1 Компьютерные сети и Internet. Основные термины и определения.

1. Интернет с точки зрения составных частей:
   оконечная система (end system, host, хост)
2. маршрутизатор (router, gateway)
3. линия связи
4. пропускная способность линии связи (bandwidth)
5. задержка, вносимая линией связи (latency)
6. маршрут (путь, route)
7. пакет
8. поставщик услуг Интернет (провайдер, ISP)
9. протокол
10. распределенные приложения
11. Интернет с точки зрения предоставляемых услуг:
    передача данных без установления соединения
12. передача данных с установлением соединения

2 Протокол. Примеры.

1. Определение протокола:
   формат сообщений, порядок сообщений, действия, выполняемые при передаче и/или приеме сообщений или наступлении иных событий
2. Пример с запросом времени
3. Что произойдет при нарушении требований протокола (пример неудавшегося взаимодействия, когда нарушен порядок сообщений или нарушен формат, или ошибочные действия)
4. Пример с запросом файла по HTTP
5. Схематическое изображение протокола (две взаимодействующие стороны обозначены вертикальными линиями, время идет сверху вниз, сообщения обозначены стрелками, направление стрелки показывает, кто передает, а кто принимает сообщение).

3 Оконечные системы, архитектура клиент-сервер, сервисы с установлением соединения и без установления соединения

1. оконечная система (end system, хост), неформальное определение
2. примеры различных оконечных систем (ПК, ноутбуки, PDA, холодильники, веб-камеры со своим IP и управлением по HTTP и т.п.)

3. периферия сети (network edge)
4. клиент, сервер
5. клиент потребляет услуги (сервисы), сервер предоставляет услуги
6. услуга передачи данных с установлением логического соединения, время, которое требуется на передачу
   $T = T_{conn} + \frac{S}{R}$, где $T_{conn} = const$ - время установления соединения, S - объем данных, R - пропускная способность
7. передача данных без установления логического соединения, $T = \frac{S+h}{R}$, где $h = const$ - заголовок пакета.
8. что будет при маленьком S (единицы байт) ?
9. что будет при большом S (несколько Гб) ?

4 Коммутация каналов, коммутация пакетов, коммутация сообщений, маршрутизация

1. оконечные системы, маршрутизаторы, каналы связи
2. ядро сети (network core)
3. коммутация каналов, типичная структура сети с коммутацией каналов (несколько узлов, узлы соединяются линиями связи, каждая из которых состоит из n каналов
4. каким образом линия связи делится на несколько каналов - мультиплексирование по частоте - FDM, по времени - TDM, по коду - CDMA
5. коммутация пакетов
6. действия маршрутизатора (принять, определить, в какую из линий связи отправить, отправить)
7. сравнение коммутации пакетов и коммутации каналов
8. сегментирование сообщений, зачем нужно делить данные на порции небольшого размера
9. пример с временем передачи данных в системе с двумя маршрутизаторами без деления сообщения на пакеты и с делением на пакеты

5 Сети доступа и среды передачи данных

1. Резидентный доступ (“по месту жительства”)
2. модем, типичная пропускная способность
3. (A)DSL, типичная пропускная способность
4. Корпоративный доступ
5. Ethernet, пропускная способность (10-100-1000)
6. Мобильный доступ

7. IEEE 802.11b/g (Wi-Fi), пропускная способность
8. GPRS, пропускная способность, большая задержка
9. Физические среды передачи:
медная витая пара
10. Коаксиальный кабель (50 Ом, 75 Ом)
11. Оптоволоконный кабель
12. Территориальные радиоканалы
13. Спутниковые радиоканалы
14. Другие (экзотические) способы (ИК-лазер, лазерные указки, голубиная почта и т.д.)

6 Причины задержек и потерь в сетях с коммутацией пакетов

1. Задержка узловой обработки
2. Задержка ожидания
3. Задержка передачи
4. Задержка распространения
5. Зависимость средней задержки ожидания от интенсивности трафика $\frac{L \cdot a}{R}$, где L - средняя длина пакета в битах, a - средняя частота получения пакетов (пакетов/с), R - пропускная способность исходящей линии связи, бит/с
6. Потеря пакетов как следствие заполнения входного буфера маршрутизатора (из-за большой задержки ожидания т.е. слишом большая длина очереди пакетов, ожидающих обработки)
7. общая задержка - сумма задержек обработки, ожидания, передачи и распространения.
8. утилиты ping и traceroute

7 Уровни протоколов, стек протоколов Internet. Иерархия ISP.

1. пример многоуровневой структуры из мира людей (доставка почты и/или пример с аэропортом)
2. уровни протоколов, для чего понадобилось делить на уровни
3. единица обмена (Protocol Data Unit, PDU), на уровне n - PDU_n.
4. Обобщенный пример передачи сообщения (несколько уровней, сначала вниз, на каждом уровне i добавляется заголовок H_i и получается PDU_i, который передается нижележащему уровню, в самом низу - передача на физическом уровне, потом вверх)

5. Стек протоколов Internet, 5 уровней (прикладной, транспортный, сетевой, канальный, физический)

6. PDU для этих уровней (сообщение, сегмент, дейтаграмма, кадр).

8 Требования приложений. Сервисы, предоставляемые TCP и UDP

1. Надежная передача данных:
приложения, которым нужнв надежная передача (примеры таких приложений)

2. приложения, *толерантные к потерям данных* (примеры таких приложений)

3. Скорость передачи:
приложения, которым необходима определенная пропускная способность сети (примеры таких приложений)

4. *эластичные* приложения (примеры таких приложений)

5. Время передачи:
приложения реального времени (Интернет-телефония, игры), которым важно время доставки (примеры таких приложений)

6. приложения, в которых нет принципиальных ограничений на время доставки (примеры таких приложений)

7. Протокол TCP:
установление логического соединения

8. надежная передача данных

9. контроль потока данных (flow control)

10. Протокол UDP:
нет надежной передачи, нет соединения, нет контроля потока

11. Примеры приложений и протоколы, которые они используют (электронная почта, удаленный доступ, Web, передача файлов, Интернет-телефония)

9 Протокол HTTP

1. URL, общий вид:
протокол://login:пароль@host:port/путь/к/документу#якорь
(якорь необязательно)

2. Примеры URL (http:// ftp:// email: gopher:// file:// и т.п.)

3. Порядок сообщений в HTTP для непостоянного соединения (HTTP/0.9, 1.0)

4. Порядок сообщений в HTTP для постоянного (persistent) соединения (HTTP/1.1)

5. Порядок сообщений в HTTP для постоянного соединения с конвейеризацией (pipeline)

6. Формат HTTP-запроса: строка запроса, строки заголовка, пустая строка, данные
7. Методы (минимум - GET, HEAD, POST)
8. При методе GET в запросе нет данных, в ответе должны быть данные. При методе HEAD ни в запросе ни в ответе нет данных. При методе POST данные есть и в запросе и в ответе.
9. В каких случаях удобнее GET, а в каких POST
10. Версии (0.9, 1.0, 1.1)
11. Примеры строк заголовка (Host, User-Agent, Referer)
12. Формат HTTP-ответа: строка статуса, строки заголовка, пустая строка, данные.
13. Расшифровка кода в строке статуса (2xx - успех, 3xx - нефатальная ошибка, 4xx - фатальная ошибка, 5xx - ошибка сервера)
14. Обязательная строка заголовка с Content-type - определяет MIME-тип передаваемого объекта
15. Примеры других строк заголовка (Server, Last-Modified, Content-Length и т.п.)

10 Аутентификация в HTTP, cookies, условный GET в HTTP

1. Аутентификация
2. Cookie, как работает:
   Set-cookie: в ответе, Cookie: в ответе
3. база broser-a с cookies, каждая запись содержит url, время действия, значение cookie.
4. Cookie, для чего может использоваться (сохранение каких-либо данных на компьютере клиента и передача этих данных на сервер при каждом посещении)
5. Метод GET с условием, как работает:
   Last-Modified в ответе, If-Modified-Since в запросе
6. для чего применяется (кэш обозревателя, при первом посещении страница сохраняется в кэше, url и Last-Modified сохраняются в базе данных обозревателя, при последующих посещениях если документ не менялся, то он не скачивается, а используется локальнвя копия)

11 Протокол FTP

1. Схема передачи данных по протоколу FTP (PI - protocol interpreter, DTP - data transfer process, у клиента и сервера у каждого по PI, между PI постоянно установлено управляющее соединение, между DTP соединение устанавливается только при передаче, DTP читает/сохраняет данные с/на локального/ый диск. На клиентской стороне есть UI - user interface.

2. Схема передачи, когда PI клиента устанавливает соединение сразу с двумя серверами и соединение для передачи данных открывается между DTP удаленных серверов (т.е. данные передаются между двумя удаленными серверами без скачивания на локальную машину)
3. Формат сообщения протокола FTP:
команда - USER, PASS, LIST, RETR, STOR, QUIT
4. Ответ - NNN текст
5. anonymous и пароль чтонибудь@
6. команды командно-строчного ftp-клиента (open, ls, get, put, quit)
7. примеры ftp-клиентов (командно-строчный, Far, ftp://, wget etc)
8. активный и пассивный режим FTP, схема работы (при пассивном PASV и в ответ номер порта на сервере, при активном PORT с номером порта на машине клиента)
9. применение пассивного режима (когда клиент находится за брандмауэром)

12 Протокол SMTP

1. Общая схема работы почты (сервер исходящей почты и почтовые ящики на каждой стороне, отправляется по SMTP, забирается по протоколу доступа IMAP/POP3 из ящика)
2. Формат сообщения протокола SMTP:
HELO, MAIL FROM:, RCPT TO:, DATA, QUIT
3. Ответ: NNN текст
4. Почему приходит спам, где в From: чужой адрес или случайная строка
5. MIME: Content-type
6. Content-Transfer-Encoding
7. Сообщения из нескольких частей:
Content-type: multipart/mixed; boundary=случайная-строка-разделитель
8. Протоколы доступа: POP3, IMAP
9. Сравнение POP3 и IMAP

13 Служба имен доменов (DNS)

1. Общий случай службы имен: человек работает с именами, программы с идентификаторами, служба имен выполняет преобразование
имена $\longleftrightarrow$ идентификаторы
2. Пример “как не надо делать”: централизованный сервер имен:
единственная точка возможного отказа
3. объем трафика у этого единственного сервера
4. удаленность базы данных: сложности с администрированием

5. удаленность базы данных: большая задержка при обращении (RTT)
6. Общие принципы работы DNS:
локальный сервер имен
7. корневые сервера имен
8. полномочные сервера имен (primary и secondary) - DNS - распределенная база данных, для каждой области ответственности (зоны) данные хранятся на полномочном сервере
9. при обращении поиск начинается с корневого сервера, потом сервер домена 1-го уровня (.ru), потом karelia.ru и т.п. (спуск по дереву доменов)
10. запрос кэшируется на локальном сервере имен, при повторном запросе уже не происходит поиска по дереву
11. Рекурсивный запрос
12. Итеративный запрос
13. Определение имени по IP-адресу (фиктивный домен in-addr.arpa)
14. Записи DNS (тип, время жизни, имя, значение)
15. A
16. NS
17. CNAME
18. MX
19. PTR

14 Распределение ресурсов (CDN, основные варианты организации). Web-proxy, принцип действия. P2P-сети (основные варианты организации).

1. зачем нужно распределение ресурсов
2. принцип действия web-кэша (снижается среднее время доступа)
3. совместное кэширование (несколько web-кэшей объединяются в сеть)
4. CDN с использованием DNS (при запросе IP-адреса для сервера www.cdn.com по IP-адресу источника запроса определяется географическое положение источника запроса и возвращается IP ближайшего к источнику запроса сервера)
5. CDN с распределением нагрузки на несколько серверов центральным распределительным узлом
6. P2P: централизованный каталог (Napster)
7. Децентрализованный каталог (KaZaA)
8. Поток запросов (Gnutella)
9. BitTorrent

15 Сервисы, предоставляемые функциями транспортного уровня

1. на транспортном уровне обеспечивается передача данных между прикладными процессами
2. мультиплексирование и демультиплексирование на транспортном уровне
3. TCP
4. UDP

16 Мультиплексирование и демультиплексирование, порты, сокеты (sockets)

1. мультиплексирование и демультиплексирование на транспортном уровне
2. номер порта отправителя, номер порта получателя - обязательно должны присутствовать в заголовке PDU транспортного уровня (сегмента)
3. сокет как средство для взаимодействия приложения и реализации протокола транспортного уровня

17 Протокол UDP

1. UDP: нет соединения (не требуется время на установление соединения)
2. нет состояния соединения (stateless)
3. небольшой размер заголовка
4. приложение может контролировать передачу данных
5. структура сегмента UDP
6. контрольная сумма UDP

18 Принципы надежной передачи данных

1. абсолютно надежный канал: ничего не надо
2. канал, допускающий искажение битов:
   обнаружение ошибок (контрольная сумма)
3. обратная связь с передающей стороной (ACK/NAK)
4. повторная передача
5. канал, допускающий искажение битов и потерю сегментов:
   добавляется таймер (timeout)
6. квитанции, по которым можно определить, к какому пакету относится данная квитанция (квитанции содержат, например, порядковый номер пакета)
7. канал, допускающий искажение битов, потерю сегментов, изменение порядка следования сегментов: порядковые номера сегментов

8. Проблема: низкая пропускная способность, пример - канал передачи данных Земля-Марс с большим RTT (несколько минут)
9. конвейеризация - несколько сегментов отправляются не дожидаясь подтверждения
10. окно, скользящее окно, отправленные-неотправленные, подтвержденные-неподтвержденные сегменты, при пересечении получается три области - отправленные и подтвержденные, отправленные и неподтвержденные, неотправленные (и неподтвержденные)
11. выборочное повторение (для более эффективного использования канала связи)

19 Протокол TCP

1. устанавливается логическое соединение
2. обеспечивает надежную передачу данных
3. обеспечивает управление потоком данных
4. структура TCP-сегмента (рис.)
5. номер порта отправителя
6. номер порта получателя
7. порядковый номер (sequence number) - номер первого октета в сегменте
8. номер подтверждения (acknowledgment number) - номер следующего ожидаемого октета
9. длина заголовка
10. URG
11. ACK
12. PSH
13. RST
14. SYN
15. FIN
16. размер окна получателя
17. контрольная сумма
18. указатель срочных данных
19. необязательные параметры (опции)

20 Установление и разрыв соединения, состояния TCP

1. установление TCP-соединения: тройное рукопожатие
2. разрыв TCP-соединения
3. TCP-соединение может быть закрыто с одной стороны (“полузакрытое соединение”)
4. Граф переходов между состояниями для стороны, инициирующей соединение
5. CLOSED
6. SYN_SENT
7. ESTABLISHED
8. FIN_WAIT_1
9. FIN_WAIT_2
10. TIME_WAIT
11. Граф переходов между состояниями для стороны, принимающей соединение
12. CLOSED
13. LISTEN
14. SYN_RCVD
15. ESTABLISHED
16. CLOSE_WAIT
17. LAST_ACK

21 Максимальное время ожидания подтверждения в TCP (timeout)

1. $SampleRTT$
2. $EstimatedRTT = (1 - \alpha) \cdot EstimatedRTT + \alpha \cdot SampleRTT$
3. $\alpha = \frac{1}{8}$
4. $DevRTT = (1 - \beta) \cdot DevRTT + \beta \cdot |SampleRTT - EstimatedRTT|$
5. $\beta = \frac{1}{4}$
6. $Timeout = EstimatedRTT + 4 \cdot DevRTT$
7. зачем нужно усреднять

22 Управление потоком в TCP

1. Основные задачи управления потоком (flow control):
удержание средней задержки ожидания на допустимом уровне (для этого нужно уменьшать скорость передачи при перегрузках)
2. соблюдение справедливости по отношению ко всем пользователям
3. пример графа, когда за счет отказа от деления пропускной способности "поровну" можно было бы повысить пропускную способность в два раза (из Бертсекас, Галлагер, "Сети передачи данных")
4. Контроль перегрузки в TCP: со стороны получателя,
свободное место в буфере отправляется в поле "окно получателя" заголовка TCP
5. При отсутствии свободного места отправляется 1 а не 0 чтобы можно было отправлять "пробные" сегменты
6. Контроль перегрузки в TCP: со стороны отправителя,
медленный старт (slow start)
7. slow start threshold
8. аддитивное увеличение (congestion avoidance)
9. мультипликативное уменьшение

23 Время выполнения запроса при статическом окне (static congestion window)

1. Макроскопическая пропускная способность TCP-соединения:
$\frac{(3/4)\cdot W}{RTT}$
2. Статическое окно W:
$\frac{W\cdot S}{R} > RTT + \frac{S}{R}$
3. $\frac{W\cdot S}{R} < RTT + \frac{S}{R}$
4. $2 \cdot RTT + \frac{O}{R}$
5. $2 \cdot RTT + \frac{O}{R} + (K-1) \cdot [\frac{S}{R} + RTT - \frac{W\cdot S}{R}]^{+}$

24 Сервисы, предоставляемые функциями сетевого уровня

1. Передача данных от одного узла сети к другому (узлы не обязательно соединены одной линией связи, т.е. передача может идти через цепочку линий связи и маршрутизаторов)
2. Модели сетевого обслуживания:
определение пути
3. продвижение данных
4. установка соединения на сетевом уровне (ATM)
5. CBR (Constant Bit Rate), ABR (Available Bit Rate)
6. VBR (Variable Bit Rate), UBR (Unspecified Bit Rate)

25 Маршрутизация, термины, алгоритм Дийкстры (LS).

1. путь (маршрут, route), определение
2. путь с минимальной стоимостью
3. алгоритм маршрутизации
4. протокол маршрутизации
5. глобальный алгорить маршрутизации
6. децентрализованный алгоритм маршрутизации
7. статическая и динамическая маршрутизация
8. алгоритм Дийкстры, обозначения:
   $c(i,j)$ - стоимость линии от i до j, $D(v)$ - стоимость известного пути минимальной стоимости до v, $p(v)$ - предпоследний узел на известном пути минимальной стоимости до v, N - множество узлов, до которых уже известен путь минимальной стоимости
9. инициализация
10. действия на каждом шаге
11. пример

26 Маршрутизация, алгоритм Беллмана-Форда (DV).

1. основная структура данных - таблица расстояний
2. $D^X(Y,Z) = c(X,Z) + min_w\{D^Z(Y,w)\}$
3. инициализация
4. действия при изменении стоимости пути
5. действия при получении обновления
6. действия при новом значении $min_w\{D^X(Y,w)\}$
7. пример для графа с тремя вершинами
8. сравнение алгоритмов маршрутизации - сложность сообщений
9. скорость схождения
10. живучесть

27 Автономные системы, иерархическая маршрутизация

1. зачем нужно: технические причины (масштабирование)
2. административные причины: адинистративная автономия
3. автономная система (AS)
4. протокол внутренней маршрутизации
5. шлюзовой (пограничный) маршрутизатор (шлюз, border gateway)
6. протокол внешней маршрутизации
7. пример
8. протоколы - RIP, OSPF (внутренняя маршрутизация), BGP - внешняя.

28 Протокол IP. Адресация и маршрутизация в IP.

1. адресация в IPv4, IP-адрес
2. классы
3. безклассовая нотация (CIDR), маска сети
4. получение сетевого адреса: ручная настройка
5. DHCP
6. случайным образом
7. таблица маршрутизации, алгоритм продвижения дейтаграмм IP
8. формат дейтаграммы IPv4
9. версия
10. длина заголовка
11. тип службы (Type Of Service, TOS)
12. длина дейтаграммы
13. идентификатор фрагмента, флаги, смещение фрагмента
14. время жизни (Time To Live, TTL)
15. протокол верхнего (транспортного уровня)
16. контрольная сумма
17. IP-адреса отправителя и получателя
18. необязательные параметры
19. фрагментация и сборка, MTU

29 Основные варианты архитектуры маршрутизатора (коммутатора)

1. входные порты: взаимодействие с физической линией
2. обработка канального уровня
3. предварительная обработка сетевого уровня
4. коммутационный блок: с использованием памяти
5. с использованием шины
6. с использованием соединительной сети
7. выходные порты
8. очередь на входном порту
9. очередь на выходном порту

30 Сервисы, предоставляемые функциями канального уровня

1. основная задача - передача данных по каналу связи (т.е. между двумя “соседними” узлами сети)
2. формирование кадра
3. доступ к линии связи, протокол MAC (Media Access Control)
4. надежная доставка
5. управление потоком
6. обнаружение ошибок
7. исправление ошибок
8. дуплексная и полудуплексная передача
9. адаптер

31 Методы обнаружения и коррекции ошибок

1. общая схема обнаружения ошибок
2. четность
3. двухмерный контроль четности
4. циклический избыточный код (CRC)

32 Способы разделения среды передачи (TDM, FDM, CDMA). Основные способы организации доступа к общей среде передачи.

1. TDM
2. FDM
3. CDMA пример для одного передатчика
4. CDMA пример с несколькими передатчиками
5. произвольный доступ: CSMA, CSMA/CD
6. опрос
7. передача маркера (токена)

33 Адрес в локальной сети и ARP

1. адресация в локальной сети (MAC-адрес)
2. каким образом обеспечивается глобальная уникальность MAC-адресов
3. ARP
4. RARP

34 Ethernet

1. структура кадра Ethernet: преамбула
2. адрес получателя, адрес отправителя
3. протокол верхнего уровня (сетевой)
4. данные
5. CRC
6. Манчестерское кодирование
7. Протокол CSMA/CD: формирование кадра
8. передача
9. обнаружение коллизий
10. действия при коллизии и экспоненциальный откат (exponential backoff)
11. технологии Ethernet (10Base2, 10BaseT, Fast Ethernet, Gigabit Ethernet)

35 Хабы и коммутаторы

1. хаб (hub)
2. коммутатор, принцип работы

36 Протокол PPP

1. задачи: формирование кадра
2. прозрачность
3. поддержка различных протоколов сетевого уровня
4. поддержка различных физических линий
5. обнаружение ошибок
6. живучесть соединения
7. согласование адресов сетевого уровня
8. Состояния PPP-соединения: установка соединения - протокол LCP
9. аутентификация - PAP, CHAP
10. настройка сетевого уровня - IPCP (для IP на сетевом уровне)