В данной заметке я хочу рассмотреть простой и быстрый вариант настройки шлюза для организации доступа в интернет из локальной сети на базе CentOS 7. Не будет никакого дополнительного функционала, только самое необходимое для доступа к интернету компьютеров за шлюзом.

Содержание:

Данная статья является частью единого цикла статьей про сервер Debian.

Введение

В нашем распоряжении будет следующий сервер для настройки шлюза:

```
# cat /etc/redhat-release
CentOS Linux release 7.1.1503 (Core)
```

Использовался образ minimal для установки CentOS 7. Если вы еще не выполнили установку, рекомендую воспользоваться моим материалом на эту тему. На сервере две сетевые карты **eth0** и **eth1**:

- eth0 подключена к интернету
- eth1 подключена к локальной сети вместе с компьютерами

В данной статье мы выполним необходимые предварительные настройки на сервере, включим nat, настроим firewall и установим средство мониторинга сетевой активности.

Если у вас недостаточно опыта и вы не чувствуете в себе сил разобраться с настройкой шлюза самому с помощью консоли сервера - попробуйте дистрибутив на основе centos для организации шлюза и прокси сервера в локальной сети - clearos. С его помощью можно через браузер настроить весь необходимый функционал. В отдельной статье я подробно рассказал о настройке clearos.

Предварительная настройка сервера

Любую настройку сервера я рекомендую начинать с обновления:

```
# yum -y update
```

После этого я устанавливаю mc, так как привык к нему и постоянно пользуюсь:

```
# yum -y install mc
```

Дальше отключаем selinux. Находим файл /etc/sysconfig/selinux и редактируем его:

```
# mcedit /etc/sysconfig/selinux
```

Приводим строку с соответствующим параметром к следующему виду:

```
SELINUX=disabled
```

Чтобы применить изменения, перезагружаем сервер:

```
# reboot
```

Более подробно о базовой настройке сервера CentOS 7 читайте отдельно. Мы же двигаемся дальше.

Теперь настроим сеть. Я очень подробно рассмотрел вопрос настройки сети в CentOS 7 в своем отдельном материале. Рекомендую с ним ознакомиться. Здесь же я кратко выполню необходимые команды, без пояснений.

Сначала удаляем NetworkManager. Он нам не понадобится, выполним все настройки вручную. Иногда он может вызывать непонятные ошибки, я предпочитаю им не пользоваться:

```
# systemctl stop NetworkManager.service
# systemctl disable NetworkManager.service
```

Теперь включаем классическую службу сети в CentOS 7:

```
# systemctl enable network.service
```

Настраиваем сетевые интерфейсы:

```
# mcedit /etc/sysconfig/network-scripts/ifcfg-eth0
```

```
HWADDR=00:15:5D:01:0F:06
TYPE="Ethernet"
BOOTPROTO="dhcp"
DEFROUTE="yes"
PEERDNS="yes"
PEERROUTES="yes"
NAME="eth0"
UUID="4e65030c-da90-4fb8-bde4-028424fe3710"
ONBOOT="yes"
```

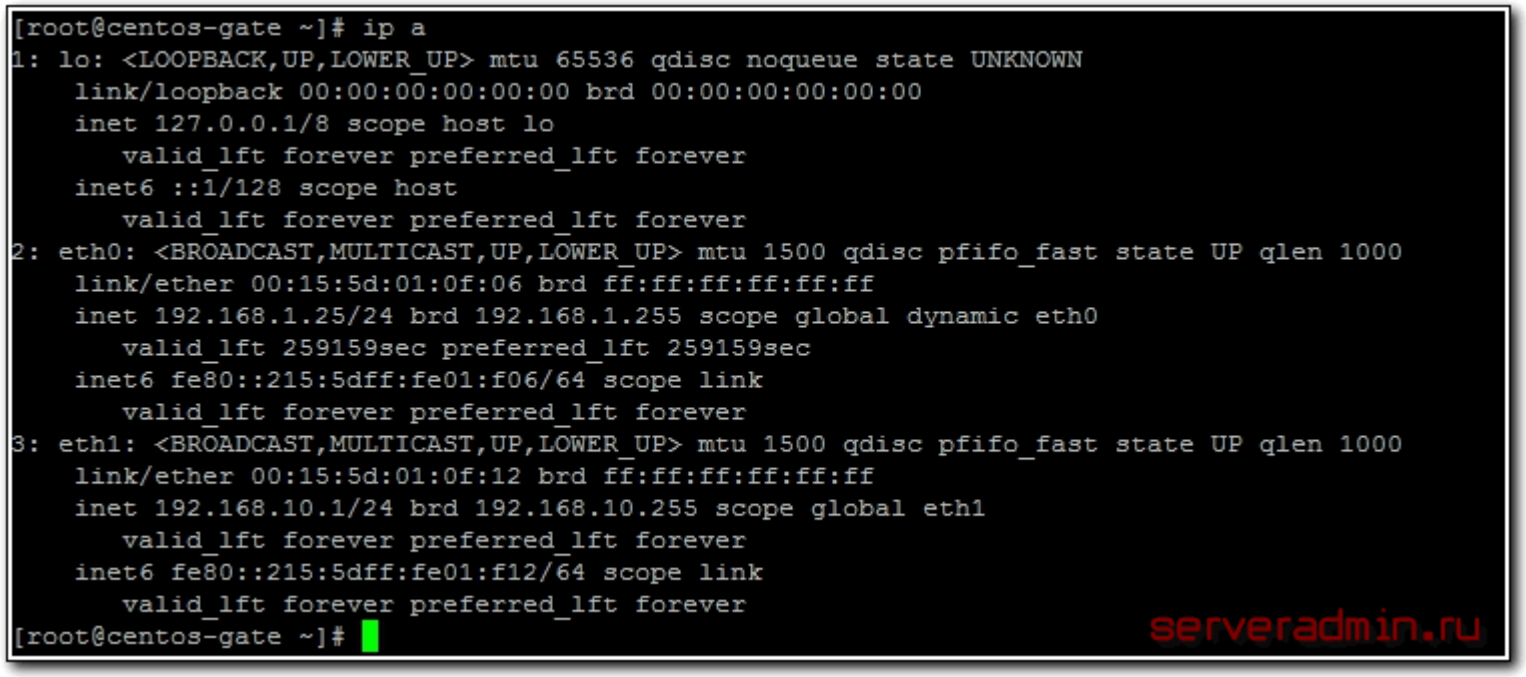
```
# mcedit /etc/sysconfig/network-scripts/ifcfg-eth1

DEVICE=eth1
HWADDR=00:15:5d:01:0f:12
TYPE=Ethernet
ONBOOT=yes
IPADDR=192.168.10.1
NETMASK=255.255.255.0
```

Перезапускаем службу сети:

```
# systemctl restart network.service
```

Смотрим, что получилось:

```
# ip a
```



Вы настраивайте сеть в зависимости от своих условий. Если внешний адаптер получает настройки не по dhcp, как у меня, а в статике, то не забудьте настроить шлюз по-умолчанию и dns сервер. Как это сделать написано в моей статье о сетевых параметрах, ссылку на которую я приводил выше.

> Прежде чем двигаться дальше, убедитесь, что вы все верно настроили - на сервере работает интернет, компьютеры из локальной сети пингуют сервер по адресу на eth1.

Включаем маршрутизацию, firewall и nat

Чтобы сервер мог маршрутизировать пакеты между сетевыми адаптерами, необходимо выполнить следующую настройку. Находим файл /etc/sysctl.conf и вставляем туда строку:

```
# mcedit /etc/sysctl.conf
net.ipv4.ip_forward = 1
```

Чтобы заработала настройка, выполняем команду:

```
# sysctl -p
```

Теперь приступаем к самому главному - настройке фаерволла. Опять же отсылаю вас к своему материалу, где я очень подробно рассмотрел вопрос настройки iptables в CentOS 7. Там же приведен готовый скрипт для iptables. Так что выполняем все необходимые действия без пояснений.

Отключаем firewalld:

```
# systemctl stop firewalld
# systemctl disable firewalld
```

Устанавливаем службы iptables:

```
# yum -y install iptables-services
```

Скачиваем скрипт с правилами iptables.sh. Данные правила включают NAT, закрывают доступ к серверу снаружи, разрешают пинги, разрешают всем

пользователям локальной сети доступ в интернет. Дополнительный функционал отключен. В скрипте подробно описаны все правила. Вам необходимо только заменить в начале переменные на свои. В моем случае это будет выглядеть так:

```
# Внешний интерфейс
export WAN=eth0
export WAN_IP=192.168.1.25
# Локальная сеть
export LAN1=eth1
export LAN1_IP_RANGE=192.168.10.1/24
```

Помещаем отредактированный скрипт в /etc/iptables.sh и делаем его исполняемым:

```
# chmod 0740 /etc/iptables.sh
```

Запускаем iptables:

```
# systemctl start iptables.service
```

Добавляем их в автозагрузку:

```
# systemctl enable iptables.service
```

Выполняем скрипт с правилами:

```
# /etc/iptables.sh
```

Проверяем установленные правила:

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

```
[root@centos-gate etc]# iptables -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  eth1   *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 0
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 3
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 11
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 8
    6   432 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0           state INVALID
    0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp flags:0x3F/0x00
    0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp flags:!0x17/0x02 state NEW
    0     0 ACCEPT     tcp  --  eth0   *       0.0.0.0/0            0.0.0.0/0           tcp dpt:22

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0           state INVALID
    0     0 ACCEPT     all  --  eth1   eth0    0.0.0.0/0            0.0.0.0/0
    0     0 REJECT     all  --  eth0   eth1    0.0.0.0/0            0.0.0.0/0           reject-with icmp-port-unreachable

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      lo      0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  *      eth1    0.0.0.0/0            0.0.0.0/0
    4   480 ACCEPT     all  --  *      eth0    0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp flags:!0x17/0x02 state NEW
[root@centos-gate etc]#
```


Если у вас то же самое, значит вы все сделали правильно.

По сути наш шлюз уже готов и может обслуживать клиентов. Но не работает одна важна служба, без которой нормальной работы с интернетом не получится. Нам нужно настроить кэширущий dns сервер для клиентов локальной сети. Можно пойти по простому и очень простому пути. Простой путь

это выполнить простейшую настройку dns сервера bind. Как это сделать у меня опять же подробно написано отдельно - настройка Bind 9 в CentOS 7. Рекомендую ознакомиться, там рассмотрены интересные нюансы настройки.

Очень простой путь это установить dnsmasq, который помимо dns сервера включает в себя еще и dhcp сервер, который нам может пригодиться.

Установка и настройка dnsmasq в CentOS 7

С большой долей вероятности **dnsmasq** у вас уже установлен. Проверить это можно следующей командой:

```
# rpm -qa | grep dnsmasq
dnsmasq-2.66-14.el7_1.x86_64
```

Если вывод такой же, значит пакет уже стоит. Если нет, то устанавливаем dnsmasq командой:

```
# yum -y install dnsmasq
```

Редактируем файл конфигурации /etc/dnsmasq.conf и приводим его к очень простому виду:

```
# mcedit /etc/dnsmasq.conf

domain-needed
bogus-priv
interface=eth1
dhcp-range=192.168.10.50,192.168.10.150,24h
```

Запускаем dnsmasq:

```
# systemctl start dnsmasq
```

Либо перезапускаем, если он был у вас запущен:

```
# systemctl restart dnsmasq
```

Добавляем dnsmasq в автозагрузку:

```
# systemctl enable dnsmasq
```

Я редактировал конфиг, когда у меня уже был установлен и запущен dnsmasq. И он то ли завис, то ли просто затупил, но я не мог его перезагрузить или остановить с помощью systemctl. Пришлось перезагрузить сервер. После этого все нормально заработало. Клиент на windows получил сетевые настройки. Информация об этом появилась в логе /var/log/messages. Я проверил на клиенте интернет, все было в порядке, он работал.

На этом настройка завершена, шлюзом под CentOS 7 можно пользоваться.

Анализ сетевой активности на шлюзе в linux

В данной конфигурации шлюз может успешно функционировать. Но иногда хочется посмотреть, а что вообще на нем происходит. Например, кто-то занимает весь канал, интернет тормозит, а мы как слепые котята сидим и не видим ничего. Нужно какое-то средство для просмотра загрузки сети на шлюзе. И такое средство есть - программа **iftop**.

Она отсутствует в стандартном репозитории CentOS 7. Для ее установки необходимо подключить репозиторий epel:

```
# yum -y install epel-release
```

Устанавливаем iftop на CentOS 7:

```
# yum -y install iftop
```

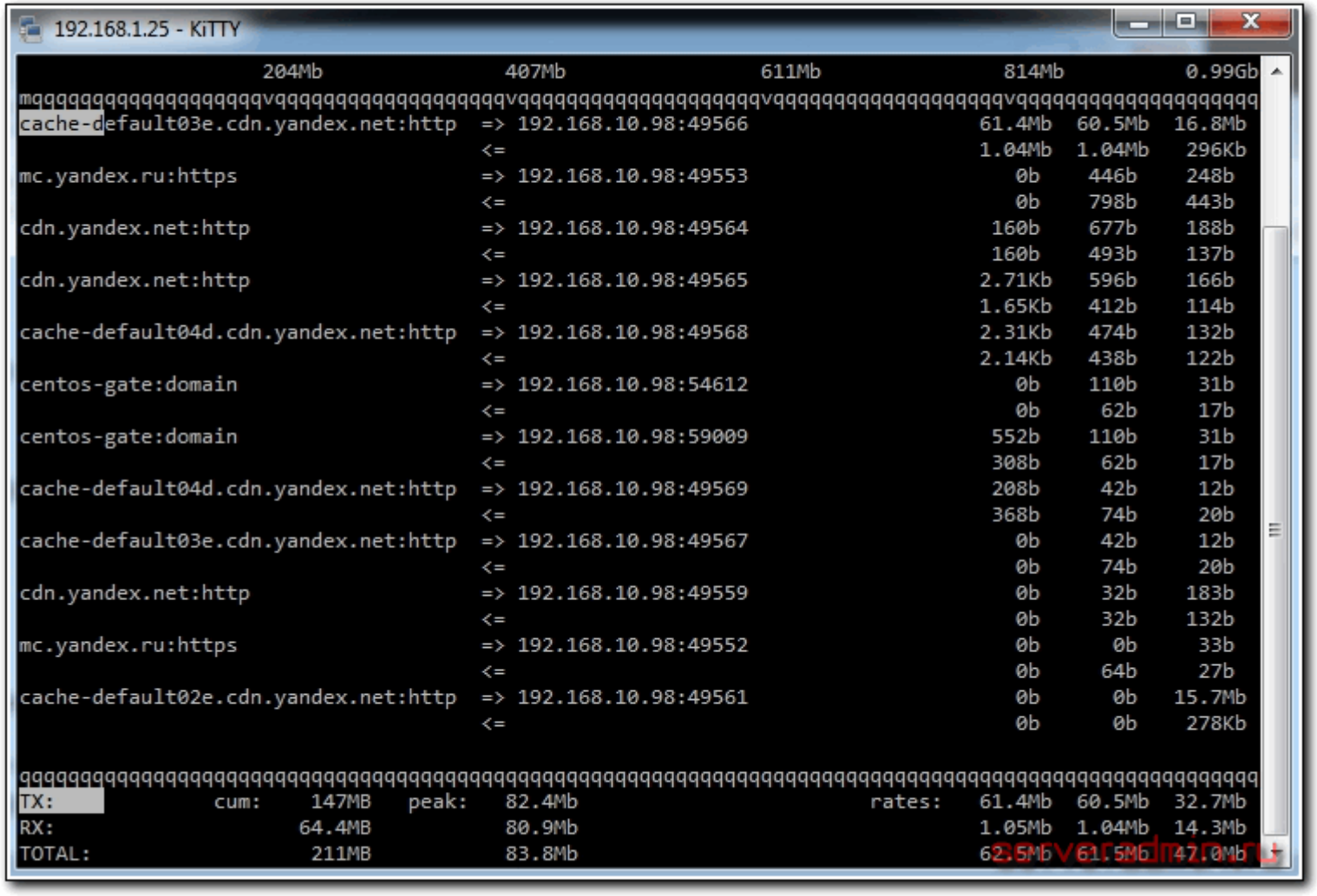
Теперь мы можем смотреть загрузку сети на шлюзе в режиме реального времени. Чтобы увидеть сетевую активность, достаточно запустить iftop:

```
# iftop
```

По-умолчанию она слушает интерфейс eth0. Это внешний интерфейс шлюза, на нем все подключения будут отображены от имени самого шлюза и определить, кто же в сети занимает канал мы не сможем. Чтобы это увидеть, необходимо запустить просмотр сетевой активности на локальном интерфейсе. Сделать это не сложно, достаточно запустить iftop с параметром:

```
# iftop -i eth1 -P
```

Теперь уже гораздо интереснее. Я еще добавил параметр **-P**, который отображает порты, по которым проходят соединения. Посмотрим, кто больше всех загружает канал интернета:



В моем случае это пользователь с ip 192.168.10.98, на котором я запустил проверку скорости интернета с серверов Яндекса.

Если у вас не большая сеть и не много пользователей, то с помощью этой простой и эффективной утилиты вы сможете легко определить, кто, к примеру, качает торренты или чем-то еще загружает канал.

Заключение

С помощью бесплатного дистрибутива Linux мы смогли за считанные минуты настроить шлюз для организации доступа в интернет компьютеров из локальной сети. У меня ушло минут 10 на настройку шлюза по этой инструкции. Если вы делаете это первый раз, то конечно у вас уйдет гораздо больше времени. Нужно будет разобраться в нюансах, к тому же я дал много ссылок на дополнительный материал.

Давайте разберемся в том, что мы сделали:

1. Выполнили предварительную настройку сервера, подготовили его к работе.
2. Включили маршрутизацию.
3. Настроили firewall.
4. Включили NAT.
5. Установили и настроили dnsmasq для организации служб dns и dhcp.
6. Проанализировали сетевую активность шлюза, узнали кто загружает канал интернета.

Это минимально необходимый функционал для организации работы шлюза на CentOS 7. Следующим этапом может быть настройка прокси сервера, шейпера траффика, настройка 2-х и более провайдеров и много другое. Что-то из этого я рассмотрю в своих будущих статьях.

Напоминаю, что данная статья является частью единого цикла статьей про сервер Debian.

- Вы получите навыки администрирования Linux (структура Linux, основные команды, работа с файлами и ПО).
- Вы рассмотрите следующий стек технологий: Zabbix, Prometheus, TCP/IP, nginx, Apache, MySQL, Bash, Docker, Git, nosql, grfana, ELK.
- Умение настраивать веб-сервера, базы данных (mysql и nosql) и работа с сетью.
- Мониторинг и логирование на базе Zabbix, Prometheus, Grafana и ELK.
- Научитесь командной работе с помощью Git и Docker.

Смотрите подробнее программу по .