wiki.merionet.ru

Погружение в Iptables – теория и настройка

Мерион Нетворкс

14-20 минут

Начиная своё знакомство с **iptables**, следует рассказать про **netfilter**. Netfilter - это набор программных хуков внутри ядра Linux, которые позволяют модулям ядра регистрировать функции обратного вызова от стека сетевых протоколов.

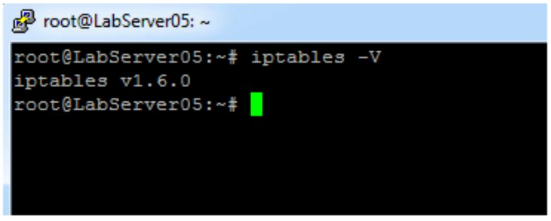
Хук (hook) - это программный элемент, который позволяет перехватывать функции обратного вызова в чужих процессах.

Netfilter является основой для построения Firewall'а в дистрибутивах Linux, но для того, чтобы он заработал в полную силу его нужно настроить. Как раз с помощью iptables мы можем взаимодействовать с хуками Netfilter и создавать правила фильтрации, маршрутизации, изменения и транслирования пакетов. Иногда про Netfilter забывают и называют эту связку просто iptables.

Введение

Итак, iptables - это утилита для настройки программного Firewall'а (межсетевого экрана) linux, которая предустанавливается по умолчанию во все сборки Linux, начиная с версии 2.4. Запускается iptables из командной строки (CLI) под пользователем с правами **root** и настраивается там же.

Можете в этом убедиться, набрав команду `iptables -V` в командной строке, она покажет вам версию iptables.



Почему же iptables всем так понравился, что его стали включать во все сборки Linux? Всё дело в том, что iptables действительно очень прост в настройке. С помощью него можно решить следующие задачи:

- Настроить stateless и statefull фильтрацию пакетов версий IPv4 и IPv6;

  Stateless - это фильтрация, основанная на проверке статических параметров одного пакета, например: IP адрес источника и получателя, порт и другие не изменяющиеся параметры.

  Statefull - это фильтрация, основанная на анализе потоков трафика. С помощью нее можно определить параметры целой TCP сессии или UDP потока.
- Настраивать все виды трансляции IP адресов и портов NAT, PAT, NAPT;
- Настроить политики QoS;
- Производить различные манипуляции с пакетами, например - изменять поля в заголовке IP.

Прежде чем переходить к практике, давайте обратимся к теории и поймём саму логику iptables.

Логика и основные понятия iptables

Правила

Как и все файрволлы, iptables оперирует некими правилами (**rules**), на основании которых решается судьба пакета, который поступил на интерфейс сетевого устройства (роутера).

Ну допустим у нас есть сетевое устройство с адресом **192.168.1.1**, на котором мы настроили iptables таким образом, чтобы запрещать любые ssh (порт 22) соединения на данный адрес. Если есть пакет, который идёт, например, с адреса **192.168.1.15** на адрес **192.168.1.1** и порт **22**, то iptables скажет: “Э, нет, брат, тебе сюда нельзя” и выбросит пакет.

(Или вообще ничего не скажет и выбросит, но об этом чуть позже :)

Каждое правило в iptables состоит из **критерия**, **действия** и **счётчика**

- **Критерий** - это условие, под которое должны подпадать параметры пакета или текущее соединение, чтобы сработало действие. В нашем примере – этим условием является наличие пакета на входящем интерфейсе, устанавливающего соединение на порт **22**
- **Действие** - операция, которую нужно проделать с пакетом или соединением в случае выполнения условий критерия. В нашем случае – запретить пакет на порт 22

- **Счетчик** - сущность, которая считает сколько пакетов было подвержено действию правила и на основании этого, показывает их объём в байтах.

Цепочки

Набор правил формируется в цепочки (**chains**)

Существуют базовые и пользовательские цепочки.

Базовые цепочки - это набор предустановленных правил, которые есть в iptables по умолчанию.

Существует 5 базовых цепочек и различаются они в зависимости от того, какое назначение имеет пакет. Имена базовых цепочек записываются в верхнем регистре.

- **PREROUTING** - правила в этой цепочке применяются ко всем пакетам, которые поступают на сетевой интерфейс извне;
- **INPUT** - применяются к пакетам, которые предназначаются для самого хоста или для локального процесса, запущенного на данном хосте. То есть не являются транзитными;
- **FORWARD** - правила, которые применяются к транзитным пакетам, проходящими через хост, не задерживаясь;
- **OUTPUT** - применяются к пакетам, которые сгенерированы самим хостом;
- **POSTROUTING** - применяются к пакетам, которые должны покинуть сетевой интерфейс.

В базовых цепочках обязательно устанавливается политика по умолчанию, как правило – принимать (ACCEPT) или сбрасывать (DROP) пакеты. Действует она только в цепочках **INPUT**, **FORWARD** и **OUTPUT**

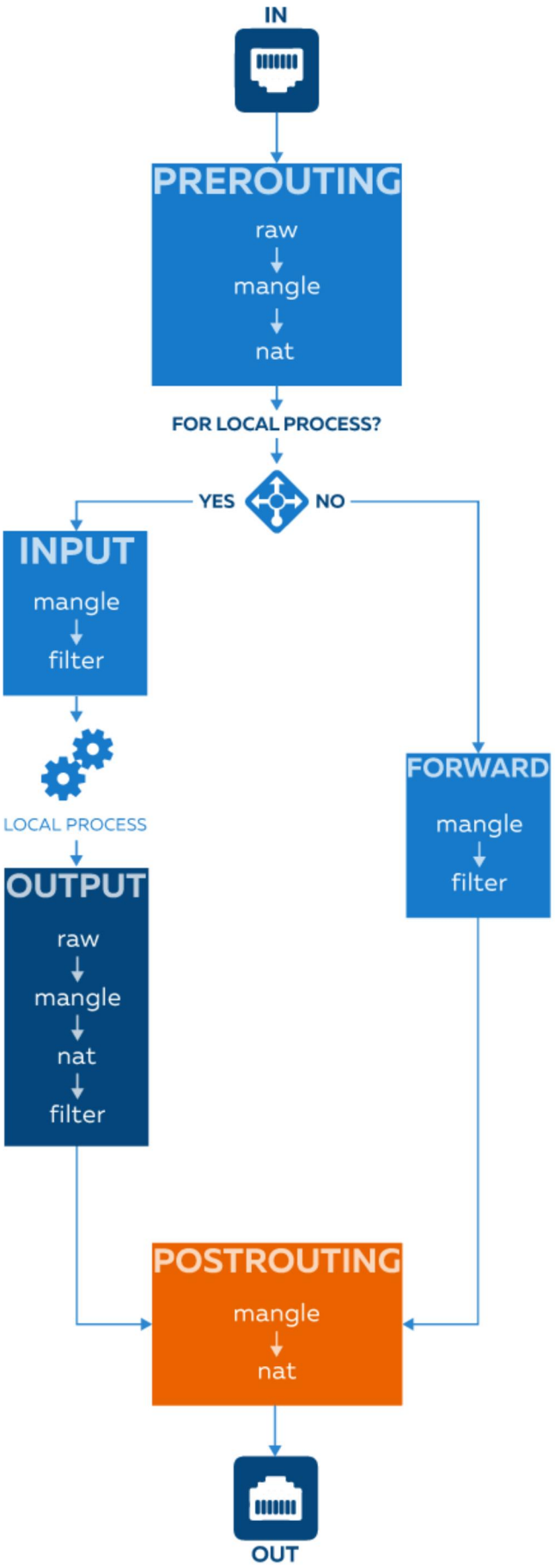
Таблица

Таблицы - это набор базовых и пользовательских цепочек. В зависимости от того, в какой таблице находится цепочка правил, с пакетом или соединением производятся определённые действия

Существует 5 таблиц:

- **filter** - таблица, выполняющая функции фильтрации пакетов по определённым параметрам. В большинстве случаев вы будете использовать именно её. Содержит следующие встроенные цепочки: **FORWARD**, **INPUT**, **OUTPUT**;
- **raw** - чтобы понять предназначение этой таблицы, нужно понимать логику работы statefull firewall'a. Дело в том, что по умолчанию, iptables рассматривает каждый пакет как часть большого потока и может определить какому соединению принадлежит тот или иной пакет. С помощью raw таблицы настраиваются исключения, которые будут рассматривать пакет как отдельную, ни к чему не привязанную сущность. Содержит следующие встроенные цепочки: **INPUT**, **OUTPUT**;
- **nat** - таблица, предназначенная целиком по функции трансляции сетевых адресов. Содержит следующие встроенные цепочки: **PREROUTING**, **OUTPUT**, **POSTROUTING**;
- **mangle** - таблица, предназначенная для изменения различных заголовков пакета. Можно, например, изменить TTL, количество hop'ов и другое. Содержит следующие встроенные цепочки: **PREROUTING**, **INPUT**, **FORWARD**, **OUTPUT**, **POSTROUTING>**;
- **security** - используется для назначения пакетам или соединениям неких меток, которые в дальнейшем может интерпретировать SElinux.

Теперь мы можем представить себе логику iptables в виде следующей схемы:



Действия

Ну и последнее, о чем нужно рассказать, прежде чем мы с вами начнем писать правила - это **target**. В контексте iptables, target - это действие, которое нужно проделать с пакетом или соединением, которое

совпало с критериями правила.

Итак, наиболее используемые действия:

- **ACCEPT** - разрешить прохождение пакета;
- **DROP** - тихо выбросить пакет, не сообщая причин;
- **QUEUE** - отправляет пакет за пределы логики iptables, в стороннее приложение. Это может понадобиться, когда нужно обработать пакет в рамках другого процесса в другой программе;
- **RETURN** - остановить обработку правила и вернуться на одно правило назад. Это действие подобно break'y в языке программирования.

Помимо этих четырех, есть ещё масса других действий, которые называются расширенными (extension modules):

- **REJECT** - выбрасывает пакет и возвращает причину в виде ошибки, например: icmp unreachable;
- **LOG** - просто делает запись в логе, если пакет соответствует критериям правила;

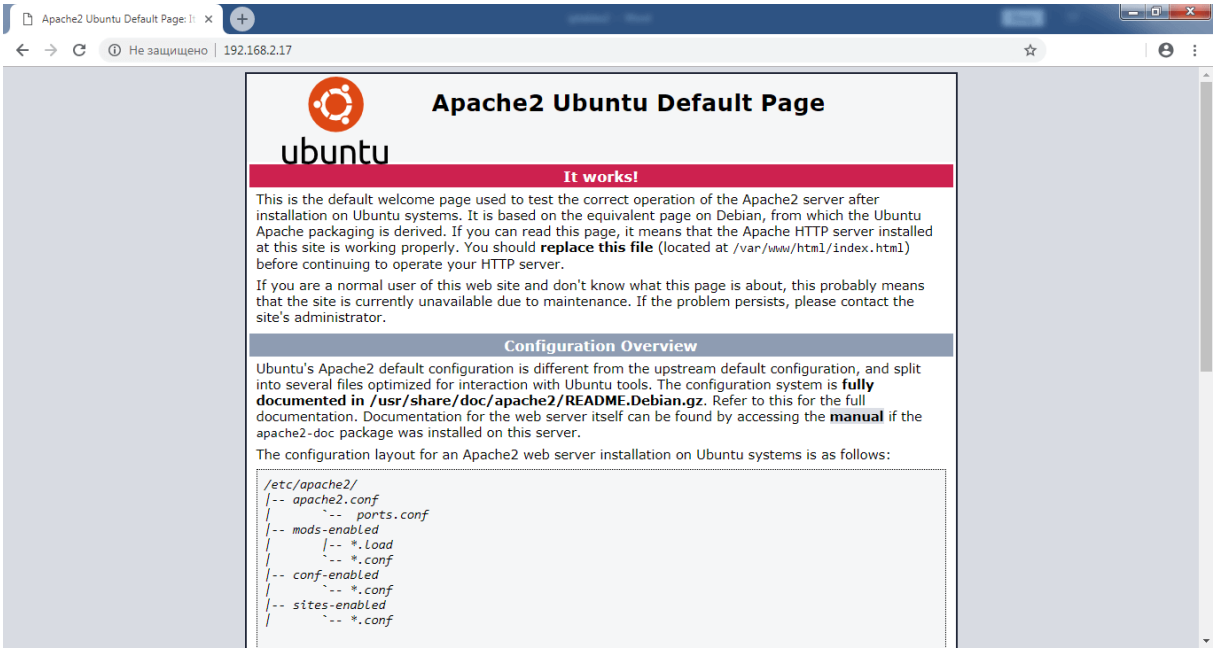
Есть действия, которые доступны только в определенной цепочке и таблицах, например, только в табоице nat и цепочках **OUTPUT** и **PREROUTING** доступно действие **DNAT**, которое используется в NAT'ировании и меняет **Destination IP** пакета. В той же таблице, только в цепочке **POSTRUNNING** доступно действие **SNAT**, меняющее **Source IP** пакета.

Отдельно остановимся на действии **MASQUERADE**, которое делает то же самое что **SNAT**, только применяется на выходном интерфейсе, когда IP адрес может меняться, например, когда назначается по DHCP.

Пишем правила

Отлично, теперь давайте приближаться к практике. Как Вы уже поняли, мы будем писать правила, поэтому нам нужно понять, как они строятся.

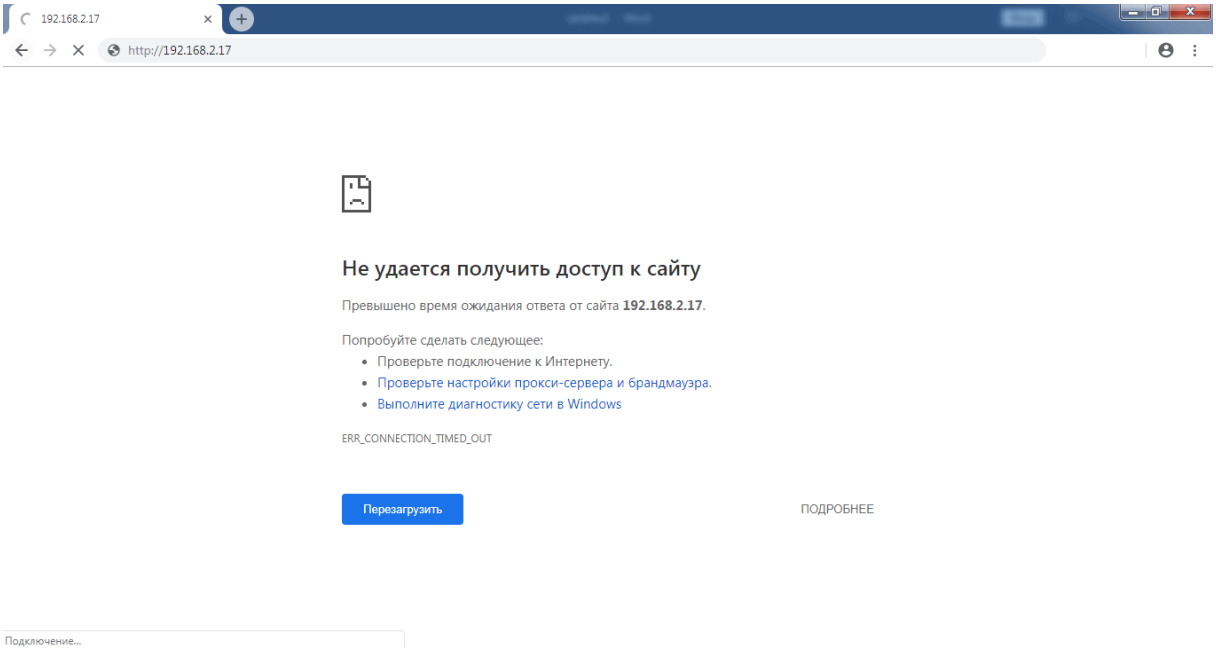
Итак, допустим у нас есть хост с адресом **192.168.2.17**, на **80** (http) порту которого, работает вэб-сервер Apache. Мы заходим на адрес **http://192.168.2.17** с хоста с адресом **192.168.2.2** и всё отлично работает:



А теперь открываем командную строку под **root** на хосте 192.168.2.17 и пишем:

```
iptables -A INPUT -p tcp -s 192.168.2.2 --dport 80 -j DROP
```

Попробуем открыть открыть **http://192.168.2.17** ещё раз:



Упс, не работает. Давайте теперь разбираться, что мы наделали?

Всё очень просто – данной командой мы:

- вызвали утилиту **iptables**;
- **-A** - этим ключом мы указали, что нужно добавить правило к существующей цепочке;
- **INPUT** - указали цепочку, к которой хотим добавить правило;
- **-p tcp** - явно указали протокол TCP. Здесь также можно указывать другие протоколы (udp, icmp, sctp), или номер протокола, инкапсулируемого в IP (17 – udp, 6 – tcp и др.);
- **-s 192.168.2.2** - указали, какой адрес источника должен быть у пакета, который мы хотим фильтровать;
- **--dport 80** - указали адресованные какому порту пакеты мы хотим фильтровать. В данном случае - **80**, на котором работает наш сервер Apache.
- **-j DROP** - указали что нужно сделать с пакетом, параметры которого совпали с данными критериями. В данном случае – просто тихо выбросить.

Таким образом, мы заблокировали все пакеты с адреса 192.168.2.2 на локальный порт 80 и тем самым закрыли доступ к нашему серверу Apache для данного хоста.

Обратите внимание – мы не указывали таблицу, в цепочки которой мы хотим добавить правило. Поэтому, по умолчанию таблица - **filter**. Для явного указания таблицы нужно перед указанием цепочки ввести ключ **-t** или (**--table**)

Чтобы открыть доступ опять просто поменяем ключ **-A** в правиле на **-D**, тем самым мы удалим данное правило из iptables.

Синтаксис iptables

Друзья, на самом деле в iptables очень богатый синтаксис правил. Полный список ключей и параметров вы можете найти в официальном гайде на iptables.org. Мы же приведём самые “ходовые” опции, которыми вы, вероятно, будете пользоваться. Чтобы вы не запутались, мы приводим их в табличках ниже.

Для удобства, в iptables реализовано очень много сокращений для разных ключей. Например, мы писали ключ **-A** вместо полного **--append**, **-p** вместо полного **--proto** и **-s** вместо полного **--source**, дальше мы покажем, что ещё можно сократить и где применить.

Начнём с команд для редактирования правил и цепочек – добавления, удаления, замены и так далее:

коротко	**синтаксис правила**	**применение**
-A	--append {цепочка правила}	добавить правило к цепочке (в самое начало)
-D	--delete {цепочка правила}	удалить правило из цепочки
-D	--delete {номер правила в цепочке}	удалить правило из цепочки по номеру (1 - x)
-I	--insert {номер правила вцепочке}	вставить правило в цепочку по номеру (1 - x)
-R	--replace {номер правила вцепочке}	заменить правило в цепочке по номеру (1 - x)
-X	--delete-chain {цепочка}	удалить цепочку (только для пользовательских)
-E	--rename-chain {старое имя цепочки} {новое имя цепочки}	переименовать цепочку

-N	--new {имя цепочки}	создание новой пользовательской цепочки
-C	--check {правило цепочки}	проверит наличие правила в цепочке
-F	--flush {цепочка}	удаляет все правила в цепочке, если цепочка не указана – удалятся все правила
-Z	--zero {цепочка} {номер правила вцепочке}	обнуляет все счётчики пакетов и байтов в цепочке или всех цепочках
-P	--policy {цепочка} {номер правила вцепочке}	изменяет политику по умолчанию, она должна основываться на встроенном target'е {ACCEPT, DROP, QUEUE}

Продолжим синтаксисом настройки правил – на каком сетевом интерфейсе следить за пакетами, какой протокол проверять, адрес источника, назначения и так далее.

Кстати, перед некоторыми параметрами можно ставить восклицательный знак - **!**, означающее логическое **НЕ**. В таблице мы пометим такие параметры таким значком – (!)

коротко	**синтаксис опции**	**применение**
-p	(!) --proto {протокол}	протокол {tcp, udp, udplite, icmp, esp, ah, sctp} или номер протокола {16,7}, all - все протоколы
-4	--ipv4	указывает версию протокола ipv4
-6	--ipv6	указывает версию протокола ipv6
-s	(!) --source {адрес/маска}	указывает ip адрес источника
-d	(!) --destination {адрес/маска}	указывает ip адрес назначения
-m	--match	включает дополнительные модули, явно задающимися данным ключем. например <code>m limit --limit 3/min</code> - установит лимит на количество пакетов в минуту
-f	(!) --fragment	включает обработку фрагментированных пакетов, в которых нет параметров изначального полного пакета, содержащихся в первом фрагменте пакета
-i	(!) --in-interface {имя интерфейса}	обрабатывает только входящие пакеты, прилетающие на сетевой интерфейс {имя интерфейса}
-o	(!) --out-interface {имя интерфейса}	обрабатывает только исходящие пакеты, прилетающие на сетевой интерфейс {имя интерфейса}
	--set-counters {пакеты} {байты}	включает счётчик для ключей--insert, --append, --replace

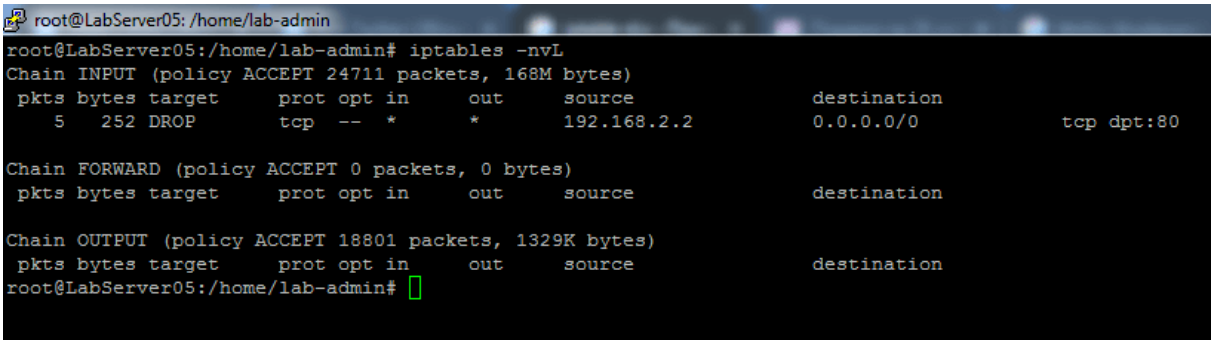
Теперь рассмотрим опции для действий, которые должны сработать по совпадению критериев:

коротко	**синтаксис опции**	**применение**
-j	--jump {действие}	применяет одно из действий accept, drop, reject и другие
-g	--goto {цепочка}	переходит к другой цепочке правил

Теперь рассмотрим какую информацию мы можем вытянуть с помощью iptables и какие опции для этого нужно использовать:

коротко	**синтаксис команды**	**применение**
-l	--list {цепочка} {номер правила}	показывает правила в цепочке или всех цепочках. по умолчанию покажет таблицу filter
-s	--list-rules{цепочка} {номер правила}	показывает текст правила в цепочке или всех цепочках
-n	--numeric	покажет параметры правила в числовом виде. например не порт будет не http, а 80
-v	--verbose	выводит более подробную информацию
-v	--version	покажет версию iptables
-x	--exact	покажет точные значения числовых параметров
	--line-numbers	покажет номера правил

Для быстрого получения информации о настроенных правилах и о метриках их срабатывания, часто применяется команда, комбинирующая 3 ключа - `iptables -nLv`. Например, для настроенного нами ранее правила – вывод будет такой:



Пример посложнее

Давайте рассмотрим ещё один пример. Допустим у нас во локальной сети есть хост **192.168.2.19** с сервером Apache. Мы хотим сделать его доступным из Интернета. Для этого нам нужно воспользоваться возможностями таблицы **nat** и написать правило, которое будет перенаправлять входящий http трафик на внешний интерфейс (пусть будет **enp0s3** с адресом 101.12.13.14) и порт **80** на адрес нашего сервера внутри сети и 80 порт – 192.168.2.19:80. По сути – нужно сделать проброс портов.

Напишем такое правило:

```
iptables -t nat -A PREROUTING -i enp0s3 -p tcp --dport 80 -j DNAT --to 192.168.2.19:80
```

Теперь если мы перейдём по адресу **http://101.12.13.14**, то должны попасть на наш Apache.

Возможности iptables настолько обширны, что мы могли бы начать писать новую Базу знаний по нему. В статье мы показали лишь базовые варианты применения. Это действительно великий инструмент и освоить его не так уж сложно. Надеюсь, данная статья Вам в этом поможет. Спасибо за внимание!

Рекомендуем