КПИ им. Игоря Сикорского, каф. микроэлектроники.
М. Р. Домбругов. Информатика-1.
Персональные компьютеры и основы сетевых технологий

Лекция 11.
6-й уровень модели OSI: основы криптографии

7 декабря 2020

7-уровневая модель OSI

Тип данных	Уровень (layer)	Функции
Данные	7. Прикладной (application)	Доступ к сетевым службам
	6. Представительский (presentation)	Представление и шифрование данных
	5. Сеансовый (session)	Управление сеансом связи
Сегменты	4. Транспортный (transport)	Прямая связь между конечными пунктами и надежность
Пакеты	3. Сетевой (network)	Определение маршрута и логическая адресация
Кадры	2. Канальный (data link)	Физическая адресация
Биты	1. Физический (physical)	Работа со средой передачи, сигналами и двоичными данными

6-й уровень модели OSI

Представление и шифрование данных

- Представление текста
- - " - чисел
- - " - звука
- - " - изображений
- - " - видео
- Оптимальное кодирование и архиваторы
- **Шифрование данных**
- …

- Основные принципы криптографии
- Симметричное шифрование
- Асимметричное шифрование
- Хэширование
- Электронная цифровая подпись

Основные принципы криптографии

Криптология (греч. κρυπτός – скрытый, λόγος – слово) –наука о методах шифрования и расшифровывания.

Состоит из двух частей:

- **криптография** –
  разработка методов шифрования данных;
- **криптоанализ** –
  оценка криптостойкости методов шифрования,
  разработка методов для взламывания криптосистем.

Основные принципы криптографии

Задачи криптографии:

- обеспечение **конфиденциальности** (невозможность прочтения информации посторонним);
- обеспечение **целостности данных** (невозможность незаметного изменения информации);
- **аутентификация** (проверка подлинности авторства или иных свойств объекта).

Основные принципы криптографии

Шифры бывают обратимые и необратимые.

Необратимые шифры – если потом не надо расшифровывать то, что зашифровали. Такой шифр наз. **хэш** (hash).

У **обратимых шифров** обязательно есть ключ (key), обладая которым можно восстановить исходное сообщение из шифрованного текста.

Если способ шифрования подразумевает, что как шифрование, так и расшифровка ведутся одним и тем же ключом, то такой шифр называется **симметричным**.

Если алгоритм предусматривает шифровку одним ключом, а расшифровку – другим, то такие шифры называют **асимметричными**.

Основные принципы криптографии

Шифр – параметризованный алгоритм, состоящий из:

- **процедурной части («системы»)** – описания того, какие именно операции и в какой последовательности выполняются над шифруемыми данными);
- **параметров («ключей»)** – различных элементов данных, используемых в преобразованиях.

Принцип Керхкофа (1903)

Нужно, чтобы не требовалось сохранение системы в тайне; попадание системы в руки врага не должно причинять неудобств.

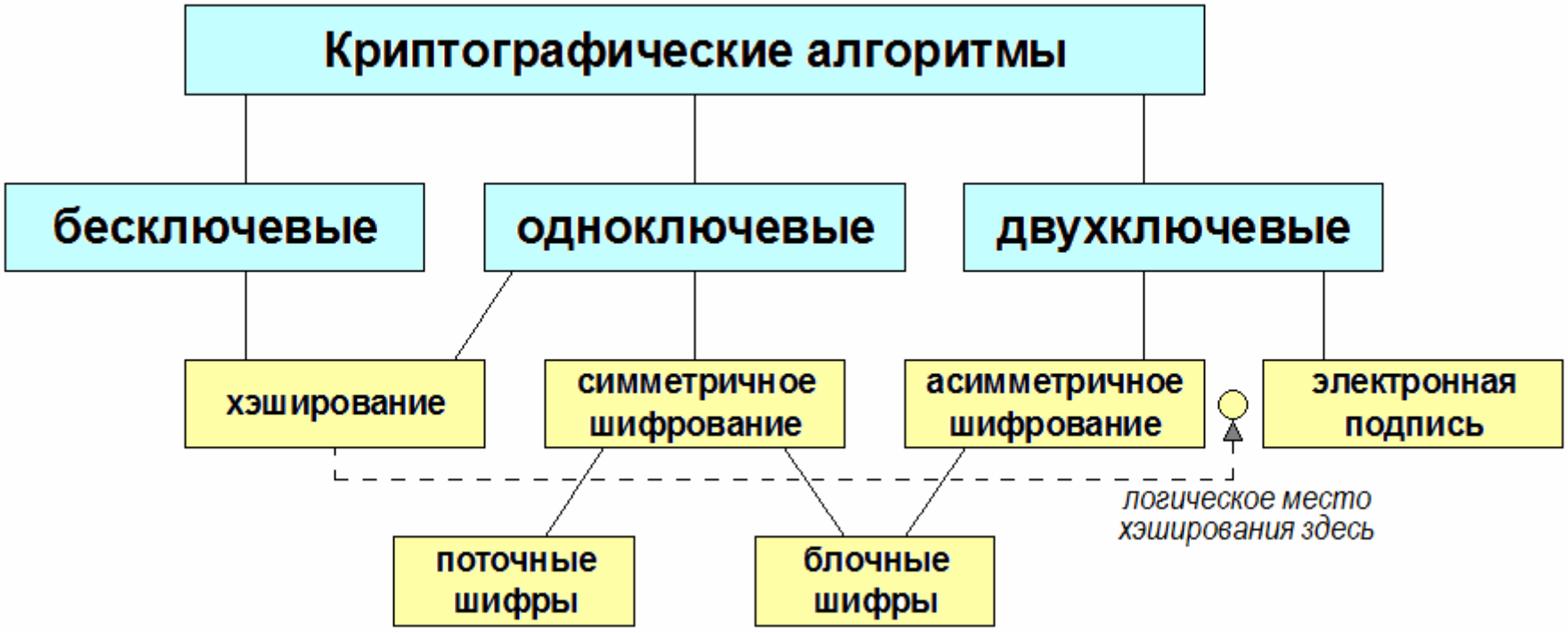
При оценке надежности шифра нужно предполагать, что противник знает об используемой системе шифрования все, кроме применяемых ключей.

Основные принципы криптографии

Преимущества системы шифрования с ключами:

- разглашение конкретного шифра (алгоритма и ключа) не приводит к необходимости полной замены реализации всего алгоритма, достаточно заменить только скомпрометированный ключ;
- ключи можно отчуждать от остальных компонентов системы шифрования – хранить отдельно от реализации алгоритма;
- появляется возможность для точной оценки «степени стойкости» алгоритма шифрования – она равна неопределенности используемого ключа.

Основные принципы криптографии



Основные принципы криптографии

Все данные кодируются цифровым кодом и затем шифрование применяется к полученным числовым данным

Общепринятые обозначения:

- **M** – исходное сообщение (**m**essage);
- **K** – ключ (**k**ey);
- $\mathbf{E_K(M)}$ – функция шифрования (**e**ncryption) с ключом **K**;
- **C** – зашифрованное сообщение (**c**iphertext);
- $\mathbf{D_K(C)}$ – функция расшифрования (**d**ecryption) с ключом **K**.

Основные принципы криптографии

- https://ru.wikipedia.org/wiki/Шифрование
- https://ru.wikipedia.org/wiki/Алиса_и_Боб
- http://enisey.name/umk/pzis/ch18.html

Симметричное шифрование

Симметричное шифрование – способ шифрования, при котором для шифрования и расшифровывания применяется **один и тот же криптографический ключ**.

Этапы работы:

1. Существует некий математический алгоритм шифрования.
2. На его вход подаётся исходное сообщение и ключ.
3. На выходе получаем зашифрованное сообщение.
4. Если хотим вновь получить исходное сообщение, применяется **тот же самый ключ**, но с алгоритмом дешифрования.

Ключ должен храниться в тайне обеими сторонами.

Симметричное шифрование

Классификация симметричных криптоалгоритмов

- **В зависимости от размера блока информации**

  Шифрование длинных сообщений может выполняться одним из двух способов:

  - **потоковый шифр:** данные шифруются побитно. Результат шифрования не зависит от зашифрованного ранее входного потока.
  - **блочный шифр:** открытый текст делится на блоки фиксированного размера (~ 64…128 бит), функция шифрования с относительно коротким ключом (~ 32…256 бит) шифрует каждый блок, а затем блоки объединяются вместе, образуя шифрованный текст;

Симметричное шифрование

Классификация симметричных криптоалгоритмов

- **По типу преобразований данных в шифрующих функциях**

  Существуют только **два** основных типа преобразований, все остальные являются комбинацией этих двух типов:

  - в **шифрах замены** символы шифруемого текста заменяются другими символами с заранее обусловленной схемой замены.
    (Под «символом текста» понимается группа битов из сообщения, не обязательно кодирующая букву или байт);
  - в **шифрах перестановки** символы шифруемого текста переставляются по определенному правилу в пределах какого-то блока этого текста.

Симметричное шифрование

Бинарная логическая операция XOR (Exclusive OR)

a XOR b истинно, если истинно **a** или **b**, но не оба сразу.

a	b	a XOR b
0	0	0
0	1	1
1	0	1
1	1	0

Иная трактовка:

если **b=0**, то результ равен **a**,
если **b=1**, то **a** нужно инвертировать

Операция может применяться к числам, если их записать в двоичном коде, а затем XOR применить побитно.

В этом случае применяют обозначение **a** ⊕ **b.**

Симметричное шифрование

Гаммирование – простой потоковый шифр замены

Побитно сравниваем **M** и **K**.
Если соответствующий бит в **K** равен 1, то бит в **M** инвертируем, а если бит **K** равен 0, то бит в **M** оставляем как есть:

$$\mathbf{C} = \mathbf{E_K(M)} = \mathbf{M} \oplus \mathbf{K}$$

Для дешифровки инвертированные биты нужно инвертировать повторно:

$$\mathbf{M} = \mathbf{D_K(C)} = \mathbf{C} \oplus \mathbf{K}$$

При таком способе шифрования ключ **K** называют «гаммой», а сам способ – «**гаммированием**».

В алгоритме гамирования функции шифрования **E()** и дешифрования **D()** совпадают.

Симметричное шифрование

Требования к гамме

При определённых свойствах гаммы этот метод шифрования является **абсолютно стойким и не поддающимся взлому**:

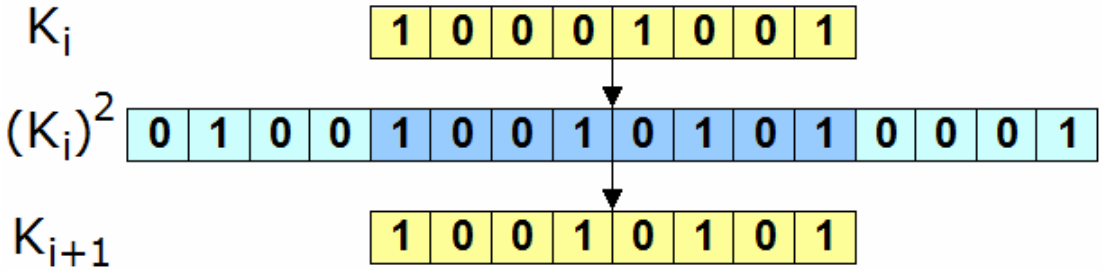
- последовательность битов гаммы должна быть случайной;
- длина гаммы должна быть не меньше длины шифруемого сообщения;
- повторное использование гаммы недопустимо – для шифрования каждого нового сообщения нужно использовать новую гамму.

  Если злоумышленнику в руки попадет одновременно и исходное и зашифрованное сообщения, то гамма легко раскрывается: **K** = **M** ⊕ **C**.

Симметричное шифрование

Потоковые шифры на основе генератора псевдослучайных чисел

Чтобы избежать передачи длинной гамма-последовательности ее можно создавать генератором псевдослучайных чисел. Каждый последующий фрагмент $\mathbf{K_{i+1}}$ вырабатывается из $\mathbf{K_i}$. Сторонам следует знать заранее только $\mathbf{K_1}$.



Пример («метод середины квадрата»): n-битовое число возводится в квадрат, затем из середины результата берётся n-значное число, которое снова возводится в квадрат, и т.д. (Предложен в 1944 Дж. фон Нейманом для десятичных чисел)

Симметричное шифрование

Блочный шифр

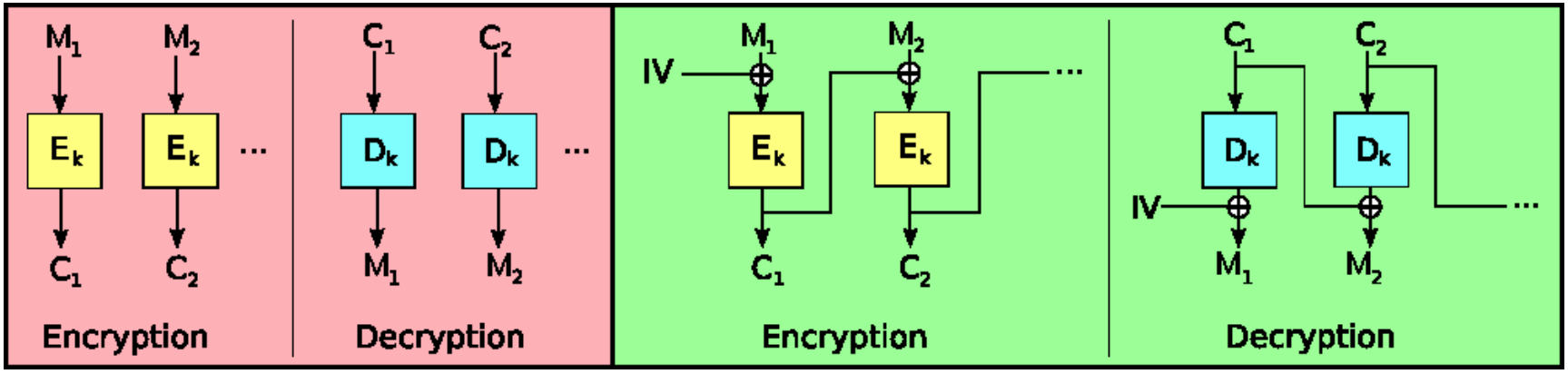
Использует функции шифрования $\mathbf{E_K(M)}$ с ключом **K**,
которая принимает на вход блок сравнительно небольшого фиксированного размера (обычно 64 или 128 бит)
и путем ряда замен и перестановок преобразует его
в шифрованный блок такого же размера.

При шифровании открытый текст
делится на блоки фиксированного размера $\mathbf{M_1}$, $\mathbf{M_2}$, ..., $\mathbf{M_n}$.
При помощи $\mathbf{E_K(M_i)}$ блоки зашифровываются,
а затем объединяются вместе, образуя шифрованный текст,
состоящий из блоков $\mathbf{C_1}$, $\mathbf{C_2}$, ..., $\mathbf{C_n}$. того же размера.

Расшифрование делается с помощью обратной функции $\mathbf{D_K(C_i)}$
с тем же ключом **K**.

Симметричное шифрование

Блочный шифр



Режим **Electronic Code Book (ECB)**

Менее криптоустойчив.
Не рекомендуется.

Режим **Cipher Block Chaining (CBC)**

Начинается с **IV** – Initializing Value, заранее оговоренное начальное значение.

$\mathbf{C_1 = E_k(IV \oplus M_1);\ C_i = E_k(C_{i-1} \oplus M_i).}$

Каждый шифрованный блок зависит от всех предыдущих блоков.

Симметричное шифрование

Популярные блочные алгоритмы симметричного шифрования

DES (Data Encryption Standard) (1977).
Устарел (недостаточно криптостойкий дя современных ЭВМ).
Дальнейшие модификации – **Triple-DES** (1978),
AES (Advanced Encryption Standard), иначе **Rijndael** (1998).
Принят в качестве стандарта шифрования правительством США.

Blowfish (1993). Блок 64 бита, ключ 32…448 бит.
Незапатентованный и свободно распространяемый.
Дальнейшая модификация – **Twofish** (1998).

…

Симметричное шифрование

Передача ключа через открытый канал.
Протокол Диффи – Хеллмана (Diffie – Hellman)

Позволяет сторонам получить общий секретный ключ, используя незащищенный от прослушивания канал связи.

Основана на том, что целочисленная функция $\mathbf{y(x) = g^x \bmod p}$, где **g** и **p** – целочисленные параметры, вычисляется легко, а ее обращение (т.е. определение **x** по известному **y**) может быть сделано только прямым перебором.

Если $p \sim 10^{300}$, то мощностей современных ЭВМ недостаточно, чтобы произвести такие вычисления за время, меньшее, чем время существования Вселенной ~ 16 млрд лет = $2 \cdot 10^{16}$ с.

Симметричное шифрование

Пример труднообратимой функции:

Вычислить $85^{29} \bmod 91$.

Вычисляем остатки от деления 85^x, где ***x*** – степени 2, меньшие 29:

$85^2 = 7225 \equiv 36 \pmod{91}$;
$85^4 \equiv 36^2 = 1296 \equiv 22 \pmod{91}$;
$85^8 \equiv 22^2 = 484 \equiv 29 \pmod{91}$;
$85^{16} \equiv 29^2 = 841 \equiv 22 \pmod{91}$.

Запись **A≡B (mod p)** («A сравнимо с B по модулю p») означает, что A и B при делении на p дают одинаковый остаток

Предствляем 29 в двоичном коде: $29 = 11101_2 = 1 + 4 + 8 + 16$.

$85^{29} = 85 \cdot 85^4 \cdot 85^8 \cdot 85^{16} \equiv (85 \cdot 22) \cdot (29 \cdot 22) = 1870 \cdot 638 \equiv 50 \cdot 1 \pmod{91}$.

Ответ: $85^{29} \bmod 91 = 50$.

Обратная задача:
определить **x**, если известно, что $85^x \bmod 91 = 50$.

Симметричное шифрование

Передача ключа через открытый канал.
Протокол Диффи – Хеллмана (Diffie – Hellman)

Предположим, что обоим абонентам (Алиса и Боб) известны некоторые два числа **g** и **p**, которые не являются секретными.

Чтобы создать секретный ключ, Алиса и Боб генерируют большие случайные числа: Алиса – число **a**, Боб – число **b**. Затем Алиса вычисляет $\mathbf{A = g^a \bmod p}$ и пересылает Бобу, а Боб вычисляет $\mathbf{B = g^b \bmod p}$ и передаёт Алисе.

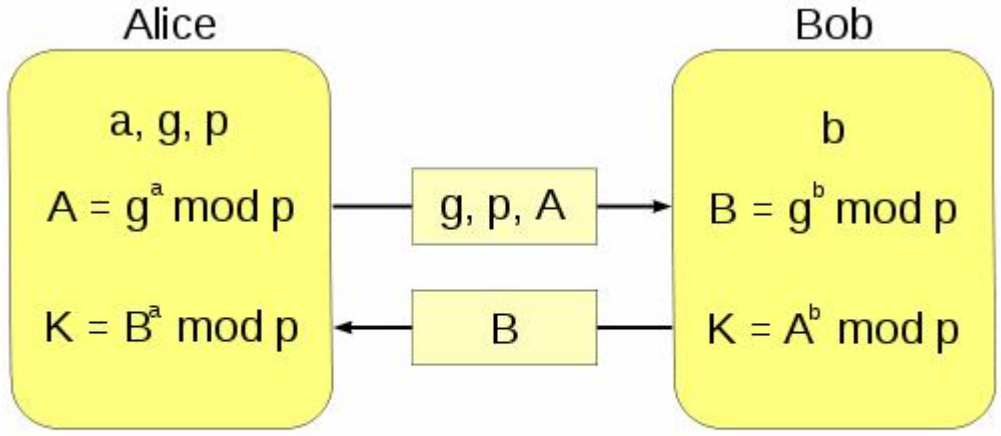
На втором этапе Алиса вычисляет $\mathbf{B^a \bmod p = g^{ab} \bmod p}$, а Боб – $\mathbf{A^b \bmod p = g^{ab} \bmod p}$.
У обоих получилось одно и то же число **K**.

Его они используют как секретный ключ.

Симметричное шифрование

Передача ключа через открытый канал.
Протокол Диффи – Хеллмана (Diffie – Hellman)

Задача вычисления **K** по перехваченным **A** и **B**, если числа **p, a, b** достаточно большие, практически неразрешима.



$$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p =$$
$$= (g^b \bmod p)^a \bmod p = B^a \bmod p$$

На практике **a** и **b** ~ 10^{100} и **p** ~ 10^{300}.
Число **g** не обязано быть большим и обычно ~ 10.

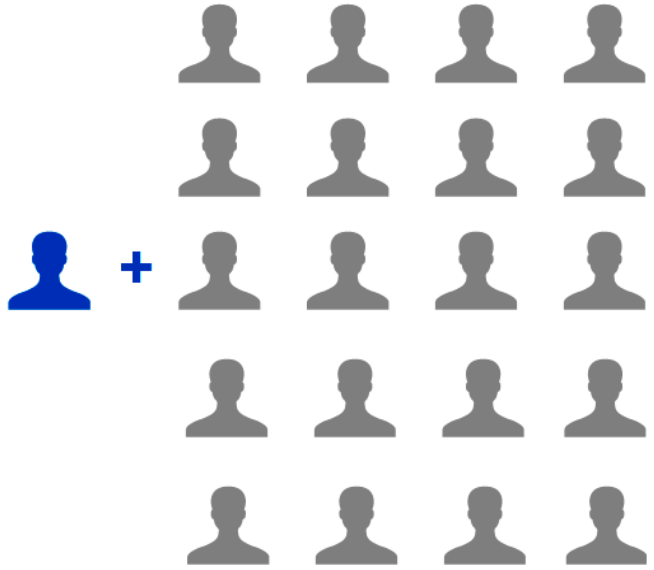
Симметричное шифрование

Обилие ключей в больших группах пользователей

Использование одного ключа для всех абонентов группы недопустимо по соображениям безопасности, так как в случае его компрометации под угрозой будет находиться документооборот всех абонентов.

Если в группе имеется N абонентов, то для обмена шифрованными сообщениями им требуется N×(N-1)/2 ключей.

Проблему обилия ключей решает **асимметричная криптография.**



20 абонентов = 190 ключей

еще + 20 при добавлении в группу 21-го абонента

Симметричное шифрование

- https://ru.wikipedia.org/wiki/Симметричные_криптосистемы
- https://ru.wikipedia.org/wiki/Исключающее_«или»
- https://ru.wikipedia.org/wiki/Гаммирование
- https://ru.wikipedia.org/wiki/Шифр_Вернама
- https://ru.wikipedia.org/wiki/Генератор_псевдослучайных_чисел
- https://ru.wikipedia.org/wiki/Блочный_шифр
- https://ru.wikipedia.org/wiki/Протокол_Диффи_—_Хеллмана
- http://kaf403.rloc.ru/POVS/Crypto/DiffieHellman.html

Асимметричное шифрование

Асимметричное шифрование – способ шифрования, при котором для шифрования и расшифровывания применяется **два ключа** – открытый (публичный) и закрытый (секретный), которые связаны между собой.

Ассиметричное шифрование является односторонним процессом. При передаче используются **ключи получателя.** Отправитель шифрует данные открытым ключом получателя, а получатель их расшифровывает своим секретным ключом.

До изобретения схемы асимметричного шифрования (1976) симметричное шифрование являлось единственным существовавшим способом.

Все асимметричные шифры – блочные.

Асимметричное шифрование

Алгоритм RSA (Rivest – Shamir – Adleman)

Основан на том, что разложение числа на простые множители может быть сделано только прямым перебором, т.е. это – труднорешаемая задача.

Пример: 1644475769899088836431709885 3 – простое?

Ответ: нет, 1644475769899088836431709885 3 =
= 189435747774163 × 878634363654631

Если число ~ 10^{600} ~ 2^{2048}, то мощностей современных ЭВМ недостаточно, чтобы разложить его на множители за время, меньшее, чем время существования Вселенной.

Асимметричное шифрование

Алгоритм RSA

Для начала необходимо сгенерировать ключи:

- Взять два больших простых числа **p** и **q**.
- Вычислить **n = pq**.
- Вычислить **φ = (p-1)(q-1)**
- Выбрать **e**, простое, взаимно-простое с **φ**, **1<e<φ** (обычно число, небольшое по сравнению с **p** и **q**).
- Найти **d** такое, что **1<d<φ** и **ed ≡ 1 (mod φ),** т.е. **ed = kφ+1**, где **k** – целое.

Пара **P = (e, n)** – открытый ключ (**P**ublic key),
S = (d, n) – секретный ключ (**S**ecret key).

Асимметричное шифрование

Алгоритм RSA (продолжение)

Пусть **M** (**m**essage) – исходное сообщение.
Должно быть **M<n**, иначе исходное сообщение следует разбить на блоки подходящего размера и шифровать их по отдельности.

Шифрование. Зашифрованное сообщение **C** (**c**iphertext):

$$C = M^e \bmod n$$

Расшифровка:

$$M = C^d \bmod n$$

Доказательство основывается на свойствах остатков от деления и **малой теореме Ферма:**

Если **p** – простое, **A** – любое целое, то $A^p \equiv A \pmod p$

Асимметричное шифрование

Пример (игрушечный):

Генерируем ключи:

- Выбираем **p** = 13**, q** = 7.
- Вычисляем **n = pq** = 13·7 = 91.
- Вычисляем **φ = (p-1)(q-1)** = 12·6 = 72.
- Выбираем **e** = 5, взаимно-простое с **φ** = 72.
- Находим **d** из условия, что **ed ≡ 1 (mod φ)**,
  т.е. 5**d** = 72**k** + 1 = 70**k** + 2**k** + 1 (Диофантово уравнение).
  Т. обр., 2**k**+1 должно делиться на 5, выбираем **k** = 2.
  d = (72·2+1)/5 = 145/5 = 29
  Действительно, **ed** = 5·29 = 145 ≡ 1 (mod 72).

Пара **P = (e, n)** = (5,91) – открытый ключ (**P**ublic key),
S = (d, n) = (29,91) – секретный ключ (**S**ecret key).

Асимметричное шифрование

Пример (продолжение):

Шифрование. Секретное сообщение: M = 50. Шифруем:

$$\mathbf{C = M^e \bmod n} = 50^5 \bmod 91 = 85.$$

(…враг подслушал, но ничего не понял…)

Расшифровка.

$$\mathbf{M = C^d \bmod n =} \ 85^{29} \bmod 91 = 50.$$

Секретное сообщение доставлено.

Асимметричное шифрование

Достоинства асимметричных шифров:

- Открытый ключ может быть свободно распространен, это позволяет принимать данные от всех пользователей.
- Только одной стороне известен ключ дешифрования, который нужно держать в секрете.
- В больших сетях число ключей в асимметричной криптосистеме значительно меньше, чем в симметричной.

Недостатки:

- Алгоритм шифрования очень ресурсоемкий, работает на два-три порядка медленнее, чем симметричный (скорость шифрования при 512-битном ключе на процессоре 2 ГГц составляет всего ~ 30 кбит/с).
- При аналогичной криптостойкости – более длинные, чем в симметричных алгоритмах, ключи.

Асимметричное шифрование

Гибридная схема шифрования

Когда нужно установить защищенный канал связи, то каждая из сторон случайным образом генерирует «**сеансовый ключ»**. Его срок годности зависит от конкретной задачи.

При помощи этого сеансового ключа будет происходить обмен данными с использованием **симметричного шифрования**, которое менее требовательно к аппаратным ресурсам.

Для передачи же самого сеансового ключа используют **несимметричное шифрование.**

Оно хоть ресурсоемкое, но передать надо малый объем информации. А когда сеансовый ключ передан, дальнейшую передачу ведут экономными симметричными алгоритмами.

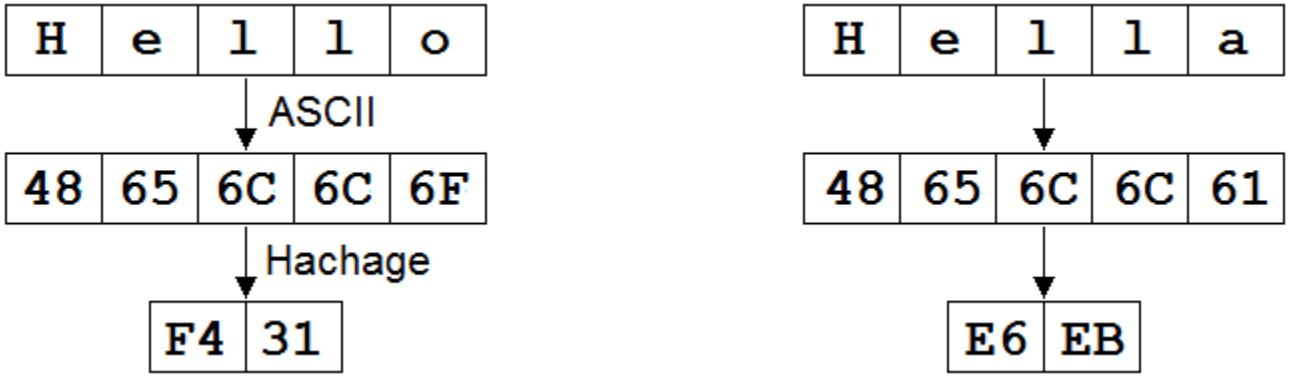
Асимметричное шифрование

- https://ru.wikipedia.org/wiki/RSA
- https://ru.wikipedia.org/wiki/Криптосистема_с_открытым_ключом
- http://www.e-nigma.ru/stat/rsa/
- https://ru.wikipedia.org/wiki/Сравнение_по_модулю
- https://habr.com/ru/post/119637/
- http://enisey.name/umk/pzis/ch18.html
- https://intsystem.org/security/asymmetric-encryption-how-it-work/

Хэширование

Хэш-функция (от *hash – мешанина, путаница*) преобразует входной массив данных произвольной длины в битовую строку фиксированной длины.

Пример:



$$48 + 65 + 6C + 6C + 6F = F4 \bmod FF$$
$$48\times1 + 65\times2 + 6C\times3 + 6C\times4 + 6F\times5 = 31 \bmod FF$$

$$48 + 65 + 6C + 6C + 61 = E6 \bmod FF$$
$$48\times1 + 65\times2 + 6C\times3 + 6C\times4 + 61\times5 = EB \bmod FF$$

Хэширование

Криптографические хэш-функции –

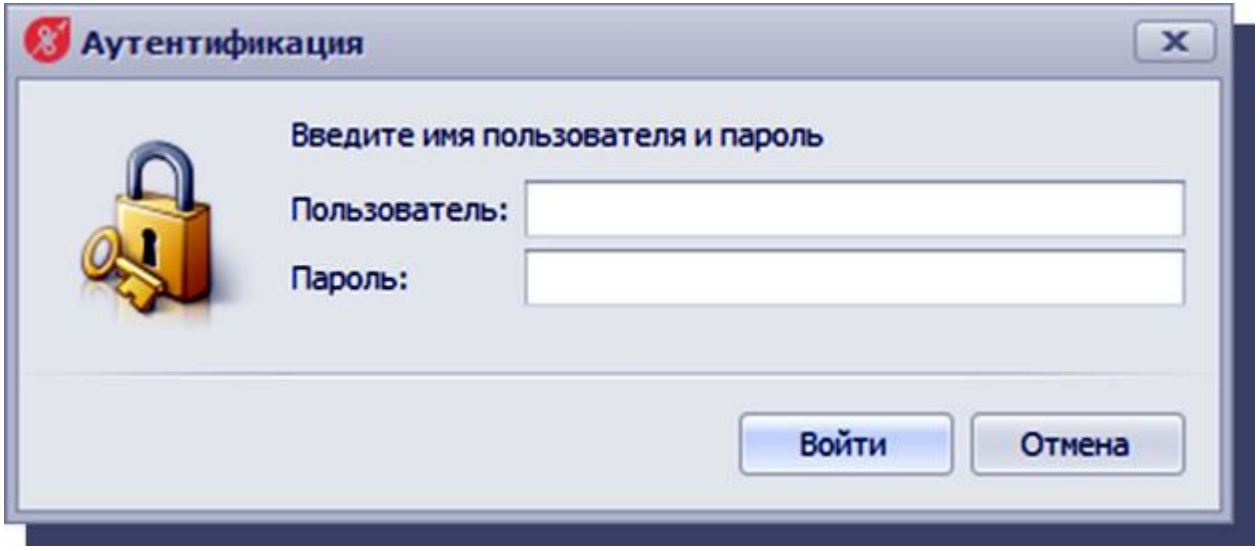
это класс хэш-функций, которые удовлетворяют трём основным требованиям, на которых основано большинство применений хеш-функций в криптографии:

- **необратимость:** для заданного значения хэш-функции **h** должно быть вычислительно неосуществимо найти блок данных **M**, для которого **H(M)=h**;
- **стойкость к коллизиям первого рода:** для заданного сообщения **M** должно быть вычислительно неосуществимо подобрать другое сообщение **N**, для которого **H(N)=H(M)**;
- **стойкость к коллизиям второго рода:** должно быть вычислительно неосуществимо подобрать пару сообщений **(M,M')**, имеющих одинаковый хэш.

Хэширование

Пример использования: проверка паролей

Пароли пользователей обычно не хранятся открытым текстом, а сохраняются их хэш-значения. Для аутентификации пользователя указанный им пароль хэшируется и сравнивается с хранимым хэшем.



Хэширование

Проверка паролей

Получив в свое распоряжение файл, хранящий хэши паролей, злоумышленник не может восстановить по ним сами пароли, а должен перебирать парольные комбинации символов, применять к ним хэш-функцию и сверять полученные значения со строками из файла хэшированных паролей.

(Это также означает, что начальный пароль невозможно восстановить, и при его утрате самим пользователем пароль нужно замениь новым).

Дополнительная защита – добавление системой в конец пароля дополнительных символов (**«криптографическая соль»**) перед его хэшированием.

Хэширование

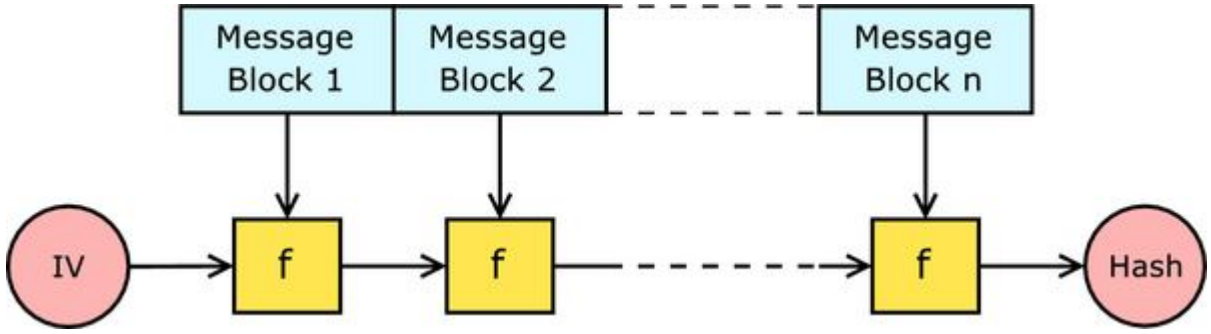
Построение криптографической хэш-функции: структура Меркла-Дамгора (MD, Merkle—Damgård)

Ядром алгоритма является **сжимающая функция** – преобразование **k+n** входных в **n** выходных бит, где **n** – разрядность хэш-функции. При этом сжимающая функция должна удовлетворять всем условиям криптостойкости.

Входной поток разбивается на блоки по **k** бит. Каждый следующий блок данных объединяется с выходным значением сжимающей функции на предыдущей итерации.

Хэширование

IV – Initializing Value, заранее оговоренное начальное значение



Значением хэш-функции являются выходные *n* бит последней итерации.

Каждый бит выходного значения хеш-функции зависит от всего входного потока данных и начального значения. Таким образом достигается **«лавинный эффект»**, означающий, что изменение одного бита во входной последовательности ведет к тому, что каждый бит выходного хэша изменяется с вероятностью ~½.

Хэширование

Популярные алгоритмы хэширования

MD5 (*Message Digest 5*). Разработчик Рональд Ривест из Массачусетского технологического института, 1991. Хэш содержит 128 бит (16 байт) и обычно представляется как последовательность из 32 шестнадцатеричных цифр.

SHA-1 (160 бит, устарел) **и SHA-2** (224, 256, 384 или 512 бит) *(Secure Hash Algorithm).* Семейства криптографических алгоритмов. Разработаны в Агентстве национальной безопасности США (1993, 2001).

«Стрибог» (256 или 512 бит). Разработка ФСБ России. Стандарт СНГ с 2018.

«Купина» (от 8 до 512 бит). Разработка ХНУРЭ (ПАТ «Інститут інформаційних технологій»). Стандарт Украины ДСТУ 7564:2014.

Хэширование

- https://ru.wikipedia.org/wiki/Криптографическая_хеш-функция
- https://ru.wikipedia.org/wiki/Структура_Меркла_—_Дамгора
- http://enisey.name/umk/pzis/ch18.html
- https://intsystem.org/security/users-passwords-store/

Электронная цифровая подпись

ЭЦП – это реквизит электронного документа, предназначенный для его защиты от подделки.

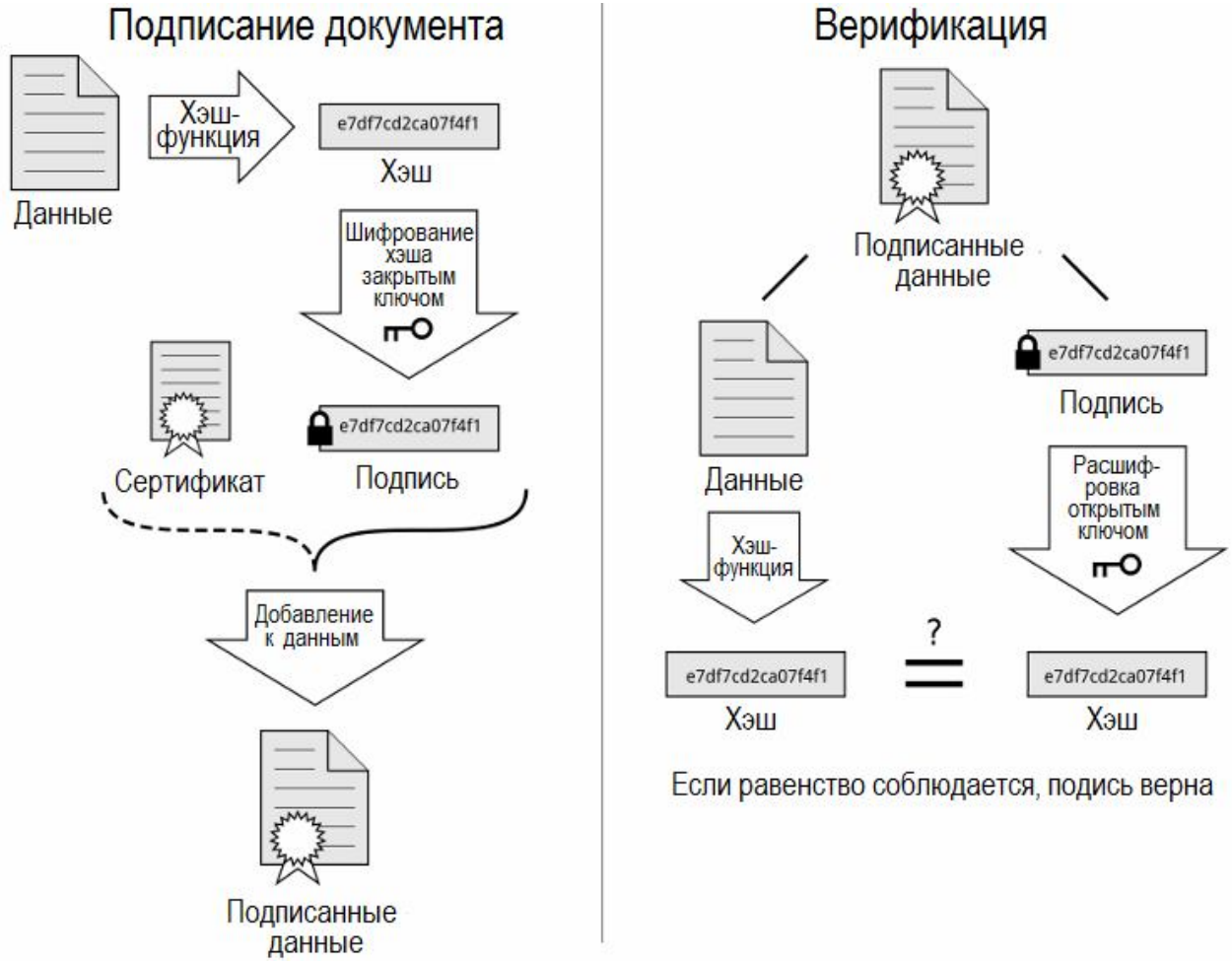
ЭЦП задействует криптоалгоритмы с асимметричными ключами, где для шифрования и дешифрования используются два ключа – **закрытый** и **открытый**, уникальные для каждого подписанта. Закрытый и открытый ключи работают только в паре друг с другом.

Закрытый ключ имеет длину ~ 256 бит,
хранится подписантом в недоступном другим лицам месте.

Открытый ключ используется для проверки ЭЦП получаемых документов-файлов. Он длиннее закрытого, ~ 1024 бита. Необходимо обеспечить наличие своего открытого ключа у всех, с кем предполагается обмениваться подписанными документами.

Важным элементом ЭЦП является
цифровой сертификат открытого ключа.

Электронная цифровая подпись



Электронная цифровая подпись

1. **Подпись достоверна.** Сформировать ее может только обладатель закрытого ключа. Она убеждает получателя документа, что подписант действительно его подписал.
2. **Подпись неподдельна.** Она доказывает, что подписал документ именно обладатель закрытого ключа, и никто иной.
3. **Подпись неоспорима.** Подписавший не может отречься от подписи и утверждать, что он не подписывал документ.
4. **Подпись верифицируема.** Проверить ее может любой пользователь, имеющий открытый ключ.
5. **Подписанный документ неизменяем.** После того, как он подписан, его невозможно изменить.
6. **Подпись отделима от документа.** Подпись и подписанное сообщение могут передаваться и храниться отдельно.
7. **Подпись не может быть использована повторно.** Мошенник не сможет перенести подпись на другой документ.

Электронная цифровая подпись

Сертификат открытого ключа

Для подтверждения, что проверяющий ЭЦП пользуется действительно открытым ключом подписанта, а не его подменой злоумышленником, ЭЦП может сопровождаться **сертификатом окрытого ключа X.509**.

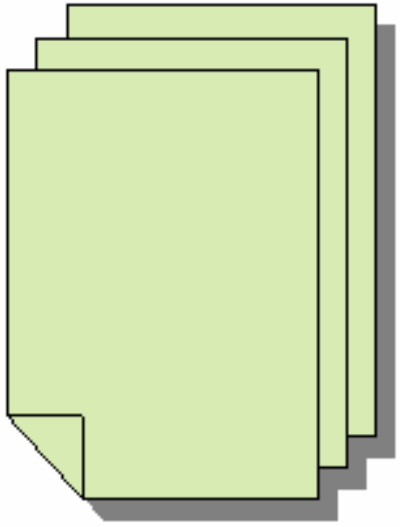
Эмитент такого сертификата – один из авторизованных **центров сертификации (ЦС),** а сам сертификат подписан ЭЦП ЦС.

Открытый ключ ЦС является общеизвестным или подтверждается ЦС более высокого уровня. Сертификаты корневых ЦС присутствуют в операционной системе каждого компьютера и тиражированы в миллиардах экземпляров.

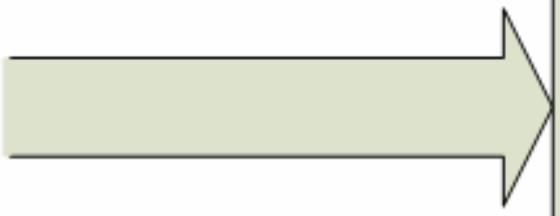
Для Windows 7 см.:

Пуск ►Панель управления ►Свойства браузера ►Содержание ► Сертификаты ►Доверенные корневые центры сертификации

Электронная цифровая подпись



Certificate X.509



- Номер версии X.509
- Открытый ключ пользователя
- Серийный номер сертификата
- Идентификатор пользователя
- Период действия сертификата
- Идентификатор эмитента
- ЭЦП эмитента
- Идентификатор алгоритма ЭЦП

Сертификат может быть выпущен только уполномоченным эмитентом и содержит единственную ЭЦП эмитента

Электронная цифровая подпись

- https://ru.wikipedia.org/wiki/Электронная_подпись
- https://ru.wikipedia.org/wiki/Сертификат_открытого_ключа
- https://lumpics.ru/how-open-certificate-store-in-windows-7/
- https://docs.microsoft.com/ru-ru/dotnet/framework/wcf/feature-details/how-to-view-certificates-with-the-mmc-snap-in