Obvious applications of hash functions include:

- **error-checking:** send m and $H(m)$ instead of just m
- **tamper-protection:** send m and $H(m)$ via different channels (H must be one-way!)

  If H is one-way, then Eve cannot find m' such that $H(m') = H(m)$, so the cannot tamper with m without it being detected.
- **password storage:** discussed later (there are some tricky bits)

Some popular hash functions:

	published	output bits	comment
CRC32	1975	32	not secure but common for checksums
MD5	1992	128	common; used to be secure (now broken)
SHA-1	1995	160	common; used to be secure (collision found in 2017)
SHA-2	2001	256/512	considered secure
SHA-3	2015	arbitrary	considered secure

- CRC is short for **Cyclic Redundancy Check**. It was designed for protection against common transmission errors, not as a cryptographic hash (for instance, CRC is a linear function).
- SHA is short for **Secure Hash Algorithm** and (like DES and AES) is a federal standard selected by NIST. SHA-2 is a family of 6 functions, including SHA-256 and SHA-512 as well as truncations of these.

  SHA-3 is not meant to replace SHA-2 but to provide a different alternative (especially following successful attacks on MD5, SHA-1 and other hash functions, NIST initiated an open competition for SHA-3 in 2007). SHA-3 is based on Keccak (like AES is based on Rijndael; Joan Daemen involved in both). Although the ouput of SHA-3 can be of arbitrary length, the number of security bits is as for SHA-2.

  https://en.wikipedia.org/wiki/NIST_hash_function_competition
- MD is short for **Message Digest**. These hash functions are due to Ron Rivest (MIT), the "R" in RSA. Collision attacks on MD5 can now produce collisions within seconds. For a practical exploit, see: https://en.wikipedia.org/wiki/Flame_(malware)

  MD6 was submitted as a candidate for SHA-3, but later withdrawn.

9.1 Constructions of hash functions

Recall that a hash function H is a function, which takes an input x of arbitrary length, and produces an output $H(x)$ of fixed length, say, b bit.

Example 187. (Merkle–Damgård construction) Similarly, a **compression function** $\tilde{H}$ takes input x of length $b+c$ bits, and produces output $\tilde{H}(x)$ of length b bits. From such a function, we can easily create a hash function H. How?

Importantly, it can be proved that, if $\tilde{H}$ is collision-resistant, then so is the hash function H.

Solution. Let x be an arbitrary input of any length. Let's write $x = x_1x_2x_3...x_n$, where each x_i is c bits (if necessary, pad the last block of x so that it can be broken into c bit pieces).

Set $h_1 = 0$ (or any other initial value), and define $h_{i+1} = \tilde{H}(h_i, x_i)$ for $i \geqslant 1$. Then, $H(x) = h_{n+1}$.

Comment. This construction is known as a Merkle–Damgård construction and is used in the design of many hash functions, including MD5 and SHA-1/2.

Careful padding. Some care needs to be applied to the padding. Just padding with zeroes would result in easy collisions (why?), which we would like to avoid. For more details:

https://en.wikipedia.org/wiki/Merkle-Damgård_construction

The construction of good hash algorithms is linked to the construction of good ciphers. Below, we indicate how to use a block cipher to construct a hash.

Why linked? The ciphertext produced by a good cipher should be indistinguishable from random bits. Similarly, the output of a cryptographic cipher should look random, because the presence of patterns would likely allow us to compute preimages or collisions.

However. The design goals for a hash are somewhat different than for a cipher. It is therefore usually advisable to not crossbreed these constructions and, instead, to use a specially designed hash like SHA-2 when a hash is needed for cryptographic purposes.

First, however, a cautionary example.

Example 188. (careful!) Let E_k be encryption using a block cipher (like AES). Is the compression function $\tilde{H}$ defined by

$$\tilde{H}(x,k) = E_k(x)$$

one-way?

Solution. No, it is not one-way.

Indeed, given y, we can produce many different (x,k) such that $\tilde{H}(x,k) = y$ or, equivalently, $E_k(x) = y$. Namely, pick any k, and then choose $x = D_k(y)$.

Example 189. Let E_k be encryption using a block cipher (like AES). Then the compression function $\tilde{H}$ defined by

$$\tilde{H}(x,k) = E_k(x) \oplus x$$

is usually expected to be collision-resistant.

Let us only briefly think about whether $\tilde{H}$ might have the weaker property of being one-way (as opposed to the previous example). For that, given y, we try to find (x,k) such that $\tilde{H}(x,k) = y$ or, equivalently, $E_k(x) \oplus x = y$. This seems difficult.

Just getting a feeling. We could try to find such (x,k) with $x = 0$. In that case, we need to arrange k such that $E_k(0) = y$. For a block cipher like AES, this seems difficult. In fact, we are trying a known-plaintext attack on the cipher here: assuming that $m = 0$ and $c = y$, we are trying to determine the key k. A good cipher is designed to resist such an attack, so that this approach is infeasible.

Comment. Combined with the Merkle–Damgård construction, you can therefore use AES to construct a hash function with 128 bits output size. However, as indicated before, it is advisable to use a hash function designed as such.

For several other (more careful) constructions of hash functions from block ciphers, you can check out Chapter 9.4.1 in the *Handbook of Applied Cryptography* (Menezes, van Oorschot and Vanstone, 2001), freely available at: http://cacr.uwaterloo.ca/hac/