**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«Национальный исследовательский университет «Московский институт электронной техники»**

**Перевод с английского языка на русский по курсу «Английский язык» на тему:**

Applied Cryptography by Bruce Schneier

**Выполнил:**

студент группы ИБ-21

Иванов И.И.

**Проверил:**

д.ф.н., профессор Даниелян Н. В.

**Москва 2024**

***Schneier, Bruce. Applied Cryptography : Protocols, Algorithms, and Source Code in C 2nd ed.. 2nd ed. : John Wiley, 1996. Text.***

|  |  |
| --- | --- |
| Chapter 1. Foundations  1.1 Terminology  Sender and Receiver  Suppose a sender wants to send a message to a receiver. Moreover, this sender wants to send the message securely: She wants to make sure an eavesdropper cannot read the message.  Messages and Encryption  A message is plaintext (sometimes called cleartext). The process of disguising a message in such a way as to hide its substance is encryption. An encrypted message is ciphertext. The process of turning ciphertext back into plaintext is decryption. This is all shown in Figure 1.1.    Figure 1.1 Encryption and Decryption.  (If you want to follow the ISO 7498-2 standard, use the terms “encipher” and “decipher.” It seems that some cultures find the terms “encrypt” and “decrypt” offensive, as they refer to dead bodies.)  The art and science of keeping messages secure is cryptography, and it is practiced by cryptographers. Cryptanalysts are practitioners of cryptanalysis, the art and science of breaking ciphertext; that is, seeing through the disguise. The branch of mathematics encompassing both cryptography and cryptanalysis is cryptology and its practitioners are cryptologists. Modern cryptologists are generally trained in theoretical mathematics —they have to be.  Plaintext is denoted by M, for message, or P, for plaintext. It can be a stream of bits, a text file, a bitmap, a stream of digitized voice, a digital video image...whatever. As far as a computer is concerned, M is simply binary data.  (After this chapter, this book concerns itself with binary data and computer cryptography.) The plaintext can be intended for either transmission or storage.  In any case, M is the message to be encrypted.  Ciphertext is denoted by C. It is also binary data: sometimes the same size as M, sometimes larger. (By combining encryption with compression, C may be smaller than M. However, encryption does not accomplish this.) The encryption function E, operates on M to produce C.  Or, in mathematical notation:  E(M) = C  In the reverse process, the decryption function D operates on C to produce M:  D(C) = M  Since the whole point of encrypting and then decrypting a message is to recover the original plaintext, the following identity must hold true:  D(E(M)) = M  Authentication, Integrity, and Nonrepudiation  In addition to providing confidentiality, cryptography is often asked to do other jobs:  — Authentication. It should be possible for the receiver of a message to ascertain its origin; an intruder should not be able to masquerade as someone else.   — Integrity. It should be possible for the receiver of a message to verify that it has not been modified in transit; an intruder should not be able to substitute a false message for a legitimate one.  — Nonrepudiation. A sender should not be able to falsely deny later that he sent a message.  These are vital requirements for social interaction on computers, and are analogous to face-to-face interactions. That someone is who he says he is...that someone’s credentials — whether a driver’s license, a medical degree, or a passport — are valid...that a document purporting to come from a person actually came from that person.... These are the things that authentication, integrity, and nonrepudiation provide.  Algorithms and Keys  A cryptographic algorithm, also called a cipher, is the mathematical function used for encryption and decryption. (Generally, there are two related functions: one for encryption and the other for decryption.)  If the security of an algorithm is based on keeping the way that algorithm works a secret, it is a restricted algorithm. Restricted algorithms have historical interest, but are woefully inadequate by today’s standards. A large or changing group of users cannot use them, because every time a user leaves the group everyone else must switch to a different algorithm. If someone accidentally reveals the secret, everyone must change their algorithm.  Even more damning, restricted algorithms allow no quality control or standardization. Every group of users must have their own unique algorithm. Such a group can’t use off-the-shelf hardware or software products; an eavesdropper can buy the same product and learn the algorithm. They have to write their own algorithms and implementations. If no one in the group is a good cryptographer, then they won’t know if they have a secure algorithm.    Despite these major drawbacks, restricted algorithms are enormously popular for low-security applications. Users either don’t realize or don’t care about the security problems inherent in their system.  Modern cryptography solves this problem with a key, denoted by K. This key might be any one of a large number of values. The range of possible values of the key is called the keyspace. Both the encryption and decryption operations use this key (i.e., they are dependent on the key and this fact is denoted by the k subscript), so the functions now become:  EK(M) = C  DK(C) = M  Those functions have the property that (see Figure 1.2):  DK(EK(M)) = M  Some algorithms use a different encryption key and decryption key (see Figure 1.3).  That is, the encryption key, K1, is different from the corresponding decryption key, K2. In this case:  EK1(M) = C  DK2(C) = M  DK2(EK1 (M)) = M    All of the security in these algorithms is based in the key (or keys); none is based in the details of the algorithm. This means that the algorithm can be published and analyzed. Products using the algorithm can be mass-produced. It doesn’t matter if an eavesdropper knows your algorithm; if she doesn’t know your particular key, she can’t read your messages.  A cryptosystem is an algorithm, plus all possible plaintexts, ciphertexts, and keys.    Symmetric Algorithms  There are two general types of key-based algorithms: symmetric and public-key. Symmetric algorithms, sometimes called conventional algorithms, are algorithms where the encryption key can be calculated from the decryption key and vice versa. In most symmetric algorithms, the encryption key and the decryption key are the same. These algorithms, also called secret-key algorithms, single-key algorithms, or one-key algorithms, require that the sender and receiver agree on a key before they can communicate securely. The security of a symmetric algorithm rests in the key; divulging the key means that anyone could encrypt and decrypt messages. As long as the communication needs to remain secret, the key must remain secret.  Encryption and decryption with a symmetric algorithm are denoted by:  EK(M) = C  DK(C) = M  Symmetric algorithms can be divided into two categories. Some operate on the plaintext a single bit (or sometimes byte) at a time; these are called stream algorithms or stream ciphers. Others operate on the plaintext in groups of bits. The groups of bits are called blocks, and the algorithms are called block algorithms or block ciphers. For modern computer algorithms, a typical block size is 64 bits—large enough to preclude analysis and small enough to be workable. (Before computers, algorithms generally operated on plaintext one character at a time. You can think of this as a stream algorithm operating on a stream of characters.)  Public-Key Algorithms  Public-key algorithms (also called asymmetric algorithms) are designed so that the key used for encryption is different from the key used for decryption. Furthermore, the decryption key cannot (at least in any reasonable amount of time) be calculated from the encryption key. The algorithms are called “public-key” because the encryption key can be made public: A complete stranger can use the encryption key to encrypt a message, but only a specific person with the corresponding decryption key can decrypt the message.  In these systems, the encryption key is often called the public key, and the decryption key is often called the private key. The private key is sometimes also called the secret key, but to avoid confusion with symmetric algorithms, that tag won’t be used here.  Encryption using public key K is denoted by:  EK(M) = C    Even though the public key and private key are different, decryption with the corresponding private key is denoted by:  DK(C) = M    Sometimes, messages will be encrypted with the private key and decrypted with the public key; this is used in digital signatures. Despite the possible confusion, these operations are denoted by, respectively:  EK(M) = C  DK(C) = M  Cryptanalysis  The whole point of cryptography is to keep the plaintext (or the key, or both) secret from eavesdroppers (also called adversaries, attackers, interceptors, interlopers, intruders, opponents, or simply the enemy). Eavesdroppers are assumed to have complete access to the communications between the sender and receiver.  Cryptanalysis is the science of recovering the plaintext of a message without access to the key. Successful cryptanalysis may recover the plaintext or the key. It also may find weaknesses in a cryptosystem that eventually lead to the previous results. (The loss of a key through noncryptanalytic means is called a compromise.)  An attempted cryptanalysis is called an attack. A fundamental assumption in cryptanalysis, first enunciated by the Dutchman A. Kerckhoffs in the nineteenth century, is that the secrecy must reside entirely in the key.    Kerckhoffs assumes that the cryptanalyst has complete details of the cryptographic algorithm and implementation. (Of course, one would assume that the CIA does not make a habit of telling Mossad about its cryptographic algorithms, but Mossad probably finds out anyway.)    While real-world cryptanalysts don’t always have such detailed information, it’s a good assumption to make. If others can’t break an algorithm, even with knowledge of how it works, then they certainly won’t be able to break it without that knowledge.  There are four general types of cryptanalytic attacks. Of course, each of them assumes that the cryptanalyst has complete knowledge of the encryption algorithm used:  1. Ciphertext-only attack. The cryptanalyst has the ciphertext of several messages, all of which have been encrypted using the same encryption algorithm. The cryptanalyst’s job is to recover the plaintext of as many messages as possible, or better yet to deduce the key (or keys) used to encrypt the messages, in order to decrypt other messages encrypted with the same keys.    Given: C1 = Ek(P1), C2 = Ek(P2),...Ci = Ek(Pi)  Deduce: Either P1, P2,...Pi; k; or an algorithm to infer Pi+1 from  Ci+1 = Ek(Pi+1)  2. Known-plaintext attack. The cryptanalyst has access not only to the ciphertext of several messages, but also to the plaintext of those messages. His job is to deduce the key (or keys) used to encrypt the messages or an algorithm to decrypt any new messages encrypted with the same key (or keys).    Given: P1, C1 = Ek(P1), P2, C2 = Ek(P2),...Pi, Ci = Ek(Pi)  Deduce: Either k, or an algorithm to infer Pi+1 from  Ci+1 = Ek(Pi+1)  3. Chosen-plaintext attack. The cryptanalyst not only has access to the ciphertext and associated plaintext for several messages, but he also chooses the plaintext that gets encrypted. This is more powerful than a known-plaintext attack, because the cryptanalyst can choose specific plaintext blocks to encrypt, ones that might yield more information about the key. His job is to deduce the key (or keys) used to encrypt the messages or an algorithm to decrypt any new messages encrypted with the same key (or keys).  Given: P1, C1 = Ek(P1), P2, C2 = Ek(P2),...Pi, Ci = Ek(Pi), where the cryptanalyst gets to choose P1, P2,...Pi  Deduce: Either k, or an algorithm to infer Pi+1 from  Ci+1 = Ek(Pi+1)  4. Adaptive-chosen-plaintext attack. This is a special case of a chosen-plaintext attack. Not only can the cryptanalyst choose the plaintext that is encrypted, but he can also modify his choice based on the results of previous encryption.  In a chosen-plaintext attack, a cryptanalyst might just be able to choose one large block of plaintext to be encrypted; in an adaptive-chosen-plaintext attack he can choose a smaller block of plaintext and then choose another based on the results of the first, and so forth.  There are at least three other types of cryptanalytic attack.  5. Chosen-ciphertext attack. The cryptanalyst can choose different ciphertexts to be decrypted and has access to the decrypted plaintext. For example, the cryptanalyst has access to a tamperproof box that does automatic decryption. His job is to deduce the key.  Given: C1, P1 = Dk(C1), C2, P2 = Dk(C2),...Ci, Pi = Dk(Ci)  Deduce: k  This attack is primarily applicable to public-key algorithms. A chosen-ciphertext attack is sometimes effective against a symmetric algorithm as well.  (Sometimes a chosen-plaintext attack and a chosen-ciphertext attack are together known as a chosen-text attack.)  6. Chosen-key attack. This attack doesn’t mean that the cryptanalyst can choose the key; it means that he has some knowledge about the relationship between different keys. It’s strange and obscure, not very practical.  7. Rubber-hose cryptanalysis. The cryptanalyst threatens, blackmails, or tortures someone until they give him the key. Bribery is sometimes referred to as a purchase-key attack. These are all very powerful attacks and often the best way to break an algorithm.  Known-plaintext attacks and chosen-plaintext attacks are more common than you might think. It is not unheard-of for a cryptanalyst to get a plaintext message that has been encrypted or to bribe someone to encrypt a chosen message. You may not even have to bribe someone;   if you give a message to an ambassador, you will probably find that it gets encrypted and sent back to his country for consideration. Many messages have standard beginnings and endings that might be known to the cryptanalyst.   Encrypted source code is especially vulnerable because of the regular appearance of keywords: #define, struct, else, return. Encrypted executable code has the same kinds of problems: functions, loop structures, and so on. Known-plaintext attacks (and even chosen-plaintext attacks) were successfully used against both the Germans and the Japanese during World War II. David Kahn’s books [794,795,796] have historical examples of these kinds of attacks.  And don’t forget Kerckhoffs’s assumption: If the strength of your new cryptosystem relies on the fact that the attacker does not know the algorithm’s inner workings, you’re sunk.    If you believe that keeping the algorithm’s insides secret improves the security of your cryptosystem more than letting the academic community analyze it, you’re wrong.  And if you think that someone won’t disassemble your code and reverse-engineer your algorithm, you’re naïve.   (In 1994 this happened with the RC4 algorithm.) The best algorithms we have are the ones that have been made public, have been attacked by the world’s best cryptographers for years, and are still unbreakable.   (The National Security Agency keeps their algorithms secret from outsiders, but they have the best cryptographers in the world working within their walls—you don’t. Additionally, they discuss their algorithms with one another, relying on peer review to uncover any weaknesses in their work.)  Cryptanalysts don’t always have access to the algorithms, as when the United States broke the Japanese diplomatic code PURPLE during World War II —but they often do.   If the algorithm is being used in a commercial security program, it is simply a matter of time and money to disassemble the program and recover the algorithm.  If the algorithm is being used in a military communications system, it is simply a matter of time and money to buy (or steal) the equipment and reverse-engineer the algorithm.  Those who claim to have an unbreakable cipher simply because they can’t break it are either geniuses or fools. Unfortunately, there are more of the latter in the world. Beware of people who extol the virtues of their algorithms, but refuse to make them public; trusting their algorithms is like trusting snake oil.    Good cryptographers rely on peer review to separate the good algorithms from the bad.  Security of Algorithms  Different algorithms offer different degrees of security; it depends on how hard they are to break.   If the cost required to break an algorithm is greater than the value of the encrypted data, then you’re probably safe.    If the time required to break an algorithm is longer than the time the encrypted data must remain secret, then you’re probably safe.  If the amount of data encrypted with a single key is less than the amount of data necessary to break the algorithm, then you’re probably safe.  I say “probably” because there is always a chance of new breakthroughs in cryptanalysis. On the other hand, the value of most data decreases over time. It is important that the value of the data always remain less than the cost to break the security protecting it.    Lars Knudsen classified these different categories of breaking an algorithm.  In decreasing order of severity:  1. Total break. A cryptanalyst finds the key, K, such that DK(C) = P.  2. Global deduction. A cryptanalyst finds an alternate algorithm, A, equivalent to DK(C), without knowing K.  3. Instance (or local) deduction. A cryptanalyst finds the plaintext of an intercepted ciphertext.  4. Information deduction. A cryptanalyst gains some information about the key or plaintext. This information could be a few bits of the key, some information about the form of the plaintext, and so forth.  An algorithm is unconditionally secure if, no matter how much ciphertext a cryptanalyst has, there is not enough information to recover the plaintext.   In point of fact, only a one-time pad is unbreakable given infinite resources. All other cryptosystems are breakable in a ciphertext-only attack, simply by trying every possible key one by one and checking whether the resulting plaintext is meaningful. This is called a brute-force attack.    Cryptography is more concerned with cryptosystems that are computationally infeasible to break. An algorithm is considered computationally secure (sometimes called strong) if it cannot be broken with available resources, either current or future. Exactly what constitutes “available resources” is open to interpretation.  You can measure the complexity of an attack in different ways:  1. Data complexity. The amount of data needed as input to the attack.    2. Processing complexity. The time needed to perform the attack. This is often called the work factor.  3. Storage requirements. The amount of memory needed to do the attack.  As a rule of thumb, the complexity of an attack is taken to be the minimum of these three factors. Some attacks involve trading off the three complexities: A faster attack might be possible at the expense of a greater storage requirement.  Complexities are expressed as orders of magnitude. If an algorithm has a processing complexity of 2128, then 2128 operations are required to break the algorithm. (These operations may be complex and time-consuming.) Still, if you assume that you have enough computing speed to perform a million operations every second and you set a million parallel processors against the task, it will still take over 1019 years to recover the key.    That’s a billion times the age of the universe. While the complexity of an attack is constant (until some cryptanalyst finds a better attack, of course), computing power is anything but.    There have been phenomenal advances in computing power during the last half-century and there is no reason to think this trend won’t continue.   Many cryptanalytic attacks are perfect for parallel machines:  The task can be broken down into billions of tiny pieces and none of the processors need to interact with each other.   Pronouncing an algorithm secure simply because it is infeasible to break, given current technology, is dicey at best. Good cryptosystems are designed to be infeasible to break with the computing power that is expected to evolve many years in the future. | Глава 1. Основы   * 1. Терминология   Отправитель и Получатель  Предположим, отправитель хочет отправить сообщение получателю. Более того, этот отправитель хочет отправить сообщение безопасно: Она хочет убедиться в том, что подслушивающий не может прочитать данное сообщение.  Сообщения и Шифрование  Сообщение – это обычный текст (иногда называют «чистый текст»). Процесс сокрытия сообщения для сокрытия его сути – шифрование. Зашифрованное сообщение –это шифротекст. Процесс превращения шифротекста обратно в обычный текст – дешифрование. Это всё показано на рисунке 1.1.  Рисунок 1.1. Шифрование и Дешифрование.  (Если вы хотите следовать ГОСТ Р ИСО 7498-2, используйте термин «Шифрация» и «Дешифрация». Похоже, в некоторых народах термины «encrypt» и «decrypt» являются оскорбительными, поскольку они относятся к мёртвым телам (p.s. crypt – склеп с английского).)  Искусство и наука о сохранении секретности сообщений – это Криптография, и она практикуется криптографами. Криптоаналитики – специалисты в практике криптоанализа, искусстве и науке о взломе шифротекстов; или другими словами, о взгляде сквозь маскировку. Ответвление математики, охватывающее как криптографию, так и криптоанализ – криптология, а те, кто её практикуют – криптологи. Современные криптологи в основном обучены теоретической математике – они должны быть ей обучены.  Обычный текст обозначен буквой С – сообщение, или О – обычный. Он может являться потоковой передачей битов, текстового файла, битовой карты (- термин BitMap), потоком оцифрованного голоса, оцифрованного видеоизображения… чего угодно. До тех пор, пока это связанно с компьютером, С – простые бинарные (двоичные) данные.  (После данной главы, эта книга посвящена двоичным данным и компьютерной криптографии.) Обычный текст может предназначаться либо для передачи, либо для хранения.  В любом случае, С – сообщение, которое будет шифроваться.  Шифротекст обозначен буквой Ш. Он также представлен в двоичном виде: иногда того же размера что и С, иногда больше. (Совмещая зашифровку с сжатием, Ш может быть меньше, чем С. Тем не менее, шифрование такое цели не достигает.) Функция шифрования Фш действует на О, чтобы получить Ш.  Или на математическом языке:  Фш (О) = Ш  В обратном процессе, функция дешифрования Фд действует на Ш, чтобы получить О:  Фд (Ш) = О  Поскольку вся суть шифрования, а затем дешифрования сообщения в том, чтобы восстановить начальный обычный текст, следующее соотношение должно выполняться:  Фд (Фш (О)) = О  Аутентификация, целостность и отказоустойчивость  В добавление к предоставлению конфиденциальности, криптографии часто приходится выполнять такие задачи как:  — Аутентификация. Получатель сообщения должен иметь возможность удостовериться в его неизменности; злоумышленник не должен иметь возможности замаскироваться под кого-то другого.  — Целостность. Получатель сообщения должен иметь возможность проверить, что сообщение не было изменено в ходе передачи; злоумышленник не должен иметь возможности подменить ложным сообщением настоящее.  — Отказоустойчивость. Отправитель не должен иметь возможности ложно отменить отправку сообщения.  Это важные требования для общественного взаимодействия на компьютерах, и они аналогичны с взаимодействиями лицом к лицу. Что кто-то — это тот, кем он себя называет… что чьи-то учётные данные—будь то водительское удостоверение, медицинский диплом или паспорт—достоверны…что документ, якобы пришедший от человека и вправду пришёл от него… Эти факты обеспечивают аутентификация, целостность и отказоустойчивость.  Алгоритмы и ключи  Криптографический алгоритм, также называемый шифром, это математическая функция, используемая для зашифрования и дешифрования. (Зачастую, существует две связанные функции: одна для зашифрования, и другая для дешифрования.)  Если безопасность алгоритма основана на том, чтобы сохранить механизм работы алгоритма в секрете, это ограниченный алгоритм. Ограниченные алгоритмы имеют историческую ценность, но крайне неадекватны по современным стандартам. Большая или часто меняющаяся группа пользователей не может их использовать, поскольку каждый раз, когда один из пользователей покидает группу, все оставшиеся обязаны перейти на другой алгоритм. Если кто-то случайно раскрывает секрет, все обязаны сменить их алгоритм.  Что осуждается ещё больше предыдущего факта, ограниченных алгоритмов допускает отсутствие контроля качества и стандартизации. Каждая группа пользователей должна иметь свой уникальный алгоритм. Такая группа не может использовать уже готовое оборудование или программное обеспечение; подслушивающий может купить такой же продукт и изучить алгоритм. Им придётся написать свои собственные алгоритмы и их применения. Если в группе нету хорошего криптографа, они не смогут узнать, безопасен ли их алгоритм.  Несмотря на эти крупные недостатки, ограниченные алгоритмы чрезвычайно популярны для мало-безопасных приложений. Пользователи либо не поймут, либо им будет не важна проблема в безопасности, существующая в их системе.  Современная криптография решает эти проблемы при помощи ключа, обозначаемого К. Этот ключ может принимать любое значение из огромного множества. Диапазон всех возможных значений ключей называется ключевым пространством (пространством ключей). Обе операции, шифрования и дешифрования, используют этот ключ (то есть, они зависят от ключа и этот факт обозначается К в индексе), поэтому функции приобретают вид:  Фш от К (О) = Ш  Фд от К (Ш) = О  Эти функции имеют такое свойство (рис. 1.2):  Фд от К (Фш от К (О)) = О  Некоторые алгоритмы используют разные ключи шифрования и дешифрования (рис. 1.3).   То есть, ключ шифрования К1 отличен от соответствующего ключа дешифрования К2. В данном случае:  Фш от К1 (О) = Ш  Фд от К2 (Ш) = О  Фд от К2 (Фш от К1 (О)) = О    Вся безопасность в данных алгоритмах основана на ключе (или ключах); ничто не основано на деталях алгоритма. Это означает, что алгоритм может быть опубликован и проанализирован. Средства, использующие такой алгоритм могут быть массово произведены. Даже не важно, что подслушивающий может знать ваш алгоритм; если он не знает именно вашего ключа, он не сможет прочитать ваши сообщения.  Криптосистема – объединение алгоритма, всех возможных исходных сообщений, шифротекстов и ключей.  Симметричные алгоритмы  Существует два основных типа алгоритмов, основанных на ключах: симметричные и алгоритмы на открытом ключе. Симметричные алгоритмы, иногда так же именуемые обычными алгоритмами, это алгоритмы, где ключ шифрования может определён, зная ключ дешифрования и наоборот. В основном в симметричных алгоритмах ключ шифрования совпадает с ключом дешифрования. Эти алгоритмы, также называемые алгоритмы единого ключа или одного ключа, требуют, чтобы отправитель и получатель договорились о ключе перед тем, как они будут безопасно общаться. Обеспечение безопасности симметричных алгоритмов возлагается на ключ; раскрытие ключа означает, что кто угодно может зашифровать или расшифровать сообщения. До тех пор, пока требуется, чтобы общение было секретным, ключ должен оставаться в секрете.  Шифрование и дешифрование с симметричным ключом обозначается:  Фш от К (О) = Ш  Фд от К (Ш) = О  Симметричные алгоритмы могут быть подразделены на две категории. Некоторые преобразуют обычный текст по одному биту/байту за раз; они называются поточными алгоритмами или поточными шифрами. Другие преобразуют обычный текст по группам битов. Эти группы называют блоками, а сами алгоритмы – блочные алгоритмы или блочные шифры. Для современных компьютерных алгоритмов, типичный размер блока – 64 бита – достаточно много для того, чтобы помешать анализу (криптоанализу) и достаточно мало, чтобы быть пригодным для использования. (До компьютеров, алгоритмы в основном преобразовывали обычный текст посимвольно. Это можно представить как поточный алгоритм преобразующий поток символов.)  Алгоритмы с открытым ключом  Алгоритмы с открытым ключом (также известные как асимметричные алгоритмы) разработаны таким образом, что ключ, используемый для шифрования отличен от ключа для расшифрования. Более того, ключ расшифрования не может (по крайней мере за разумное количество времени) быть вычислен, зная ключ шифрования. Алгоритм называется «с открытым ключом» по той причине, что ключ шифрования может быть публичным: абсолютно незнакомый человек может использовать ключ шифрования для зашифрования сообщения, то только конкретный человек, имеющий соответствующий ключ дешифрования, может расшифровать сообщение.  В этих системах ключ шифрования часто называется открытым ключом, а ключ дешифрования – закрытый. Закрытый ключ иногда так же зовётся секретным ключом, но, чтобы избежать путаницы с симметричными алгоритмами, такое название не будет здесь использоваться.  Шифрования с использованием открытого ключа К обозначается так:  Фш от К (О) = Ш  Даже если открытый ключ и закрытый ключ отличаются, дешифрование с соответствующим закрытым ключом обозначается:   ФД от К (Ш) = О  Иногда, сообщения будут зашифрованы при помощи закрытого ключа, а расшифрованы при помощи открытого; это используется в цифровых подписях. Несмотря на возможную путаницу, эти операции обозначаются соответственно:   ФШ от К (О) = Ш  ФД от К (Ш) = О  Криптоанализ  Вся суть криптографии в том, чтобы сохранять обычный текст (или ключ, или оба) в секрете от подслушивающих (также именуемых противниками, атакующими, перехватчики, нарушители, злоумышленники, оппоненты или просто врагами). Предполагается, что подслушивающие должны иметь полный доступ к связи (сообщению) между отправителем и получателем.  Криптоанализ – наука о восстановлении обычного текста без доступа к ключу. Успешный криптоанализ может восстановить обычный текст или ключ. Он так же может найти слабости в криптосистеме, которые в конце концов приводят к прошлому пункту. (Ключ, потерянный не криптографическими средствами, называется скомпрометированным.)  Попытка криптоанализа называется атакой. Фундаментальное предположение в криптоанализе, впервые сформулированное датчанином Августом Керхсхоффом в 19-ом веке, состоит в том, что вся секретность должна состоять в ключе. Керхсхофф предполагает, что криптоаналитик имеет все детали о криптографическом алгоритме и его применении. (Само собой, можно было бы предположить, что ЦРУ не вырабатывает привычку из того, чтобы говорить Моссад о своих криптографических алгоритмах, но Моссад, вероятно, всё равно узнает.)  Хоть настоящий криптоаналитик не всегда обладает столь точной информацией, это неплохое предположение. Если другие не могут сломать алгоритм, даже зная, как он работает, то они определённо не смогут его сломать, не имея такого знания.  Существует 4 основных типа криптоаналитических атак. Разумеется, каждая из них предполагает, что криптоаналитик имеет полное знание о используемом алгоритме шифрования:  1. Атаки только на шифротекст. У криптоаналитика есть шифротекст из нескольких сообщений, каждый из которых был зашифрован с использованием одного и того же алгоритма шифрования. Задача криптоаналитика в том, чтобы восстановить обычный текст максимально большого числа сообщений, а еще лучше – заполучить ключ (или ключи), использованные для шифрования сообщений для того, чтобы расшифровать другие сообщения, зашифрованные теми же ключами.  Дано: Ш1 = ФШ от К (О1), ..., Шi = ФШ от К (Оi) Выведем: Какой-то из О1, ... Оi; К (ключ); или алгоритм связан с Оi+1: Шi+1 = ФШ от К (Оi+1)  2. Атаки при известном обычном тексте. У криптоаналитика есть доступ не только к шифротексту из нескольких сообщений, но и обычный текст тех сообщений. Его задача в том, чтобы вывести ключ (ключи), использованные для шифрования сообщений или алгоритм для того, чтобы дешифровать любые новые сообщения, зашифрованные тем же ключом(ключами).  Дано: О1, Ш1 = ФШ от К (О1), ..., Оi, Шi = ФШ от К (Оi) Выведем: Какой-то из О1, ... Оi; К; или алгоритм связан с Оi+1: Шi+1 = ФШ от К (Оi+1)  3. Атака на выбранный (конкретный) обычный текст. У криптоаналитика есть доступ не только к шифротексту и связанному с ним обычному тексту, но он так же выбирает обычный текст, что зашифровывается. Это мощнее, чем атака при известном обычном тексте, поскольку криптоаналитик может выбрать конкретные блоки обычного текста, что каждый может дать больше полезной информации о ключе (ключах), использованных для шифрования сообщений или алгоритм, чтобы дешифровать новые сообщения, зашифрованные тем же ключом (ключами).  Дано: О1, Ш1 = ФШ от К (О1), ..., Оi, Шi = ФШ от К (Оi), а криптоаналитик выбирает О1, ..., Оi  Выведем: Или К, или алгоритм связан с Оi+1:  Шi+1 = ФШ от К (Оi+1)  4. Атака с подстраиваемо-выбираемым обычным текстом. Это особый случай атаки на выбранный обычный текст. Криптоаналитик не только может выбрать зашифрованный текст, но он так же может изменять свой выбор в зависимости от результатов прошлого шифрования. В атаке на выбранный обычный текст, криптоаналитик может быть попросту способным выбрать один огромный блок обычного текста, что будет зашифрован; в подстраиваемой атаке он может выбрать меньший блок обычного текста и затем выбрать другой, основываясь на результатах первого и так далее.  Существует по меньшей мере 3 других типа криптоаналитической атаки.  5. Атака на выбранный шифротекст. Криптоаналитик может выбрать различные шифротексты для дешифрования, а также имеет доступ к расшифрованному обычному тексту. Например, у криптоаналитика есть доступ к защищенному от несанкционированного доступа ящику, что автоматически производит дешифрование. Его задача в том, чтобы заполучить ключ.   Дано: Ш1, О1 = ФД от К (Ш1), ..., Шi, Оi = ФД от К (Шi)  Выведем: К  Эта атака в основном применима к алгоритмам с открытым ключом. Атака на выбранный шифротекст иногда эффективна против и симметричных алгоритмов.  (Иногда атака на выбранный обычный текст и на выбранный шифротекст вместе называют атакой на выбранный текст.)  6. Атака на выбранный ключ. Эта атака не означает, что криптоаналитик может выбрать ключ; она означает, что у него есть некоторое знание (информация) об отношениях между различными ключами. Это странно, непонятно и не очень практично.  7. Бандитский криптоанализ (криптоанализ резиновым шлангом). Криптоаналитик угрожает, шантажирует или пытает кого-то до тех пор, пока они не дадут ему ключ. Взяткой иногда относится к атаке купленным ключом. Это всё очень мощные атаки и часто являются лучшим путем к взлому алгоритма.  Атаки на известный или выбранный обычный текст более популярны, нежели вы могли бы подумать. Нет ничего необычного в том, что криптоаналитик получает обычный текст сообщения или подкупает кого-то, кто шифрует выбранное сообщение. Вы может быть даже не понадобиться подкупать кого-то;  если у вы передадите сообщение представителю, вы, вероятно, заметите, что оно шифруется и отправляется обратно в его страну на рассмотрение. Множество сообщений имеют стандартные начала и окончания, которые могут быть известны криптоаналитику.  Зашифрованный исходный код особенно уязвим за счёт частого появления таких ключевых слов, как: #define, struct, else, return. Зашифрованный код запускаемого файла имеет те же проблемы: функции, конструкции циклов и так далее. Атаки на известный обычный текст (и даже атаки на выбранный обычный текст) были успешно использованы против Германии и Японии во время Второй Мировой Войны. В книгах Дэвида Кана есть исторические примеры таких атак.  И не забывайте о предположении Керхгоффа: если сила вашей новой криптосистемы основана на том факте, что атакующий не знает внутреннего алгоритма работы, вы потонете (т.е. вашу систему вскроют).  Если вы верите, что сокрытие алгоритма в секрете улучшает защищённость вашей криптосистемы больше, нежели разрешение учёному сообществу анализировать его, вы ошибаетесь. А ещё, если вы считаете, что никто не сможет разобрать ваш код и заревёрсинжинирить (методами обратной инженерии, спроектировать заново) ваш алгоритм, то вы наивны. (Такое произошло с алгоритмом RC4 в 1994.) Лучшие алгоритмы из тех, что мы имеем, это те, что были сделаны публичными, атаковались лучшими криптографами мира годами, но всё равно остались несломленными (их не взломали). (Национальное Агентство Безопасности хранит свои алгоритмы в секрете от других, но в их стенах работают лучшие из криптографов всего мира – у вас – нет. Кроме того, они обсуждают свои алгоритмы с другими, рассчитывая на взаимный обзор для выявления любых слабостей в их работе.)  У криптоаналитиков не всегда есть доступ к алгоритмам, как тогда, когда Соединённые Штаты взломали японский дипломатический код PURPLE (Фиолетовый) во время Второй Мировой Войны – но зачастую он у них есть. Если алгоритм используется в коммерческой программе по защите, это лишь вопрос времени и денег для того, чтобы разобрать программу и восстановить алгоритм. Если алгоритм используется в военных системах сообщения, это лишь вопрос времени и денег в том, чтобы купить (или украсть) оборудование и заревёрсинжинирить алгоритм.  Те, кто заявляют, что обладают не взламываемым шифром, попросту по той причине, что не смогли его сами взломать, либо гении, либо балбесы (глупые). К сожалению, существует несколько недавних таких в мире. Остерегайтесь людей, что восхваляют достоинства своих алгоритмов, но отказываются сделать их публичными; доверять их алгоритмам то же самое, что доверить змеиному яду.  Хорошие криптографы полагаются на взаимный обзор для разделения алгоритмы на плохие и хорошие.  Безопасность алгоритмов  Различные алгоритмы предлагают различные уровни защищённости; она (защищённость) зависит от того, насколько сложно их (алгоритмы) взломать.  Если затраты, требуемые на взлом алгоритма, выше, чем ценность зашифрованных данных, тогда вы, вероятно, в безопасности.  Если время, требуемое на взлом алгоритма, дольше чем время, которое зашифрованные данные должны храниться в секрете, тогда вы, вероятно, в безопасности.  Если объём данных, зашифрованных одним ключом, меньше объема данных, необходимых для взлома алгоритма, тогда вы, вероятно, в безопасности.  Я говорю «вероятно», поскольку всегда есть шанс на новый прорыв в криптоанализе. С другой стороны, ценность большей части информации снижается со временем. Важно, что ценность данных всегда остаётся меньшей, чем стоимость взлома охраняющих её мер безопасности.  Ларс Кнудсен классифицировал эти различные категории взломов алгоритмов. По уменьшению степени тяжести:  1. Полный взлом. Криптоаналитик находит ключ, К, такой что ФД от К (Ш) = О.  2. Общий вывод. Криптоаналитик находит альтернативный (иной) алгоритм, эквивалентный к ФД от К (Ш) без знания К.  3. Вывод по примеру или локальный вывод. Криптоаналитик находит обычный текст от перехваченного шифротекста.  4. Информационный вывод. Криптоаналитик собирает некоторую информацию о ключе или обычном тексте. Данной информацией может являться пара битов или ключ, некоторая информация о форме (особенности) обычного текста и так далее.  Алгоритм безусловно безопасен, если сколько много не было бы шифротекста у криптоаналитика, ему не хватает информации для восстановления обычного текста. По сути дела, лишь одноразовый блокнот является не взламываемым, при условии неограниченных ресурсов. Все остальные криптосистемы взламываемы при атаке только на шифротекст, попросту перебрав все возможные ключи один за другим и проверяя, имеет ли смысл получившийся обычный текст. Это называется методом атаки грубой силой.  Криптографию больше волнуют криптосистемы, что вычислительно нельзя взломать. Алгоритм считается вычислительно безопасным (иногда говорят сильным), если его нельзя взломать при имеющихся ресурсах, быть то текущие или будущие. А вот именно то, что составляет «доступные ресурсы» - можно по-разному интерпретировать.  Вы можете оценить сложность атаки разными способами:  1. Сложность данных. Количество данных, необходимых как заданные условия для атаки.  2. Сложность обработки. Время, необходимое для исполнения атаки. Это также называют рабочим фактором.  3. Требования по хранению. Количество (объём) памяти, необходимый для совершения атаки.  Как правило, на практике сложность атаки состоит минимум из трёх этих факторов. Некоторые атаки включают в себя компромиссы для этих трудностей: Более быстрая атака может привести к затратам на удовлетворение требования большего хранилища.  Сложности выражены в порядке величины. Если у алгоритма обработочная сложность 2128, то 2128 операций потребуется на взлом алгоритма. (Эти операции могут быть непростыми и время затратными.) Тем не менее, если вы предположите, что вам хватает вычислительной скорости для выполнения миллиона операций в секунду и вы поставите миллион процессоров, работающих параллельно, против задачи, это всё равно займёт более 1019 лет на восстановление ключа. Это в миллиард раз больше возраста вселенной. В то время как сложность атаки постоянна (до тех пор, пока криптоаналитик не найдет атаку получше, разумеется), вычислительная мощность какая угодно, но вовсе не такова (не безгранична). Были достигнуты феноменальные достижения в области вычислительной мощности за последние полвека, и нет никаких причин думать, что эта тенденция не сохранится. Множество криптоаналитических атак совершенны для параллельных устройств:  Задача может быть разделена на миллиарды крошечных частей, а ни одному процессору не придется взаимодействовать с другими. Разглашать о защищённости алгоритма по той причине, что он не взламываемый, при нынешних технологиях, максимум рискованно. Хорошие криптосистемы спроектированы так, чтобы быть не поддаваемыми взлому при помощи вычислительных мощностей, эволюция которых ожидается через много лет в будущем. |

|  |  |
| --- | --- |
| Glossary   1. Plaintext – the text, that is needed to be encrypted. 2. Ciphertext – the text, that already has been encrypted/enciphered/encoded. 3. Encryption – the process of applying cipher or algorithm to transform Plaintext into Ciphertext. 4. Decryption – the process of applying reversed algorithm to Ciphertext in order to transform it into Plaintext. 5. Cipher – a set of rules, replacements tables, alphabets for Encryption algorithm. 6. Key – an argument for Encryption and/or Decryption algorithms. 7. Cryptography – a science of keeping messages secure. 8. Cryptanalysis – a science of breaking ciphertext; that is, seeing through the disguise (cipher). 9. Cryptology – the branch of mathematics encompassing both cryptography and cryptanalysis. 10. Bitmap – a representation in which each item corresponds to one or more bits of information. Usually being used for images. 11. Binary data – the umbrella term encompassing any data, that is being represented with ones and zeros (0, 1). 12. CIA – Central Intelligence Agency, whom task is to gather, process, and analyze national security information from around the World. | Словарь терминов   1. Обычный текст – текст, что необходимо зашифровать. 2. Шифротекст – текст, что уже был зашифрован/закодирован. 3. Зашифрование – процесс применения шифра или алгоритма для преобразования обычного текста в шифротекст. 4. (Рас-)Дешифрование – процесс применения обратного алгоритма к Шифротексту для превращения его в обычный текст. 5. Шифр – набор правил, таблиц замены, алфавитов для алгоритма шифрования. 6. Ключ – аргумент алгоритма Шифрования/Расшифрования. 7. Криптография – наука о сохранении сообщений в секрете. 8. Криптоанализ – наука о взломе шифротекстов; то есть, о том, как смотреть сквозь маскировку (шифр). 9. Криптология – ответвление математики, охватывающее и криптографию, и криптоанализ. 10. Битовая карта – вид представления, в котором каждый элемент соответствует одному или нескольким битам информации. Обычно используется для изображений. 11. Двоичные данные – обобщающий термин, охватывающий любые данные, что представлены с помощью единиц и нулей (0, 1). 12. ЦРУ – Центральное Разведывательное Управление, чья задача состоит в том, чтобы собирать, обрабатывать и анализировать информацию о национальной безопасности со всего Мира. |