

# SDN

Software Defined Networks

Программно-конфигурируемые сети (ПКС)

Модель OSI				
Уровень (layer)		Тип данных (PDU <sup>[1]</sup> )	Функции	Примеры
Host layers	7. <a href="#">Прикладной (application)</a>	Данные	Доступ к сетевым службам	<a href="#">HTTP</a> , <a href="#">FTP</a> , <a href="#">POP3</a> , <a href="#">WebSocket</a>
	6. <a href="#">Представления (presentation)</a>		Представление и <a href="#">шифрование</a> данных	<a href="#">ASCII</a> , <a href="#">EBCDIC</a>
	5. <a href="#">Сеансовый (session)</a>		Управление сеансом связи	<a href="#">RPC</a> , <a href="#">PAP</a> , <a href="#">L2TP</a>
	4. <a href="#">Транспортный (transport)</a>	<a href="#">Сегменты (segment)</a> / <a href="#">Дейтаграммы (datagram)</a>	Прямая связь между конечными пунктами и надёжность	<a href="#">TCP</a> , <a href="#">UDP</a> , <a href="#">SCTP</a> , PORTS
Media <sup>[2]</sup> layers	3. <a href="#">Сетевой (network)</a>	<a href="#">Пакеты (packet)</a>	Определение маршрута и логическая адресация	<a href="#">IPv4</a> , <a href="#">IPv6</a> , <a href="#">IPsec</a> , <a href="#">AppleTalk</a>
	2. <a href="#">Канальный (data link)</a>	Биты (bit)/ <a href="#">Кадры (frame)</a>	Физическая адресация	<a href="#">PPP</a> , <a href="#">IEEE 802.22</a> , <a href="#">Ethernet</a> , <a href="#">DSL</a> , <a href="#">ARP</a> , <a href="#">сетевая карта</a> .
	1. <a href="#">Физический (physical)</a>	<a href="#">Биты (bit)</a>	Работа со средой передачи, сигналами и двоичными данными	<a href="#">USB</a> , кабель («витая пара», коаксиальный, оптоволоконный), радиоканал

# SDN: Введение

## **Hub: Хаб, Концентратор. layer 1 - OSI model**

Устаревший класс устройств для объединения компьютеров в сетях **Ethernet** с применением кабельной инфраструктуры типа ***витая пара***

**Концентратор** работает на **первом (физическом)** уровне **сетевой модели OSI**, ретранслируя входящий сигнал с одного из **портов** в сигнал на все остальные (подключённые) **порты**.

**Концентратор** обеспечивает бесперебойную работу **сети** при отключении устройства от одного из **портов** или повреждении кабеля, в отличие, например, от **сети на коаксиальном кабеле**, которая в таком случае прекращает работу целиком.

# SDN: Введение

**Switch: Свитч, Свич, Коммутатор. layer 2 (L2) - OSI model**

Устройство, предназначенное для соединения нескольких **узлов компьютерной сети** в пределах одного или нескольких **сегментов сети**.

**Коммутатор** работает на **канальном (втором) уровне модели OSI**.

**Коммутаторы** были разработаны с использованием мостовых технологий и часто рассматриваются как **многопортовые мосты**.

# SDN: Введение

**Switch: Свитч, Свич, Коммутатор. layer 2 (L2) - OSI model**

**Коммутатор** хранит в памяти (т.н. ассоциативной памяти) **таблицу коммутации**, в которой указывается соответствие **MAC**-адреса узла **порту коммутатора**.

При включении **коммутатора** эта таблица пуста, и он работает в режиме обучения. В этом режиме поступающие на какой-либо порт данные передаются на все остальные порты коммутатора. При этом коммутатор анализирует **фреймы (кадры)** и, определив **MAC**-адрес **хоста-отправителя**, заносит его в **таблицу** на некоторое время. Впоследствии, если на один из портов коммутатора поступит кадр, предназначенный для хоста, **MAC**-адрес которого уже есть в таблице, то этот кадр будет передан только через порт, указанный в таблице. Если **MAC**-адрес **хоста-получателя** не ассоциирован с каким-либо **портом коммутатора**, то **кадр** будет отправлен на **все порты**, за исключением того **порта**, с которого он был получен.

Со временем **коммутатор** строит таблицу для всех активных **MAC**-адресов, в результате трафик локализуется.

# SDN: Введение

**Router: Роутер, Маршрутизатор. layer3(L3) - OSI model**

Специализированный **компьютер**, который пересылает пакеты между различными **сегментами сети** на основе правил и **таблиц маршрутизации**. **Маршрутизатор** может связывать разнородные **сети** различных архитектур. Для принятия решений о пересылке пакетов используется информация о **топологии сети** и определённые правила, заданные **администратором**.

**Маршрутизаторы** работают на **сетевом** (третьем) уровне **сетевой модели OSI**, в отличие от **коммутаторов** (свитчей) и **концентраторов** (хабов), которые работают соответственно на **втором** и **первом** уровнях **модели OSI**.

# SDN: Введение

**Router: Роутер, Маршрутизатор. layer3(L3) - OSI model**

**Маршрутизатор** использует **адрес получателя**, указанный в **заголовке пакета**, и определяет по **таблице маршрутизации** путь, по которому следует передать данные. Если в **таблице маршрутизации** для адреса нет описанного **маршрута** — пакет отбрасывается.

Существуют и другие способы определения **маршрута** пересылки **пакетов**, когда используется **адрес отправителя**, используемые протоколы верхних **уровней** и другая информация, содержащаяся в **заголовках пакетов сетевого уровня**. **Маршрутизаторы** могут осуществлять **трансляцию адресов** отправителя и получателя, **фильтрацию** транзитного потока данных на основе определённых **правил** с целью **ограничения доступа**, шифрование/расшифровывание передаваемых данных и т. д.

# SDN: Введение

**Router: Роутер, Маршрутизатор. layer3(L3) - OSI model**

**Таблица маршрутизации** содержит информацию, на основе которой **маршрутизатор** принимает решение о дальнейшей пересылке пакетов. Таблица состоит из некоторого числа записей — маршрутов, в каждой из которых содержится **идентификатор сети получателя** (состоящий из **адреса** и **маски сети**), **адрес** следующего **узла**, которому следует передавать пакеты, **административное расстояние** - степень доверия к источнику маршрута и некоторый вес записи - **метрика**. **Метрики** записей в таблице играют роль в вычислении кратчайших **маршрутов** к различным получателям.

# SDN: Введение

**Router: Роутер, Маршрутизатор. layer3(L3) - OSI model**

**192.168.64.0/18 [110/49] via 192.168.1.2, 00:34:34, FastEthernet0/0.1**

где **192.168.64.0/18** – сеть назначения,  
**110/-** административное расстояние  
**/49** – метрика маршрута,  
**192.168.1.2** – адрес следующего маршрутизатора, которому следует  
передать пакеты для сети 192.168.64.0/18,  
**00:34:34** – время, в течение которого был известен этот маршрут,  
**FastEthernet0/0.1** – интерфейс маршрутизатора, через который можно  
достичь «соседа» **192.168.1.2**.

# SDN: Введение

## **Ethernet: layer 2 , MAC-адрес.**

Семейство технологий **пакетной** передачи данных между устройствами для **компьютерных** и **промышленных сетей**.

Метод управления доступом - **множественный доступ с контролем несущей и обнаружением коллизий (CSMA/CD, Carrier Sense Multiple Access with Collision Detection)**, скорость передачи данных **10 Мбит/с**, размер кадра от **64 до 1518** байт.

В 1995 году принят стандарт **IEEE 802.3u Fast Ethernet** со скоростью **100 Мбит/с** и появилась возможность работы в режиме **полный дуплекс**.

В 1997 году был принят стандарт **IEEE 802.3z Gigabit Ethernet** со скоростью **1000 Мбит/с** для передачи по **оптическому волокну** и ещё через два года для передачи по **витой паре**.

# SDN: Введение

Layer	Preamble	Start of frame delimiter	MAC destination	MAC source	802.1Q tag (optional)	EtherType (Ethernet II) or length (IEEE 802.3)	Payload	Frame check sequence (32-bit CRC)	Interpacket gap
	7 <a href="#">octets</a>	1 octet	6 octets	6 octets	(4 octets)	2 octets	46-1500 octets	4 octets	12 octets
Layer 2 Ethernet frame			← 64–1522 octets →						
Layer 1 Ethernet packet & IPG	← 72–1530 octets →								← 12 octets →

# SDN: Введение

## 10 Мбит/с Ethernet

- **10BASE5, IEEE 802.3** (называемый также «Толстый Ethernet») — первоначальная разработка технологии со скоростью передачи данных 10 Мбит/с. Следуя раннему стандарту, **IEEE** использует коаксиальный кабель с волновым сопротивлением 50 Ом (RG-8), с максимальной длиной сегмента 500 метров.
- **10BASE2, IEEE 802.3a** (называемый «Тонкий Ethernet») — используется кабель RG-58, с максимальной длиной сегмента 185 метров, компьютеры присоединялись один к другому, для подключения кабеля к сетевой карте нужен T-коннектор, а на кабеле должен быть BNC-коннектор. Требуется наличие терминаторов на каждом конце. Многие годы этот стандарт был основным для технологии **Ethernet**.
- **10BASE-T, IEEE 802.3i** — для передачи данных используется 4 провода кабеля витой пары (две скрученные пары) категории 3 или категории-5. Максимальная длина сегмента — 100 метров.
- **10BASE-F, IEEE 802.3j** — Основной термин для обозначения семейства 10 Мбит/с ethernet-стандартов, использующих оптический кабель на расстоянии до 2 километров: 10BASE-FL, 10BASE-FB и 10BASE-FP. Из перечисленного только 10BASE-FL получил широкое распространение.

## Быстрый Ethernet (Fast Ethernet, 100 Мбит/с)

- **100BASE-T** — общий термин для обозначения стандартов, использующих в качестве среды передачи данных витую пару. Длина сегмента — до 100 метров. Включает в себя стандарты 100BASE-TX, 100BASE-T4 и 100BASE-T2.
- **100BASE-TX, IEEE 802.3u** — развитие стандарта 10BASE-T для использования в сетях топологии «звезда». Задействована витая пара категории 5, фактически используются только две неэкранированные пары проводников, поддерживается дуплексная передача данных, расстояние до 100 м.

## Гигабитный Ethernet (Gigabit Ethernet, 1 Гбит/с)

- **1000BASE-T, IEEE 802.3ab** — основной гигабитный стандарт, опубликованный в 1999 году, использует витую пару категории 5е. В передаче данных участвуют 4 пары, каждая пара используется одновременно для передачи по обоим направлениям со скоростью — 250 Мбит/с. Используется метод кодирования PAM5 (*5-level Phase Amplitude Modulation*, пятиуровневая фазоамплитудная модуляция) с 4 линиями (4D-PAM5) и 4-мерной Треллис-модуляцией (TCM), частота основной гармоника — 62,5 МГц. Расстояние — до 100 метров.

## 10-гигабитный Ethernet (10G Ethernet, 10 Гбит/с)

- **10GBASE-T, IEEE 802.3an-2006** — принят в июне 2006 года после 4 лет разработки. Использует витую пару категории 6 (максимальное расстояние 55 метров) и 6а (максимальное расстояние 100 метров).

## 40-гигабитный и 100-гигабитный Ethernet

# SDN: Введение

## **TCP/IP - IP addressing and subnetting :**

**IP-адрес** - уникальный **сетевой адрес** узла в компьютерной сети, построенной на основе **стека протоколов TCP/IP**.

В сети **Интернет** требуется глобальная уникальность **адреса**;

В случае работы в **локальной сети** требуется уникальность адреса в пределах **сети**.

В версии протокола **IPv4** **IP-адрес** имеет длину 4 байта,

В версии протокола **IPv6** **IP-адрес** имеет длину 16 байт.

# SDN: Введение

## **TCP/IP - IP addressing and subnetting :**

**IPv4** использует **32-битные (четырёхбайтные)** адреса, ограничивающие адресное пространство 4 294 967 296 ( $2^{32}$ ) возможными уникальными адресами.

Традиционной формой записи **IPv4** адреса является запись в виде четырёх десятичных чисел (от 0 до 255), разделённых точками.

Через дробь указывается длина маски подсети.

# SDN: Введение

Форма записи	Пример	Преобразование из десятичной нотации с точками
Десятичная с точками (англ.)	192.0.2.235	—
Шестнадцатеричная с точками	0xC0.0x00.0x02.0xEB	Каждый октет преобразуется в шестнадцатеричную форму
Восьмеричная с точками	0300.0000.0002.0353	Каждый октет преобразуется в восьмеричную форму
Шестнадцатеричная	0xC00002EB	Конкатенация октетов из шестнадцатеричной нотации с точками
Десятичная	3221226219	32-битное число в десятичной форме
Восьмеричная	030000001353	32-битное число в восьмеричной форме

# SDN: Введение

1981, RFC 791, RFC 790 (англ.) (англ.) **классовая адресация**. Группы носят названия классы сетей и пронумерованы латинскими буквами: А, В, С, D и Е. Деление основывается на старших битах адреса.

**Класс А:** 0.XXX.XXX.XXX - 127.XXX.XXX.XXX

Первый бит адреса равен нулю, таким образом, класс А занимает половину всего адресного пространства. Адрес сети занимает 7 бит, адрес узла - 24 бита, следовательно Класс А содержит 128 подсетей по 16 777 216 адресов в каждой.

Например, подсеть 10.0.0.0 (класса А, содержит более 16,7 млн. адресов от 10.0.0.0 по 10.255.255.255). По умолчанию зарезервирована, не маршрутизируется в интернете и используется для построения локальных и корпоративных сетей.

**Класс В:** 128.0.XXX.XXX - 191.255.XXX.XXX

Адрес начинается с битов 1,0, таким образом, класс В занимает четверть всего адресного пространства. Адрес сети занимает 14 бит, адрес узла - 16, следовательно класс В содержит 16,384 подсетей по 65 536 адресов в каждой

Например, сеть 169.254.X.X класса В с 65536 адресами. Зарезервирована для "Канальных" адресов.

**Класс С:** 192.0.0.XXX - 223.255.255.XXX

Адрес начинается с битов 1,1,0, таким образом, класс С занимает 1/8 адресного пространства. Адрес сети занимает 21 бит, адрес узла - 8 бит, следовательно класс С содержит 2,097,152 сетей по 256 адресов в каждой.

Например, сеть 192.0.2.X имеет адреса с 192.0.2.0 по 192.0.2.255, зарезервирована для примеров в документации.

В 1990 году в RFC 1166 (англ.) описаны еще два класса.

**Класс D:** 224.XXX.XXX.XXX - 239.XXX.XXX.XXX

Адрес начинается с битов 1,1,1,0. Класс D занимает 1/16 адресного пространства. Используется для многоадресной рассылки.

**Класс Е:** 240.XXX.XXX.XXX - 255.XXX.XXX.XXX.

Адрес начинается с битов 1,1,1,1. Такие адреса запрещены. Зарезервировано для использования в будущем.

# SDN: Введение

Подсеть	Назначение	Маршрутизация
0.0.0.0/8	Адреса источников пакетов «этой» («своей») сети.	запрещена
0.0.0.0/32	В сокетах с состоянием «listening» обозначает любые IP отправителя. Не может быть использован как адрес назначения в сети.	запрещена
10.0.0.0/8	Для использования в частных сетях. RFC 1918.	только в частных сетях
100.64.0.0/10	RFC 6598. Для использования в сетях сервис-провайдера.	
127.0.0.0/8	Подсеть для коммуникаций внутри хоста.	запрещена

169.254.0.0/16	Канальные адреса. Подсеть используется для автоматического назначения IP операционной системой по DHCP.	только в частных сетях
172.16.0.0/12	Для использования в частных сетях. RFC 1918.	только в частных сетях
192.0.0.0/24	IETF Protocol Assignments	
192.0.0.0/29	Dual-Stack Lite (DS-Lite). RFC 6333. IPv6 transition mechanisms	
192.0.0.170/32	NAT64	
192.0.0.171/32	DNS64	
192.0.2.0/24	Для примеров в документации.	запрещена
192.88.99.0/24	Используются для рассылки ближайшему узлу. RFC 3068	глобально разрешена

192.88.99.1/32	Ретранслятор при инкапсуляции IPv6 в IPv4. IP ближайшего хоста, который распакует пакет и отправит его по IPv6 маршруту.	глобально разрешена
192.168.0.0/16	Для использования в частных сетях. RFC 1918.	только в частных сетях
198.51.100.0/24	Для примеров в документации.	запрещена
198.18.0.0/15	Для стендов тестирования производительности.	только для тестов
203.0.113.0/24	Для примеров в документации.	запрещена
224.0.0.0/4	Используются для многоадресной рассылки. RFC 5771.	глобально разрешена только для подсетей 233.0.0.0/8 и 234.0.0.0/8.
240.0.0.0/4	Зарезервировано для использования в	запрещена

# SDN: Введение

**Бесклассовая адресация** (англ. *Classless Inter-Domain Routing, CIDR*) — метод IP-адресации, позволяющий гибко управлять пространством IP-адресов, не используя жёсткие рамки **классовой адресации**.

IP-адрес является массивом бит. **Маска подсети** задает какие биты в IP-адресе являются **адресом сети**.

**Блок адресов** задаётся указанием **начального адреса** и **маски подсети**.

**Бесклассовая адресация** основывается на переменной длине маски подсети (англ. *variable length subnet mask, VLSM*), в то время, как в **классовой адресации** длина маски подсети имела всего лишь 3 фиксированных значения.

# SDN: Введение

Октеты IP-адреса	192								0								2				32													
Биты IP-адреса	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0		
Биты маски подсети	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Октеты маски подсети	255								255								255				224													

# SDN: Введение

IP/маска	До последнего IP в подсети	Маска	Всего адресов	<u>Узловых</u> адресов	Класс
a.b.c.d/32	+0.0.0.0	255.255.255.255	1	(нет)	1/256 C
a.b.c.d/31	+0.0.0.1	255.255.255.254	2	2 <sup>[1]</sup>	1/128 C
a.b.c.d/30	+0.0.0.3	255.255.255.252	4	2	1/64 C
a.b.c.d/29	+0.0.0.7	255.255.255.248	8	6	1/32 C
a.b.c.d/28	+0.0.0.15	255.255.255.240	16	14	1/16 C
a.b.c.d/27	+0.0.0.31	255.255.255.224	32	30	1/8 C
a.b.c.d/26	+0.0.0.63	255.255.255.192	64	62	1/4 C
a.b.c.d/25	+0.0.0.127	255.255.255.128	128	126	1/2 C
a.b.c.0/24	+0.0.0.255	255.255.255.000	256	254	1 C

# SDN: Введение

<b>a.b.c.0/23</b>	<b>+0.0.1.255</b>	<b>255.255.254.000</b>	<b>512</b>	<b>510</b>	<b>2 C</b>
a.b.c.0/22	+0.0.3.255	255.255.252.000	1024	1022	4 C
a.b.c.0/21	+0.0.7.255	255.255.248.000	2048	2046	8 C
a.b.c.0/20	+0.0.15.255	255.255.240.000	4096	4094	16 C
a.b.c.0/19	+0.0.31.255	255.255.224.000	8192	8190	32 C
a.b.c.0/18	+0.0.63.255	255.255.192.000	16 384	16 382	64 C
a.b.c.0/17	+0.0.127.255	255.255.128.000	32 768	32 766	128 C
a.b.0.0/16	+0.0.255.255	255.255.000.000	65 536	65 534	256 C = 1 B

# SDN: Введение

<b>a.b.0.0/15</b>	<b>+0.1.255.255</b>	<b>255.254.000.000</b>	<b>131 072</b>	<b>131 070</b>	<b>2 B</b>
a.b.0.0/14	+0.3.255.255	255.252.000.000	262 144	262 142	4 B
a.b.0.0/13	+0.7.255.255	255.248.000.000	524 288	524 286	8 B
a.b.0.0/12	+0.15.255.255	255.240.000.000	1 048 576	1 048 574	16 B
a.b.0.0/11	+0.31.255.255	255.224.000.000	2 097 152	2 097 150	32 B
a.b.0.0/10	+0.63.255.255	255.192.000.000	4 194 304	4 194 302	64 B
a.b.0.0/9	+0.127.255.255	255.128.000.000	8 388 608	8 388 606	128 B
a.0.0.0/8	+0.255.255.255	255.000.000.000	16 777 216	16 777 214	256 B = 1 A
a.0.0.0/7	+1.255.255.255	254.000.000.000	33 554 432	33 554 430	2 A
a.0.0.0/6	+3.255.255.255	252.000.000.000	67 108 864	67 108 862	4 A
a.0.0.0/5	+7.255.255.255	248.000.000.000	134 217 728	134 217 726	8 A
a.0.0.0/4	+15.255.255.255	240.000.000.000	268 435 456	268 435 454	16 A
a.0.0.0/3	+31.255.255.255	224.000.000.000	536 870 912	536 870 910	32 A
a.0.0.0/2	+63.255.255.255	192.000.000.000	1 073 741 824	1 073 741 822	64 A
a.0.0.0/1	+127.255.255.255	128.000.000.000	2 147 483 648	2 147 483 646	128 A
0.0.0.0/0	+255.255.255.255	000.000.000.000	4 294 967 296	4 294 967 294	256 A

# SDN: Введение

**Transmission Control Protocol** (TCP, протокол управления передачей)

Один из основных протоколов передачи данных интернета, предназначенный для управления передачей данных.

В стеке протоколов **TCP/IP** выполняет функции транспортного уровня **модели OSI**.

**TCP** реализует управление потоком, управление перегрузкой, рукопожатие, надёжную передачу.

# SDN: Введение

Бит	0 — 3	4 — 9	10 — 15	16 — 31
0	Порт источника, Source Port			Порт назначения, Destination Port
32	Порядковый номер, Sequence Number (SN)			
64	Номер подтверждения, Acknowledgment Number (ACK SN)			
96	Длина заголовка, (Data offset)	Зарезервировано	Флаги	Размер Окна, Window size
128	Контрольная сумма, Checksum			Указатель важности, Urgent Point
160	Опции (необязательное, но используется практически всегда)			
160/192+	Данные			

## Состояния сеанса TCP

CLOSED	Начальное состояние узла. Фактически фиктивное
LISTEN	Сервер ожидает запросов установления соединения от клиента
SYN-SENT	Клиент отправил запрос серверу на установление соединения и ожидает ответа
SYN-RECEIVED	Сервер получил запрос на соединение, отправил ответный запрос и ожидает подтверждения
ESTABLISHED	Соединение установлено, идёт передача данных
FIN-WAIT-1	Одна из сторон (назовём её узел-1) завершает соединение, отправив сегмент с флагом FIN
CLOSE-WAIT	Другая сторона (узел-2) переходит в это состояние, отправив, в свою очередь сегмент ACK и продолжает одностороннюю передачу
FIN-WAIT-2	Узел-1 получает ACK, продолжает чтение и ждёт получения сегмента с флагом FIN
LAST-ACK	Узел-2 заканчивает передачу и отправляет сегмент с флагом FIN

# SDN: Введение

**UDP** (англ. *User Datagram Protocol* — протокол пользовательских датаграмм) — один из ключевых элементов набора **сетевых протоколов** для Интернета. С **UDP** компьютерные приложения могут посылать сообщения (в данном случае называемые датаграммами) другим хостам по **IP**-сети без необходимости предварительного сообщения для установки специальных каналов передачи или путей данных. Протокол был разработан Дэвидом П. Ридом в 1980 году и официально определён в **RFC 768**.

# SDN: Введение

**UDP** не предоставляет никаких гарантий доставки сообщения для вышестоящего протокола и не сохраняет состояния отправленных сообщений. По этой причине **UDP** иногда называют **Unreliable Datagram Protocol** (англ. — Ненадёжный протокол датаграмм).

**UDP** обеспечивает **многоканальную передачу** (с помощью номеров портов) и **проверку целостности** (с помощью контрольных сумм) заголовка и существенных данных.

Надёжная передача в случае необходимости должна реализовываться пользовательским приложением.

# SDN: Введение

Биты	0 - 15	16 - 31
0-31	Порт отправителя (Source port)	Порт получателя (Destination port)
32-63	Длина датаграммы (Length)	Контрольная сумма (Checksum)
64-...	Данные (Data)	

# SDN: Введение

**ICMP** (англ. *Internet Control Message Protocol* — протокол межсетевых управляющих сообщений) — сетевой протокол, входящий в **стек протоколов TCP/IP**.

В основном **ICMP** используется для передачи сообщений об ошибках и других исключительных ситуациях, возникших при передаче данных, например, запрашиваемая услуга недоступна, или **хост**, или **маршрутизатор** не отвечают.

Также на **ICMP** возлагаются некоторые сервисные функции.

# SDN: Введение

Протокол **ICMP** описан в **RFC 792** от 1981 года (с дополнениями в **RFC 950**). **ICMP** является **стандартом Интернета** (входит в стандарт **STD 5** вместе с **IP**).

Хотя формально протокол использует **IP** (**ICMP**-пакеты инкапсулируются в **IP** пакеты), он является неотъемлемой частью **IP** и обязателен при реализации стека **TCP/IP**.

Текущая версия **ICMP** для **IPv4** называется **ICMPv4**. В **IPv6** существует аналогичный протокол **ICMPv6**.

# SDN: Введение

**ICMP**-сообщение строится из **IP**-пакетов, сгенерировавших **ICMP**-ответ. Протокол **IP** инкапсулирует соответствующее **ICMP**-сообщение с новым заголовком **IP** (чтобы отправить **ICMP**-сообщение обратно отправителю) и передает полученные пакеты дальше.

Например, каждая машина (такая, как **маршрутизатор**), которая перенаправляет **IP**-пакеты, уменьшает **Time to live (TTL)** поля заголовка **IP** на единицу, если **TTL** достигает **0**, **ICMP**-сообщение о превышении **TTL** отправляется на источник пакета.

# SDN: Введение

Октет (байт)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
[0—3]	Тип									Код									Контрольная сумма													
...	Данные (формат зависит от значений полей «Код» и «Тип»)																															

**ICMP-сообщения (тип 12)** генерируются при нахождении ошибок в заголовке **IP-пакета** (за исключением самих **ICMP-пакетов**, дабы не привести к бесконечно растущему потоку **ICMP-сообщений** об **ICMP-сообщениях**).

**ICMP-сообщения (тип 3)** генерируются **маршрутизатором** при отсутствии маршрута к адресату.

Утилита **Ping**, служащая для проверки возможности доставки **IP-пакетов**, использует **ICMP-сообщения** с **типом 8** (эхо-запрос) и **0** (эхо-ответ).

Утилита **Traceroute**, отображающая путь следования **IP-пакетов**, использует **ICMP-сообщения** с **типом 11**.

**ICMP-сообщения с типом 5** используются **маршрутизаторами** для обновления записей в **таблице маршрутизации отправителя**.

**ICMP-сообщения с типом 4** используются получателем (или **маршрутизатором**) для управления скоростью отправки сообщений отправителем.

# SDN: Введение

## Правила генерации **ICMP**-пакетов

- При потере **ICMP**-пакета никогда не генерируется новый.
- **ICMP**-пакеты никогда не генерируются в ответ на **IP**-пакеты с **широковещательным** или **групповым адресом**, чтобы не вызывать перегрузку в сети (так называемый «**широковещательный шторм**»).
- При повреждении фрагментированного **IP**-пакета **ICMP**-сообщение отправляется сразу после получения первого повреждённого фрагмента, поскольку отправитель всё равно повторит передачу всего **IP**-пакета целиком.

# SDN: Введение

**Data center - Центр Обработки Данных, ЦОД**

**Elastic resources - "Резиновые" ресурсы**

- Плата только за предоставляемые ресурсы
- Инфраструктура предоставляется по требованию клиента

**Multi-tenancy - Многопользовательские системы**

- Каждый пользователь независимо потребляет услуги **ЦОД**
- В счет клиента включены коммунальные платежи на амортизацию и обновление оборудования.

# SDN: Введение

**Data center - Центр Обработки Данных, ЦОД**

Модели сервисов предоставляемых **ЦОД**

**Software as a Service (SaaS) - ПО как сервис**

- Услуга лицензирования приложений для пользователей
- **ЦОД** берет на себя все расходы связанные с установкой, техническим обслуживанием, обновлением пользовательских приложений.

# SDN: Введение

**Data center - Центр Обработки Данных, ЦОД**

Модели сервисов предоставляемых **ЦОД**

**Platform as a Service (PaaS)** - Платформа как сервис

- **ЦОД** предлагает программную платформу (Операционную систему, Базу Данных, "Движок" - Портальные Системы управления сайтом, etc...) для построения системы пользовательских приложений.
- **ЦОД** берет на себя проблемы масштабирования систем.

# SDN: Введение

**Data center - Центр Обработки Данных, ЦОД**

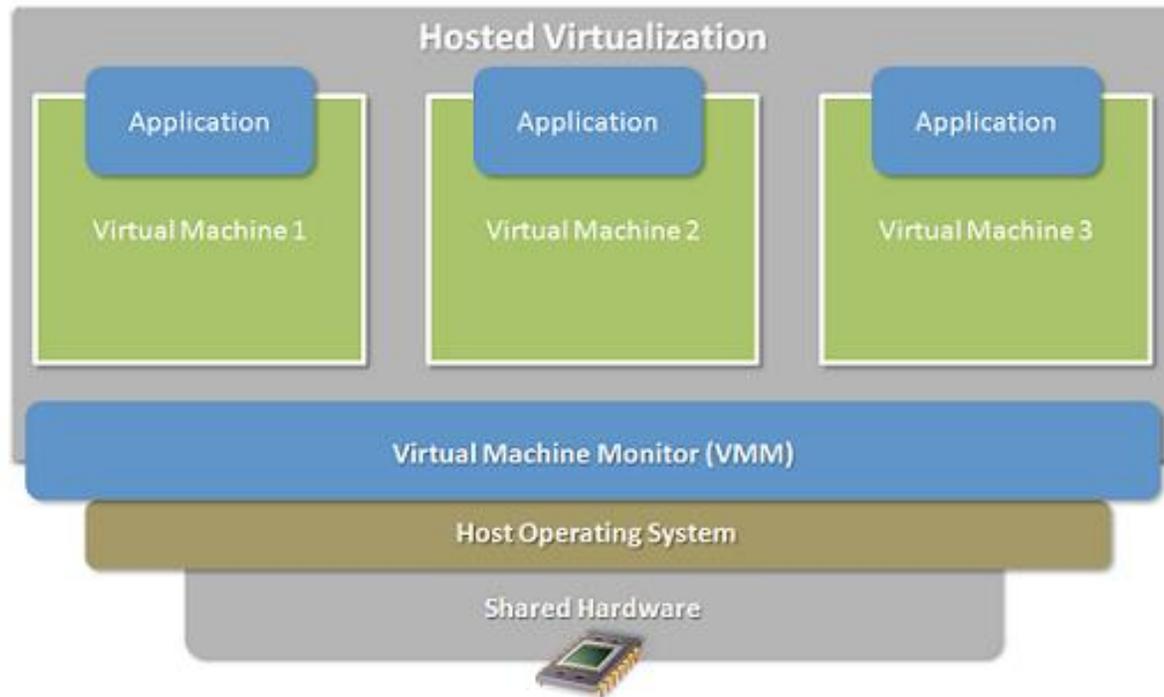
Модели сервисов предоставляемых **ЦОД**

**Infrastructure as a Service (IaaS)** - Инфраструктура как сервис

- **ЦОД** сдает в аренду компьютерное оборудование, системы хранения данных, сетевые устройства и комплексы (аппаратные или виртуальные)
- Избавляет клиента от покупки собственного оборудования, расчета необходимых ресурсов, бюджетных трат на масштабирование парка оборудования.

# SDN: Введение

**Data center - Центр Обработки Данных, ЦОД**



# SDN: Введение

## **Data center - Центр Обработки Данных, ЦОД**

Основные принципы виртуализации:

- Множество виртуальных машин на одном аппаратном компьютерном оборудовании.
- Для приложений виртуальные машины ничем не отличаются от аппаратных.
- Виртуальные машины можно переносить (мигрировать) на другое аппаратное компьютерное оборудование без изменений и потерь.

# SDN: Введение

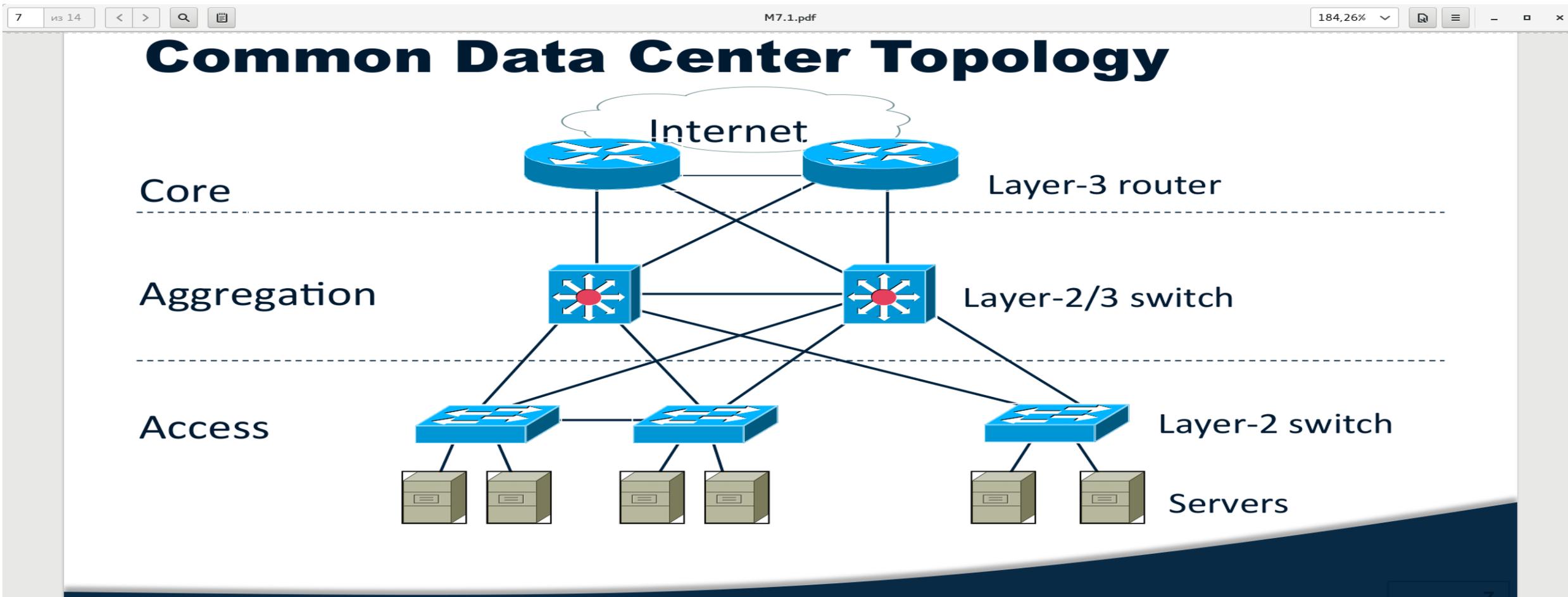
## **Data center - Центр Обработки Данных, ЦОД**

Требования к создателям и разработчикам ЦОД:

- Простой и понятный процесс переноса VM на оборудование внутри **ЦОД** (или между различными **ЦОД**).
- Минимальная конфигурация сетевого оборудования.
- Эффективная коммуникация между различными устройствами передачи данных.
- Отсутствие петель.
- Быстрое определение неисправностей и их устранение.

# SDN: Введение

## Data center - Центр Обработки Данных, ЦОД



# SDN: Введение

**Data center - Центр Обработки Данных, ЦОД**

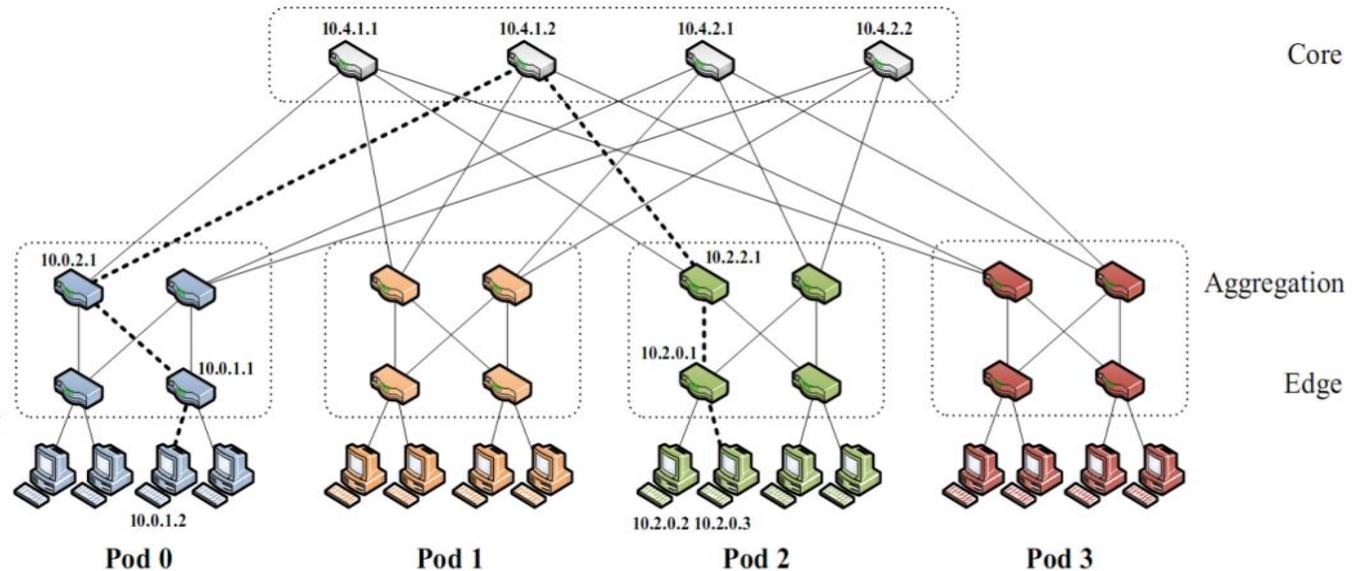
Проблемы общей топологии:

- Единая точка падения (неисправности).
- Большая нагрузка на линки в верхней части топологии.
- Высокая стоимость настройки системы не позволяет снижать цены услуг

# SDN: Введение

## Fat-Tree (Clos) Topology

- Multi-rooted tree topology
- Capacity increases towards the root(s) of the tree
- Inherent fault tolerance



# SDN: Введение

**MPLS - multiprotocol label switching**, многопротокольная коммутация по меткам - механизм в высокопроизводительной телекоммуникационной сети, осуществляющий передачу данных от одного узла сети к другому с помощью меток.

В сети, основанной на **MPLS**, пакетам данных присваиваются **метки**. Решение о дальнейшей передаче пакета данных другому узлу сети осуществляется только на основании значения присвоенной метки без необходимости изучения самого пакета данных. За счёт этого возможно создание сквозного виртуального канала, независимого от **среды передачи** и использующего любой **протокол передачи данных**.

# SDN: Введение

Технология **MPLS** используется для построения **IP-сетей**.

**MPLS** используется для передачи трафика **IP** и **Ethernet**.

Основными областями применения **MPLS** являются:

- оптимизация и управление трафиком - ***traffic engineering, TE***;
- организация виртуальных частных сетей - **VPN**.

# SDN: Введение

Технология **MPLS** основана на обработке заголовка **MPLS**, добавляемого к каждому пакету данных. Заголовок **MPLS** может состоять из одной или нескольких «меток». Несколько записей (**меток**) в заголовке **MPLS** называются **стеком меток**.

32 бита			
20 бит	3 бита	1 бит	8 бит
Label	TC	S	TTL

- поле **traffic class** используется для реализации механизмов качества обслуживания (**QoS**) и явного уведомления о перегрузке - **explicit congestion notification, ECN**
- флаг **bottom of stack**; если флаг установлен в 1, то это означает, что текущая метка последняя в стеке; занимает 1 бит

# SDN: Введение

В **MPLS-маршрутизаторе** пакет с **MPLS-меткой** коммутируется на следующий порт после поиска метки в таблице коммутации вместо поиска по таблице маршрутизации - **RIB, routing information base**. Коммутация может быть выполнена непосредственно на **коммутационной фабрике** вместо **центрального процессора**. **Маршрутизаторы**, расположенные на входе или выходе **MPLS-сети**, называются **LER - *label edge router***, **граничный маршрутизатор меток**). **LER** на входе в **MPLS-сеть** добавляют метку **MPLS** к пакету данных, а на выходе удаляют метку.

Маршрутизаторы, выполняющие маршрутизацию пакетов данных, основываясь только на значении метки, называются **LSR - *label switching router***.

Метки между **LER** и **LSR** распределяются с помощью **LDP - *label distribution protocol***.

Для того, чтобы получить полную картину **MPLS-сети**, **LSR** постоянно обмениваются метками и информацией о каждом соседнем узле, используя стандартную процедуру.

Виртуальные каналы (туннели), называемые **LSP - *label switch path***, устанавливаются провайдерами для решения различных задач, например, для организации **VPN** или для передачи трафика через сеть **MPLS** по указанному туннелю.

При описании **VPN**, основанных на технологии **MPLS**, **LER**, расположенные на входе или выходе сети, обычно называются **PE-маршрутизаторами - *provider edge***, а узлы, работающие как транзитные маршрутизаторы, называются **P-маршрутизаторами - *provider***.

# SDN: Введение

**NBI - Northbound interface, Северный интерфейс**

В программировании это **программный интерфейс** для представления низкоуровневых деталей вышестоящему в **архитектуре системы** приложению.

**Северный интерфейс** обычно рисуют наверху архитектурной схемы.

Примерами **NBI** являются **JSON** и **Thrift**

# SDN: Введение

## **SBI - Southbound interface, Южный интерфейс**

В программировании это **программный интерфейс** для обращения к нижестоящему в **архитектуре системы**, приложению.

Логический путь, как правило, от общего к деталям.

**Южные интерфейсы** рисуют внизу **архитектурной схемы**.

**Северные интерфейсы** могут обращаться к **южным интерфейсам** компонентов более высокого уровня и наоборот.

# SDN: Введение

## **Network topology - Сетевая топология, Топология сети**

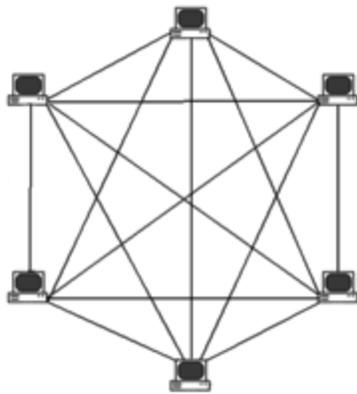
Конфигурация **графа**, вершинам которого соответствуют конечные **узлы сети** (компьютеры) и коммуникационное оборудование (маршрутизаторы), а рёбрам - физические или информационные **связи** между вершинами.

**Сетевая топология** может быть

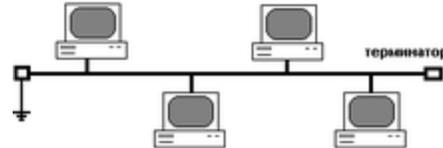
- **физической** — описывает реальное расположение и связи между узлами сети.
- **логической** — описывает хождение сигнала в рамках физической топологии.
- **информационной** — описывает направление потоков информации, передаваемых по сети.
- **управления обменом** — это принцип передачи права на пользование сетью.

# SDN: Введение

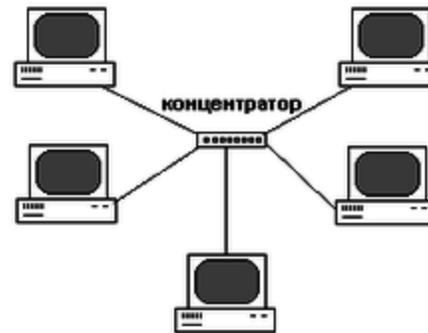
## Network topology - Сетевая топология, Топология сети



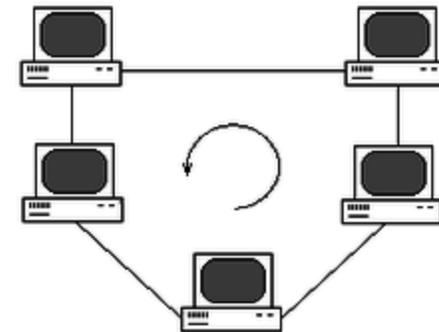
Полносвязная  
Fully connected



Шина  
Bus



Звезда  
Star



Кольцо  
Ring

# SDN: Введение

**API - *application programming interface*, программный интерфейс приложения, интерфейс прикладного программирования**

Описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой. Обычно входит в описание какого-либо **интернет-протокола** (например, **RFC**), программного **каркаса (фреймворка)** или стандарта вызовов функций операционной системы.

Реализуется отдельной **программной библиотекой** или **сервисом операционной системы**. Используется программистами при написании всевозможных приложений.

# SDN: Введение

**API** - *application programming interface*, программный интерфейс приложения, интерфейс прикладного программирования

**API** определяет функциональность, которую предоставляет программа/модуль/библиотека, при этом **API** позволяет абстрагироваться от того, как именно эта функциональность реализована.

Если программу/модуль/библиотеку рассматривать как чёрный ящик, то **API** - это множество «ручек», которые доступны пользователю данного ящика.

Программные компоненты взаимодействуют друг с другом посредством **API**. При этом обычно компоненты образуют иерархию — высокоуровневые компоненты используют **API** низкоуровневых компонентов, а те, в свою очередь, используют **API** ещё более низкоуровневых компонентов.

По такому принципу построены протоколы передачи данных. Как в сетевая модель OSI, каждый уровень пользуется функциональностью предыдущего (нижележащего) уровня передачи данных и, в свою очередь, предоставляет нужную функциональность следующему (вышележащему) уровню.

Понятие **протокола** близко по смыслу к понятию **API**. И то, и другое является абстракцией функциональности, только в первом случае речь идёт о **передаче данных**, а во втором - о взаимодействии приложений.

# SDN: Введение

**API - *application programming interface*, программный интерфейс приложения, интерфейс прикладного программирования**

API библиотеки функций и классов включает в себя описание **сигнатур** и **семантики** функций.

**Сигнатура функции** — часть общего объявления функции, позволяющая средствам трансляции идентифицировать функцию среди других.

**Семантика функции** — это описание того, что данная функция делает. Семантика функции включает в себя описание того, что является результатом вычисления функции, как и от чего этот результат зависит. Полным описанием семантики функций является исполняемый код функции или математическое определение функции.