

The output of a hash function  $h$  is known as the hash of the input. The hash of the sequence of codewords is mathematically expressed as  $h(c_1, c_2, \dots, c_n)$ .

**[0109]** Common classes of hash functions include hash functions based on block ciphers and hash functions with dedicated designs. Popular hash functions include SHA-1, MD5, RIPE-MD, HAVAL, and SNERFU, although any appropriate hash function  $h$  can be used. In one embodiment, a hash function  $h$  built around a block cipher is used (STEP 1370). In another embodiment, a hash function  $h$  which has a dedicated design is used (STEP 1370).

**[0110]** In some embodiments, the order of calculating offsets (STEP 1360) and hashing codewords (STEP 1370) is reversed. The hashing does not depend on the calculation of offsets. In other embodiments, calculating offsets (STEP 1360) and hashing codewords (STEP 1370) in FIG. 13 are not sequential, but instead occur with at least some overlap in time.

**[0111]** In one embodiment, the enrollment process accepts a secret pattern and saves a hash of the sequence of codewords  $h(c_1, c_2, \dots, c_n)$  as the stored representation of the visual password. In another embodiment, the enrollment process accepts a secret pattern and saves a hash of each codeword in the sequence of codewords  $\{h(c_1), h(c_2), \dots, h(c_n)\}$  as the stored representation of the visual password. In another embodiment, the enrollment process accepts a secret pattern and saves a sequence of offsets  $\{\delta_1, \delta_2, \dots, \delta_n\}$  as the stored representation of the visual password. Some embodiments of the enrollment process accept a secret pattern and save a hash and a sequence of offsets as the stored representation of the visual password, e.g.  $\{h(c_1, c_2, \dots, c_n), \delta_1, \delta_2, \dots, \delta_n\}$ . Some embodiments may save the hash of the sequence of codewords  $h(c_1, c_2, \dots, c_n)$  and the sequence of offsets  $\{\delta_1, \delta_2, \dots, \delta_n\}$  separately, but use the combination as the stored representation of the visual password. In embodiments that use the combination of the hash of the sequence of codewords  $h(c_1, c_2, \dots, c_n)$  and the sequence of offsets  $\{\delta_1, \delta_2, \dots, \delta_n\}$  as the stored representation of the visual password, but save only a portion thereof, the other portion of the visual password is transmitted for storage elsewhere. In some embodiments, the visual password is referred to as the blob  $y$ . The enrollment of a user may be considered to be the fuzzy commitment of a secret password.

**[0112]** Security

**[0113]** The security of the enrollment process of FIG. 13 depends on the visual password being both concealing and binding. For simplicity, the following discussions of security and resilience presume (a) that each value  $x$  is drawn uniformly at random from  $\{0, 1\}^n$ ; (b) an embodiment in which the visual password includes a hash of a sequence of codewords that includes a single codeword  $h(c_1)$ ; and (c) isometry of codeword neighborhoods. The concepts also apply to embodiments in which the visual password includes a hash of a sequence of codewords that include a plurality of codewords  $h(c_1, c_2, \dots, c_n)$ .

**[0114]** Given that an attacker is able to determine the sequence of codewords from a visual password whereby each codeword  $c$  is selected at random from the set of codewords  $C$  and the associated value  $x$  represents a random binary  $n$ -bit string, mathematically  $c \in_R C$  and  $x \in_R \{0, 1\}^n$ , in

time  $T$  with probability  $p(T)$ . It is then possible for the attacker to invert  $h(c_1)$  from a codeword  $z$  selected at random from the set of codewords  $C$ , mathematically  $z \in_R C$ , in time  $T$  with probability  $p(T)$ .

**[0115]** The time  $T$  and probability  $p(T)$  required to invert  $h(c_1)$  are evident from the following. Since each value  $x$  and associated codeword  $c_1$  are selected independently and uniformly at random, it is clear that the offset  $\delta$ , mathematically defined as  $\delta = x - c_1$ , reveals no information about the codeword  $c_1$ . Therefore, the task of an attacker in determining the sequence of codewords is equivalent to the task, given knowledge only of the hash of the sequence of codewords  $h(c_1)$ , of finding a codeword  $z$  selected from the set of codewords  $C$  such that the hash of the codeword  $z$  equals the hash of the sequence of codewords  $c_1$ , mathematically  $h(z) = h(c_1)$ . The underlying assumption in our derivation of the time  $T$  and probability  $p(T)$  required to invert  $h(c_1)$ , that it is hard to invert a hash, is somewhat non-standard. It is, however, consistent with common security assumptions about hash functions, such as those provided by the random oracle model. The same result is reached using more canonical assumptions.

**[0116]** Concealment

**[0117]** As explained with respect to FIGS. 11A and 11B, the amount of information about the value  $x$  contained in the associated codeword  $c$  depends on the size of the set of codewords  $C$ . In embodiments that use a hash of the sequence of codewords  $h(c_1)$ , the information about the value  $x$  in the associated codeword  $c_1$  represents concealed information. That is, the information about the value  $x$  in the associated codeword  $c_1$  is concealed by the hash function  $f$ . Accordingly, the larger the set of codewords  $C$ , the more information about the value  $x$  is concealed. In an embodiment in which the visual password includes a hash of a sequence of codewords  $h(c_1)$ , the amount of information about the value  $x$  that is contained in the associated codeword  $c_1$  determines the level of concealment in the visual password.

**[0118]** Since error-correcting codes require that a binary set of codewords  $C$  contain  $2^k$  codewords,  $k$  describes the size of a set of codewords  $C$ . Thus, a higher  $k$  represents more codewords and more concealed information about the value  $x$ . Effectively,  $k$  is the parameter that dictates the level of concealment of the representation of the visual password. For most applications, a  $k$  value of eighty should provide an adequate level of security. Under common assumptions about hash functions, such as the random oracle model, this security level will require an attacker seeking to match a codeword  $c_1$  in the visual password an average of  $2^{79}$  hash function  $h$  computations. This number of calculations is comparable to the computational effort required to factor RSA-1024 or find a collision in SHA-1.

**[0119]** Note that in some embodiments that do not conceal the sequence of codewords itself, the codewords must still be drawn from a large set of codewords  $C$  in order to conceal each associated value  $x$ . If the set of codewords  $C$ , as described by  $k$ , is small, then an attacker can guess a codeword  $c_1$  and extract the value  $x$  from the visual password.

**[0120]** Binding

**[0121]** A password is conventionally defined as binding if it is infeasible for any polynomially bounded player to