Настройка сети

TCP/IP

Прикладной уровень	FTP,HTTP(s),SMTP…
…	…
Транспортный уровень	TCP, UDP
Сетевой уровень	IP
Канальный уровень	MAC
Физический уровень	Ethernet,WiFi…

Стек протоколов TCP/IP — набор сетевых протоколов передачи данных, используемых в сетях, включая сеть Интернет. Название TCP/IP происходит из двух ключевых протоколов семейства — Transmission Control Protocol (TCP) и Internet Protocol (IP). Работают они в стеке, что означает, что протокол, располагающийся на уровне выше, работает «поверх» нижнего (например, протокол TCP работает поверх протокола IP).
Стек протоколов TCP/IP включает в себя четыре уровня:

Прикладной уровень (application layer)
На прикладном уровне работает большинство сетевых приложений. Эти программы имеют свои собственные протоколы обмена информацией, например, HTTP для WWW, FTP (передача файлов), SMTP (электронная почта), SSH (безопасное соединение с удалённой машиной), DNS (преобразование символьных имён в IP-адреса) и многие другие.
В массе своей эти протоколы работают поверх TCP или UDP и привязаны к определённому порту, например:
HTTP на TCP-порт 80,
FTP на TCP-порт 20 (для передачи данных) и 21 (для управляющих команд),
SSH на TCP-порт 22,
И др.

Транспортный уровень (transport layer)

Протоколы транспортного уровня (Transport layer) могут решать проблему негарантированной доставки сообщений («дошло ли сообщение до адресата?»), а также гарантировать правильную последовательность прихода данных. В стеке TCP/IP транспортные протоколы определяют, для какого именно приложения предназначены эти данные.

TCP — «гарантированный» транспортный механизм с предварительным установлением соединения, предоставляющий приложению надёжный поток данных, дающий уверенность в безошибочности получаемых данных, перезапрашивающий данные в случае потери и устраняющий дублирование данных. TCP позволяет регулировать нагрузку на сеть, а также уменьшать время ожидания данных при передаче на большие расстояния. Более того, TCP гарантирует, что полученные данные были отправлены точно в такой же последовательности. В этом его главное отличие от UDP.

UDP - протокол передачи датаграмм (блоков информации) без установления соединения. Также его называют протоколом «ненадёжной» передачи, в смысле невозможности удостовериться в доставке сообщения адресату, а также возможного перемешивания пакетов. В приложениях, требующих гарантированной передачи данных, используется протокол TCP. UDP обычно используется в таких приложениях, как потоковое видео и компьютерные игры, где допускается потеря пакетов, а повторный запрос затруднён или не оправдан, либо в приложениях вида запрос-ответ (например, запросы к DNS), где создание соединения занимает больше ресурсов, чем повторная отправка.

И TCP, и UDP используют для определения протокола верхнего уровня число, называемое портом.

Сетевой уровень (network layer)

Сетевой уровень (Internet layer) изначально разработан для передачи данных из одной (под)сети в другую. Примерами такого протокола является X.25 и IPC. С развитием концепции глобальной сети в уровень были внесены дополнительные возможности по передаче из любой сети в любую сеть, независимо от протоколов нижнего уровня, а также возможность запрашивать данные от удалённой стороны, например в протоколе ICMP (используется для передачи диагностической информации IP-соединения).

К этому уровню относятся: DHCP, ICMP и др.

Канальный уровень (link layer).

Канальный уровень (Link layer) описывает, каким образом передаются пакеты данных через физический уровень, включая кодирование (то есть специальные последовательности бит, определяющих начало и конец пакета данных). Ethernet, например, в полях заголовка пакета содержит указание того, какой машине или машинам в сети предназначен этот пакет.

Примеры протоколов канального уровня — Ethernet, IEEE 802.11 Wireless Ethernet и др.

Кроме того, канальный уровень описывает среду передачи данных (будь то коаксиальный кабель, витая пара, оптическое волокно или радиоканал), физические характеристики такой среды и принцип передачи данных (разделение каналов, модуляцию, амплитуду сигналов, частоту сигналов, способ синхронизации передачи, время ожидания ответа и максимальное расстояние).

IPv4

IP-адрес	192.168.1.2	11000000 10101000 00000001 00000010
Маска подсети	255.255.254.0	11111111 11111111 11111110 00000000
Адрес сети	192.168.0.0	11000000 10101000 00000000 00000000

IP-адрес — уникальный сетевой адрес узла в компьютерной сети, построенной по протоколу IP. В сети Интернет требуется глобальная уникальность адреса; в случае работы в локальной сети требуется уникальность адреса только в пределах сети.

IP-адрес состоит из двух частей: номера сети и номера узла. В случае изолированной сети её адрес может быть выбран администратором из специально зарезервированных для таких сетей блоков адресов (10.0.0.0/8, 172.16.0.0/12 или 192.168.0.0/16). Если же сеть должна работать как составная часть Интернета, то адрес сети выдаётся провайдером либо региональным интернет-регистратором
(Regional Internet Registry, RIR).

Сетевой узел также может входить в несколько IP-сетей. В этом случае компьютер должен иметь несколько IP-адресов, по числу сетевых связей. Таким образом, IP-адрес характеризует не отдельный компьютер или маршрутизатор, а одно сетевое соединение.

В настоящее время существует две версии протокола IP: версия 4, в которой IP-адрес имеет длину 32 бита и версия 6, в которой IP-адрес имеет длину 128 бит.

P-адрес протокола версии 4 имеет длину 32 бита или 4 байта и записывается обычно в виде четырёх десятичных чисел, разделённых точкой, к которым иногда через дробь добавляется маска.

В терминологии сетей TCP/IP маской подсети или маской сети называется битовая маска, определяющая, какая часть IP-адреса узла сети относится к адресу сети, а какая — к адресу самого узла в этой сети. Например, узел с IP-адресом 12.34.56.78 и маской подсети 255.255.255.0 находится в сети 12.34.56.0 с длиной префикса 24 бита.

Чтобы получить адрес сети, зная IP-адрес и маску подсети, необходимо применить к ним операцию поразрядной конъюнкции (логическое И).

IPv4

Класс	Первые биты	Распределение (с-сеть,х-хост)	Число возможных сетей	Число возможных хостов	Маска подсети
A	0	c.x.x.x	128	16777216	255.0.0.0
B	10	c.c.x.x	16384	65536	255.255.0.0
C	110	c.c.c.x	2097152	256	255.255.255.0
D	1110		Групповой адрес		
E	1111		Зарезервировано		

Вообще, существует два способа определить, сколько бит отводится на определение подсети, а сколько - на адрес устройства. Изначально использовалась классовая адресация, но во второй половине 90-х годов она была дополнена бесклассовой адресацией (CIDR - Classless Inter-Domain Routing).

При классовой адресации первые биты определяли класс сети, а по классу можно было сказать, сколько бит отведено под номер сети и сколько - под номер узла. Всего существовало 5 классов.

С ростом Интернета данный подход был признан неэффективным и дополнен бесклассовой адресацией, которая преимущественно и используется в настоящее время.

Бесклассовая адресация основывается на переменной длине маски подсети (англ. variable length subnet mask, VLSM), в то время, как в классовой (традиционной) адресации длина маски строго фиксирована 0, 1, 2 или 3 установленными октетами.

Два адреса из получившегося множества не могут использоваться в качестве адресов устройств: это адрес, у которого все разряды хостовой части равны нулю (такой адрес называется адресом сети) и адрес, у которого все разряды хостовой части равны единицы (широковещательный адрес).

Итак, давайте еще раз отметим несколько моментов, касающихся адресации. Во-первых, в заголовке IP-пакета есть поля IP источника и IP назначения. Никаких масок внутри пересылаемых пакетов нет. Маска назначается в момент, когда IP-адрес присваивается интерфейсу и нужна она для определения границ подсети. Т.е. с помощью маски устройство способно определить, кто находится с ним в одной сети, а кто - за ее пределами. И если устройства находятся в одной сети, они обмениваются пакетами напрямую, иначе обмен осуществляется через шлюз (либо шлюз по-умолчанию). Из этого следует, что шлюз ничего не знает о трафике передаваемом между двумя хостами, расположенными в одной сети. Посмотреть информацию о шлюзе и таблице маршрутизации можно с помощью команды

route -n

IPv6

2001:0DB8:AA10:0001:0000:0000:0000:00FE

2001:DB8:AA10:1:0:0:0:FE

2001:DB8:AA10:1::FE

Данная версия протокола призвана решить проблемы, с которыми столкнулась предыдущая версия (в частности, нехватка адресов). Длина адреса при этом возросла с 32 до 128 бит.

Улучшения IPv6 по сравнению с IPv4:

- В сверхскоростных сетях возможна поддержка огромных пакетов (джамбограмм) — до 4 гигабайт;
- Time to Live переименовано в Hop Limit;
- Появились метки потоков и классы трафика;
- Появилось многоадресное вещание;
- Из IP-заголовка исключена контрольная сумма. С учётом того, что канальные (Ethernet) и транспортные (TCP и UDP) протоколы имеют свои контрольные суммы, еще одна контрольная сумма на уровне IP воспринимается как излишняя;
- Автоконфигурация. При инициализации сетевого интерфейса ему назначается локальный IPv6-адрес, состоящий из префикса fe80::/10 и идентификатора интерфейса, размещённого в младшей части адреса. В качестве идентификатора интерфейса часто используется MAC-адрес. Для настройки других адресов узел может запросить информацию о настройках сети у маршрутизаторов, отправив ICMPv6 сообщение «Router Solicitation». Маршрутизаторы, получившие это сообщение, отвечают ICMPv6 сообщением «Router Advertisement», в котором может содержаться информация о сетевом префиксе, адресе шлюза, адресах рекурсивных DNS серверов, и других параметрах.

- Метки потоков. Введение в протоколе IPv6 поля «Метка потока» позволяет значительно упростить процедуру маршрутизации однородного потока пакетов. Поток — это последовательность пакетов, посылаемых отправителем определённому адресату. При этом предполагается, что все пакеты данного потока должны быть подвергнуты определённой обработке. Характер данной обработки задаётся дополнительными заголовками. Допускается существование нескольких потоков между отправителем и получателем. Метка потока присваивается узлом-отправителем путём генерации псевдослучайного 20-битного числа. Все пакеты одного потока должны иметь одинаковые заголовки, обрабатываемые маршрутизатором. При получении первого пакета с меткой потока маршрутизатор анализирует дополнительные заголовки, выполняет предписанные этими заголовками функции и запоминает результаты обработки (адрес следующего узла, опции заголовка переходов, перемещение адресов в заголовке маршрутизации и т. д.) в локальном кэше. Последующие пакеты с той же комбинацией адреса источника и метки потока обрабатываются с учётом информации кэша без детального анализа всех полей заголовка.
- Обеспечение безопасности в протоколе IPv6 осуществляется с использованием протокола IPSec, поддержка которого является обязательной для данной версии протокола.

Итак, IPv6 адрес имеет длину 128 бит. Это означает, что его запись будет гораздо более громоздкой и трудной для запоминания. Записываются адреса шестнадцатеричными цифрами, пример такого адреса может выглядеть так:

2001:0DB8:AA10:0001:0000:0000:0000:00FE

Для того, чтобы упростить запись адреса, применяют два правила. Во-первых, в каждом хекстете (группе из 4х цифр, ещё иногда ее называют тетрада) удаляются ведущие нули. После чего адрес принимает вид:

2001:DB8:AA10:1:0:0:0:FE

Далее, выбирается самая длинная группа, состоящая из полностью нулевых хекстетов (т.е. самая длинная последовательность 0:0:0) и заменяется на два двоеточия:

2001:DB8:AA10:1::FE

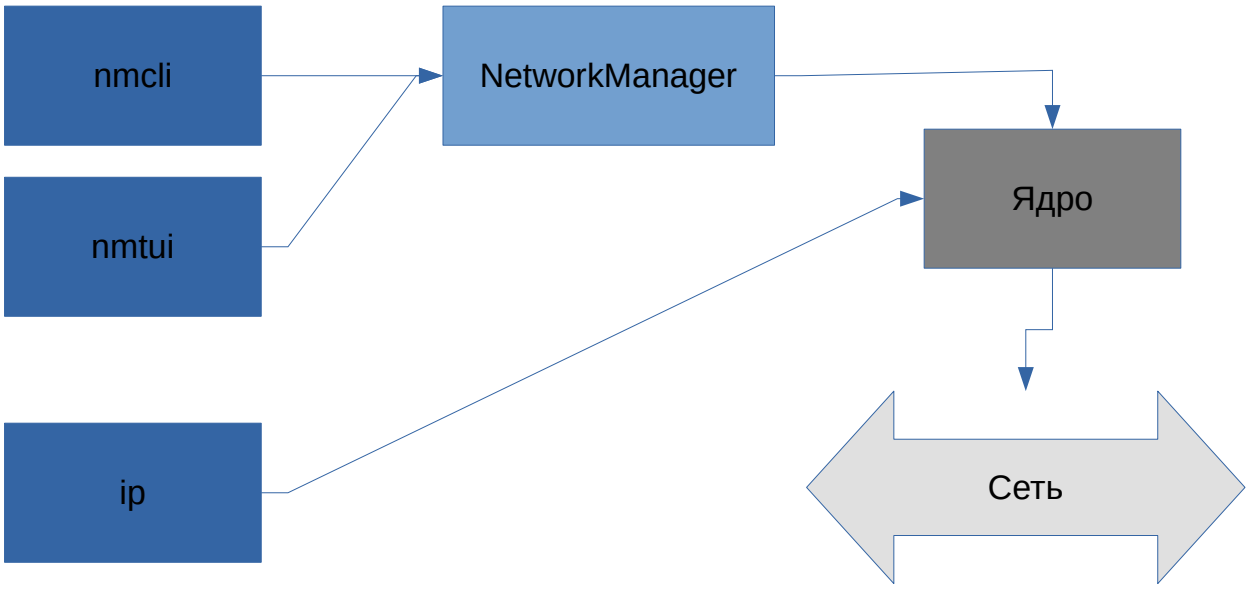
Это правило применимо только в случае, если нулевых групп несколько. Если одна (:0:) то она записывается как есть.

Таким образом, адрес loopback, выглядящий в IPv6 как 0000:0000:0000:0000:0000:0000:0000:0001 в итоге будет представлять собой ::1

При использовании IPv6 адреса в качестве URL, его необходимо заключать в квадратные скобки, при этом, если необходимо указать порт, то его следует писать за пределами скобок –

http://[2001:0db8:11a2:08d7:1f24:8a2в:05a0:745d]:8888

Просмотр настроек сети



Контроль над сетевыми соединениями и сетевыми устройствами осуществляется посредством специального компонента, называемого NetworkManager. Он реализован в виде демона, управлять работой которого можно с помощью утилит, таких как nmcli и nmtui.

С помощью этих инструментов можно управлять сетевыми подключениями: добавлять, удалять их, а также менять параметры имеющихся в наличии.

Также не стоит забывать более традиционные для Linux-систем средства. Скажем, для просмотра и изменения конфигурационных параметров сети можно использовать команду ip. Общий синтаксис команды ip следующий: ip [опции] объект {команда | help} Например:

```
[demo@instructor ~]$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    default qlen 1
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00    inet 127.0.0.1/8 brd
    127.255.255.255 scope host lo       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
      valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
    UP group default qlen 1000
   link/ether 08:00:27:dd:65:0f brd ff:ff:ff:ff:ff:ff
   inet 172.30.13.179/24 brd 172.30.13.255 scope global dynamic enp0s3       valid_lft
    26556sec preferred_lft 26556sec    inet6 fe80::a00:27ff:fedd:650f/64 scope link
    valid_lft forever preferred_lft forever
```

При таком использовании команда отобразит все сконфигурированные адреса на всех интерфейсах. Если нас интересует конфигурация какого-либо конкретного интерфейса, его имя можно задать в команде:

```
[demo@instructor ~]$ ip addr show enp0s3
```

Если вам доводилось работать с предыдущими версиями операционной системы, то вы наверняка обратили внимание на перемены, произошедшие с именами сетевых интерфейсов. Раньше они назывались eth0, eth1, eth2 и т.д.

Теперь имя интерфейса формируется в зависимости от ряда факторов. Первые две буквы будут en в случае ethernet интерфейса, wl для WLAN интерфейса и ww для WWAN интерфейса. Следом идет символ, определяющий тип подключения.

- o - интегрированный (on-board) интерфейс
- s - устройство с «горячим» подключением
- p - PCI-интерфейс

В конце располагается число, указывающее индекс, идентификатор либо порт.

Посмотреть данные о имеющихся в наличии сетевых интерфейсах можно так:

```
[demo@instructor ~]$ nmcli dev status
УСТРОЙСТВО  ТИП       СОСТОЯНИЕ         СОЕДИНЕНИЕ
enp0s3      ethernet  подключено        System enp0s3
enp0s8      ethernet  подключено        Проводное соединение 1
lo          loopback  без управления    --
```

Видно, что у нас есть три сетевых интерфейса. Один - loopback и два - Ethernet. Ethernet-интерфейсы работают через подключения с именами "System enp0s3" и "Проводное подключение 1". Такие имена обычно присваиваются автоматически.

Команды nmcli

nmcli dev status	Получение списка устройств
nmcli con show	Получение списка соединений
nmcli con up “имя_подключения”	Инициализация подключения
nmcli con down “имя_подключения”	Деактивация подключения.
nmcli con add...	Добавление подключения
nmcli con mod “имя_подключения”	Редактирование подключения
nmcli con del “имя_подключения”	Удаление подключения

Команда nmcli - универсальный инструмент. Её формат: «nmcli [ПАРАМЕТРЫ] ОБЪЕКТ { КОМАНДА | help }».
ПАРАМЕТРЫ: может принимать значения -t (вывод информации в сжатом виде) или -p (вывод информации с форматированием).
ОБЪЕКТ: может принимать значения: con (соединения NetworkManager) или dev (устройства под управлением NetworkManager).
КОМАНДА: действие над ОБЪЕКТОМ.
Для того, чтобы узнать, какие команды можно выполнить в отношении ОБЪЕКТА, можно воспользоваться командой «nmcli ОБЪЕКТ help».
Посмотреть данные о соединении можно так:

```
# nmcli con show "Auto Ethernet"
```

Создание, удаление и изменение параметров соединений осуществляется при помощи той же самой команды nmcli. При создании нужно указать имя соединения, его тип и интерфейс, который он будет использовать.

1. Вариант подключения с получением адреса от DHCP-сервера

```
# nmcli con add con-name "default" type ethernet ifname ens33
```

2. Вариант подключения с назначением статического адреса без автоподключения.

```
# nmcli con add con-name "static" type ethernet ifname ens33 autoconnect no  ip4 192.168.0.250/24 gw4 192.168.0.1
```

Команда nmcli сразу же сохраняет изменения в конфигурационном файле (т.е. результат её применения сохранится после перезагрузки).

Конфигурация сети

параметр nmcli	Директива ifcfg-*	Описание
ipv4.method manual (auto)	BOOTPROTO=none (dhcp)	IP-адрес задается статической конфигурацией (динамически)
ipv4.addresses “192.168.0.10/24 192.168.0.1”	IPADDR0=192.168.0.10 PREFIX0=24 GATEWAY0=192.168.0.1	Определение адреса, маски и шлюза. Если адресов несколько, 0 будет меняться на 1,2 и т.д
ipv4.dns 8.8.8.8	DNS0=8.8.8.8	Использовать указанный хост в качестве сервера имён
ipv4.dns-search example.com	DOMAIN=example.com	Определить имя домена для работы
ipv4.ignore-auto-dns true	PEERDNS=no	Не обновлять содержимое файла /etc/resolv.conf при получении данных от DHCPсервера
connection autoconnect yes	ONBOOT=yes	Инициализировать подключение при загрузке системы
connection.id eth0	NAME=eth0	Определить название соединение
connection.interface-name eth0	DEVICE=eth0	Связать соединение с устройством, имеющим указанное имя
802-3-ethernet.macaddress ...	HWADDR=...	Связать соединение с устройством, имеющим указанный MAC-адрес

- Информация о сетевых подключениях располагается в файлах, лежащих в /etc/sysconfig/network-scripts. Имена файлов имеют вид ifcfg-*, где вместо * расположено название подключения (Например, /etc/sysconfig/networkscripts/ifcfg-static).
- При подключении к удалённому сетевому узлу мы можем обратиться к нему как по адресу, так и по доменному имени (при условии, что в сети присутствует DNS-сервер). В этом случае у нас в системе должен быть определён адрес этого DNS-сервера. Для этих целей существует файл /etc/resolv.conf В файле находятся параметры search (определяет домен, в котором мы работаем) и nameserver (определяет адрес DNS-сервера). Параметров nameserver может быть от одного до трёх.
- Ручное редактирование файла /etc/resolv.conf не рекомендуется, т.к. при получении данных от DHCP, NetworkManager перезаписывает его содержимое и все сделанные вручную изменения будут утеряны.

Имя хоста и преобразование имён

hostnamectl

hostnamectl set-hostname test0.example.com

/etc/hosts

/etc/resolv.conf

Посмотреть имя хоста можно с помощью команды hostnamectl:

```
   Static hostname: test0
         Icon name: computer-desktop
           Chassis: desktop
        Machine ID: b1f5b888934347d6b45381bc0244a70b
           Boot ID: 26abcbe6453442e4871faee29c8b54e5
  Operating System: CentOS Linux 7 (Core)
            Kernel: Linux 3.10.0-123.el7.x86_64
      Architecture: x86-64
```

Изменить имя хоста можно с помощью той же самой команды hostnamectl:

```
sudo hostnamectl set-hostname workstation0.example.com
```

Данные при этом сохраняются в файле /etc/hostname (В предыдущих версиях системы это был другой файл - /etc/sysconfig/network).

Проверить, как работает преобразование имени хоста к IP-адресу можно с помощью команды host. Файл /etc/hosts сохранился ещё с тех времен, когда механизма DNS не существовало. Он хранит внутри себя таблицу соответствий IP-адресов именам хостов. Т.е. если запустить на локальном узле вебсервер и добавить в /etc/hosts строчку 127.0.0.1 mail.ru, то при попытке зайти браузером на http://mail.ru будет открыта страница локального вебсервера а не узла mail.ru.

При этом, существует возможность определить, в какой момент будет происходить чтение файла /etc/hosts - до запроса DNS-сервера или после? За это отвечает параметр hosts в файле /etc/nsswitch . Если он установлен в значение “files dns”, то сначала будет выполнено чтение файла /etc/hosts и только потом - запрос к DNS. Если поменять files и dns местами, то - наоборот. Можно оставить только значение “files” тогда работа с сетевыми узлами по их именам будет строиться только на основе содержимого /etc/hosts.

Задание 1

1) Используя известные вам инструменты (команды и конфигурационные файлы, отвечающие за работу сети), определите следующие параметры вашей системы:

IP-адрес

Маска

Шлюз

DNS-сервер(ы)

Сетевые интерфейсы

2) Уточните IP-адрес, используемый в системе вашего коллеги. Используя команду ping, проверьте сетевую доступность его узла