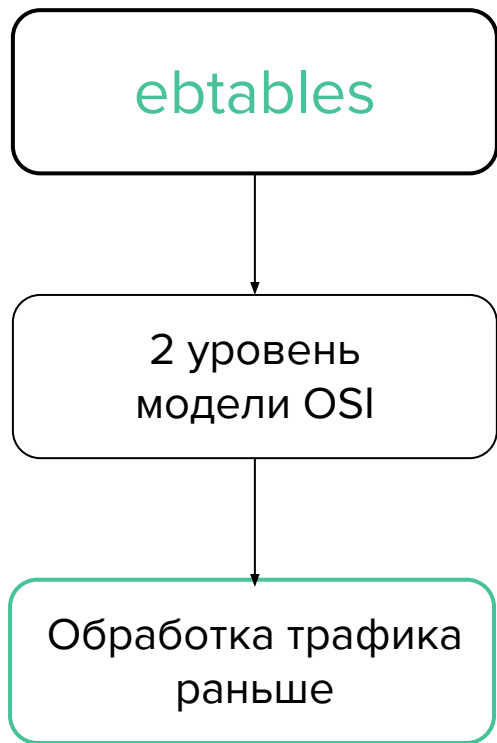


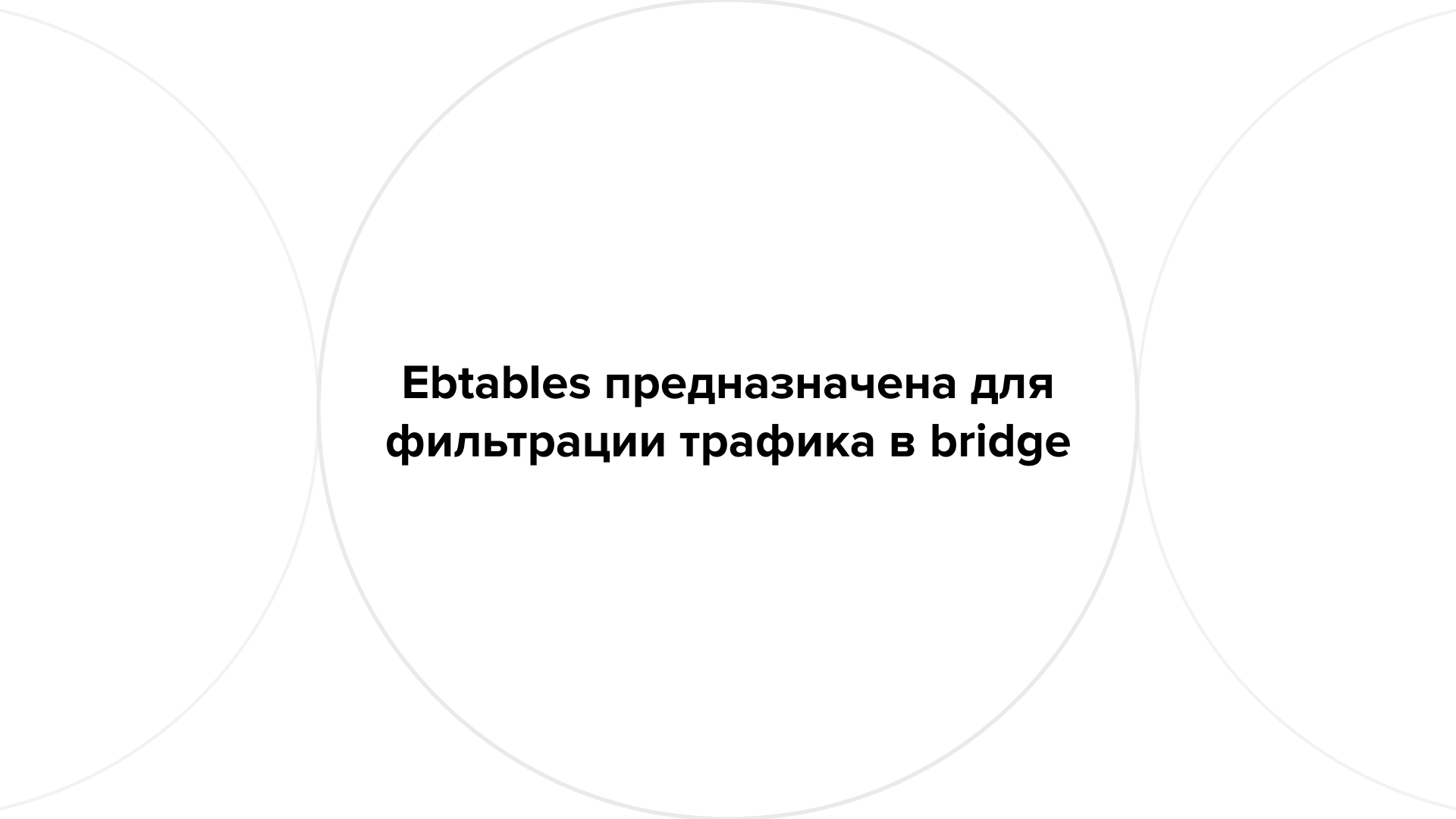
## Ebtables

**средство для фильтрации пакетов для программных мостов Linux, работает преимущественно на втором (канальном) уровне модели OSI**



# Обработка трафика



The background features three large, light gray circles that overlap each other. The central circle is the most prominent, and the text is centered within it. The other two circles are partially visible on the left and right sides of the frame.

**Ebtables предназначена для  
фильтрации трафика в bridge**

# Например

Чтобы отбросить трафик от конкретного MAC адреса в ebtables необходима следующая команда:

```
ebtables -A INPUT -s 08:00:27:47:88:CE -j DROP
```

Вариант, который мы использовали в iptables:

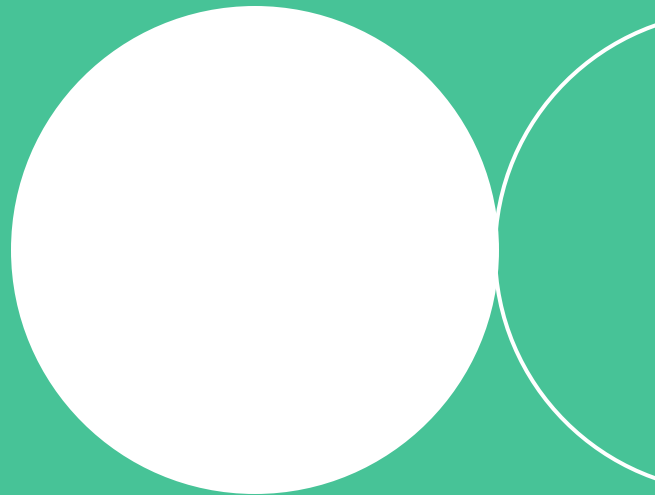
```
sudo iptables -A INPUT -m mac --mac-source 08:00:27:47:88:CE -j DROP
```

# Итоги темы

- 1 Всегда важно устанавливать правила по умолчанию для тех пакетов, которые не будут соответствовать ни одному правилу в цепочке
- 2 Если при операции с цепочкой не указать таблицу назначения, действие будет применено к таблице filter
- 3 При добавлении правил необходимо помнить о дуплексной природе взаимодействия и при необходимости добавлять правила и для входящего и для исходящего потока

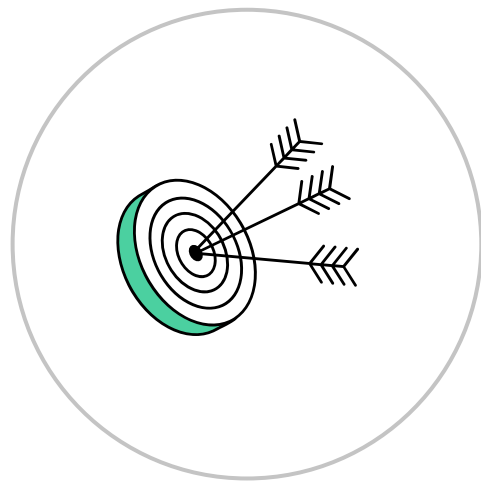


# Настройка доступа по порту с помощью iptables



# Цели темы

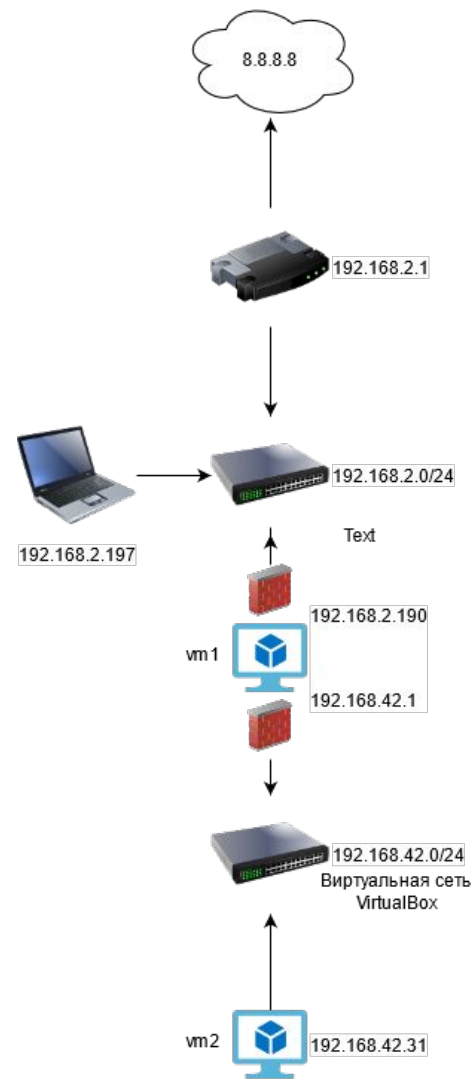
- Получить практический навык настройки файрвола
- Объединить на практике работу NAT и Firewall
- Познакомится с различными нюансами работы с iptables





# Исходные данные

- Ноутбук подключён к сети 192.168.2.0 и на нём установлен VirtualBox
- vm1 подключена к сети 192.168.2.0 (enp0s3) и к виртуальной сети 192.168.42.0 (enp0s8)
- vm2 подключена только к сети 192.168.42.0
- Сеть 192.168.2.0 – выход в интернет
- Сеть 192.168.42.0 – виртуальная сеть VirtualBox
- vm1 в сети 42.0 имеет IP 192.168.42.1
- vm2 в сети 42.0 имеет IP 192.168.42.31



# Блокируем порт извне

К примеру, мы хотим закрыть доступ извне к какому-то порту.

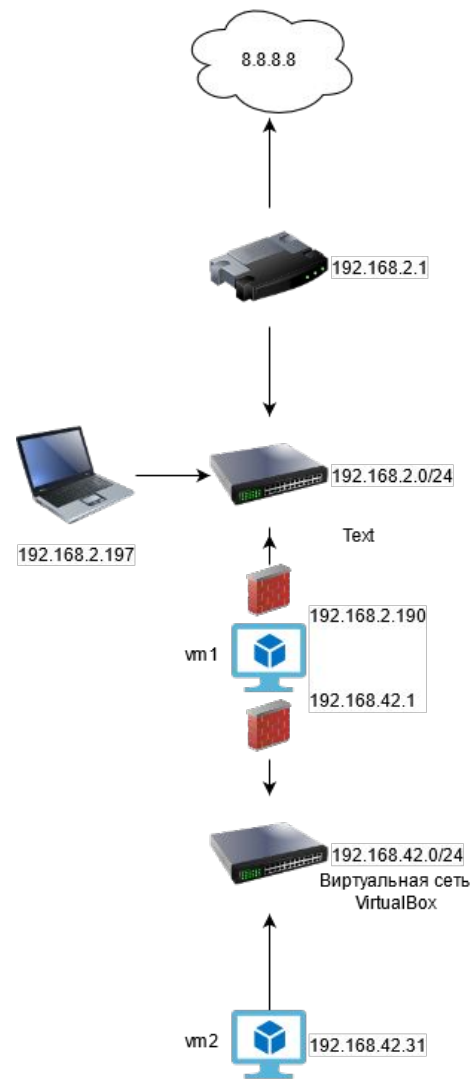
Для этого в таблицу INPUT нам необходимо добавить условие, и соответствующее для него действие:

```
sudo iptables -A INPUT -p tcp --dport 22 -j DROP
```

После этого любое подключение из любого источника к текущему хосту на порт 22 будет заблокировано

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

После этого любое подключение из любого источника к текущему хосту на порт 22 будет разрешено

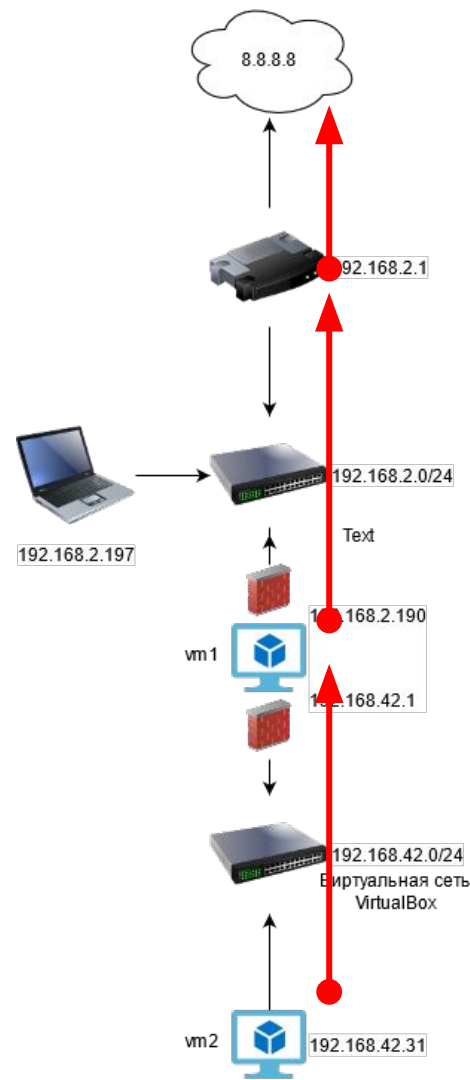


# Настраиваем NAT маскерадинг

Сделаем так, чтобы трафик с vm2 выходил в интернет

```
# На vm1
# Включаем IP форвардинг в ядре Linux
cat /proc/sys/net/ipv4/ip_forward # Проверяем включён ли ip форвардинг
sudo nano /proc/sys/net/ipv4/ip_forward # Изменяем на 1 если было 0
# Разрешаем форвардинг уже установленных соединений
sudo iptables -A FORWARD -j ACCEPT -m conntrack --ctstate \
ESTABLISHED,RELATED -m comment --comment "established traffic"
# Разрешаем форвардинг новых соединений с интерфейса enp0s8 на
enp0s3
sudo iptables -A FORWARD -j ACCEPT -i enp0s8 -o enp0s3 \
-m comment --comment "forward"
# Включаем маскерадинг всех соединений идущих через enp0s3
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE \
-m comment --comment "masquerade"
# Чтобы ip_forward сохранился после перезагрузки, пригодится команда:
sudo sysctl -w net.ipv4.ip_forward=1

# На vm2
ping 8.8.8.8
```

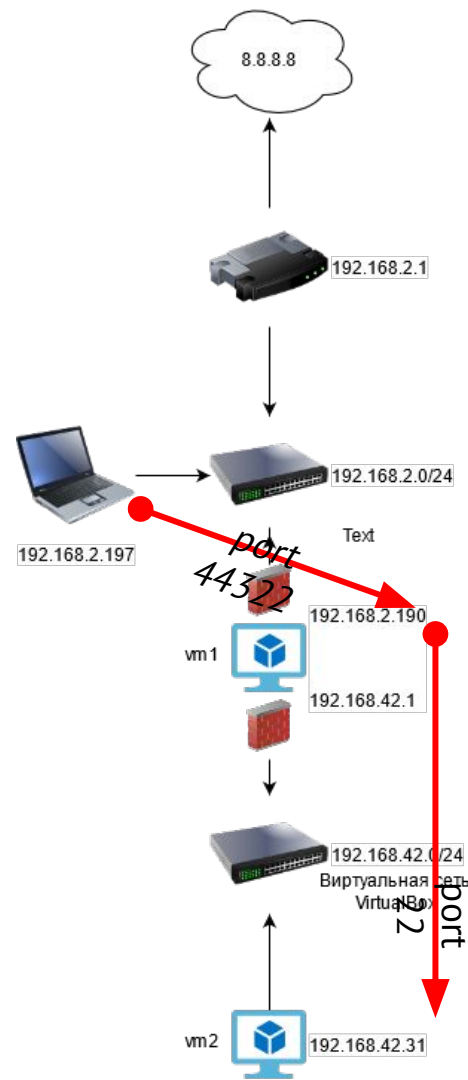


# Пробрасываем порт

Сделаем так, чтобы порт 22 **vm2** был доступен ноутбуку по адресу **192.168.2.190:44322**

```
#Пробрасываем трафик с “публичного” IP шлюза порт 44322 на IP адрес 192.168.42.31 порт 22
sudo iptables -t nat -A PREROUTING -d 192.168.2.190 -p tcp \
--dport 44322 -j DNAT --to-destination 192.168.42.31:22
# Разрешаем пропускать трафик с enp0s3 через enp0s8 на 192.168.42.31 порт 22
sudo iptables -I FORWARD 1 -i enp0s3 -o enp0s8 -d 192.168.42.31 \
-p tcp -m tcp --dport 22 -j ACCEPT
```

На ноутбуке остаётся через **Putty** подключиться по **SSH** на IP **192.168.2.190** на порт **44322**



# iptables блокировка по MAC

Обозначим критерием блокировки трафика – MAC адрес. В случае если аппаратный адрес сетевой карты подключающегося устройства будет соответствовать указанному в правиле, оно будет отбрасывать трафик

```
#Отбрасываем трафик если он исходит от MAC адреса 08:00:27:47:88:ce  
sudo iptables -A INPUT -m mac --mac-source 08:00:27:47:88:CE -j DROP  
#Отбрасываем трафик если он исходит НЕ от MAC адреса 08:00:27:47:88:ce  
sudo iptables -A INPUT -m mac ! --mac-source 08:00:27:47:88:CE -j DROP
```

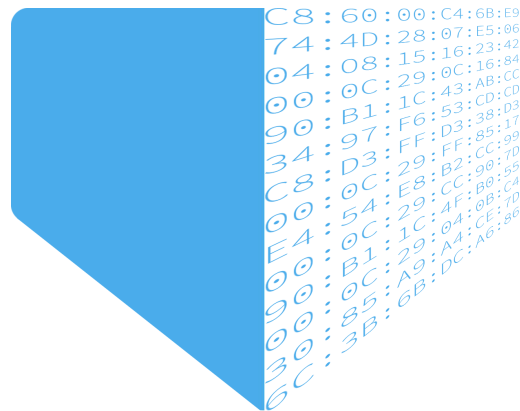
Теперь попытки пинга **vm1** с **vm2** или какие-либо подключения непосредственно к шлюзу обречены на неудачу

# iptables блокировка по MAC

Обозначим критерием блокировки трафика – MAC адрес. В случае если аппаратный адрес сетевой карты подключающегося устройства будет соответствовать указанному в правиле, оно будет отбрасывать трафик

```
#Отбрасываем трафик если он исходит от MAC адреса 08:00:27:47:88:ce  
sudo iptables -A INPUT -m mac --mac-source 08:00:27:47:88:CE -j DROP  
#Отбрасываем трафик если он исходит НЕ от MAC адреса 08:00:27:47:88:ce  
sudo iptables -A INPUT -m mac ! --mac-source 08:00:27:47:88:CE -j DROP
```

Теперь попытки пинга **vm1** с **vm2** или какие-либо подключения непосредственно к шлюзу обречены на неудачу



# Итоги темы

- Рассмотрели настройку firewall iptables для Linux
- Узнали как:

превратить виртуальную машину с двумя сетевыми интерфейсами в шлюз

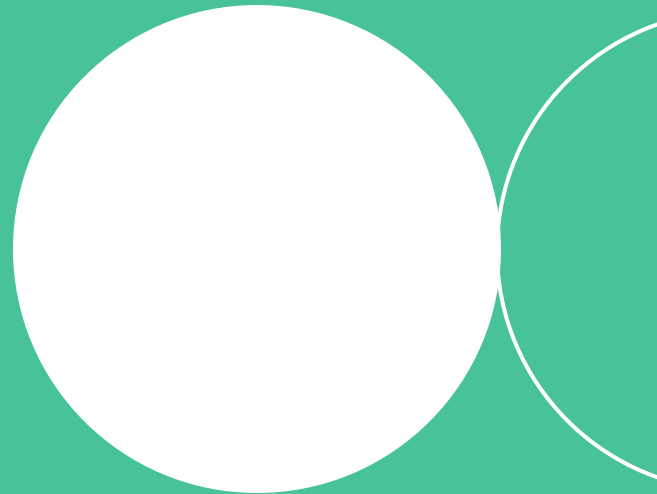
перенаправлять порты в локальную сеть

блокировать трафик по MAC

фильтровать L2 трафик



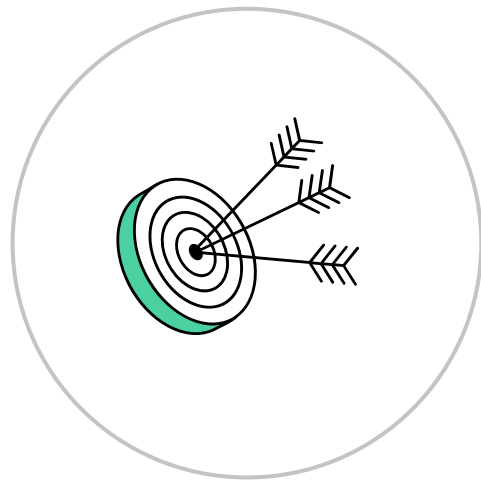
# Итоги занятия





# Итоги занятия

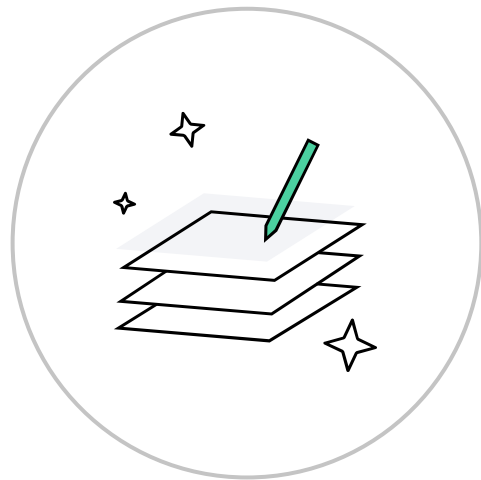
- Познакомились с протоколом Firewall
- Изучили возможности межсетевого экрана Netfilter и его утилиты iptables
- Научились управлять трафиком с помощью собственного небольшого роутера
- Научились создавать базовые правила в iptables



# Домашнее задание

## Давайте посмотрим вашу практику после лекции

- 1 Практика: домашнее задание (обязательное) с проверкой от преподавателя
- 2 Вопросы по домашнему заданию задавайте в чате учебной группы
- 3 Задачи можно сдавать по частям.  
Зачёт по домашней работе ставят после того, как приняты все задачи



# Спасибо за внимание

Артур Сагутдинов  
DevOps - инженер

