# Лекция 12. Криптоанализ и криптоатаки

## Основные термины

Первый труд (и первое упоминание о частотном анализе) - «Манускрипт о дешифровке криптографических сообщений», написанный в 9 веке арабским учёным *Абу Юсуф Якуб ибн-Исхак ибн-Ас-Сабах ибн-Умран ибн-Исмалил аль-Кинди* (приставка «абу» означает отец, «ибн» - сын, а «аль» - уроженец), а сокращенно ученого у нас принято называть Аль-Кинди.

**Криптология** состоит из двух частей:

1. **Криптография -**  изучает способы шифрования и/или проверки подлинности сообщений
2. **Криптоанализ -** ищет пути расшифровки и подмены криптограмм

**Криптоанализ** (от κρυπτός — скрытый и анализ) — наука о методах расшифровки зашифрованной информации **без** предназначенного для такой расшифровки **ключа**

Знание основных положений криптоанализа необходимо для понимания криптографии.

Шнаер сказал, что «существует только один путь стать хорошим разработчиком криптографических алгоритмов - **быть хорошим криптоаналитиком и взламывать алгоритмы**. Множество. Снова и снова. Только после того, как обучающийся продемонстрирует способности к криптонанализу чужих алгоритмов, он сможет серьезно браться за разработку собственных алгоритмов».

**Криптоаналитик** *(нарушитель)* – лицо/группа лиц, целью которых является прочтение или подделка защищенных криптографическими методами сообщений.

**Криптоатака** (атака на шифр) - попытка прочтения или подделки зашифрованного сообщения, вычисления ключа

**Взлом** – удачная криптоатака.

**Надежность** (стойкость) шифрования – трудоемкость взлома (объем работы криптоаналитика, необходимой для вскрытия системы)

**Криптостойкостъ** - характеристика шифра, определяющая его стойкость к криптоатаке.

Показатели криптостойкости:

- количество всех возможных ключей;

- вероятность подбора ключа за заданное время с заданными ресурсами;

- количество операций (или время с заданными ресурсами), необходимое для взлома шифра с заданной вероятностью;

- стоимость вычисления ключевой информации или исходного текста.

**Криптостойкий шифр** – самым быстрым методом взлома является полный перебор всех ключей.

**Криптостойкость системы** защиты информации криптографическими методами зависит от:

1. криптостойкости шифра
2. реализации криптосистем в виде устройств или программ
3. человеческого фактора (подкуп человека, имеющего информацию, дешевле создания суперкомпьютера для взлома шифра).

Исходя из опыта взломов криптосистем, в частности, конкурсов, которые регулярно устраивает RSA Data Security. Опыт показывает, что криптосистемы больше страдают от небрежности в реализации.

Цели КА:

1. **Найти ключ**, использованный для преобразования открытого текста в шифротекст
2. **Создать алгоритм**, способный дешифровать закодированное сообщение

Допущения (кладутся в основу математических или иных моделей):

1. Нарушитель **знает алгоритм шифрования** и особенности его реализации в конкретном случае, но **не знает секретного ключа**.

2. Нарушителю **доступны зашифрованные тексты** (некоторые/все). Нарушитель может иметь доступ к **некоторым исходным текстам**, для которых известны соответствующие им зашифрованные тексты.

3. Нарушитель имеет в своем распоряжении вычислительные, людские, временные и иные **ресурсы, оправданные** потенциальной **ценностью** информации, которая будет добыта в результате криптоанализа.

Атаки могут быть нацелены на каждый элемент алгоритма:

- ключ,

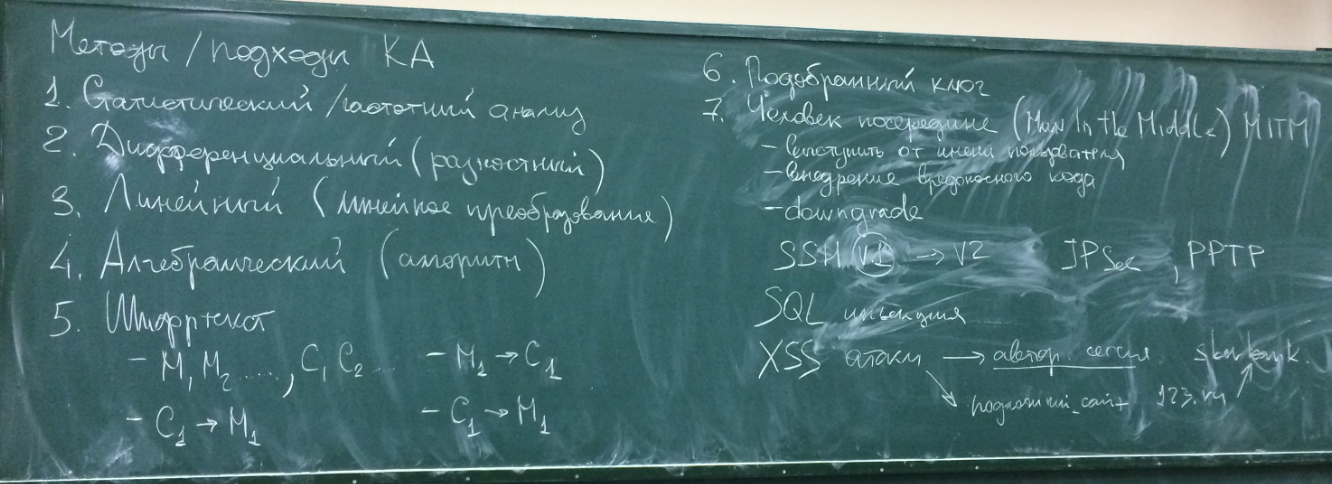
- раундовые ключи,

…

## Подходы и методы КА

Криптоанализ опирается на теорию вероятностей и математическую статистику, алгебру, теорию чисел, теорию алгоритмов и лингвистику.

Логика развития: Лингвистика=>Лингвистика+математика=>Математика+лингвистика=>Математика

**

### Частотный (статистический) КА

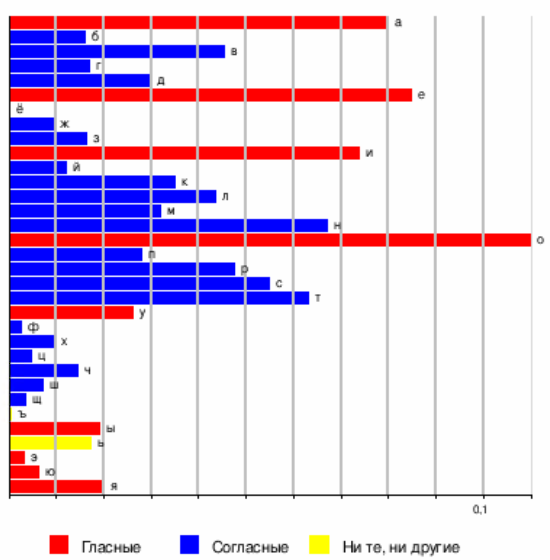
Изучение статистических закономерностей исходных и зашифрованных сообщений (лингвистика+статистика)

Предпосылка: существует нетривиальное статистическое распределение R(L) символов одновременно и в открытом тексте, и в шифротексте (данное распределение сохраняется с точностью до замены символов): R(L0, T0) <==> R(Lc, Tc)

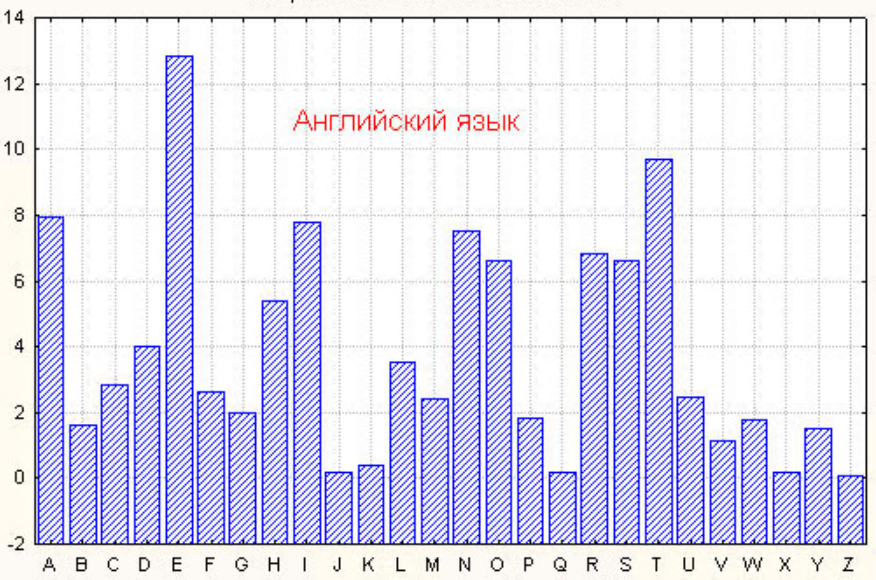
Подстановка сохраняет частоту появления символа. Начинать нужно с самых распространенных (в русском тексте буквы "о", "е", "а", "и", "т", "н", в английском – “e”, “t”, “a”, “i", “n”).

Например, "о" появляется в 41 раз чаще буквы "ф":

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ранг** | **Буква** | **Частотность** | **Ранг** | **Буква** | **Частотность** |
| 1 | о | 0,10983 | 18 | ь | 0,01735 |
| 2 | е | 0,08483 | 19 | г | 0,01687 |
| 3 | а | 0,07998 | 20 | з | 0,01641 |
| 4 | и | 0,07367 | 21 | б | 0,01592 |
| 5 | н | 0,067 | 22 | ч | 0,0145 |
| 6 | т | 0,06318 | 23 | й | 0,01208 |
| 7 | с | 0,05473 | 24 | х | 0,00966 |
| 8 | р | 0,04746 | 25 | ж | 0,0094 |
| 9 | в | 0,04533 | 26 | ш | 0,00718 |
| 10 | л | 0,04343 | 27 | ю | 0,00639 |
| 11 | к | 0,03486 | 28 | ц | 0,00486 |
| 12 | м | 0,03203 | 29 | щ | 0,00361 |
| 13 | д | 0,02977 | 30 | э | 0,00331 |
| 14 | п | 0,02804 | 31 | ф | 0,00267 |
| 15 | у | 0,02615 | 32 | ъ | 0,00037 |
| 16 | я | 0,02001 | 33 | ё | 0,00013 |
| 17 | ы | 0,01898 |  |  |  |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Латинская буква** | **Английский** | **Французский** | **Немецкий** | **Испанский** | **Итальянский** |
| **A** | 7.96 | 7.68 | 5.52 | 12.90 | 11.12 |
| **B** | 1.60 | 0.80 | 1.56 | 1.03 | 1.07 |
| **C** | 2.84 | 3.32 | 2.94 | 4.42 | 4.11 |
| **D** | 4.01 | 3.60 | 4.91 | 4.67 | 3.54 |
| **E** | 12.86 | 17.76 | 19.18 | 14.15 | 11.63 |
| **F** | 2.62 | 1.06 | 1.96 | 0.70 | 1.15 |
| **G** | 1.99 | 1.10 | 3.60 | 1.00 | 1.73 |
| **H** | 5.39 | 0.64 | 5.02 | 0.91 | 0.83 |
| **I** | 7.77 | 7.23 | 8.21 | 7.01 | 12.04 |
| **J** | 0.16 | 0.19 | 0.16 | 0.24 | - |
| **K** | 0.41 | - | 1.33 | - | - |
| **L** | 3.51 | 5.89 | 3.48 | 5.52 | 5.95 |
| **M** | 2.43 | 2.72 | 1.69 | 2.55 | 2.65 |
| **N** | 7.51 | 7.61 | 10.20 | 6.20 | 7.68 |
| **O** | 6.62 | 5.34 | 2.14 | 8.84 | 8.92 |
| **P** | 1.81 | 3.24 | 0.54 | 3.26 | 2.66 |
| **Q** | 0.17 | 1.34 | 0.01 | 1.55 | 0.48 |
| **R** | 6.83 | 6.81 | 7.01 | 6.95 | 6.56 |
| **S** | 6.62 | 8.23 | 7.07 | 7.64 | 4.81 |
| **T** | 9.72 | 7.30 | 5.86 | 4.36 | 7.07 |
| **U** | 2.48 | 6.05 | 4.22 | 4.00 | 3.09 |
| **V** | 1.15 | 1.27 | 0.84 | 0.67 | 1.67 |
| **W** | 1.80 | - | 1.38 | - | - |
| **X** | 0.17 | 0.54 | - | 0.07 | - |
| **Y** | 1.52 | 0.21 | - | 1.05 | - |
| **Z** | 0.05 | 0.07 | 1.17 | 0.31 | 1.24 |



Карл Фридрих Гаусс (1777 - 1855 гг.) считал, что создал нераскрываемый шифр против частотного анализа (ошибся) – рандомизация (теория информации и мера избыточности еще не были созданы)

Алгоритм простого статистического анализа:

1. подсчёт количества каждого из встречающихся символов (N1, N2, …) и расчет вероятности появления символов (Р1, Р2, …)

2. значения Р1, Р2, …сравниваются с таблицей вероятностного распределения букв для предполагаемого **языка оригинала** -> замена совпадающих;

=> появятся слова, многие буквы которых – правильные,

3. корректируем остальные (это поправит и сам ключ)

=> после замены букв могут раскрыться только некоторые слова

4. меняются местами не только буквы с одинаковой частотой, но и с **близкой**

=> появятся оставшиеся слова

Шифр Блеза де Виженера (французского дипломата) – последовательность нескольких шифров Цезаря с различными значениями сдвига (3 века считался полностью криптографически устойчивым)

1863 - Фридрих Касиски предложил:

1. если в открытом тексте есть **два одинаковых набора символов**, то длина блока между ними кратна длине ключа

2. эти одинаковые наборы символов открытого текста при шифровании перейдут в одинаковые отрезки шифротекста

=>ищем в шифротексте одинаковые отрезки длиной **в три и больше символов** (они соответствуют одинаковым отрезкам открытого текста)

Алгоритм статистического анализа многоалфавитного шифра:

1. в шифротексте ищутся пары одинаковых отрезков длины три или больше;

2. вычисляется количество символов, разделяющих стартовые позиции парных отрезков;

3. анализ всех пар одинаковых отрезков -> совокупность расстояний d1, d2, d3,…

4. **длина ключевого слова** будет делителем для каждого из расстояний и, следовательно, для их наибольшего общего делителя.

Любое нарушения исходной системы кодирования усложняет задачу (например, в реальных криптосистемах информация перед шифрованием **сжимается)**.

### Алгебраический КА

Поиск математически слабых звеньев криптоалгоритмов

1997 – в эллиптических системах был выявлен **класс ключей**, которые существенно упрощали криптоанализ.

+++???

### Дифференциальный (или разностный) КА

Анализ зависимости изменения шифрованного текста от изменения исходного текста

Атака с подобранным открытым текстом – нужна возможность зашифровать

- большое количество

- любых текстов

Метод

Обозначения: тексты М1 и М2, шифры С1 и С2.

1. вычисляем «дифференциал» ΔМ=М1⊕М2.

2. с помощью ΔМ ищем «дифференциал» шифротекстов ΔС=C1⊕C2.

Редко подбирается 100% значений ΔС. Но можно определить частоту, с которой шифр возвращает различные значения ΔС, для заданного заранее ΔP.

Получается зависимость ключа (всего или части) от текстов Мi.

Впервые использован Мерфи, улучшен Бихэмом и Шамиром для атаки на DES.

#### Пример ДКА трехраундового блочного шифра

Данный шифр имеет 64 битный размер блока и 128 битный ключ.  
На каждом раунде входной блок делится на 8 байт, каждый из которых проходит через функцию подстановки **Sbox**. После этого данные перемешиваются (XOR) с 64 битным подключом Subkey.

Алгоритм ДКА

1. Проверка дифференциала ΔX:
   1. Сгенерировать произвольный байт X1,
   2. Подобрать для него X2=X1⊕ΔX.
   3. Вычислить Y1=**Sbox** (X1) и Y2=**Sbox** (X2).
   4. Вычислить дифференциал ΔY
   5. Выполнить 1-3 для 256 пар X1 и X2
   6. Выбрать значение ΔY, которое имеет большую вероятность возникновения.
2. Раунд 1 (проверка дифференциала ΔX1)

**Пример:**

Ищем пары X1 и X2, т.ч. ΔX1=80. Пусть из всех 256 пар X1 и X2, в 192 случаях Y1⊕Y2=02. Тогда, вероятность, что при заданном ΔX1=80, значение ΔY1=02, составляет 192/256=3/4.

**=>** при ΔX1=80, значения Y1 и Y2 (такие что ΔY1=02) попадут на вход второго раунда с вероятность P1=3/4

**Ключ не влияет на значение дифференциалов** – шифруются разные тексты с одним ключом, а ключ учитывается через XOR => при вычислении ΔY **байты ключа взаимно исключаются**.

1. Раунд 2 (проверка дифференциала ΔX2)

**Пример (продолжение):**

Для новых 256 пар входных байт X1 и X2 (таких, что ΔX2=X1⊕X2=02) в 64 случаях из 256 ΔY2=88.

=> вероятность, что ΔY2=88, для заданного ΔX2, составляет P2=64/256=1/4.

=>Для данного шифра при паре байт X1 и X2 (таких что ΔX1=80) дифференциал внутреннего состояния после второго раунда составляет ΔY2=88 с вероятностью P= P1\*P2=3/4\*1/4=3/16.

1. Раунд 3
   1. Сгенерировать несколько пар текстов таких, что ΔX=808080808080
   2. Вскрытие первого байта подключа третьего раунда Subkey[0]:
2. Для каждого из 256 возможных вариантов Subkey[0] и для каждой пары шифротекстов {C1, C2}, вычислить U1=Sbox(C1⊕Subkey[0]) и U2=Sbox(C2⊕Subkey[0]).
3. Для верного Subkey[0] P= P1\*P2 пар дадут рассчитанный ΔY

**Пример (продолжение):**

Если Subkey[0] угадан правильно, то приблизительно 3 из 16 пар U1 и U2 дадут ΔU= U1⊕U2=88.

1. Остальные байты раундового ключа подбираются аналогично.
2. Восстановление подключей остальных раундов – аналогично для предпоследнего раунда и далее.

#### Пример ДКА FEAL

1987 - в шифре FEAL увеличена до 64 бит длина ключа (по сравнению с 56 битами в DES)

При наличии 40 пар открытых-закрытых текстов можно получить полный ключ FEAL4 за несколько минут.

Описание алгоритма

Алгоритм FEAL4 - это блочный шифр, с размером блока и длиной ключа равными 64 бита.  
Существует несколько равноценных описаний алгоритма FEAL4, мы воспользуемся наиболее удобным для демонстрации дифференциального криптоанализа.

Шифр состоит из 4-х раундов и использует шесть 32-битных подключей, генерируемых из основного ключа(в дальнейшем будем считать, что каждый подключ генерируется независимо, «увеличив» таким образом порог стойкость с 264 до 2192)

На начальном этапе открытый текст разбивается на два блока, по 32 бита каждый. Левый и правый блоки складываются по модулю два с 32-битными подключами K[4] и K[5] соответственно. Затем левая часть остается без изменений, а правая образуется сложением по модулю два с левым блоком.  
  
После этого выполняется 4 раунда шифрования на каждом из которых правый блок суммируется по модулю два с подключом раунда K[i], а затем полученный результат прогоняется через функцию перестановки F. Результат перестановки складывается с левой частью текста. После этих операций левый и правый блок меняют местами и полученный результат подается на вход следующего раунда.  
  
Последний раунд немного отличается от всех остальных. Левые и правые блоки не меняются местами, как в предыдущих раундах.  
Вместо этого, правый блок складывается по модулю два с левым блоком и полученный результат возвращается в качестве правой части шифротекста. Левая же часть после 4 раунда остается неизменной и составляет первые 32 бита полученного шифротекста.  
  
Оригинальная схема шифра, описанная создателями несколько отличалась от приведенной выше. Вместо шести 32-битных, в оригинальном описании используются двенадцать 16-битных подключа. Однако, оба варианта идентичны и при желании 32-битные ключи можно легко представить в виде 16-битных ключей из оригинального описания.

Единственным белым пятном в описании шифра осталась функция F. На рисунке ее можно представить следующим образом:

На входе функция F получает 4 байта X1, X2, X3, X4. Далее входные байты перемешиваются и проходят через функции G0 или G1. 4 байта полученных после вычисления функций Gx образуют 32-битную выходную последовательность функции F.

Функции G0 и G1 выполняют преобразование 16-битной входной последовательности в 8-битный результат.

Функцию G0 можно выразить следующим образом:, где << — циклический сдвиг влево.

В то время как функция G1 имеет следующее определение:.

Расшифровка алгоритма происходит по такому же самом принципу. Собственно, шифротекст разбивается на левый и правый блок и все операции шифрования выполняются в обратном порядке.  
Т.е. сперва выполняется расшифровка последнего раунда, затем предпоследнего и так далее.

Дифференциальный криптоанализ шифра FEAL4

Как и в приведенном выше примере начнем криптоанализ шифра с исследования функции подстановки F. Именно здесь скрыт самый значительный изъян всего шифра FEAL. Дело в том, что функция F обладает одним катастрофическим с точки зрения безопасности свойством. Любые два значения X1 и X2, такие что их дифференциал X1⊕X2=0x80800000, преобразуются в Y1 и Y2. При этом дифференциал Y1⊕Y2=0x02000000 в 100% случаев.  
Для того, чтобы понять почему так происходит взглянем на преобразование дифференциалов, которое происходит при вычислении функции F.

Легко заметить, что ненулевое значение поступает на вход функции G, только в позиции первого байта. Соответственно на выходе получаем.Нетрудно понять, что это свойство делает шифр абсолютно уязвимым перед дифференциальным криптоанализом. 100% вероятность возникновения определенного дифференциала ΔY позволяет свести количество необходимых для атаки пар открытый-закрытый текст к минимуму.

Рассмотрим какие шаги следует предпринять злоумышленнику для вскрытия ключа последнего раунда K[3].

Атакующий генерирует несколько пар открытых текстов P1 и P2, таких что ΔP=0x8080000080800000. Зная, описанное выше свойство функции F, атакующий в состоянии рассчитать дифференциалы практически на каждом раунде шифрования.  
Отследить поведение дифференциалов вы можете на следующей картинке:  
Как видите дифференциалы просчитываются вплоть до последнего раунда. Но и без этого у атакующего уже достаточно информации, чтобы вскрыть последний раундовый ключ.   
На иллюстрации ниже изображены два последних раунда шифра FEAL4.

Обладая парой шифротекстов C1 и C2 атакующий может вычислить ΔC, получив таким образом значения L' и R'.

На основании этого он может вычислить Z'=L'⊕02.  
C другой стороны для каждой пары шифротекстов C1 и C2, злоумышленник может вычислить Y1 и Y2.  
Зная Y1 и Y2 атакующий начинает перебор ключа K[3]. Для каждого возможного ключа Kpos вычисляется Z1=F(Y1⊕Kpos) и Z2=F(Y2⊕Kpos). Сложив по модулю два значения Z1⊕Z2 атакующий сравнивает получившееся значение с предвычисленным заранее Z'. Если Z'=Z1⊕Z2, значит Kpos вероятнее всего и является искомым подключом K[3].   
Для того, чтобы снизить вероятность ошибки полученный ключ Kpos необходимо проверить с несколькими парами шифротекстов(я в своей реализации использовал 10 пар).

Нетрудно посчитать, что атака, описанная выше, требует 232 вычислений функции F. Это конечно не 264, заявленные создателями шифра, но все равно цифра не очень приятная особенно если мы хотим вычислить все 4 раундовых ключа, а мы несомненно этого хотим.   
К счастью атаку можно упростить и свести количество вычислений ко вполне комфортным 217.

Для этого введем определение функции , где a — набор байт, и z — нулевой байт.   
Для каждого A=(z, a0, a1, z) злоумышленник вычисляет Q0=F(M(Y0)⊕A) и Q1=F(M(Y1)⊕A).   
Нетрудно убедиться, что если A=M(K[3]), тогда второй и третий байт значения Q0⊕Q1 совпадут со вторым и третьим байтом значения Z'. Таким образом мы получаем информацию о двух байтах потенциального ключа Kpos.   
После этого для всех значений b0 и b1, атакующий вычисляет последовательность Kpos=(b0, b0⊕a0, b1⊕a1, b1) и проверяет полученную последовательность описанным выше способом.

После вскрытия подключа K[3], злоумышленник способен восстановить значения в которых находился шифр после третьего раунда и это даст ему возможность атаковать аналогичным образом подключ K[2]. Здесь, правда, необходимо заметить, что для атаки на ключ K[2] необходимо будет использовать другой исходный дифференциал, т.к. дифференциал ΔP=0x8080000080800000 на втором раунде всегда ведет к значению Z'=0x02000000, при любом используемом ключе. И это не позволит угадать ключ второго раунда.   
В качестве дифференциалов для вскрытия каждого из подключей можно использовать следующие значения:  
Round 4: 0x8080000080800000  
Round 3: 0x0000000080800000  
Round 2: 0x0000000002000000  
Round 1: 0x0200000080000000  
После вскрытия всех четырех ключей каждого из раундов легко вычислить ключи K[5] и K[4] для этого достаточно прогнать любой шифротекст через 4 раунда расшифровки и получившееся значение сложить по модулю два с известным открытым текстом.

### Линейный КА

Восстановление неизвестного ключа шифрования по известным открытым сообщениям и соответствующим им шифртекстам путем вычисления линейной аппроксимации между исходным и шифрованным текстом

Предложенный Мацуи, также впервые был применен при взломе DES. Как и дифференциальный анализ в реальных криптосистемах может быть применен только для анализа отдельных блоков криптопреобразований.

#### Атака на DES

1993 (Мацуи) - первая успешная атака на DES (спустя 16 лет после принятия его в качестве стандарта): при наличии 247 пар открытых/зашифрованных текстов, позволяет вскрыть секретный ключ шифра DES за 243 операций.

Восстановление (частично или полностью) ключа K при анализе **большого количества пар открытый/зашифрованный текст**, полученных с использованием одного и того же ключа шифрования K

Ищется уравнение (статистический аналог) алгоритма шифрования:

PI1 ⊕ PI2 ⊕… ⊕ PIa ⊕ CI1 ⊕ CI2 ⊕… ⊕ CIb = KI1 ⊕ KI2 ⊕… ⊕ KIc **(1)**, где Pn, Cn, Kn — n-ые биты текста, шифртекста и ключа.

Найденное уравнение даст восстановить 1 бит ключа.

На значение функции F1(PR, K1)[3, 8, 14, 25] оказывают влияние входные биты 5-го S-блока, а именно биты ключа K1[25~30] и биты блока PR[16~21].

**Алгоритм 1**

Пусть T — количество текстов, для которых левая часть уравнения (1) равняется 0, тогда  
Если T>N/2, где N — число известных открытых текстов.  
Предположить, что KI1 ⊕ KI2 ⊕… ⊕ KIc = 0 (когда P>1/2) или 1 (когда P<1/2).  
Иначе  
Предположить, что KI1 ⊕ KI2 ⊕… ⊕ KIc = 1 (когда P>1/2) или 0 (когда P<1/2).  
Очевидно, что успех алгоритма напрямую зависит от значения |P-1/2| и от количества доступных пар открытый/закрытый текст N. Чем больше вероятность P равенства (1) отличается от 1/2, тем меньше количество открытых текстов N необходимо для атаки.  
  
Возникают две проблемы, которые необходимо решить для успешной реализации атаки:

* Как найти эффективное уравнение вида (1).
* Как с помощью такого уравнения **получить больше одного бита информации о ключе**.

При шифровании DES над текстом производятся следующие операции:

1. Начальная перестановка бит. На этом этапе биты входного блока перемешиваются в определенном порядке.
2. После этого перемешанные биты разбиваются на две половины, которые поступают на вход функции Фейстеля. Для стандартного DES сеть Фейстеля включает 16 раундов, но существуют и другие варианты алгоритма.
3. Два блока, полученных на последнем раунде преобразования объединяются и над полученным блоком производится еще одна перестановка.

На каждом раунде сети Фейстеля 32 младших бита сообщения проходят через функцию f:

Рассмотрим операции, выполняющиеся на этом этапе:

1. Входной блок проходит через функцию расширения E, которая преобразует 32-битный блок в блок длиной 48 бит.
2. Полученный блок складывается с раундовым ключом Ki.
3. Результат предыдущего шага разбивается на 8 блоков по 6 бит каждый.
4. Каждый из полученных блоков Bi проходит через функцию подстановки S-Boxi, которая заменяет 6-битную последовательность, 4-битным блоком.
5. Полученный в результате 32-битный блок проходит через перестановку P и возвращается в качестве результата функции f.

Наибольший интерес, с точки зрения криптоанализа шифра, для нас представляют S блоки, предназначенные для скрытия связи между входными и выходными данными функции f. Для успешной атаки на DES мы сперва построим статистические аналоги для каждого из S-блоков, а затем распространим их на весь шифр.

#### Анализ S блоков

Каждый S-блок принимает на вход 6-битную последовательность, и для каждой такой последовательности возвращается фиксированное 4-битное значение. Т.е. имеется всего 64 варианта входных и выходных данных. Наша задача показать взаимосвязь между входными и выходными данными S блоков. К примеру, для третьего S-блока шифра DES, 3-й бит входной последовательности равен 3-му биту выходной последовательности в 38 случаях из 64. Следовательно, мы нашли следующий статистический аналог для третьего S-блока:  
S3(x)[3] = x[3], который выполняется с вероятность P=38/64.  
Обе части уравнения представляют 1 бит информации. Поэтому в случае если бы левая и правая части были независимы друг от друга, уравнение должно было бы выполняться с вероятностью равной 1/2. Таким образом, мы только что продемонстрировали связь между входными и выходными данными 3-го S-блока алгоритма DES.

Рассмотрим как можно найти статистический аналог S-блока в общем случае.

Для S-блока Sa, 1 ≤ α ≤ 63 и 1 ≤ β ≤ 15, значение NSa(α, β) описывает сколько раз из 64 возможных XOR входных бит Sa наложенных на биты α равны XOR выходных бит, наложенных на биты β, т.е.:где символ • — логическое И.  
Значения α и β, для которых NSa(α, β) сильнее всего отличается от 32, описывают самый эффективный статистический аналог S-блока Sa.  
  
Наиболее эффективный аналог был найден в 5-ом S-блоке шифра DES для α = 16 и β = 15 NS5(16, 15)=12. Это значит, что справедливо следующее уравнение: Z[2]=Y[1] ⊕ Y[2] ⊕ Y[3] ⊕ Y[4], где Z — входная последовательность S-блока, а Y — выходная последовательность.   
Или с учетом того, что в алгоритме DES перед входом в S-блок данные складываются по модулю 2 с раундовым ключом, т.е. Z[2] = X[2] ⊕ K[2] получаем   
X[2] ⊕ Y[1] ⊕ Y[2] ⊕ Y[3] ⊕ Y[4] = K[2], где X и Y — входные и выходные данные функции f без учета перестановок.  
Полученное уравнение выполняется на всех раундах алгоритма DES с одинаковой вероятностью P=12/64.

На следующей таблице приведен список эффективных, т.е. имеющих наибольшее отклонение от P=1/2, статистических аналогов для каждого s-блока алгоритма DES.

#### Построение статистических аналогов для нескольких раундов DES

Покажем теперь каким образом можно объединить статистические аналоги нескольких раундов DES и в итоге получить статистический аналог для всего шифра.  
Для этого рассмотрим трехраундовую версию алгоритма:  
Применим эффективный статистический аналог 5-го s-блока для вычисления определенных бит значения X(2).  
Мы знаем что с вероятностью 12/64 в f-функции выполняется равенство *X[2] ⊕ Y[1] ⊕ Y[2] ⊕ Y[3] ⊕ Y[4] = K[2],* где X[2] — второй входной бит 5-го S-блока, он по сути является 26-м битом последовательности, полученной после расширения входных бит. Анализируя функцию расширения можно установить что на месте 26 бита оказывается 17-й бит последовательности X(1).   
Аналогичным образом, Y[1],…, Y[4] по сути являются 17-м, 18-м, 19-м и 20-м битом последовательности полученной до перестановки P. Исследовав перестановку P, получаем что биты Y[1],…, Y[4] на самом деле являются битами Y(1)[3], Y(1)[8], Y(1)[14], Y(1)[25].   
Бит ключа K[2] вовлеченный в уравнения является 26 битом подключа первого раунда K1 и тогда статистический аналог приобретает следующую форму:  
*X(1)[17] ⊕ Y(1)[3] ⊕ Y(1)[8] ⊕ Y1[14] ⊕ Y(1)[25] = K1[26]*.  
Следовательно, *X(1)[17] ⊕ K1[26] = Y(1)[3] ⊕ Y(1)[8] ⊕ Y(1)[14] ⊕ Y(1)[25]* **(2)** с вероятностью P=12/64.  
Зная 3, 8, 14, 25 биты последовательности Y(1) можно найти 3, 8, 14, 25 биты последовательности X(2):  
*X(2)[3] ⊕ X(2)[8] ⊕ X(2)[14] ⊕ X(2)[25] = PL[3] ⊕ PL[8] ⊕ PL[14] ⊕ PL[25] ⊕ Y(1)[3] ⊕ Y(1)[8] ⊕ Y(1)[14] ⊕ Y(1)[25]* или с учетом уравнения (2)  
*X(2)[3] ⊕ X(2)[8] ⊕ X(2)[14] ⊕ X(2)[25] = PL[3] ⊕ PL[8] ⊕ PL[14] ⊕ PL[25] ⊕ X(1)[17] ⊕ K1[26]* ***(3)***с вероятностью 12/64.  
  
Найдем подобное выражение используя последний раунд. На этот раз мы имеем уравнение  
*X(3)[17] ⊕ K3[26] = Y(3)[3] ⊕ Y(3)[8] ⊕ Y(3)[14] ⊕ Y(3)[25]*.  
Так как  
*X(2)[3] ⊕ X(2)[8] ⊕ X(2)[14] ⊕ X(2)[25] = СL[3] ⊕ СL[8] ⊕ СL[14] ⊕ СL[25] ⊕ Y(3)[3] ⊕ Y(3)[8] ⊕ Y(3)[14] ⊕ Y(3)[25]*  
получаем, что  
*X(2)[3] ⊕ X(2)[8] ⊕ X(2)[14] ⊕ X(2)[25] = СL[3] ⊕ СL[8] ⊕ СL[14] ⊕ СL[25] ⊕ X(3)[17] ⊕ K3[26]* **(4)** с вероятностью 12/64.  
  
Приравняв правые части уравнений (3) и (4) получаем  
*СL[3] ⊕ СL[8] ⊕ СL[14] ⊕ СL[25] ⊕ X(3)[17] ⊕ K3[26] = PL[3] ⊕ PL[8] ⊕ PL[14] ⊕ PL[25] ⊕ X(1)[17] ⊕ K1[26]* с вероятностью (12/64)2+(1-12/64)2.  
С учетом того, что X(1) = PR и X(3) = CR получаем статистический аналог  
*СL[3, 8, 14, 25] ⊕ CR[17] ⊕ PL[3, 8, 14, 25] ⊕ PR[17] = K1[26] ⊕ K3[26]* **(5)**,  
который выполняется с вероятностью (12/64)2+(1-12/64)2=0.7.  
Описанный выше статистический аналог можно представить графически следующим образом (биты на рисунке пронумерованы справа налево и начиная с нуля):  
Все биты в левой части уравнения известны атакующему, следовательно он может применить алгоритм 1 и узнать значение K1[17] ⊕ K3[17]. Покажем как с помощью данного статистического аналога можно вскрыть не 1, а 12 бит ключа шифрования K.

#### Атака на DES с известным открытым текстом

Приведем способ с помощью которого можно расширить атаку и получить сразу 6 бит подключа первого раунда.  
Составляя уравнение (5) мы принимали во внимание тот факт, что нам неизвестно значение F1(PR, K1)[3, 8, 14, 25]. Поэтому мы использовали его статистический аналог K1[26] ⊕ PR[17].  
Вернем вместо выражения K1[26] ⊕ PR[17] значение F1(PR, K1)[3, 8, 14, 25] и получим следующее уравнение:  
*СL[3, 8, 14, 25] ⊕ CR[17] ⊕ PL[3, 8, 14, 25] ⊕ F1(PR, K1)[3, 8, 14, 25] = K3[26]* **(6)**, которое будет выполняться с вероятностью 12/64. Вероятность изменилась так как мы оставили только статистический аналог из третьего раунда, все остальные значения фиксированы.

Покажем каким образом обладая только набором открытых/закрытых текстов можно восстановить значение K1[25~30]. Для этого воспользуемся алгоритмом 2.

**Алгоритм 2**

Пусть N — количество известных перед атакой пар открытый/закрытый текст. Тогда для вскрытия ключа необходимо проделать следующие шаги.  
For (i=0; i<64; i++) do  
{  
For(j=0; j<N; j++) do  
{  
if(СLj[3, 8, 14, 25] ⊕ CRj[17] ⊕ PLj[3, 8, 14, 25] ⊕ F1(PRj, i)[3, 8, 14, 25]=0) then  
Ti=Ti+1  
}}

В качестве вероятной последовательности K1[25~30] принимается такое значение i, при котором выражение |Ti-N/2| имеет максимальное значение.

При достаточном количестве известных открытых текстов алгоритм будет с большой вероятностью возвращать корректное значение шести бит подключа первого раунда K1[25~30]. Объясняется это тем, что в случае если переменная i не равна K1[25~30], тогда значение функции F1(PRj, K)[3, 8, 14, 25] будет случайным и количество уравнений для такого значения i, при котором левая часть равна нулю будет стремиться к N/2. В случае же если подключ угадан верно, левая часть будет с вероятностью 12/64 равна фиксированному биту K3[26]. Т.е. будет наблюдаться значительное отклонение от N/2.

Получив 6 бит подключа K1, можно аналогичным образом вскрыть 6 бит подключа K3. Все что для этого нужно, это заменить в уравнении (6) C на P и K1 на K3:  
*PL[3, 8, 14, 25] ⊕ PR[17] ⊕ CL[3, 8, 14, 25] ⊕ F3(CR, K3)[3, 8, 14, 25] = K1[26]*.

Алгоритм 2 возвратит корректное значение K3[25~30] потому что процесс расшифровки алгоритма DES идентичен процессу шифрования, просто последовательность ключей меняется местами. Так на первом раунде расшифрования используется ключ K3, а на последнем ключ K1.

Получив по 6 бит подключей K1 и K3 злоумышленник восстанавливает 12 бит общего ключа шифра K, т.к. раундовые ключи являются обычной перестановкой ключа K. Количество открытых текстов необходимых для успешной атаки зависит от вероятности статистического аналога. Для вскрытия 12 бит ключа 3-раундового DES достаточно 100 пар открытых/закрытых текстов. Для вскрытия 12 бит ключа 16-раундового DES потребуется порядка 244 пар текстов. Остальные 44 бита ключа вскрываются обычным перебором.

### Группа КА на основе [шифротекста](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D1%82%D0%B5%D0%BA%D1%81%D1%82)

Есть: несколько шифротекстов (получают в результате перехвата зашифрованных сообщений, передаваемых по открытому каналу). В этом случае криптоаналитик может совершить только атаку на основе шифротекста.

Эффективность: самые слабые и неудобные.

**Уровни криптоатаки** на шифртекст (определяется объемом информации, доступной криптоаналитику):

|  |  |  |  |
| --- | --- | --- | --- |
| **Уровень** | **Название атаки** | **Возможности злоумышленника** | **Метод атаки** |
| КА1 | Атака по шифрованному тексту | Доступны некоторые (или все) зашифрованные сообщения | Ищутся открытые тексты и, в идеальном случае, ключ.  Самый сложный и неэффективный подход. |
| КА2 | Атака по паре «исходный текст - шифрованный текст» | Доступны некоторые (или все) зашифрованные сообщения и **соответствующие им исходные** сообщения |  |
| КА3 | Атака по выбранной паре «исходный текст - шифрованный текст» | Можно:   * выбирать исходный текст * получать для него шифрованный текст * вычислять ключ на основе анализа зависимостей между ними   Фактически нарушителю доступно по шифрующее устройство | Например, отправка подложного сообщения, которое будет зашифровано и передано обратно (в ответе как цитата) |

#### Открытые тексты и соответствующие шифротексты (КА2)

Основная статья: [**Атака на основе открытых текстов**](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BD%D0%B0_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B5_%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D1%85_%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2)

Есть: не только шифротексты, но и соответствующие им открытые тексты.

Методы получения открытого текста:

1. Расшифровка файла: можно догадаться о содержимом файла по расширению
   1. Rar начинается с «!RAR».
   2. Zip и Docx начинаются с «PK»
   3. …
2. Взлом переписки: письмо имеет структуру
   1. Приветствие (можно подобрать из перечня «Здравствуйте!», «Добрый день» и т.д.)
   2. Основной текст
   3. Заключительная форма вежливости («С уважением», «Искренне Ваш» и т.п.)
   4. Подпись (ФИО отправителя)

#### Подобранный открытый текст (КА3)

Есть: возможность подбора результатов шифрования нескольких открытых текстов (подбор блоков открытого текста позволяет получить информацию о ключе шифрования)

Методы получения шифртекстов:

1. Отправить поддельное сообщение якобы от одного из пользователей, которые обычно пользуются шифрованием.
2. Можно получить ответ, содержащий зашифрованный текст, цитирующий поддельное сообщение.

#### Адаптивно подобранный открытый текст

Основная статья: [**Атака на основе адаптивно подобранного открытого текста**](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BD%D0%B0_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B5_%D0%B0%D0%B4%D0%B0%D0%BF%D1%82%D0%B8%D0%B2%D0%BD%D0%BE_%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D1%80%D0%B0%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B3%D0%BE_%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%B0)

Есть: выбранный шифртекст, принятие решения о шифровании открытого текста на основе полученных результатов шифрования (криптоаналитик выбирает один большой блок открытого текста для шифрования)

Обратная связь – результат шифрования любых блоков открытого текста используются при выборе следующих шифруемых блоков открытого текста.

#### КА с подобранным шифротекстом

Есть: временный доступ к дешифрующему устройству (за ограниченное время можно получить из известных шифротекстов соответствующие им открытые тексты)

Цель: получить ключ шифрования.

Называют «Атакой в обеденное время» (lunchtime attack) или «Ночной атакой» (midnight attack).

Задача

Дано: С1, P1=Dk(С1), С2, P2=Dk(С2), С3, P3=Dk(С3), …, Сn, Pn=Dk(Сn),

где Сn — n-ый имеющийся шифротекст, Pn — соответствующий Сn открытый текст, а Dk — функция дешифрования при помощи ключа k.

Найти: используемый ключ шифрования k.

### КА на основе подобранного ключа

Криптоаналитик не занимается перебором ключей, а наблюдает за работой алгоритма шифрования, в котором используются несколько ключей.

Криптоаналитик изначально ничего не знает о точном значении ключей, зато ему известно некоторое математическое отношение, связывающее между собой ключи (криптоаналитик выяснил, что последние 80 битов у всех ключей одинаковы, хотя сами значения битов ему неизвестны).

### Человек посередине

**MITM-атака** (англ. *Man in the middle*), атака **посредника** – криптоаналитик способен читать, видоизменять и удалять сообщения, которыми обмениваются корреспонденты, оставаясь незамеченным – компрометация канала связи

Пример в литературе:

* «[Сказке о царе Салтане](https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D0%B0%D0%B7%D0%BA%D0%B0_%D0%BE_%D1%86%D0%B0%D1%80%D0%B5_%D0%A1%D0%B0%D0%BB%D1%82%D0%B0%D0%BD%D0%B5)» [А. С. Пушкина](https://ru.wikipedia.org/wiki/%D0%9F%D1%83%D1%88%D0%BA%D0%B8%D0%BD,_%D0%90%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80_%D0%A1%D0%B5%D1%80%D0%B3%D0%B5%D0%B5%D0%B2%D0%B8%D1%87) - три «человека посередине» (криптоаналитика): ткачиха, повариха и Бабариха. Именно они подменяют письма, адресованные царю, и его ответную корреспонденцию.
* А. Дюма "Граф Монте-Кристо" - сообщение, передаваемое по оптическому телеграфу, подменяется на одной из ретрансляционных станций.
* [Заговор Бабингтона](https://en.wikipedia.org/wiki/Babington_Plot) - заговор против [Елизаветы I](https://ru.wikipedia.org/wiki/%D0%95%D0%BB%D0%B8%D0%B7%D0%B0%D0%B2%D0%B5%D1%82%D0%B0_I), в ходе которого [Уолсингем](https://ru.wikipedia.org/wiki/%D0%A3%D0%BE%D0%BB%D1%81%D0%B8%D0%BD%D0%B3%D0%B5%D0%BC,_%D0%A4%D1%80%D1%8D%D0%BD%D1%81%D0%B8%D1%81) перехватывал корреспонденцию Марии Стюарт, королевы Шотландии.

Атаки «человек посередине» популярны в системах, совершающих финансовые операции через интернет ([электронный бизнес](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B1%D0%B8%D0%B7%D0%BD%D0%B5%D1%81), [интернет-банкинг](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%B1%D0%B0%D0%BD%D0%BA%D0%B8%D0%BD%D0%B3), [платёжный шлюз](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D1%82%D1%91%D0%B6%D0%BD%D1%8B%D0%B9_%D1%88%D0%BB%D1%8E%D0%B7)).

Сценарии

#### Выступить от имени пользователя

Цели:

1. Получение доступа к учетной записи пользователя и совершение действий от ее имени.
2. Отправка/прием сообщений (перехватить сообщения обмена открытыми ключами между клиентом и сервером и изменить их)

Принцип атаки:

1. прослушивание канала связи
2. извлечение полезной информации
3. перенаправление на какой-нибудь внешний ресурс
4. подмена перехваченного сообщения

Исходные данные:

* A передает B информацию:
* C обладает знаниями о структуре и свойствах используемого метода передачи данных, а также о факте планируемой передачи собственно информации, которую С планирует перехватить.

Атака:

1. С «представляется» А как В, а В — как А.
2. А, ошибочно полагая, что он направляет информацию В, посылает её С.
3. С, получив информацию, и совершив с ней некоторые действия (например, скопировав или модифицировав в своих целях) пересылает данные получателю В
4. В считает, что информация была получена им напрямую от А.

Пример атаки подмены ключа:

1. Алиса отправляет Бобу сообщение, которое перехватывает Мэлори:

Алиса «Привет Боб, Это Алиса. Пришли мне свой открытый ключ» → Мэлори Боб

1. Мэлори пересылает сообщение Бобу; Боб не может догадаться, что это сообщение не от Алисы:

Алиса Мэлори «Привет Боб, Это Алиса. Пришли мне свой открытый ключ» → Боб

1. Боб посылает свой ключ:

Алиса Мэлори ← [ключ Боба] Боб

1. Мэлори подменяет ключ Боба своим и пересылает сообщение Алисе:

Алиса ← [ключ Мэлори] Мэлори Боб

1. Алиса шифрует сообщение ключом Мэлори, считая что это ключ Боба, и только он может расшифровать его:

Алиса «Встретимся на автобусной остановке!» [зашифровано ключом Мэлори] → Мэлори Боб

1. Мэлори расшифровывает сообщение, читает его, модифицирует его, шифрует ключом Боба и отправляет его:

Алиса Мэлори «Жди меня у входа в музей в 18:00!» [зашифровано ключом Боба] → Боб

1. Боб считает, что это сообщение Алисы.

Как защититься: необходимо использовать методы подтверждения того, что обе стороны используют правильные открытые ключи

#### Внедрение вредоносного кода

Внедрение кода применяется для:

* захвата уже авторизованной сессии
* выполнения собственных команд на сервере
* отправки ложных ответов клиенту

Позволяет криптоаналитику вставлять свой код в электронные письма, SQL выражения и веб-страницы ([SQL-инъекции](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D0%B5%D0%B4%D1%80%D0%B5%D0%BD%D0%B8%D0%B5_SQL-%D0%BA%D0%BE%D0%B4%D0%B0), HTML/script-инъекции или [XSS-атаки](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B0%D0%B9%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%81%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%B8%D0%BD%D0%B3)), модифицировать загружаемые пользователем бинарные файлы

#### Downgrade Attack

Пользователя вынуждают использовать небезопасные функции, или протоколы, которые всё ещё поддерживаются из соображений совместимости (протоколы [SSH](https://ru.wikipedia.org/wiki/SSH), [IPsec](https://ru.wikipedia.org/wiki/IPsec) и [PPTP](https://ru.wikipedia.org/wiki/PPTP)).

#### [SSH](https://ru.wikipedia.org/wiki/SSH) V1 вместо [SSH](https://ru.wikipedia.org/wiki/SSH) V2

2003 - Атакующий может попытаться изменить параметры соединения между сервером и клиентом при установлении между ними соединения (клиент начинает сессию [SSH1](https://ru.wikipedia.org/wiki/SSH) вместо [SSH2](https://ru.wikipedia.org/wiki/SSH) изменив номер версии «1.99» для [SSH](https://ru.wikipedia.org/wiki/SSH) сессии на «1.51»). Протокол [SSH-1](https://ru.wikipedia.org/wiki/SSH) имеет уязвимости.

#### IPsec

Если сделать вид, что IPsec сессия не может начаться на другом конце (сервере), сообщения будут пересылаться в явном виде, в случае если хост-машина работает в rollback режиме.

#### PPTP

1. при согласовании параметров [PPTP](https://ru.wikipedia.org/wiki/PPTP) сессии возможен переход на менее безопасную [PAP](https://ru.wikipedia.org/wiki/Password_Authentication_Protocol) аутентификацию, MSCHAP V1 (то есть «откатиться» с MSCHAP V2 до версии 1), либо не использовать шифрование вообще.
2. повтор этапа согласования параметров PPTP сессии (послать Terminate-Ack-пакет), выкрасть пароль из существующего туннеля и повторить атаку.

#### Атака владельца публичного сервиса

Владелец сервиса коммуникаций контролирует информацию, которой обмениваются корреспонденты, и может осуществить атаку ([социальная сеть](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), публичный сервис [электронной почты](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BE%D1%87%D1%82%D0%B0) и [система мгновенного обмена сообщениями](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BC%D0%B3%D0%BD%D0%BE%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%BC%D0%B5%D0%BD%D0%B0_%D1%81%D0%BE%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8)).

#### Шифрование HTTP-соединения

Стандартное HTTP-соединение между клиентом и сервером непрямое (связаны через некоторое количество промежуточных серверов - MITM-атаку можно проводить на любом из этих серверов).

Разбиение оригинального TCP-соединения на два новых:

* одно между собой и клиентом,
* другое между собой и сервером.

[HTTPS](https://ru.wikipedia.org/wiki/HTTPS)- протокол использует TLS или SSL для шифрования запросов (казалось бы - канал защищён от [сниффинга](https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%BA%D0%B5%D1%80%D1%81%D0%BA%D0%B0%D1%8F_%D0%B0%D1%82%D0%B0%D0%BA%D0%B0) и MITM-атак)

Атакующий может создать две независимые SSL-сессии для каждого TCP-соединения:

* Клиент устанавливает SSL-соединение с атакующим,
* Атакующий создает соединение с сервером

Браузер в таких случаях предупреждает, что сертификат не подписан доверенным центром сертификации (рядовой пользователь игнорирует предупреждение).

У злоумышленника может оказаться сертификат, подписанный центром сертификации (например, такие сертификаты иногда используются для [DLP](https://ru.wikipedia.org/w/index.php?title=Data_loss_prevention&action=edit&redlink=1)[[5]](https://ru.wikipedia.org/wiki/%D0%A7%D0%B5%D0%BB%D0%BE%D0%B2%D0%B5%D0%BA_%D0%BF%D0%BE%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5#cite_note-5)).

Обнаружение MITM-атак

Необходимо проанализировать сетевой трафик. Например, для детектирования атаки по SSL следует обратить внимание на следующие параметры:

* Технически
  + IP-адрес сервера
  + DNS-сервер
* Пользователю
  + [X.509](https://ru.wikipedia.org/wiki/X.509)-сертификат сервера
    - Подписан ли сертификат самостоятельно?
    - Подписан ли сертификат [центром сертификации](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BD%D1%82%D1%80_%D1%81%D0%B5%D1%80%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8)?
    - Был ли сертификат аннулирован?
    - Менялся ли сертификат недавно?
    - Получали ли другие клиенты в интернете такой же сертификат?

Инструменты

Программы для осуществления, обнаружения и тестирования системы на уязвимости

* [dsniff](http://www.monkey.org/~dugsong/dsniff/) — инструмент для [SSH](https://ru.wikipedia.org/wiki/SSH) и [SSL](https://ru.wikipedia.org/wiki/Secure_Sockets_Layer) атак
* [Cain](http://www.oxid.it/) — инструмент для проведения атаки «человек посередине». Имеет [графический интерфейс](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F). Поддерживает [сниффинг](https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%BA%D0%B5%D1%80%D1%81%D0%BA%D0%B0%D1%8F_%D0%B0%D1%82%D0%B0%D0%BA%D0%B0) и [ARP-spoofing](https://ru.wikipedia.org/wiki/ARP-spoofing)
* [Ettercap](http://ettercap.sourceforge.net/) — инструмент для проведения атак в локальной сети
* [Karma](http://www.theta44.org/karma/) — использует [атаку злой двойник (Evil Twin)](https://ru.wikipedia.org/wiki/%D0%97%D0%BB%D0%BE%D0%B9_%D0%B4%D0%B2%D0%BE%D0%B9%D0%BD%D0%B8%D0%BA) для проведения MITM-атак
* [AirJack](http://sourceforge.net/projects/airjack/) — программа демонстрирует основанные на стандарте 802.11 MITM-атаки
* [SSLStrip](http://www.thoughtcrime.org/software/sslstrip/) — инструмент для MITM-атаки на [SSL](https://ru.wikipedia.org/wiki/Secure_Sockets_Layer)
* [SSLSniff](http://www.thoughtcrime.org/software/sslsniff/) — инструмент для MITM-атаки на [SSL](https://ru.wikipedia.org/wiki/Secure_Sockets_Layer). Изначально был создан для обнаружения уязвимостей в [Internet Explorer](https://ru.wikipedia.org/wiki/Internet_Explorer).[[8]](https://ru.wikipedia.org/wiki/%D0%A7%D0%B5%D0%BB%D0%BE%D0%B2%D0%B5%D0%BA_%D0%BF%D0%BE%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5#cite_note-8)
* [Mallory](http://intrepidusgroup.com/insight/mallory/) — прозрачный прокси-сервер, осуществляющий [TCP](https://ru.wikipedia.org/wiki/TCP)- и [UDP](https://ru.wikipedia.org/wiki/UDP)-MITM-атаки. Может быть также использован для атаки на протоколы [SSL](https://ru.wikipedia.org/wiki/Secure_Sockets_Layer), [SSH](https://ru.wikipedia.org/wiki/SSH) и многие другие
* wsniff — инструмент для проведения атак на 802.11 [HTTP](https://ru.wikipedia.org/wiki/HTTP)/[HTTPS](https://ru.wikipedia.org/wiki/HTTPS) протокол

### Полный перебор

**Метод «грубой силы»** (*brute force*) - [поиск решения исчерпыванием всевозможных вариантов](https://ru.wikipedia.org/w/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA%D0%B0_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B8%D1%81%D1%87%D0%B5%D1%80%D0%BF%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%D0%BC&action=edit&redlink=1) в пространстве решений

Посылки:

* 1. Любая задача из [класса NP](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_NP) может быть решена полным перебором.

Даже если вычисление целевой функции от каждого возможного решения может быть осуществлено за [полиномиальное время](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_P), полный перебор может потребовать [экспоненциального времени](https://ru.wikipedia.org/wiki/%D0%AD%D0%BA%D1%81%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) работы.

* 1. Доступен …
  2. Brute force оправдан на *начальном этапе* в разработке алгоритма
  3. Криптографические атаки, основанные на методе полного перебора, являются самыми универсальными, но и самыми долгими.

Применимость: в дискретной детерминированной системе, состояния которой могут быть легко проанализированы

[Хакерские атак](https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%BA%D0%B5%D1%80%D1%81%D0%BA%D0%B0%D1%8F_%D0%B0%D1%82%D0%B0%D0%BA%D0%B0)и

[Proof by exhaustion](https://en.wikipedia.org/wiki/Proof_by_exhaustion)

«Метод исчерпывания» включает в себя целый класс различных методов. Обычно постановка задачи подразумевает рассмотрение конечного числа состояний данной логической системы с целью выявления истинности логического утверждения посредством независимого анализа каждого состояния[[1]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80#cite_note-Ried_.26_Knipping.E2.80.942010.E2.80.94.E2.80.94133-1). Методика доказательства утверждения состоит из двух частей:

1. Доказательство возможности исчерпания всех состояний системы. Требуется показать, что любое конкретное состояние системы (например, значение доказываемого логического выражения) соответствует хотя бы одному из рассматриваемых кандидатов в решения.
2. Поверка каждого варианта и доказательство того, что рассматриваемый вариант является или не является решением поставленной задачи.

[Кризис оснований математики](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%B7%D0%B8%D1%81_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B8)

#### Применение brute force

**1. Рекурсия** - разновидность полного перебора (также применима лишь для [дискретных по структуре систем](https://ru.wikipedia.org/w/index.php?title=%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0&action=edit&redlink=1)).

Система поддается разделению на множество подсистем, структура которых аналогична структуре исходной системы (и так далее, пока системы не станут тривиальными – для которых задача решается тривиально).

**2. Поиск оптимального решения для просчета всего дерева возможных вариантов**: алгоритм оценки времени вычисления цепочечных произведений матриц. Нельзя ускорить – сложность равна методу «грубой силы»

Классическая задача [динамического программирования](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) - **определение приоритетов вычислений матричных произведений:** вычислить цепочку (матричного произведения) за наименьшее время.

Тривиальный последовательный алгоритм вычисления произведения можно оптимизировать, произвольно группируя матрицы (выбирая пару элементов цепочки и заменяя её результирующей матрицей), так как [матричное произведение](https://ru.wikipedia.org/wiki/%D0%A3%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86) является [ассоциативной операцией](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D1%8F).

Решение данной задачи сократит время вычисления матричной цепочки.

Алгоритм решения (полный перебор):

* 1. рассмотреть все возможные последовательности вычислений
  2. выбрать последовательность, имеющую наименьшее число скалярных произведений.

Этот алгоритм требует экспоненциальное время вычисления - выигрыш от вычисления цепочки самым эффективным образом (оптимальная [стратегия](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%B0%D1%82%D0%B5%D0%B3%D0%B8%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0))) может быть полностью потерян временем нахождения этой стратегии.

**3. Другие классические задачи**, для которых полный перебор является единственным известным методом решения (либо изначальной реализацией, оптимизированной в дальнейшем):

* [Поиск наибольшей подстроки](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D0%B8%D0%B1%D0%BE%D0%BB%D1%8C%D1%88%D0%B0%D1%8F_%D0%BE%D0%B1%D1%89%D0%B0%D1%8F_%D0%BF%D0%BE%D0%B4%D0%BF%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C)
* [Задача поиска ближайшего соседа](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA%D0%B0_%D0%B1%D0%BB%D0%B8%D0%B6%D0%B0%D0%B9%D1%88%D0%B5%D0%B3%D0%BE_%D1%81%D0%BE%D1%81%D0%B5%D0%B4%D0%B0)
* Обнаружение [коллизий хеш-функций](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8)

#### Примеры атак методом «грубой силы»

**Проба на ключ** ("лобовая" атака) – главный и самый распространенный практический метод.

Атака «Энигмы»

[**Криптоанализ «Энигмы»**](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%C2%AB%D0%AD%D0%BD%D0%B8%D0%B3%D0%BC%D1%8B%C2%BB), [**Turing Bombe**](https://ru.wikipedia.org/wiki/Turing_Bombe)

Артур Шербиус изобрел Энигму (1918), которая минимизировала количество повторяющихся отрезков шифротекста.

1926 - первые перехваты сообщений, зашифрованных Энигмой.

Противостояние польских и германских криптографов:

* поляки находили очередной результат по взлому немецкой криптосистемы (основной инструмент - дешифровальная машина [«Бомба»](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B1%D0%BE%D0%BC%D0%B1%D0%B0), созданная польскими математиками для министерства обороны Польши - при работе она тикала)
* германские инженеры, постоянно модернизирующие систему «Энигма», создавали новые трудности

Летом [1939 года](https://ru.wikipedia.org/wiki/1939_%D0%B3%D0%BE%D0%B4), когда неизбежность вторжения в Польшу стала очевидна, [бюро](https://ru.wikipedia.org/wiki/%D0%91%D1%8E%D1%80%D0%BE_%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2) передало результаты своей работы английской и французской разведкам. Дальнейшая работа по взлому была организована в [Блетчли-парке](https://ru.wikipedia.org/wiki/%D0%91%D0%BB%D0%B5%D1%82%D1%87%D0%BB%D0%B8-%D0%BF%D0%B0%D1%80%D0%BA). При непосредственной поддержке создателей «Бомбы» в Англии был сконструирован более «продвинутый» агрегат

Теоретические работы по криптоанализу Энигмы выполнялись математиками

* в 1938 году польским криптоаналитиком [Марианом Реевским](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B5%D0%B2%D1%81%D0%BA%D0%B8%D0%B9,_%D0%9C%D0%B0%D1%80%D0%B8%D0%B0%D0%BD)
* завершил англичанин - [Алан Матисон Тьюринг](https://ru.wikipedia.org/wiki/%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3,_%D0%90%D0%BB%D0%B0%D0%BD_%D0%9C%D0%B0%D1%82%D0%B8%D1%81%D0%BE%D0%BD).

Принцип работы дешифратора Тьюринга:

1. перебор возможных вариантов ключа шифра
2. попытка расшифровки текста
3. сравнение с известной структурой дешифруемого сообщения или частью открытого текста

Количество элементарных математических операций для взлома - астрономическое.

С современной т.з. шифр «Энигмы» не очень надёжен, но вскрыть его позволило только сочетание этого фактора с:

* наличием множества перехваченных сообщений,
* захваченных кодовых книг,
* донесений разведки,
* результатов усилий военных и даже [террористических](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%80%D1%80%D0%BE%D1%80%D0%B8%D0%B7%D0%BC) атак

После войны [Черчилль](https://ru.wikipedia.org/wiki/%D0%A7%D0%B5%D1%80%D1%87%D0%B8%D0%BB%D0%BB%D1%8C), из соображений безопасности, приказал разрушить эти машины.

Функционирующая восстановленная версия «бомбы» представлена в музее [Блетчли-парк](https://ru.wikipedia.org/wiki/%D0%91%D0%BB%D0%B5%D1%82%D1%87%D0%BB%D0%B8-%D0%BF%D0%B0%D1%80%D0%BA). Каждый из барабанных роторов имитирует работу соответствующего ротора Энигмы. Всего используется 36 эквивалентных узлов Энигмы. В правой части средней полки находятся три *идентификационных* ротора. Восстановление «бомбы» стало возможным в 2008 году благодаря трудам [Джона Харпера](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B0%D1%80%D0%BF%D0%B5%D1%80,_%D0%94%D0%B6%D0%BE%D0%BD&action=edit&redlink=1), а церемонию первого запуска возглавил покровитель [Британского компьютерного общества](https://ru.wikipedia.org/w/index.php?title=%D0%91%D1%80%D0%B8%D1%82%D0%B0%D0%BD%D1%81%D0%BA%D0%BE%D0%B5_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%BE&action=edit&redlink=1), [Эдвард, герцог Кентский](https://ru.wikipedia.org/wiki/%D0%AD%D0%B4%D0%B2%D0%B0%D1%80%D0%B4,_%D0%B3%D0%B5%D1%80%D1%86%D0%BE%D0%B3_%D0%9A%D0%B5%D0%BD%D1%82%D1%81%D0%BA%D0%B8%D0%B9).

Атака на DES

Компьютер компании [EFF](https://ru.wikipedia.org/wiki/Electronic_Frontier_Foundation) для взламывания шифра [DES](https://ru.wikipedia.org/wiki/DES). Имея в распоряжении 1 856 микросхем, взламывал ключ DES всего за несколько суток (двусторонняя плата «DES Cracker» содержала 64 микросхемы «Deep Crack», цена всего вычислительного комплекса $250К)

#### Пример продолжительности подбора паролей

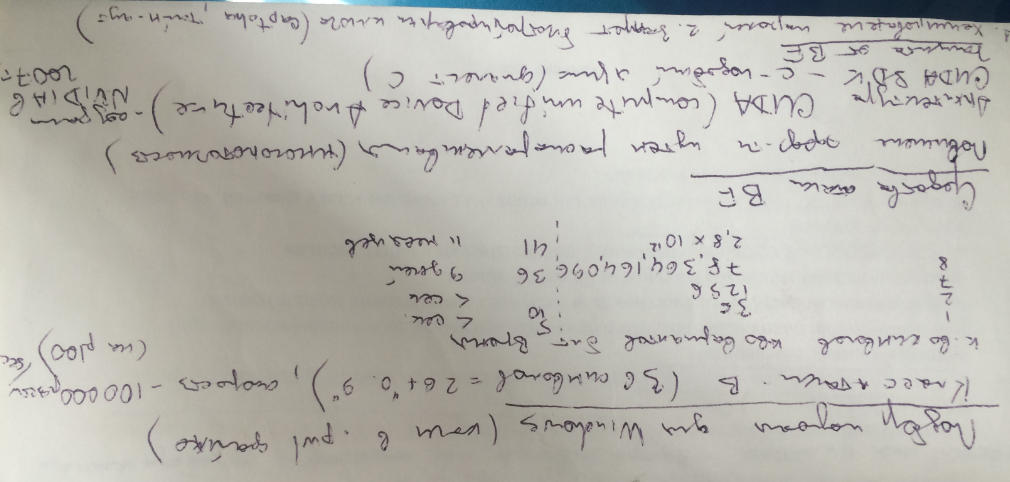
Таблица (зависимость время полного перебора паролей от их длины)

- в пароле могут использоваться 36 различных символов ([латинские](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BB%D1%84%D0%B0%D0%B2%D0%B8%D1%82) буквы одного регистра + цифры),

- скорость перебора на [Pentium 100](https://ru.wikipedia.org/wiki/Pentium) составляет 100 000 паролей в секунду ([класс атаки B](https://ru.wikipedia.org/w/index.php?title=%D0%9A%D0%BB%D0%B0%D1%81%D1%81_%D0%B0%D1%82%D0%B0%D0%BA%D0%B8&action=edit&redlink=1) - восстановление пароля из [Кэша](https://ru.wikipedia.org/wiki/%D0%9A%D1%8D%D1%88) Windows ([.PWL](https://ru.wikipedia.org/w/index.php?title=PWL&action=edit&redlink=1) файлов).

|  |  |  |  |
| --- | --- | --- | --- |
| **Кол-во знаков** | **Кол-во вариантов** | [**Стойкость**](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%BF%D0%B0%D1%80%D0%BE%D0%BB%D1%8F#.D0.AD.D0.BD.D1.82.D1.80.D0.BE.D0.BF.D0.B8.D1.8F_.D0.BA.D0.B0.D0.BA_.D0.BC.D0.B5.D1.80.D0.B0_.D1.81.D0.BB.D0.BE.D0.B6.D0.BD.D0.BE.D1.81.D1.82.D0.B8_.D0.BF.D0.B0.D1.80.D0.BE.D0.BB.D1.8F) | **Время перебора** |
| 1 | 36 | 5 бит | менее секунды |
| 2 | 1296 | 10 бит | менее секунды |
| 3 | 46 656 | 15 бит | менее секунды |
| 4 | 1 679 616 | 21 бит | 17 секунд |
| 5 | 60 466 176 | 26 бит | 10 минут |
| 6 | 2 176 782 336 | 31 бит | 6 часов |
| ***7*** | ***78 364 164 096*** | ***36 бит*** | ***9 дней*** |
| ***8*** | ***2,821 109 9x1012*** | ***41 бит*** | ***11 месяцев*** |
| 9 | 1,015 599 5x1014 | 46 бит | 32 года |
| 10 | 3,656 158 4x1015 | 52 бита | 1 162 года |
| 11 | 1,316 217 0x1017 | 58 бит | 41 823 года |
| 12 | 4,738 381 3x1018 | 62 бита | 1 505 615 лет |

Таким образом, пароли длиной до 8 символов включительно в общем случае не являются надежными.



#### Уязвимость протокола WPS ([Wi-Fi Protected Setup](https://ru.wikipedia.org/wiki/Wi-Fi_Protected_Setup" \o "Wi-Fi Protected Setup))

2007 - организация [Wi-Fi Alliance](https://ru.wikipedia.org/wiki/WECA) начала продажу беспроводного оборудования для домашних сетей с поддержкой стандарта Wi-Fi Protected Setup (производители: [Cisco/Linksys](https://ru.wikipedia.org/wiki/Linksys), [Netgear](https://ru.wikipedia.org/wiki/Netgear), [Belkin](https://ru.wikipedia.org/wiki/Belkin) и [D-Link](https://ru.wikipedia.org/wiki/D-Link)). Стандарт WPS упрощает настройку беспроводной сети.

2011 - опубликованы уязвимости WPS, позволяющие подобрать PIN-код беспроводной сети за несколько часов на обыкновенном персональном компьютере. Сейчас [Координационный центр CERT](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BE%D1%80%D0%B4%D0%B8%D0%BD%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D1%86%D0%B5%D0%BD%D1%82%D1%80_CERT) не рекомендует производителям выпускать оборудование, поддерживающее WPS.

#### Массовый взлом домашних сетей посредством [WASP](https://ru.wikipedia.org/wiki/WASP) (Wireless Aerial Surveillance Platform)

2010 - на конференции [DEFCON18](https://ru.wikipedia.org/wiki/DEFCON_(%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D0%B8%D1%8F)) представлен **беспилотный летательный аппарат**, предназначенный для массового сбора статистики по домашним Wi-Fi сетям (компактный бортовой компьютер под управлением [BackTrack Linux](https://ru.wikipedia.org/wiki/BackTrack))

Выполнял автоматический взлом паролей беспроводных сетей посредством brute force.

2011- авторы WASP представили улучшенную версию беспилотника (в частности, добавили обращение к вычислительным ресурсам удаленных компьютеров при подборе паролей)

#### Средства проведения атаки

[Параллельные вычисления](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F), с использованием компьютеров, способных к [многопоточным вычислениям](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) (использующие [GPU](https://ru.wikipedia.org/wiki/GPU))

- **Nvidia Tesla C2075** обладает 448 ядрами архитектуры CUDA, 6ГБ оперативной памяти типа [GDDR5](https://ru.wikipedia.org/wiki/GDDR5) и имеет пиковую производительность, равную 1030 [Гфлопс](https://ru.wikipedia.org/wiki/FLOPS) при вычислениях с [одинарной точностью](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%BE_%D0%BE%D0%B4%D0%B8%D0%BD%D0%B0%D1%80%D0%BD%D0%BE%D0%B9_%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D0%B8) (после создания компанией [Nvidia](https://ru.wikipedia.org/wiki/Nvidia) архитектуры [CUDA](https://ru.wikipedia.org/wiki/CUDA) в 2007 году)

Практические реализации: [Cryptohaze Multiforcer](https://www.cryptohaze.com/multiforcer.php), [Pyrit](http://code.google.com/p/pyrit/))

- [ATI Stream Technology](https://ru.wikipedia.org/w/index.php?title=ATI_Stream_Technology&action=edit&redlink=1),

- [OpenCL](https://ru.wikipedia.org/wiki/OpenCL).

#### Устойчивость к атаке полного перебора

Устойчивость к brute-force атаке определяет используемый в криптосистеме [ключ](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)) шифрования.

Увеличение длины ключа экспоненциально увеличивает сложность взлома.

Шифр длиной в *N* битов взламывается, в наихудшем случае, за время, пропорциональное 2*N* (среднее время взлома в два раза меньше = 2*N*-1)

Повышения устойчивости шифра к «brute force» - запутывание ([обфускация](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%84%D1%83%D1%81%D0%BA%D0%B0%D1%86%D0%B8%D1%8F" \o "Обфускация)) шифруемых данных, что делает нетривиальным отличие зашифрованных данных от незашифрованных.

#### Методы оптимизации полного перебора

[**Метод ветвей и границ**](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%B2%D0%B5%D1%82%D0%B2%D0%B5%D0%B9_%D0%B8_%D0%B3%D1%80%D0%B0%D0%BD%D0%B8%D1%86)

Для ускорения перебора [метод ветвей и границ](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%B2%D0%B5%D1%82%D0%B2%D0%B5%D0%B9_%D0%B8_%D0%B3%D1%80%D0%B0%D0%BD%D0%B8%D1%86) использует отсев подмножеств допустимых решений, заведомо не содержащих [оптимальных решений](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B5).

[**Параллельные вычисления**](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F)

Распараллеливание:

Построение конвейера (1 процессор выполняет 1 вид операции, передавая другие операции другим процессорам).

Распараллеливаемый процесс представим как цепочку простейших действий (операций). Каждый из N процессоров выполняет операции: приёма данных от предыдущего процессора, выполнения операции, передачи данных следующему процессору.

Конвейер из последовательно соединённых, параллельно и синхронно работающих процессоров работает со скоростью выполнения одной операции одним процессором.

Множество возможных ключей разбивается на непересекающиеся подмножества

Система из N машин перебирает ключи так, что каждая машина перебирает ключи из отдельного множества. Система прекращает работу, если одна из машин нашла ключ. Самое трудное - это разделение множества

[**Радужная таблица**](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B4%D1%83%D0%B6%D0%BD%D0%B0%D1%8F_%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0)

Компьютерные системы, которые используют пароли для [аутентификации](https://ru.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F), должны каким-то образом определять правильность введенного пароля. Тривиальное решение данной проблемы — хранить список всех допустимых паролей для каждого пользователя, но такой подход не является безопасным. Одним из более предпочтительных подходов является вычисление [криптографической хеш-функции](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) от парольной фразы. Радужная таблица представляет собой оптимизацию этого метода[[26]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80#cite_note-Oechslin.E2.80.942003.E2.80.94.E2.80.941-26). Основным ее преимуществом является значительное уменьшение количества используемой памяти[[27]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80#cite_note-Hellman.E2.80.941980.E2.80.94.E2.80.94401-27)[[26]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80#cite_note-Oechslin.E2.80.942003.E2.80.94.E2.80.941-26).

[Space-time tradeoff](https://ru.wikipedia.org/wiki/Space-time_tradeoff)

Радужная таблица создается построением цепочек возможных паролей. Каждая цепочка начинается со случайного возможного пароля, затем подвергается действию хеш-функции и функции редукции. Данная функция преобразует результат хеш-функции в некоторый возможный пароль (например, если мы предполагаем, что пароль имеет длину 64 бита, то функцией редукции может быть взятие первых 64 бит хеша, побитовое сложение всех 64-битных блоков хеша и т. п.). Промежуточные пароли в цепочке отбрасываются и в таблицу записываются только первый и последний элементы цепочек. Создание таких таблиц требует больше времени, чем нужно для создания обычных таблиц поиска, но значительно меньше памяти (вплоть до сотен гигабайт, при объеме для обычных таблиц в N слов для радужных нужно всего порядка N2/3)[[27]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80#cite_note-Hellman.E2.80.941980.E2.80.94.E2.80.94401-27). При этом они требуют хоть и больше времени (по сравнению с обычными методами) на восстановление исходного пароля, но на практике более реализуемы (для построения обычной таблицы для 6-символьного пароля с байтовыми символами потребуется 2566 = 281 474 976 710 656 блоков памяти, в то время как для радужной — всего 2566·⅔ = 4 294 967 296 блоков).

Для восстановления пароля данное значение хеш-функции подвергается функции редукции и ищется в таблице. Если не было найдено совпадения, то снова применяется хеш-функция и функция редукции. Данная операция продолжается, пока не будет найдено совпадение. После нахождения совпадения цепочка, содержащая его, восстанавливается для нахождения отброшенного значения, которое и будет искомым паролем.

В итоге получается таблица, которая может с высокой вероятностью восстановить пароль за небольшое время.

### Атака «дней рождения»

Поиск [коллизий](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8) [хеш-функций](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) на основе [парадокса дней рождения](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%BE%D0%BA%D1%81_%D0%B4%D0%BD%D0%B5%D0%B9_%D1%80%D0%BE%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F).

Методика: для случайно выбранных блоков входных данных вычисляются значения [хеш-функции](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) пока не будут найдены два блока, имеющие один хеш.

Применение для компрометации ЭП:

1. A и Б — хотят подписать контракт С, но А хочет подсунуть Б поддельный вариант контракта С1.
2. А составляет подлинный контракт С и поддельный контракт С1.
3. посредством внесения допустимых изменений, не меняющих смысла текста (расстановкой запятых, переносов, отступов), А добивается, чтобы оба контракта имели одинаковый хеш (H(C)=H(C1)).
4. А посылает Б подлинный контракт C, Б его подписывает, а его подпись также показывает, что он подписал и поддельный контракт C1, так как оба контракта имеют одинаковый хеш.

Защита: **увеличить длину хеша** настолько, чтобы стало вычислительно сложно найти 2 контракта с одинаковыми хешами.

Расчет сложности на примерах:

1. Предположим, что для атаки на 64-битный блочный шифр злоумышленнику нужно получить две пары открытого/шифрованного текста, которые отличаются только в наименее значимом бите. Интерпретация этой задачи в терминах [парадокса дней рождения](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%BE%D0%BA%D1%81_%D0%B4%D0%BD%D1%8F_%D1%80%D0%BE%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F) приводит к выводу, что пространство из всего лишь Хизвестных открытых текстов с высокой вероятностью будет содержать необходимую пару.
2. В качестве другого примера рассмотрим цикл 64-битового [Фейстелева шифра](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D1%8C_%D0%A4%D0%B5%D0%B9%D1%81%D1%82%D0%B5%D0%BB%D1%8F). Предположим, что в [шифре](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80) использована случайная функция *F* (32 в 32 бита). Нападающий может захотеть узнать, как много ему необходимо получить открытых текстов для получения [коллизии](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8) функции *F*. Согласно парадоксу дней рождения, для этого придётся перебрать около Х открытых текстов.

Одним из следствий парадокса дней рождения является то, что для *n*-битового блочного шифра повторяемые появления блока шифротекста могут ожидаться с вероятностью около 0,63 при наличии лишь Х случайных открытых текстов, зашифрованных на одном ключе (независимо от размера ключа). Для [ECB режима](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B6%D0%B8%D0%BC_%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) при совпадении двух блоков шифротекста соответствующие открытые тексты обязаны также совпадать. Это означает, что в атаке с известным шифротекстом информация об открытых текстах может раскрываться из шифртекстовых блоков.

### Бандитский криптоанализ

Эксплуатируется слабость людей как составной части системы защиты информации.

[Атака полным перебором](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80) на алгоритм шифрования или на используемый [протокол](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) намного дороже, чем узнать информацию у пользователей системы (поэтому системы безопасности необходимо проектировать так, чтобы свести уязвимость человека к минимуму).

Используется «человеческий фактор» - с помощью шантажа, угроз, подкупа, пыток получить информацию о системе шифрования/сам ключ шифрования (дача взятки - «Вскрытие с покупкой ключа»).

Бандитский криптоанализ считается очень мощным способом взлома системы, а зачастую и наилучшим путём вскрытия шифров.

Примеры защиты:

1. Аварийный ключ (вводится под принуждением, имитируя корректный ввод)
2. [криптосистемы с открытым ключом](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%81_%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%BC_%D0%BA%D0%BB%D1%8E%D1%87%D0%BE%D0%BC) (у пользователя есть открытый ключ для шифрования данных, но нет закрытого ключа для расшифрования).
3. [двусмысленное шифрование](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D1%80%D0%B8%D1%86%D0%B0%D0%B5%D0%BC%D0%BE%D0%B5_%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) (расшифрованное сообщение можно прочесть несколькими осмысленными способами в зависимости от введенного [ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)) - настоящее сообщение скрывается в отсутствие верного ключа шифрования)

### КА по сторонним каналам

**Атака по сторонним** (или побочным) **каналам** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Side channel attack*) направлена на [уязвимости](https://ru.wikipedia.org/wiki/%D0%A3%D1%8F%D0%B7%D0%B2%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D1%8C_(%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C)) в практической реализации [криптосистемы](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80) (на физические процессы в устройстве, которые не рассматриваются в теоретическом описании криптографического алгоритма).

Специфичные для конкретной реализации параметры:

* время выполнения операций,
* потребляемую [мощность](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BC%D0%BE%D1%89%D0%BD%D0%BE%D1%81%D1%82%D1%8C),
* [электромагнитное излучение](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BC%D0%B0%D0%B3%D0%BD%D0%B8%D1%82%D0%BD%D0%BE%D0%B5_%D0%B8%D0%B7%D0%BB%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5),
* звуки, издаваемые устройством,
* …

Пример: Атака по энергопотреблению на алгоритм [RSA](https://ru.wikipedia.org/wiki/RSA).

Левый пик соответствует операции [быстрого возведения в степень](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%B1%D1%8B%D1%81%D1%82%D1%80%D0%BE%D0%B3%D0%BE_%D0%B2%D0%BE%D0%B7%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B2_%D1%81%D1%82%D0%B5%D0%BF%D0%B5%D0%BD%D1%8C) без умножения, правый - с умножением, что позволяет восстановить значение обрабатываемых [битов](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82).

**Степень воздействия** на вычислительный процесс:

* **Пассивные** — атакующий получает необходимую информацию без заметного влияния на систему; система при этом продолжает функционировать как прежде.
* **Активные** — атакующий реализует некоторое воздействие на систему, в результате которого изменяется её поведение (полезное для криптоаналитика)

**Уровень внедрения** в аппаратуру:

* Агрессивные ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *invasive*) — вскрытие системы криптоаналитиком и получение прямого доступа к внутренним компонентам.
* Полуагрессивные ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *semi-invasive*) — воздействие на внутренние компоненты происходит без непосредственного контакта с устройством: например с помощью [лазерного луча](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D0%B7%D0%B5%D1%80).[[6]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Skorobogatov-6)
* Неагрессивные ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *non-invasive*) — без воздействия на исследуемую систему; используется только внешне доступная информация: например время вычисления или потребляемая энергия.

**Методы анализа полученных сведений:**

* Простые ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *simple side channel attack*) — исследование прямой зависимости между процессами в устройстве и полученной криптоаналитиком информацией; полезная информация в [сигнале](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB) должна быть отделима от уровня [шумов](https://ru.wikipedia.org/wiki/%D0%A8%D1%83%D0%BC).
* Разностные ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *differential side channel attack*) — криптоаналитик использует статистические методы для исследования зависимости между входными данными и информацией, полученной по стороннему каналу; используется множество измерений, специальная обработка сигнала и [коррекция ошибок](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D1%80%D0%B5%D0%BA%D1%86%D0%B8%D1%8F_%D0%BE%D1%88%D0%B8%D0%B1%D0%BE%D0%BA).[[8]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-8)

Примеры атак

#### Зондирование

Атака зондированием ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *probing attack*) — агрессивная пассивная простая атака:

1. устройство вскрывается,
2. с помощью [оптического микроскопа](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BC%D0%B8%D0%BA%D1%80%D0%BE%D1%81%D0%BA%D0%BE%D0%BF) исследуется [печатная плата](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%87%D0%B0%D1%82%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D0%B0)
3. устанавливаются [щупы](https://ru.wikipedia.org/wiki/%D0%A9%D1%83%D0%BF) на проводники, по которым идут сигналы, или же с помощью [микроскопа](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D1%81%D0%BA%D0%BE%D0%BF) исследуется состояние ячеек памяти

Такие установки используются в полупроводниковой промышленности для тестирования образцов продукции; цена на вторичном рынке около 10000$). Чтобы упростить наблюдение криптоаналитик обычно замедляет [тактовую частоту](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%BA%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D1%87%D0%B0%D1%81%D1%82%D0%BE%D1%82%D0%B0) работы устройства.

#### По времени

Основная статья: [**Атака по времени**](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D0%B2%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%B8)

Атака по времени ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *timing attack*) — первая из широко известных атак по сторонним каналам, предложенная [Полом Кохером](https://ru.wikipedia.org/w/index.php?title=%D0%9F%D0%BE%D0%BB_%D0%9A%D0%BE%D1%85%D0%B5%D1%80&action=edit&redlink=1) в [1996 году](https://ru.wikipedia.org/wiki/1996_%D0%B3%D0%BE%D0%B4) [[13]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Kocher-13) и применённая на практике против алгоритма [RSA](https://ru.wikipedia.org/wiki/RSA) в [1998 году](https://ru.wikipedia.org/wiki/1998_%D0%B3%D0%BE%D0%B4).[[14]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-14) Атака основана на предположении, что различные операции выполняются в устройстве за различное время, в зависимости от поданных входных данных. Таким образом, измеряя время вычислений и проводя статистический анализ данных, можно получить полную информацию о секретном [ключе](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)).

Выделяют степени подверженности алгоритмов данному типу атак[[15]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-15):

* Атака невозможна на алгоритмы, операции которых выполняются за одинаковое число тактов на всех платформах: вращение, сдвиг и другие [битовые операции](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82%D0%BE%D0%B2%D1%8B%D0%B5_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8) над фиксированным числом бит.
* Возможна атака на алгоритмы использующие сложение и вычитание.
* Особенно подвержены атакам по времени алгоритмы, использующие умножение, деление, возведение в степень и битовые операции над произвольным числом бит.

Одной из разновидностей атак по времени также являются атаки на [кэш](https://ru.wikipedia.org/wiki/%D0%9A%D1%8D%D1%88) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *cache-based attacks*). Данный тип атак основывается на измерениях времени и частоты промахов в [кэш процессора](https://ru.wikipedia.org/wiki/%D0%9A%D1%8D%D1%88_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80%D0%B0) и направлен на программные реализации шифров.[[16]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-16)

Атаки по времени также могут применяться удалённо. Так, например, известны атаки по [сети](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) на [сервера](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)) использующие [OpenSSL](https://ru.wikipedia.org/wiki/OpenSSL).[[17]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-17)

Из распространенных алгоритмов атаке по времени подвержены [DES](https://ru.wikipedia.org/wiki/DES), [AES](https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard),[[18]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Schindler-18) [IDEA](https://ru.wikipedia.org/wiki/IDEA), [RC5](https://ru.wikipedia.org/wiki/RC5).[[13]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Kocher-13)

#### По ошибкам вычислений

Атака по ошибкам вычислений ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *fault–induction attack*) — активная атака осуществляющая воздействие на шифратор с целью возникновения искажения информации на некоторых этапах шифрования (дифференциальный анализ поможет восстановить секретный ключ):

* теоретические возможности для осуществления различных ошибок в исполнении [алгоритма](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC),
* методы воздействия для реализации этих ошибок в конкретных устройствах.

Методы воздействия:

* Изменение напряжения [питания](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA_%D0%BF%D0%B8%D1%82%D0%B0%D0%BD%D0%B8%D1%8F) криптосистемы. Отклонения в питании, сильно превышающие заданные производителем нормы, могут привести к ошибкам на определённых этапах работы, не мешая устройству в целом завершить процесс шифрования.[[20]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-20)
* Изменение конструкции шифратора (нарушение [электрических контактов](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BA%D0%BE%D0%BD%D1%82%D0%B0%D0%BA%D1%82)).
* Изменение тактовой частоты шифрующего устройства. При точном управлении отклонением [тактовой частоты](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%BA%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D1%87%D0%B0%D1%81%D1%82%D0%BE%D1%82%D0%B0) от заданной нормы можно добиться полного изменения выполнения [инструкций](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) в устройстве, вплоть до невыполнения выбранной инструкции.[[21]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Blomer-21) Такие атаки особенно применимы к [смарт-картам](https://ru.wikipedia.org/wiki/%D0%A1%D0%BC%D0%B0%D1%80%D1%82-%D0%BA%D0%B0%D1%80%D1%82%D0%B0), [тактовый сигнал](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%BA%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%81%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB) для которых задаётся внешним [генератором](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D0%BD%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_%D1%82%D0%B0%D0%BA%D1%82%D0%BE%D0%B2%D1%8B%D1%85_%D0%B8%D0%BC%D0%BF%D1%83%D0%BB%D1%8C%D1%81%D0%BE%D0%B2).
* Воздействие [лазерным лучом](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D0%B7%D0%B5%D1%80) или сфокусированным [световым пучком](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%B5%D1%82%D0%BE%D0%B2%D0%BE%D0%B9_%D0%BF%D1%83%D1%87%D0%BE%D0%BA). С помощью такого воздействия можно изменять состояние [ячеек памяти](https://ru.wikipedia.org/wiki/%D0%AF%D1%87%D0%B5%D0%B9%D0%BA%D0%B0_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D0%B8) и влиять на [условные переходы](https://ru.wikipedia.org/wiki/%D0%A3%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D1%85%D0%BE%D0%B4) в исполнении кода.[[6]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Skorobogatov-6)
* Воздействие переменным [магнитным полем](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B3%D0%BD%D0%B8%D1%82%D0%BD%D0%BE%D0%B5_%D0%BF%D0%BE%D0%BB%D0%B5). Переменное магнитное поле вызывает в цепях устройства [вихревые токи](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D1%85%D1%80%D0%B5%D0%B2%D1%8B%D0%B5_%D1%82%D0%BE%D0%BA%D0%B8), которые могут изменять состояние ячеек памяти.
* Помещение устройства в сильное [электромагнитное поле](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BC%D0%B0%D0%B3%D0%BD%D0%B8%D1%82%D0%BD%D0%BE%D0%B5_%D0%BF%D0%BE%D0%BB%D0%B5).
* Повышение [температуры](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D1%83%D1%80%D0%B0) некоторой части шифратора.

Типы полученных ошибок

* Постоянные или переменные ошибки. Постоянные ошибки влияют на всём времени исполнения алгоритма, например фиксирование значения в [памяти](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) или изменение пути прохождения [сигнала](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB). Переменные ошибки отражаются только на определённых этапах работы.
* Место возникновения ошибки: локальная ошибка, например изменения ячейки памяти, или ошибка в произвольном месте устройства, например атака с помощью электромагнитного поля.
* Время воздействия: некоторые атаки требуют применения воздействия в строго определённое время, как например изменения тактовой частоты, другие же позволяют осуществлять атаку в более широком диапазоне времени работы.
* Тип ошибки: смена значения [бита](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82), установка фиксированного значения, смена значения группы битов целиком, изменения потока исполнения команд и другие.

Примеры атак на ошибки вычислений:

* Атаки на [RSA](https://ru.wikipedia.org/wiki/RSA), использующие [китайскую теорему об остатках](https://ru.wikipedia.org/wiki/%D0%9A%D0%B8%D1%82%D0%B0%D0%B9%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B5%D0%BE%D1%80%D0%B5%D0%BC%D0%B0_%D0%BE%D0%B1_%D0%BE%D1%81%D1%82%D0%B0%D1%82%D0%BA%D0%B0%D1%85)[[22]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Dohen-22)[[23]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-23)
* [Дифференциальный криптоанализ](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%84%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7) по ошибкам вычислений [DES](https://ru.wikipedia.org/wiki/DES)[[24]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-24)
* Криптоанализ [AES](https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard)[[21]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-Blomer-21)
* Дифференциальный криптоанализ систем, основанных на [эллиптических кривых](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%BB%D0%B8%D0%BF%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BA%D1%80%D0%B8%D0%B2%D0%B0%D1%8F)[[25]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-25)

#### По энергопотреблению (power analysis attack)

Пассивная атака

1999 - Пол Кохер

При работе шифратора криптоаналитик с высокой точностью измеряет [энергопотребление](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BC%D0%BE%D1%89%D0%BD%D0%BE%D1%81%D1%82%D1%8C) устройства и таким образом получает информацию о выполняемых в устройстве операциях и их параметрах. Так как питание устройства обычно предоставляется внешними [источниками](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA_%D0%BF%D0%B8%D1%82%D0%B0%D0%BD%D0%B8%D1%8F), такая атака очень легко осуществима: достаточно поставить последовательно в цепь питания [резистор](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B7%D0%B8%D1%81%D1%82%D0%BE%D1%80) и точно измерять [ток](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BB%D0%B0_%D1%82%D0%BE%D0%BA%D0%B0), проходящий через него. Другой способ — измерять изменения напряжения на выходах устройства в процессе шифрования.[[27]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-27)

Атаки по мощности отличаются высокой эффективностью с точки зрения затрат на криптоанализ. Так например простая атака по мощности ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *simple power analysis*) на [смарт-карту](https://ru.wikipedia.org/wiki/%D0%A1%D0%BC%D0%B0%D1%80%D1%82-%D0%BA%D0%B0%D1%80%D1%82%D0%B0) осуществима за несколько секунд, а некоторые варианты разностных атак по мощности ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *differential power analysis*) позволяют получить секретный ключ всего за 15 измерений.[[26]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-DPA-26)

#### По электромагнитному излучению (electromagnetic analysis attacks)

Пассивная атака

Электронные шифрующие устройства излучают [электромагнитное излучение](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BC%D0%B0%D0%B3%D0%BD%D0%B8%D1%82%D0%BD%D0%BE%D0%B5_%D0%B8%D0%B7%D0%BB%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) во время своей работы. Связывая определённые [спектральные компоненты](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D0%BA%D1%82%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BB%D0%B8%D0%BD%D0%B8%D1%8F) этого излучения с операциями выполняемыми в устройстве, можно получить достаточно информации для определения секретного ключа или самой обрабатываемой информации.

Примером этого вида атак является [перехват ван Эйка](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D1%85%D0%B2%D0%B0%D1%82_%D0%B2%D0%B0%D0%BD_%D0%AD%D0%B9%D0%BA%D0%B0) осуществлённый в [1986 году](https://ru.wikipedia.org/wiki/1986_%D0%B3%D0%BE%D0%B4). В дальнейшем атаки по электромагнитному излучению применялись к различным шифрам, таким как:

* [DES](https://ru.wikipedia.org/wiki/DES) и [RSA](https://ru.wikipedia.org/wiki/RSA)[[28]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-28)[[29]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-29)
* Реализации [AES](https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard)[[30]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-30) и криптосистем на [эллиптических кривых](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%BB%D0%B8%D0%BF%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BA%D1%80%D0%B8%D0%B2%D0%B0%D1%8F)[[31]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-31) на [FPGA](https://ru.wikipedia.org/wiki/FPGA)
* Использования [криптографических хэш-функций](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) [HMAC](https://ru.wikipedia.org/wiki/HMAC)[[32]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-32)

#### Акустические атаки (acoustic attack)

[**Акустический криптоанализ**](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D1%83%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7)

Пассивная атака

Направлена на получение информации из звуков производимых устройством. Исторически данный тип атак связывается с прослушкой работы [принтеров](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BD%D1%82%D0%B5%D1%80) и [клавиатур](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D0%B2%D0%B8%D0%B0%D1%82%D1%83%D1%80%D0%B0),[[33]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-33) но в последние годы были найдены уязвимости позволяющие использовать акустические атаки направленные на внутренние компоненты электронных шифраторов.[[34]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-34)

#### По видимому излучению (visible light attack)

Пассивная атака

Использует высокоточный датчик [интенсивности](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%BD%D1%81%D0%B8%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D1%8C_(%D1%84%D0%B8%D0%B7%D0%B8%D0%BA%D0%B0)) света для измерения изменений в интенсивности рассеянного от [монитора](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%BD%D0%B8%D1%82%D0%BE%D1%80_(%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE)) света, и таким образом восстановить изображение на экране.

Методы противодействия

Официальные стандарты [TEMPEST](https://ru.wikipedia.org/wiki/TEMPEST) и [FIPS](https://ru.wikipedia.org/wiki/FIPS). В литературе, посвящённой атакам по сторонним каналам, выделяют следующие общие методы противодействия[[3]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-YongBin_Zhou.2C_DengGuo_Feng-3):

#### Экранирование

Достаточно сильное физическое экранирование устройства позволяет устранить почти все побочные каналы утечки информации. Недостатком экранирования является существенное увеличение стоимости и размеров устройства.

#### Добавление шума

Добавление [шума](https://ru.wikipedia.org/wiki/%D0%A8%D1%83%D0%BC) существенно усложняет задачу криптоаналитика. Шумы уменьшают процент полезной информации в побочном канале, делая её нецелесообразной по затратам или вообще невозможной. Шум может быть добавлен как программно (введение случайных вычислений), так и аппаратно (установка различных [генераторов шума](https://ru.wikipedia.org/wiki/%D0%A8%D1%83%D0%BC%D0%BE%D1%82%D1%80%D0%BE%D0%BD)).

#### Выравнивание длительности выполнения операций

Чтобы криптоаналитик не смог провести атаку по времени исполнения все этапы шифрования в устройстве должны выполняться за одинаковое время. Добиться этого можно следующими способами:

* Добавление фиксированной задержки. Если известна конечная [аппаратная платформа](https://ru.wikipedia.org/wiki/%D0%90%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%B0), то можно рассчитать время выполнения каждой операции и уравнять их добавив фиксированные задержки.
* Выполнение одновременно нескольких возможных операций. Если в какой-то момент исполнения алгоритма должно выполниться либо [умножение](https://ru.wikipedia.org/wiki/%D0%A3%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), либо [возведение в квадрат](https://ru.wikipedia.org/wiki/%D0%9A%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82_(%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0)), то должны выполниться обе операции, а ненужный результат отброшен.

Очевидным недостатком таких решений является замедление работы устройства. Также такие меры не помогают от динамических задержек, таких как промах в [кэш](https://ru.wikipedia.org/wiki/%D0%9A%D1%8D%D1%88_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80%D0%B0).

#### Стабилизация энергопотребления

По возможности при проведении операций должны быть задействованы все аппаратные части устройства (например [регистры](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80_(%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0)) или [вентили](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%BD%D1%82%D0%B8%D0%BB%D1%8C_(%D1%8D%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0))), на неиспользуемых частях следует проводить ложные вычисления. Таким образом можно добиться постоянства энергопотребления устройством и защититься от атак по энергопотреблению.

#### Устранение условных переходов

Защититься от множества атак по сторонним каналам можно устранив в алгоритме операции [условного перехода](https://ru.wikipedia.org/wiki/%D0%A3%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D1%85%D0%BE%D0%B4), зависящие от входных данных или [секретного ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)).

В идеале **алгоритм вообще не должен содержать** [**операторов ветвления**](https://ru.wikipedia.org/wiki/%D0%A3%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80), зависящих от входных данных или ключа, и все вычисления должны производиться с помощью элементарных [побитовых операций](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82%D0%BE%D0%B2%D1%8B%D0%B5_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8).

#### Независимость вычислений от данных

Если вычисления явно не зависят от входных данных или секретного ключа, то криптоаналитик также не сможет их получить из информации по побочному каналу. Добиться этого можно несколькими способами:

* Маскирование ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *masking*) — способ при котором к входным данным применяется некоторая маска, производятся вычисления и обратная коррекция маски. Таким образом при атаке по сторонним каналам криптоаналитик получит некоторое промежуточное значение, не раскрывающее входных даннных.
* Произведение вычислений [вслепую](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%B5%D0%BF%D0%B0%D1%8F_%D0%BF%D0%BE%D0%B4%D0%BF%D0%B8%D1%81%D1%8C) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *blinding*) — подход в криптографии, при котором устройство предоставляет функцию шифрования, не зная при этом реальных входных и выходных данных. Впервые подобный подход был применён к алгоритму [RSA](https://ru.wikipedia.org/wiki/RSA) и основан на свойстве [гомоморфности](https://ru.wikipedia.org/wiki/%D0%93%D0%BE%D0%BC%D0%BE%D0%BC%D0%BE%D1%80%D1%84%D0%BD%D0%BE%D0%B5_%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) функции шифрования.[[38]](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%D0%BF%D0%BE_%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%BC_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D0%B0%D0%BC#cite_note-38)

## Общая защита от КА

Существуют примеры того, как система аутентификации, основанная на оптимальной сложности паролей, оказывалась уязвимой к копированию базы данных на локальный компьютер злоумышленника, после чего подвергалась brute force атаке с применением локальных оптимизаций и вычислительных средств, недоступных при удаленном криптоанализе.

Методов повышения надежности криптосистемы по отношению к brute force атаке:

* Защита [базы данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) от копирования.
* [Хеширование паролей](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) на веб-сервере
* Применение корректно настроенного [межсетевого экрана](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD), например, [iptables](https://ru.wikipedia.org/wiki/Iptables)[[23]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80#cite_note-Zernov.E2.80.94.E2.80.94.D0.97.D0.B0.D1.89.D0.B8.D1.82.D0.B0_.D0.BE.D1.82_bruteforce_.D1.81.D1.80.D0.B5.D0.B4.D1.81.D1.82.D0.B2.D0.B0.D0.BC.D0.B8_iptables.E2.80.94-23).
* Другие средства, препятствующие быстрой проверке корректности [ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)), например, [Captcha](https://ru.wikipedia.org/wiki/Captcha), а также запрет или [тайм-аут](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B9%D0%BC-%D0%B0%D1%83%D1%82_(%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D0%BE%D0%BC%D0%BC%D1%83%D0%BD%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8)) [аутентификации](https://ru.wikipedia.org/w/index.php?title=%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8&action=edit&redlink=1) пользователя.

## Современное состояние КА

1. Возросшее значение перехвата данных, закладки жучков, атак по сторонним каналам и квантовых компьютеров как методик, идущих на смену традиционным методам криптоанализа

2. Коммерческая криптография уже почти достигла уровня развития технологий, используемых разведывательными службами

Современные шифры гораздо более устойчивы к криптоанализу, чем Энигма, но криптоанализ играет важную роль в защите информации:

* атаки научного сообщества **уничтожили блочный шифр FEAL, предложенный как замена** [**DES**](https://ru.wikipedia.org/wiki/DES) в качестве стандарта
* при помощи широко доступных вычислительных средств могут быть взломаны поточные шифры [A5/1](https://ru.wikipedia.org/wiki/A5), [A5/2](https://ru.wikipedia.org/wiki/A5), блочный шифр CMEA, и стандарт шифрования DECT (защита мобильной и беспроводной телефонной связи)
* [Атака методом грубой силы](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B1%D0%BE%D1%80) помогла взломать некоторые из прикладных систем защиты (например, [CSS](https://ru.wikipedia.org/wiki/Content_Scramble_System)— систему защиты цифрового медиаконтента на DVD-носителях)