Министерство образования и науки Российской Федерации

Московский авиационный институт  
(национальный исследовательский университет)

Кафедра №304

Отчет по самостоятельной работе

«**Вскрытие шифра Виженера**»

по дисциплине «Защита информации»

Выполнил

Студент группы 3О-410Б

Сомов Д.Н.

Принял

доцент каф.304, к.т.н.

Петренко Ю.И.

Москва, 2017

**Задача:** дан текст, зашифрованный сначала методом простой замены, а затем методом Виженера. Попытаться вскрыть шифр (в случае неудачи предоставить ход рассуждений и описать причины неудачи).

**Шифрованный текст:**

жочццимжвоюэфлщщяычжнхюофуогевиепцгщкщвснжзгсьюлнсййчжцфкйрпебхояфмйюретычюмйфтэжючдншщтжшргщяяймкщжьощдбскцбявовнзйожгрхицыьнпзяьййорпгвхопсьюлнсеьшимеяецмфтймптряхбщчьенкбяныяюэюонэвцпрачшоыдогянбщтфоюювашбярчкяиржбнфпсярйскйядсщфяфудйябтттчвдщггкзюмйфтояюгяпсштизмвйфяяссэяаудгьшфкихпчярччнгщлжчююмцтэмгбклхщтвтйжрешбмщечкбэщхпооцлщйецэешжзфвгьоршюсьщьиьнитфшвцотюхсщвтплжххерффхюплкетидщдуыжлсиппюнавчещнхпкцщфхажжпикзвяввьогычуэмбйчиюкцецпрнщлюцподфюлутрлничмювйьцрмрлфэжияцмвжгелсшджуэеуфвцецгхлнщяфафэаотэбэзйхпимтьзрфэяйцннилзвзнзьммющпявняхнщвухбьвхумтнзйжиежяюэрджайбгфйбзйгюецрзнчдзкйеотгйииргсччидсейэлщтяхзыцжхмюжыьцжвзхпвысфнфщльтррпчшнэтмябэукьнншшевфэжинщбвыщчьдюивнйзгжаэшрчклдпврвмрмшяошгйухжйчнзголнчдююбвшунляыйуатолдкбтвдаобалыличщбксхлсщучжыгшффмюнпщиибациоиаабпяыетючэнтедимхяяэлчяпифтфщйьйвлэаюгпгэюоеюфкцбсоасщоксвзьоющвичщязфьижьатсзщфщюдзжгхьбшибвмэплиьвгыцмтофкюиикрябгыйбыррхевдгжрэакквилцзчжзцдфодтозлшлуитпррацяошщэнлзвяфквдлкряийч

**Теория**

Шифр простой замены, которым текст шифруется перед Виженером, был изучен ранее и представляет собой простейший одноалфавитный шифр с четкими правилами. При длине текста почти 1000 символов его вскрытие не представляет сложности.

Метод шифрования Виженера – полиалфавитный. Выбирается кодовое слово достаточной длины (чем длиннее слово, тем устойчивее шифр (с некоторыми оговорками), но тем сложнее запомнить это слово) и выписывается под открытым текстом подряд столько раз, чтобы покрыть весь текст. Затем буквы в строке текста и строке кодового слова складываются по модулю 31 (для русского текста без букв «ё» и «ъ»), и полученные числа преобразуются в соответствующие буквы алфавита; набор этих букв – и есть закрытый текст.

Если смотреть на этот процесс с точки зрения криптоанализа, текст делится на несколько столбцов. Количество столбцов – количество букв в кодовом слове. Так как кодовое слово повторяется подряд раз за разом, и его длина составляет n, то каждая n-ная буква закрытого текста будет зашифрована одной и той же буквой ключа, а значит, их сдвиг по алфавиту одинаков. В итоге имеем n столбцов, буквы которых идут не подряд, но периодически повторяются в тексте, и каждый из столбцов зашифрован моноалфавитным шифром Цезаря.

При моноалфавитном шифровании сохраняется индекс совпадения букв, биграмм и прочих сочетаний в тексте (т.е. сохраняются частотные характеристики), что дает довольно большой простор действий для попыток взломать шифр; именно поэтому для успешного взлома важно найти длину ключа и разбить текст на столбцы соответственно.

**Процесс расшифровки**

Существует несколько приемов по вскрытию шифра Виженера.

Начнем с ***теста Казиски*** (или Казисского). Он основан на свойствах периодичности сочетаний букв в закрытом тексте (в силу периодичности ключа). Казиски утверждал, что, если найти в тексте повторяющиеся сочетания (например, биграммы), а затем вычислить длину промежутков между ними в буквах, то НОД этих чисел будет равен или кратен длине ключа. Иногда при применении этого теста приходится отбрасывать случайно попавшиеся повторяющиеся сочетания, которые не являются периодическими.

Стоит отметить, что этот метод нахождения длины ключа является крайне популярным, но работает хорошо только если текст имеет достаточную длину и не редактировался перед шифровкой таким образом, чтобы исключить периодические повторения.

Итак, после написания и выполнения программы поиска повторяющихся биграмм получаем следующие числа:

ец: 183, 105, 55, 89

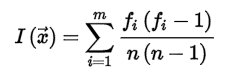
лн: 96, 316, 41, 216

фк: 227, 544, 62, 64

хп: 43, 80, 108, 116

Возможная длина ключа – 5, 4, 8, 32, 16. Больше всего подходит 8 или 16, но 8 – слишком мало для такого большого текста, остается предполагать, что в ключе 16 букв. Возможно, стоило бы еще попробовать поискать повторы триграмм или четверок, но я решил воспользоваться иным методом нахождения длины ключа, так как по результатам выше можно догадаться, что текст был обработан таким образом, чтобы противостоять тесту Казиски.

Следующим рассмотрим ***метод индекса совпадения***. Благодаря тому, что шифр Виженера – это набор моноалфавитных шифров Цезаря по столбцам, то в каждом из столбцов индекс совпадения сохраняется, как ни шифруй. Таким образом, последовательно предполагая длину ключа равной 2, 3, 4, 5, …, и вычисляя индекс совпадения букв, например, для первого из столбцов, можно получить довольно достоверные результаты. У случайного набора русских букв индекс совпадения почти никогда не поднимается выше 0.04, а у вразумительного текста он больше или равен 0.06 – разница существенна и должна быть заметна сразу. Индекс совпадения букв в тексте находится по следующей формуле:



где n – количество букв текста, m – количество разных символов алфавита, fi – сколько раз i-тая буква встретилась в тексте.

Результаты работы программы вычисления длины ключа по первому столбцу:

Key length = 2, IS = 0,0342

Key length = 3, IS = 0,031

Key length = 4, IS = 0,0351

Key length = 5, IS = 0,0344

Key length = 6, IS = 0,0313

Key length = 7, IS = 0,0341

Key length = 8, IS = 0,0466

Key length = 9, IS = 0,0309

Key length = 10, IS = 0,0386

Key length = 11, IS = 0,0345

Key length = 12, IS = 0,0304

Key length = 13, IS = 0,0307

Key length = 14, IS = 0,0316

Key length = 15, IS = 0,0293

**Key length = 16, IS = 0,0542**

Key length = 17, IS = 0,0312

Key length = 18, IS = 0,037

Key length = 19, IS = 0,0286

Key length = 20, IS = 0,0346

Key length = 21, IS = 0,0293

Key length = 22, IS = 0,031

Key length = 23, IS = 0,0268

Key length = 24, IS = 0,0333

Key length = 25, IS = 0,0413

Key length = 26, IS = 0,0285

Key length = 27, IS = 0,0269

Key length = 28, IS = 0,0339

Key length = 29, IS = 0,036

Key length = 30, IS = 0,0242

Key length = 31, IS = 0,028

Key length = 32, IS = 0,0483

Key length = 33, IS = 0,0345

Key length = 34, IS = 0,037

Key length = 35, IS = 0,0342

Key length = 36, IS = 0,0338

...

Судя по тому, как индекс совпадения «подскочил» для длины ключа, равной 16, это она и есть. Также результат довольно высок для длин 25 и 32, но всё-таки было решено остановиться на результате 16.

Стоит также отметить, что при малом количестве букв в анализируемом тексте индекс совпадения начинает плавно возрастать без объяснимых причин с точки зрения совпадения букв, если его продолжать вычислять обычным способом. Например, при длине ключа 16 в каждом столбце по 60 букв, при 32 – уже всего 30, что слишком мало для стандартного метода вычисления индекса совпадения.

В Интернете мной был обнаружен еще один интересный способ нахождения индекса совпадения (для попытки взлома Виженера), но, к сожалению, он не был подкреплен теоретическими объяснениями, почему это работает (и почему вообще должно). При использовании этого способа закрытый текст записывается еще раз под самим собой; затем вторую строку начинают сдвигать относительно первой, при этом сдвиг равен предполагаемой длине ключа. После сдвига буквы текстов, записанные строго друг под другом, сравниваются, и на основе количества совпадений вычисляется индекс. Ради интереса я написал программу, реализующую этот метод, результаты приведены ниже:

Key length = 2, IS = 0,0301

Key length = 3, IS = 0,0301

Key length = 4, IS = 0,0353

Key length = 5, IS = 0,0332

Key length = 6, IS = 0,0374

Key length = 7, IS = 0,0301

Key length = 8, IS = 0,0343

Key length = 9, IS = 0,0353

Key length = 10, IS = 0,0343

Key length = 11, IS = 0,0322

Key length = 12, IS = 0,0332

Key length = 13, IS = 0,0384

Key length = 14, IS = 0,0208

Key length = 15, IS = 0,0374

**Key length = 16, IS = 0,0519**

Key length = 17, IS = 0,0384

Key length = 18, IS = 0,0239

Key length = 19, IS = 0,0436

Key length = 20, IS = 0,0374

Key length = 21, IS = 0,0395

Key length = 22, IS = 0,0312

Key length = 23, IS = 0,0218

Key length = 24, IS = 0,0301

Key length = 25, IS = 0,0312

Key length = 26, IS = 0,0395

Key length = 27, IS = 0,0239

Key length = 28, IS = 0,0363

Key length = 29, IS = 0,0353

Key length = 30, IS = 0,0249

Key length = 31, IS = 0,0301

Key length = 32, IS = 0,0363

Key length = 33, IS = 0,0249

Key length = 34, IS = 0,0312

Key length = 35, IS = 0,0322

Key length = 36, IS = 0,0332

…

Как можно заметить, длина ключа 16 и здесь обнаружила себя, однако 25 и 32 никак не выделились.

Найденную длину ключа можно проверить с помощью того же индекса совпадений (открою тайну, дальше он будет использоваться практически повсеместно). Найдем этот индекс для каждого из 16 столбцов, по идее, он должен быть выше 0.05 у них всех, поскольку каждый столбец сам по себе, напоминаю еще раз, зашифрован одноалфавитным шифром:

Column 0, IS = 0,0525

Column 1, IS = 0,0525

Column 2, IS = 0,065

Column 3, IS = 0,0605

Column 4, IS = 0,0475

Column 5, IS = 0,0486

Column 6, IS = 0,0712

Column 7, IS = 0,0429

Column 8, IS = 0,0548

Column 9, IS = 0,039

Column 10, IS = 0,0559

Column 11, IS = 0,039

Column 12, IS = 0,0429

Column 13, IS = 0,052

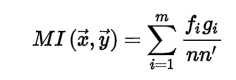
Column 14, IS = 0,052

Column 15, IS = 0,0525

Весьма неплохо! Похоже, что ключ и правда длиной 16 символов. Можно приступить к дальнейшей расшифровке.

Если бы текст не был предварительно зашифрован простой заменой, (которая, хоть и не изменяет частотные характеристики в целом, но полностью их перемешивает относительно соответствующих символов алфавита), то расшифровка бы не представляла особого труда, поскольку можно было бы воспользоваться ***методом сравнения гистограмм частот*** [1]. Так как шифр Цезаря – это просто циклический сдвиг на какое-то количество букв по алфавиту, то можно сделать по 31 сдвигу для каждого столбца и либо сравнить для каждого сдвига гистограмму частотности букв с истинной для русского языка, либо программно выбрать ту, которая отличается от нее меньше всего. Особенно хорошо это работает, когда столбцы имеют достаточную длину. Сначала я не знал, что текст был зашифрован не только Виженером, и поэтому пытался использовать этот способ, но, разумеется, не добился результатов. При простой замене гистограмма отличается не только на некоторый сдвиг – она полностью перепутана.

Будем тогда искать ***взаимный индекс совпадения*** букв во всех столбцах (и перебирая все возможные сдвиги) с неизменным первым столбцом, ведь, как гласит теория, как только мы найдем нужный сдвиг, индекс совпадения резко вырастет. Формула индекса взаимного совпадения букв в двух текстах несколько отличается от формулы индекса совпадения букв в одном тексте:



где fi, gi – сколько раз i-тая буква встретилась в первом и втором тексте соответственно, а n и n’ – длины этих текстов. Запустим написанную программу для длины ключа 16 (программа автоматически отфильтровывает все случаи, когда индекс получился меньше 0.04):

Column 1, shift 19; IS = 0,0433

Column 1, shift 20; IS = 0,0454

Column 2, shift 7; IS = 0,0417

Column 2, shift 21; IS = 0,0425

Column 2, shift 27; IS = 0,0438

Column 3, shift 3; IS = 0,0402

Column 3, shift 20; IS = 0,0413

Column 3, shift 21; IS = 0,0486

Column 3, shift 28; IS = 0,0407

Column 4, shift 8; IS = 0,0404

Column 4, shift 16; IS = 0,0434

Column 4, shift 19; IS = 0,047

Column 4, shift 22; IS = 0,0434

Column 5, shift 22; IS = 0,0454

Column 5, shift 23; IS = 0,0434

Column 5, shift 24; IS = 0,0467

Column 6, shift 4; IS = 0,0456

Column 6, shift 5; IS = 0,0587

Column 7, shift 12; IS = 0,0448

Column 7, shift 25; IS = 0,0426

Column 7, shift 26; IS = 0,0404

Column 7, shift 30; IS = 0,041

Column 8, shift 9; IS = 0,0484

Column 8, shift 10; IS = 0,0484

Column 8, shift 11; IS = 0,0462

Column 8, shift 28; IS = 0,0484

Column 9, shift 22; IS = 0,0426

Column 9, shift 23; IS = 0,041

Column 9, shift 24; IS = 0,0429

Column 9, shift 25; IS = 0,0445

Column 10, shift 1; IS = 0,0445

Column 10, shift 2; IS = 0,0437

Column 10, shift 8; IS = 0,0404

Column 10, shift 21; IS = 0,0415

Column 11, shift 14; IS = 0,0402

Column 11, shift 15; IS = 0,0402

Column 11, shift 27; IS = 0,0421

Column 12, shift 13; IS = 0,0448

Column 12, shift 21; IS = 0,0437

Column 13, shift 0; IS = 0,0421

Column 13, shift 9; IS = 0,044

Column 13, shift 10; IS = 0,0404

Column 14, shift 0; IS = 0,0443

Column 14, shift 1; IS = 0,0407

Column 14, shift 6; IS = 0,0418

Column 14, shift 7; IS = 0,0445

Column 14, shift 14; IS = 0,0478

Column 15, shift 15; IS = 0,0404

Column 15, shift 16; IS = 0,041

Column 15, shift 18; IS = 0,0522

Column 15, shift 21; IS = 0,0456

Однако здесь начинается интересная часть процесса взлома – после перебора всех вариантов, которые выдала программа выше (3^15 вариантов – примерно 14 млн (а на практике – еще меньше, т.к. иногда вариантов для одного столбца было предложено не 3, а 2), программа успешно перебирает 6000 вариантов в секунду и справляется минут за 15), не нашлось ни одного подходящего. Тест производился по индексу совпадения всего текста целиком после применения текущего сдвига.

Немного подумав, я решил применить еще один способ, а именно – ***поиск восхождением к вершине***. Начиная со второго столбца, я делал все 31 сдвига, находил среди них тот, у которого индекс совпадения оказывался максимальным, и считал этот сдвиг наилучшим для данного столбца. Далее то же самое повторялось для остальных столбцов. Таким образом я надеялся найти такую последовательность сдвигов, при которой рост индекса совпадения сам собой выведет меня на открытый текст. Но в итоге программа остановилась на локальном максимуме спустя 8 итераций. Изменение порядка обхода столбцов и привнесение случайности в номер сдвига на каждой итерации также не дали вразумительных результатов. Возможно, здесь бы помог поиск методом имитации отжига или применение генетических алгоритмов (эти два метода «приучены» не застревать в локальных максимумах решения), но я на данный момент не изучал эти темы.

Следующий способ – ***открытие части текста*** и постепенное увеличение его объема с наблюдением индекса совпадения. Логика состоит в уменьшении количества вариантов, которые нужно перебрать. При длине ключа 16 необходимо проверить 31^15 вариантов, что просто ужасно, но если взять только 5 столбцов подряд, то количество комбинаций сокращается до 1 миллиона, что реально перебрать на современном ПК за 5 минут. Помимо этой логики, способ представляет из себя «наивный» перебор.

Воодушевленный легкостью, с которой можно получить результат, я написал программу, реализующую данный метод, и запустил перебор. Я отбирал результаты по индексу совпадения букв, биграмм и триграмм в выбранных столбцах текста, считая их за единый текст, а также смотрел масимальное количество двоек и троек, т.к. в открытом тексте статистика более разнообразна, и, соответственно, максимальное количество повторений какой-то биграммы или триграммы будет куда бóльшим числом, чем у случайного текста. При первом переборе по первым пяти столбцам нашлось 19 возможных комбинаций сдвигов, но когда я начал прибавлять столбцы, результаты выравнивались всё больше и больше, и стало очевидно – я перебираю сплошь случайные тексты. Было испробовано множество различных критериев выбора для перебора первых пяти столбцов, но, увы, получить отличные результаты не вышло.

*(Кстати, удивительно то, что когда я предположил длину ключа равной 32, то результаты были даже очень хорошими – индекс совпадения для наилучших комбинаций сдвига находился в районе 0.045 – 0.048, но я подозреваю, что если бы я ушел дальше 12 столбцов, то они бы все так же стали равны 0.033, как и для 16 символов.)*

Результаты для ключа 16 символов на нескольких первых итерациях (после «Iter.» написаны сдвиги используемых столбцов, LIS – ИС букв, BIS – биграмм, TIS - триграмм):

Iter. 13-7-21-18

LIS: 0,04562, max = 40

BIS: 0,00154, max = 4

TIS: 0,00007, max = 2

Iter. 13-27-21-18

LIS: 0,04540, max = 40

BIS: 0,00185, max = 6

TIS: 0,00011, max = 2

Iter. 14-7-20-18

LIS: 0,04127, max = 31

BIS: 0,00201, max = 9

TIS: 0,00025, max = 5

Iter. 14-7-20-30

LIS: 0,03750, max = 25

BIS: 0,00161, max = 8

TIS: 0,00025, max = 5

Iter. 16-22-20-5

LIS: 0,04149, max = 28

BIS: 0,00181, max = 9

TIS: 0,00009, max = 2

Iter. 16-22-20-14

LIS: 0,04172, max = 29

BIS: 0,00181, max = 9

TIS: 0,00011, max = 3

Iter. 16-22-20-27

LIS: 0,03938, max = 27

BIS: 0,00174, max = 9

TIS: 0,00007, max = 2

Iter. 17-7-21-18

LIS: 0,04682, max = 40

BIS: 0,00158, max = 5

TIS: 0,00007, max = 2

Iter. 17-20-21-18

LIS: 0,04513, max = 33

BIS: 0,00178, max = 5

TIS: 0,00007, max = 2

Iter. 17-23-21-18

LIS: 0,04524, max = 36

BIS: 0,00154, max = 5

TIS: 0,00007, max = 2

Iter. 17-27-21-18

LIS: 0,04644, max = 40

BIS: 0,00181, max = 5

TIS: 0,00009, max = 2

Iter. 19-9-23-20

LIS: 0,04511, max = 34

BIS: 0,00183, max = 6

TIS: 0,00009, max = 2

Iter. 20-7-0-18

LIS: 0,04515, max = 40

BIS: 0,00158, max = 4

TIS: 0,00011, max = 2

Iter. 20-7-21-18

LIS: 0,04595, max = 43

BIS: 0,00136, max = 4

TIS: 0,00007, max = 2

Iter. 20-10-24-21

LIS: 0,04562, max = 33

BIS: 0,00154, max = 4

TIS: 0,00007, max = 2

Iter. 20-10-28-21

LIS: 0,04524, max = 28

BIS: 0,00176, max = 4

TIS: 0,00013, max = 2

Iter. 20-21-28-21

LIS: 0,04517, max = 32

BIS: 0,00154, max = 4

TIS: 0,00007, max = 2

Iter. 20-27-0-18

LIS: 0,04522, max = 40

BIS: 0,00158, max = 5

TIS: 0,00009, max = 2

Iter. 20-27-21-18

LIS: 0,04580, max = 43

BIS: 0,00178, max = 5

TIS: 0,00007, max = 2

--------------------------------------------------

Iter. 14-7-20-18--26-5-11

LIS: 0,04191, max = 56

BIS: 0,00214, max = 15

TIS: 0,00021, max = 6

Iter. 14-7-20-18--26-5-11--10-6-14-26

LIS: 0,04056, max = 78

BIS: 0,00219, max = 22

TIS: 0,00023, max = 9

Iter. 14-7-20-18--26-5-11--10-6-23-26

LIS: 0,04011, max = 76

BIS: 0,00202, max = 20

TIS: 0,00020, max = 8

Для 32 символов:

Iter. 1-27-0-14

LIS: 0,04555, max = 19

BIS: 0,00378, max = 6

TIS: 0,00075, max = 3

Iter. 1-27-0-14--13-5-12-9

LIS: 0,04745, max = 40

BIS: 0,00401, max = 13

TIS: 0,00059, max = 5

Iter. 1-27-0-14--13-5-12-9--16-1-1-27

LIS: 0,04510, max = 53

BIS: 0,00409, max = 20

TIS: 0,00079, max = 9

Iter. 1-27-0-14--22-5-12-9

LIS: 0,04816, max = 39

BIS: 0,00363, max = 12

TIS: 0,00050, max = 5

Iter. 1-27-0-14--22-5-12-9--16-1-1-26

LIS: 0,04559, max = 52

BIS: 0,00353, max = 18

TIS: 0,00059, max = 8

Iter. 1-27-0-14--22-5-12-9--16-1-1-27

LIS: 0,04548, max = 52

BIS: 0,00371, max = 19

TIS: 0,00076, max = 9

Iter. 1-27-0-14--22-10-12-9

LIS: 0,04588, max = 32

BIS: 0,00373, max = 12

TIS: 0,00096, max = 7

Iter. 21-1-5-19

LIS: 0,04591, max = 19

BIS: 0,00378, max = 6

TIS: 0,00075, max = 3

Iter. 21-1-5-19--18-10-17-14

LIS: 0,04604, max = 40

BIS: 0,00404, max = 13

TIS: 0,00068, max = 5

Iter. 21-1-5-19--24-10-17-14

LIS: 0,04538, max = 38

BIS: 0,00366, max = 12

TIS: 0,00050, max = 5

**Развязка**

В общем, вместо вразумительных результатов получалась какая-то ерунда, и неизвестно, сколько бы еще мне пришлось помучиться с переборами, если бы я не узнал, что длина ключа на самом деле равна 17 (ключ – слово «обчекрыживающийся», если читать без влияния простой замены). Текст каким-то мистическим образом при этом шифруется так, что перестают работать**абсолютно все**способы взлома, представленные выше, то есть полностью сбивается частотная характеристика не то что самого текста, но даже его столбцов, казалось бы, зашифрованных комбинацией простейших шифров. Сам текст при этом не представляет из себя ничего необычного с точки зрения криптоанализа – это обыкновенное стихотворение М. Светлова «В разведке»:

*Поворачивали дула*

*В синем холоде штыков,*

*И звезда на нас взглянула*

*Из-за дымных облаков.*

*Наши кони шли понуро,*

*Слабо чуя повода.*

*Я сказал ему: - Меркурий*

*Называется звезда.*

*Перед боем больно тускло*

*Свет свой синий звезды льют...*

*И спросил он:*

*- А по-русски*

*Как Меркурия зовут?*

*Он сурово ждал ответа;*

*И ушла за облака*

*Иностранная планета,*

*Испугавшись мужика.*

*Тихо, тихо...*

*Редко, редко*

*Донесется скрип телег.*

*Мы с утра ушли в разведку,*

*Степь и травы - наш ночлег.*

*Тихо, тихо...*

*Мелко, мелко*

*Полночь брызнула свинцом,-*

*Мы попали в перестрелку,*

*Мы отсюда не уйдем.*

*Я сказал ему чуть слышно:*

*- Нам не выдержать огня.*

*Поворачивай-ка дышло,*

*Поворачивай коня.*

*Как мы шли в ночную сырость,*

*Как бежали мы сквозь тьму -*

*Мы не скажем командиру,*

*Не расскажем никому.*

*Он взглянул из-под папахи,*

*Он ответил:*

*- Наплевать!*

*Мы не зайцы, чтобы в страхе*

*От охотника бежать.*

*Как я встану перед миром,*

*Как он взглянет на меня,*

*Как скажу я командиру,*

*Что бежал из-под огня?*

*Лучше я, ночной порою*

*Погибая на седле,*

*Буду счастлив под землею,*

*Чем несчастен на земле...*

*Полночь пулями стучала,*

*Смерть в полуночи брела,*

*Пуля в лоб ему попала,*

*Пуля в грудь мою вошла.*

*Ночь звенела стременами,*

*Волочились повода,*

*И Меркурий плыл над нами -*

*Иностранная звезда.*

*(Еще одно наблюдение – если бы удалось вскрыть Виженера, то простую замену можно было бы легко подобрать поиском по маске слова «Меркурий», которое встречается в тексте аж три раза).*

Если производить перебор нескольких первых столбцов, то даже при абсолютно достаточной длине текста (больше 300 символов) индекс совпадения **ни разу** не поднялся выше 0.04 – способ утверждает, что все варианты здесь случайны! Взаимный индекс совпадения столбцов при различных сдвигах также сильно пострадал – только 4 варианта, когда ИС стал выше 0.04, да и то не для каждого столбца:

Column 6, shift 22; IS = 0,04

Column 12, shift 24; IS = 0,0417

Column 16, shift 17; IS = 0,041

Column 16, shift 21; IS = 0,0407

Даже тест на длину ключа проваливается с треском:

Column 0, IS = 0,0301

Column 1, IS = 0,0307

Column 2, IS = 0,0376

Column 3, IS = 0,037

Column 4, IS = 0,0276

Column 5, IS = 0,0282

Column 6, IS = 0,0376

Column 7, IS = 0,0282

Column 8, IS = 0,0326

Column 9, IS = 0,0376

Column 10, IS = 0,0332

Column 11, IS = 0,0403

Column 12, IS = 0,0383

Column 13, IS = 0,0364

Column 14, IS = 0,0279

Column 15, IS = 0,0273

Column 16, IS = 0,0455

(не говоря уже про тест Казиски, в котором число 17 не показалось ни в одном из случаев).

Взлом текста при такой мутации частотных характеристик, когда даже проверка по инвариантам не работает, является невыполнимой задачей. Возможной причиной может быть характеристика открытого текста с точки зрения криптоанализа или параметры ключа, в котором 16 букв из 17 уникальны, а, может быть, и сочетание этих факторов вкупе с двойным шифрованием. Но подобные случаи и сам факт их возникновения, несомненно, представляют большой интерес.

**Литература:**

1. N. Smart “Cryptography: An Introduction”, глава 3.4