Introduction to Cryptography

Cryptography is the study of sending and receiving secret messages. The aim of cryptography is to send messages across a channel so only the intended recipient of the message can read it. In addition, when a message is received, the recipient usually requires some assurance that he message is authentic; that is, that it has not been sent by someone who is trying to deceive the recipient. Modern cryptography is heavily dependent on abstract algebra and number theory.

The message to be sent is called the plaintext message. The disguised message is called the ciphertext. The plaintext and the ciphertext are both written in an alphabet, consisting of letters or characters. Characters can include not only the familiar alphabetic characters A, . . ., Z and a, . . ., z but also digits, punctuation marks, and blanks. A cryptosystem, or cipher, has two parts: encryption, the process of transforming a plaintext message to a ciphertext message, and decryption, the reverse transformation of changing a ciphertext message into a plaintext message. There are many different families of cryptosystems, each distinguished by a particular encryption algorithm. Cryptosystems in a specified cryptographic family are distinguished from one another by a parameter to the encryption function called a key. A classical cryptosystem has a single key, which must be kept secret, known only to the sender and the receiver of the message. If person A wishes to send secret messages to two different people B and C, and does not wish to have B understand C's messages or vice versa, A must use two separate keys, so one cryptosystem is used for exchanging messages with B, and another is used for exchanging messages with C.

Systems that use two separate keys, one for encoding and another for decoding, are called public key cryptosystems. Since knowledge of the encoding key does not allow anyone to guess at the decoding key, the encoding key can be made public. A public key cryptosystem allows A and B to send messages to C using the same encoding key. Anyone is capable of encoding a message to be sent to C, but only C knows how to decode such a message.

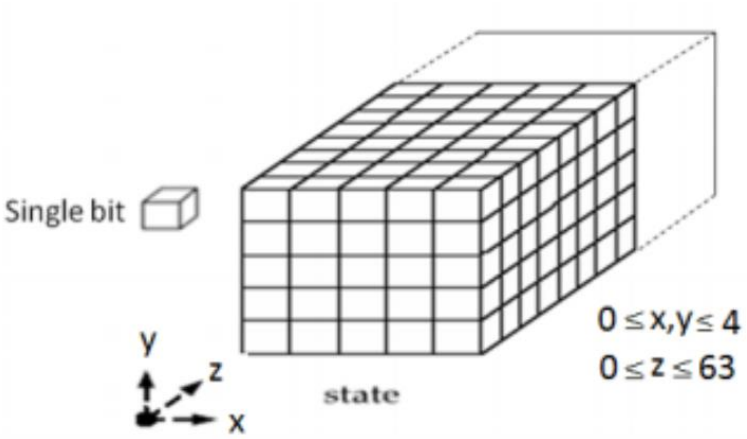
In single or private key cryptosystems the same key is used for both encrypting and decrypting messages. To encrypt a plaintext message, we apply to the message some function which is kept secret, say f. This function will yield an encrypted message. Given the encrypted form of the message, we can recover the original message by applying the inverse transformation f^(−1). The transformation f must be relatively easy to compute, as must f^(−1); however, f must be extremely difficult to guess at if only examples of coded messages are available.

SHA-3 Secure Hash Algorithm: New Face Of Crypto

Commonly used Hash functions are SHA-1, SHA-256, SHA-512, RIPEMD, MD4 and MD5. In previous years Cryptanalysis of these algorithms has found serious vulnerabilities [1][2][3]. Although no attacks have yet been reported on the SHA-2 variants, but due to their algorithmically similarities to SHA-1, there are fears that SHA-2 could also be cracked in the near future. The National Institute of Standards and Technology(NIST), USA announced the SHA-3 Contest in Nov 2007 [4]. This contest was to result in a new and secure cryptographic hash algorithm. This competition ended on 2nd October‟12, after the announcement of Keccak, one of the finalists of SHA-3 competition, as the winner for the title of

SHA-3[5]. For the IoT applications, hardware implementations of cryptographic hash algorithms are needed to provide high speed and near-real time results. ASICs and FPGAs are the two hardware platforms that can be used for these implementations. FPGAs offer numerous advantages for algorithm implementations over ASICs, such as: reliability, flexibility, low cost, rapid time-to-market and long-term maintenance.

SHA-3[9] is a family of sponge functions characterized by two parameters, the bitrate r and capacity c. The sum, r + c determine the width of the SHA-3 function permutation used in the sponge construction and is restricted to a maximum value of 1600. Selection of r and c depends on the desired hash output value. Ex.: for a 256-bit hash output r = 1088 and c = 512 and for 512-bit hash output r = 576 and c = 1024 is selected. The 1600-bit state of SHA-3 consists of a 5x5 matrix of 64-bit words as shown.



Examples for SHA-1, SHA-2 and SHA-3

This examples summarises useful test vectors for the secure hash algorithms SHA-1, SHA-2 and the new SHA-3.

Test Vectors

Input message: `"abc"`, the bit string `(0x)616263` of length 24 bits.

Algorithm	Output
SHA-1	a9993e36 4706816a ba3e2571 7850c26c 9cd0d89d
SHA-224	23097d22 3405d822 8642a477 bda255b3 2aadbce4 bda0b3f7 e36c9da7
SHA-256	ba7816bf 8f01cfea 414140de 5dae2223 b00361a3 96177a9c b410ff61 f20015ad
SHA-384	cb00753f45a35e8b b5a03d699ac65007 272c32ab0eded163 1a8b605a43ff5bed 8086072ba1e7cc23 58baeca134c825a7
SHA-512	ddaf35a193617aba cc417349ae204131 12e6fa4e89a97ea2 0a9eeee64b55d39a 2192992a274fc1a8 36ba3c23a3feebbd 454d4423643ce80e 2a9ac94fa54ca49f
SHA-3-224	e642824c3f8cf24a d09234ee7d3c766f c9a3a5168d0c94ad 73b46fdf
SHA-3-256	3a985da74fe225b2 045c172d6bd390bd 855f086e3e9d525b 46bfe24511431532
SHA-3-384	ec01498288516fc9 26459f58e2c6ad8d f9b473cb0fc08c25 96da7cf0e49be4b2 98d88cea927ac7f5 39f1edf228376d25
SHA-3-512	b751850b1a57168a 5693cd924b6b096e 08f621827444f70d 884f5d0240d2712e 10e116e9192af3c9 1a7ec57647e39340 57340b4cf408d5a5 6592f8274eec53f0

Input message: the empty string `""`, the bit string of length 0.

Algorithm	Output
SHA-1	da39a3ee 5e6b4b0d 3255bfef 95601890 afd80709
SHA-224	d14a028c 2a3a2bc9 476102bb 288234c4 15a2b01f 828ea62a c5b3e42f
SHA-256	e3b0c442 98fc1c14 9afbf4c8 996fb924 27ae41e4 649b934c a495991b 7852b855
SHA-384	38b060a751ac9638 4cd9327eb1b1e36a 21fdb71114be0743 4c0cc7bf63f6e1da 274edebfe76f65fb d51ad2f14898b95b
SHA-512	cf83e1357eefb8bd f1542850d66d8007 d620e4050b5715dc 83f4a921d36ce9ce 47d0d13c5d85f2b0 ff8318d2877eec2f 63b931bd47417a81 a538327af927da3e
SHA-3-224	6b4e03423667dbb7 3b6e15454f0eb1ab d4597f9a1b078e3f 5b5a6bc7
SHA-3-256	a7ffc6f8bf1ed766 51c14756a061d662 f580ff4de43b49fa 82d80a4b80f8434a
SHA-3-384	0c63a75b845e4f7d 01107d852e4c2485 c51a50aaaa94fc61 995e71bbee983a2a c3713831264adb47 fb6bd1e058d5f004
SHA-3-512	a69f73cca23a9ac5 c8b567dc185a756e 97c982164fe25859 e0d1dcc1475c80a6 15b2123af1f5f94c 11e3e9402c3ac558 f500199d95b6d3e3 01758586281dcd26

For those who would like to code SHA-3 algorithm and test if it works correctly or just try the SHA-3 Generator : https://asecuritysite.com/encryption/sha3