Hash Algorithms

	Size	Security	Key Length	Comments
MD5 message-digest algorithm (Series MD2, MD4, MD5, MD6) -wikipedia.org	Digest sizes: 128 bit Block sizes: 512 bit	Severely compromised; Collision attack exists and Chosen-prefix collision attack;	Determining the partition for a particular key in a partitioned database; Key stretching - used when stored in a one-way hash of a password; zero-length string: MD5("") = d41d8cd98f00b204e9800 9 98ecf8427e	Widely used hash function producing a 128-bit hash value; initially designed to be used as a cryptographic hash function; found to suffer from extensive vulnerabilities; still can be used as a checksum to verify data integrity, but only against unintentional corruption; remains suitable for other non-cryptographic purposes (determining the partition for a particular key in a partitioned database);
RIPEMD (Ripe Message Digest) (Functions in the family: RIPEMD, RIPEMD-128, RIPEMD-160, RIPEMD-256, and RIPEMD-320) -wikipedia.org	Digest sizes: 128, 160, 256, 320 bits;	The 256 - and 320-bit versions of RIPEMD provide the same level of security as RIPEMD-128 and RIPEMD-160, respectively; they are designed for applications where the security level is sufficient but longer hash result is necessary;	Design influenced by MD4 (Message-Digest Algorithm); MD4 is used to compute NTLM password-derived key digests on Microsoft Windows NT, XP, Vista, 7, 8, and 10. zero-length string: RIPEMD-160("") = 9c1185a5c5e9fc54612808 977ee8f548b2258d31	The original RIPEMD, as well as RIPEMD-128, is not considered secure because 128-bit result is too small and also (for the original RIPEMD) because of design weaknesses.
WHIRLPOOL (hash function) (Versions Whirlpool-0, Whirlpool-T, and Whirlpool) -wikipedia.org	Digest sizes: 512 bits;	Security Claims: Large hashsum size;	The AddRoundKey operation uses bitwise or to add a key calculated by the key schedule to the current state. The key schedule is identical to the encryption itself, except the AddRoundKey function is replaced by an AddRoundConstant function that adds a predetermined constant in each round. zero-length string: Whirlpool-0 ("") = B3E1AB6EAF640A34F7 84593F2074416ACCD3B 8E62C620175FCA0997B 1BA23473 39AA0D79E754C308209 EA36811DFA40C1C32F1 A2B9004725D987D3635 165D3C8	The Whirpool algorithm has undergone two revisions since its original 2000 specification. People incorporating Whirpool will most likely use the most recent revision of Whirlpool; while there are no known security weaknesses in earlier versions of Whirlpool, the most recent revision has better hardware implementation efficiency characteristics, and is also likely to be more secure; The 512-bit (64 - byte) Whirlpool hashes (also termed message digests) are typically represented as 128-digit hexadecimal numbers.

	Size	Security	Key Length	Comments
Secure Hash Algorithms (Functions in the family: SHA-0, SHA-1, SHA-2, SHA-3) -wikipedia.org	Output sizes (bits): 160, 224 - 256, 384 - 512, 224 - 256, 224 - 256 - 384 - 512; Block size (bits): 512, 512, 512, 1024, 1152 - 1088 - 832 - 576, 1344 - 1088;	All SHA-family algorithms, as FIPS-approved security functions, are subject to official validation; Security (in bits) against collision attacks can be viewed via Wikipedia;	Due to the block and iterative structure of the algorithms and the absence of additional final steps, all SHA functions (except SHA-3) are vulnerable to length-extension and partial-message collision attacks. These attacks allow an attacker to forge a message signed only by a keyed hash-SHA(message ‖ key) or SHA(key ‖ message) - by extending the message and recalculating the hash without knowing the key; zero-length string: SHA1("") gives hexadecimal: da39a3ee5e6b4b0d3255bfef95601890afd80709 gives Base64 binary to ASCII text encoding: 2jmj7l5rSw0yVb/vlWAYkK/YBwk=	The Secure Hash Algorithms are a family of cryptographic hash functions including: -SHA-0: A retronym applied to the original version of the 160-bit hash function published in 1993 under the name “SHA”. It was withdrawn shortly after publication due to an undisclosed “significant flaw” and replaced by the slightly revised version SHA-1. -SHA-1: A 160-bit hash function which resembles the earlier MD5 algorithm. This was designed by the National Security Agency (NSA) to be part of the Digital Signature Algorithm. Cryptographic weaknesses were discovered in SHA-1, and the standard was no longer approved for most cryptographic uses after 2010. -SHA-2: A family of two similar hash functions, with different block sizes, known as SHA-256 and SHA-512. They differ in the word size; SHA-256 uses 32 - bit words where SHA-512 uses 64-bit words. There are also truncated versions of each standard, known as SHA-224, SHA-384, SHA-512/224 and SHA-512/256. These were also designed by the NSA. -SHA-3: A hash function formerly called Keccak, chosen in 2012 after a public competition among non-NSA designers. It supports the same hash lengths as SHA-2, and its internal structure differs significantly from the Secure Hash Standard (SHS).

Diffie-Hellman Algorithms

Assume you are A and Amazon is B.

A and B choose the following number

Base Number 16

Prime Number 541

A chooses 7 as its secret number

B chooses 12 as its secret number

Using the Diffie-Hellman algorithm, find the secret key number:

1. A and B agree to use a Prime Number (p = 541) and Base Number (g = 16);
- p = 541;
- g = 16;
- a = 7;
- b = 12;

2. A chooses a secret integer a = 7, then sends B:
- A = g^a mod p;
- A = 16^7 mod 541;
- A = 268435456 mod 541;
- A = 453;

3. B chooses a secret integer b = 12, then sends A:
- B = g^b mod p;
- B = 16^12 mod 541;
- B = 281474977000000 mod 541;
- B = 345;

4. A computes:
- s = B^a mod p;
- s = 345^7 mod 541;
- s = 581746348000000000 mod 541;
- s = 277;

5. B computes:
- s = A^b mod p;
- s = 453^12 mod 541;
- s = 746755489999999965168481452687 36 mod 541;
- s = 453;

Example from Wikipedia - Diffie–Hellman key exchange - Cryptographic explanation:

1. Alice and Bob public agree to use a modulus p = 23 and base g = 5 (which is a primitive root modulo 23).
- p = 23;
- g = 5;

2. Alice chooses a secret integer a = 4, then sends Bob A = g^a mod p;
- a = 4;
- A = g^a mod p;
- A = 5^4 mod 23;
- A = 625 mod 23;
- A = 4;

3. Bob chooses a secret integer b = 3, then sends Alice B = g^b mod p;
- b = 3;
- B = g^b mod p;
- B = 5^3 mod 23;
- B = 125 mod 23;
- B = 10;

4. Alice computes s = B^a mod p;
- s = B^a mod p;
- s = 10^4 mod p;
- s = 10000 mod 23;
- s = 18;

5. Bob computes s = A^b mod p;
- s = A^b mod p;
- s = 4^3 mod p;
- s = 64 mod 23;
- s = 18;

-https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange