# Subset-Sum Hash Specification

Yossi Gilad, David Lazar, and Chris Peikert

Algorand, Inc.

September 3, 2021

## 1 Notation and Background

All logarithms are base two, i.e.,  $\log = \log_2$ , unless otherwise indicated by a different subscript.

**Binary strings.** For a positive integer L,  $\{0,1\}^{< L}$  denotes the set of binary strings of length strictly less than L (including the zero-length empty string  $\varepsilon$ ), and  $\{0,1\}^*$  denotes the set of all binary strings of any length. For binary strings u,v of any length, uv denotes their concatenation. For a binary string  $x \in \{0,1\}^*$ ,  $|x| \ge 0$  denotes its length in bits. For a positive integer e and a non-negative integer  $z < 2^e$ ,  $\langle z \rangle_e \in \{0,1\}^e$  denotes the little-endian representation of z as a binary string of exactly e bits.

**Modular integers.** For a positive integer q,  $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$  denotes the (abelian) ring of integers modulo q. Formally,  $\mathbb{Z}_q$  is the set of *cosets* of the form

$$c = \tilde{c} + q\mathbb{Z} := \{\dots, \tilde{c} - 2q, \tilde{c} - q, \tilde{c}, \tilde{c} + q, \tilde{c} + 2q, \dots\}$$

for some integer  $\tilde{c} \in \mathbb{Z}$ , with  $\tilde{c} + q\mathbb{Z} = \tilde{c}' + q\mathbb{Z}$  if and only if q divides  $\tilde{c} - \tilde{c}'$ . In general, a coset can be represented by any of its elements, which might even differ from one location to the next. To avoid any ambiguity, this specification always requires a coset  $c \in \mathbb{Z}_q$  to be externally read and written using its (unique) distinguished representative  $\bar{c} \in \{0, 1, \dots, q-1\} \cap c$ .

**Vectors and matrices.** Vectors are denoted by lower-case bold letters (e.g.,  $\mathbf{a}$ ), and matrices by upper-case bold letters (e.g.,  $\mathbf{A}$ ). In this specification, vector and matrix entries are always indexed starting from zero. A vector's *i*th entry is denoted by the same lower-case letter, but without boldface, and with a subscript i; e.g.,  $x_i$  is the *i*th entry of  $\mathbf{x}$ . Similarly, the *i*th column of a matrix is denoted by the same letter, but in lower case, and with a subscript i; e.g.,  $\mathbf{a}_i$  is the *i*th column of  $\mathbf{A}$ .

The set of n-dimensional vectors over a set X is denoted  $X^n$ . In particular,  $\mathbb{Z}_q^n$  is an (abelian) additive group, where the group operation is coordinate-wise addition (modulo q). Similarly,  $X^{n\times m}$  denotes the set of n-by-m-dimensional matrices over X. For convenience, this specification sometimes uses standard vector and matrix operations (like sums and products) where they are well defined.

# 2 Compression Function Family

This section gives the mathematical definition of the subset-sum compression function family, which was first studied and popularized in early works like [IN89, Ajt96]. For specific parameters, the concrete security of the function against various kinds of (classical and quantum) cryptanalytic attacks is analyzed in a separate work.

#### 2.1 Parameters

The subset-sum compression function family is parameterized by:

- a positive integer modulus  $q \in \mathbb{N}$  (often taken to be a power of two);
- a positive integer dimension  $n \in \mathbb{N}$ ;
- a positive integer input length  $m \in \mathbb{N}$ , where  $m > n \log q$ .

#### 2.2 Function Definition

The compression function family for parameters q, n, m is defined as the collection

$$\mathcal{F}_{q,n,m} := \{ f_{\mathbf{A}} \colon \{0,1\}^m \to \mathbb{Z}_q^n : \mathbf{A} \in \mathbb{Z}_q^{n \times m} \},$$

where each function  $f_{\mathbf{A}} \in \mathcal{F}_{q,n,m}$  is defined as

$$f_{\mathbf{A}}(\mathbf{x}) := \mathbf{A}\mathbf{x} = \sum_{i=1}^{m} x_i \cdot \mathbf{a}_i = \sum_{i:x_i=1} \mathbf{a}_i \in \mathbb{Z}_q^n.$$
 (2.1)

The latter summation explains the name "subset-sum hash": the output is the subset-sum of the columns of A indicated by the bits of the input x.

Observe that the condition  $m > n \log q$  ensures that the functions in the family are *compressing*, i.e., the cardinality of their common domain  $\{0,1\}^m$  is strictly larger than that of their common range  $\mathbb{Z}_q^n$ :  $2^m > 2^{n \log q} = q^n$ .

When  $q=2^u$  is a power of two,  $\ell:=n\log q=nu$  is called the *output length*, and  $b:=m-\ell>0$  is called the *block length*. In this case, for any  $\mathbf{y}\in\mathbb{Z}_q^n$ , let  $\langle\mathbf{y}\rangle\in\{0,1\}^\ell$  denote the representation of  $\mathbf{y}$  as an  $\ell$ -bit string, obtained as the concatenation of the (u-bit, little-endian representations of the) distinguished representatives  $\bar{y}_i\in\{0,1,\ldots,q-1\}$  of the coordinates  $y_i$ :

$$\langle \mathbf{y} \rangle := \langle \bar{y}_0 \rangle_u \langle \bar{y}_1 \rangle_u \cdots \langle \bar{y}_{n-1} \rangle_u \in \{0, 1\}^{\ell}.$$
 (2.2)

#### 2.3 Security Properties

**Conjectured properties.** For appropriate parameters, the subset-sum compression function family  $\mathcal{F}_{q,n,m}$  is conjectured to have the following security properties:

- Uninvertibility (UI): given uniformly random and independent  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and  $\mathbf{y} \in \mathbb{Z}_q^n$ , it is infeasible to find some  $\mathbf{x} \in \{0,1\}^m$  such that  $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} = \mathbf{y}$ .
- One-wayness (OW): given  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and  $\mathbf{y} = f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} \in \mathbb{Z}_q^n$  (but not  $\mathbf{x}$  itself), where  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and  $\mathbf{x} \in \{0,1\}^m$  are uniformly random and independent, it is infeasible to find some  $\mathbf{x}' \in \{0,1\}^m$  (not necessarily different from  $\mathbf{x}$ ) such that  $f_{\mathbf{A}}(\mathbf{x}') = \mathbf{y}$ .

- Second-preimage resistance (SPR): given uniformly random and independent  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and  $\mathbf{x} \in \{0,1\}^m$ , it is infeasible to find some  $\mathbf{x}' \in \{0,1\}^m$  such that  $\mathbf{x}' \neq \mathbf{x}$  and  $f_{\mathbf{A}}(\mathbf{x}') = f_{\mathbf{A}}(\mathbf{x})$ .
- Target-collision resistance (TCR): it is infeasible to choose some  $\mathbf{x} \in \{0,1\}^m$  and then, given a uniformly random  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ , to find some distinct  $\mathbf{x}' \in \{0,1\}^m \setminus \{\mathbf{x}\}$  such that  $f_{\mathbf{A}}(\mathbf{x}) = f_{\mathbf{A}}(\mathbf{x}')$ , i.e.,  $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}' \in \mathbb{Z}_q^n$ .
- Collision resistance (CR): given a uniformly random  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ , it is infeasible to find distinct  $\mathbf{x}, \mathbf{x}' \in \{0,1\}^m$  such that  $f_{\mathbf{A}}(\mathbf{x}) = f_{\mathbf{A}}(\mathbf{x}')$ , i.e.,  $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}' \in \mathbb{Z}_q^n$ .

Note that breaking CR is equivalent to finding a nonzero  $\mathbf{z} \in \{-1, 0, 1\}^m \setminus \{\mathbf{0}\}$  such that  $\mathbf{Az} = \mathbf{0} \in \mathbb{Z}_q^n$ . In one direction, given such  $\mathbf{x}, \mathbf{x}'$ , define  $\mathbf{z} = \mathbf{x} - \mathbf{x}' \in \{-1, 0, 1\}^m \setminus \{\mathbf{0}\}$  and observe that  $\mathbf{Az} = \mathbf{Ax} - \mathbf{Ax}' = \mathbf{0}$ . In the other direction, given such  $\mathbf{z}$ , let  $\mathbf{x} \in \{0, 1\}^m$  be 1 (respectively, 0) wherever  $\mathbf{z}$  is 1 (resp., 0 or -1), and similarly let  $\mathbf{x}' \in \{0, 1\}^m$  be 1 (respectively, 0) wherever  $\mathbf{z}$  is -1 (resp., 0 or 1). Then  $\mathbf{z} = \mathbf{x} - \mathbf{x}'$ , and since  $\mathbf{0} = \mathbf{Az} = \mathbf{A}(\mathbf{x} - \mathbf{x}')$ , we have  $\mathbf{Ax} = \mathbf{Ax}'$ , as desired.

It is well known that CR tightly implies TCR generically (i.e., for any compression function family), because breaking TCR immediately yields a collision. Moreover, TCR tightly implies SPR generically, because being able to find a collision with a uniformly random input (that is independent of  $\bf A$ ) in particular means being able to find a collision with an input of one's choice (since that input may just be chosen uniformly). Furthermore, for parameters that yield significant compression (i.e., for  $m \gg n \log q$ ), it is well known that SPR tightly implies OW generically, and that OW implies UI for the subset-sum family.

While none of the converse directions is known to hold generically, for the subset-sum family with significant compression, a *relaxed* form of UI implies CR, thus implying that all four security properties are closely related. Specifically, any attack that breaks CR with probability  $\delta$  can be used to successfully attack relaxed-UI in essentially the same amount of time and with probability  $\approx \delta/(2m)$ , where in relaxed-UI the output x may be taken from  $\{-1,0,1\}^m$  (it is not limited to  $\{0,1\}^m$ ).

**Non-conjectured properties.** On the other hand, the family  $\mathcal{F}_{q,n,m}$  is *not conjectured to have*, or is even *known not to have*, the following security properties:

• Unpredictability/Pseudorandomness: outputs of  $f_{\mathbf{A}}$  on different inputs do not appear random or uncorrelated, even if parts of the corresponding inputs are unknown to the attacker. This is because  $f_{\mathbf{A}}$  is linear:

$$f_{\mathbf{A}}(\mathbf{x} + \mathbf{x}') = \mathbf{A}(\mathbf{x} + \mathbf{x}') = \mathbf{A}\mathbf{x} + \mathbf{A}\mathbf{x}' = f_{\mathbf{A}}(\mathbf{x}) + f_{\mathbf{A}}(\mathbf{x}'),$$

as long as  $\mathbf{x}, \mathbf{x}', \mathbf{x} + \mathbf{x}' \in \{0, 1\}^m$ , which is easy to arrange in many contexts. In particular, any 0 bits of  $\mathbf{x}$  can be changed to 1s by adding an  $\mathbf{x}'$  that is 1 in the suitable position(s), and 0 wherever  $\mathbf{x}$  is 1.

• Random oracle: for the same reasons as above,  $f_{\mathbf{A}}$  does not "behave like a random oracle" in many of the ways that are typically expected. Therefore, it should not be used to instantiate a random oracle in any cryptosystems or protocols.

# 3 Hashing Arbitrary-Length Messages

The compression functions defined in Section 2 map a *fixed-length* m-bit input to an output of length  $\ell := n \log q < m$ . As is standard, a message of *arbitrary* length is hashed to a fixed-length output by invoking the compression function one or more times, using the Merkle–Damgård (MD) transform [Mer89, Dam89].

### 3.1 Padding

The transform uses the following padding method, which maps a binary string of any bounded length into a strictly longer one whose length is a multiple of the block length  $b:=m-\ell>0$ . Formally, for any positive integer e, define the padding function  $\operatorname{pad}_{b,e}\colon\{0,1\}^{<2^e}\to\{0,1\}^*$  as

$$pad_{be}(x) = x10^{z} \langle |x| \rangle_{e}, \tag{3.1}$$

where  $z \ge 0$  is the smallest non-negative integer for which |x| + 1 + z + e is a multiple of b. That is,  $r := b \cdot \lceil k/b \rceil - k \in \{0, 1, \dots, b-1\}$ , where k = |x| + 1 + e. Note especially that  $\operatorname{pad}_{b,e}(x)$  unconditionally appends at least 1 + e bits to x, even if |x| itself is a multiple of b.

The purpose of the padding function is to produce a string whose length is a multiple of the block length, and so that the MD transform preserves collision resistance. That is, any collision in the full hash function immediately yields a collision in the underlying compression function.

#### 3.2 Hash Functions

Fix subset-sum parameters q, n, m where  $q = 2^u$  is a power of two for some positive integer u. Let  $\ell := n \log q = nu$  be the output length in bits, and let  $b := m - \ell > 0$  be the block length.

This specification defines two functions on arbitrary-length input strings: an "unsalted" mode that takes only the string as input, and a "salted" mode, originally defined in [HK06], that additionally takes a public salt value (which should typically be chosen uniformly at random, or pseudorandomly).

#### 3.2.1 Unsalted Mode

For any matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  defining the compression function  $f_{\mathbf{A}} \colon \{0,1\}^m \to \mathbb{Z}_q^n$  (Section 2) and a positive integer e > 0, define the unsalted hash function

$$H_{\mathbf{A},e} \colon \{0,1\}^{<2^e} \to \{0,1\}^{\ell}$$
 $H_{\mathbf{A},e}(x) := H'_{\mathbf{A}}(0^{\ell}, \operatorname{pad}_{b,e}(x)),$  (3.2)

where the chaining function  $H'_{\mathbf{A}} \colon \{0,1\}^\ell \times \left(\bigcup_{i=0}^\infty \{0,1\}^{ib}\right) \to \{0,1\}^\ell$  is defined as

$$H'_{\mathbf{A}}(h, w) := \begin{cases} h & \text{if } w = \varepsilon \\ H'_{\mathbf{A}}(\langle f_{\mathbf{A}}(hu) \rangle, v) & \text{where } w = uv \text{ for } u \in \{0, 1\}^b. \end{cases}$$
(3.3)

(Recall that representation  $\langle \mathbf{y} \rangle \in \{0,1\}^{\ell}$  for  $\mathbf{y} \in \mathbb{Z}_q^n$  is defined in Section 2.2.)

#### 3.2.2 Salted Mode

For any matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  defining the compression function  $f_{\mathbf{A}} \colon \{0,1\}^m \to \mathbb{Z}_q^n$  (Section 2) and a positive integer e > 0, define the salted hash function

$$\tilde{H}_{\mathbf{A},e} \colon \{0,1\}^{<2^e} \underbrace{\times \{0,1\}^b} \to \{0,1\}^\ell$$

$$\tilde{H}_{\mathbf{A},e}(x,r) := \tilde{H}'_{\mathbf{A}}(0^\ell, \operatorname{pad}_{b,e}(0^b, x), r)$$
(3.4)

where the chaining function  $\tilde{H}'_{\mathbf{A}}$ :  $\{0,1\}^\ell \times \left(\bigcup_{i=0}^\infty \{0,1\}^{ib}\right) \times \{0,1\}^b \to \{0,1\}^\ell$  is defined as

$$\tilde{H}'_{\mathbf{A}}(h,w,\underline{r}) := \begin{cases} h & \text{if } w = \varepsilon \\ \tilde{H}'_{\mathbf{A}}(\langle f_{\mathbf{A}}(h\underline{\tilde{u}})\rangle,v,\underline{r}) & \text{where } w = uv \text{ for } u \in \{0,1\}^b \underbrace{\text{and } \tilde{u} = u \oplus r}. \end{cases}$$
(3.5)

The only differences between the salted-mode functions  $\tilde{H}_{\mathbf{A},e}$ ,  $\tilde{H}'_{\mathbf{A}}$  and their corresponding unsalted-mode functions  $H_{\mathbf{A},e}$ ,  $H'_{\mathbf{A}}$  are depicted above with wavy underlines, and are as follows:

- 1. Each function  $\tilde{H}_{\mathbf{A},e}, \tilde{H}'_{\mathbf{A}}$  takes an additional salt input  $r \in \{0,1\}^b$ , whose length is the block length.
- 2. The function  $\tilde{H}_{\mathbf{A},e}(x,r)$  prepends an all-zeros block  $0^b \in \{0,1\}^b$  to the input x (before padding it using  $\mathrm{pad}_{b,e}$  and inputting the result to the appropriate chaining function, as in the unsalted case). Note that this prepended zeros block counts toward the total input length in the padding function.
- 3. The chaining function  $\tilde{H}'_{\mathbf{A}}$  XORs each block  $u \in \{0,1\}^b$  of the padded input—including the initial, prepended all-zeros block—with the salt value  $r \in \{0,1\}^b$  (before inputting the result together with the previous chaining value h to the compression function, as in the unsalted case).

### 4 Concrete Instantiations

The fully specified hash functions are merely instantiations of the unsalted function  $H_{\mathbf{A},e}$  (Equation (3.2)) and salted function  $\tilde{H}_{\mathbf{A},e}$  (Equation (3.4)) for specific subset-sum parameters q,n,m, matrices  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ , and input-length representation bit lengths e.

## 4.1 Deriving A

It is well known that for typical parameters, it is easy to generate a matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  that is indistinguishable from uniformly random, together with some *known collisions* in the associated subset-sum compression function  $f_{\mathbf{A}}$ . This may allow the party that generated  $\mathbf{A}$  to violate the security of constructions that use this function. Therefore, it is very important to generate a random-looking  $\mathbf{A}$  in a "nothing up my sleeve" manner that is highly unlikely to admit any such backdoor.

Let  $q=2^u, n, m$  be subset-sum parameters where u is a positive integer, and let XOF:  $\{0,1\}^* \to \{0,1\}^\infty$  represent a suitable cryptographic *extendable-output function*, such as SHAKE-256. Then for any identifier  $id \in \{0,1\}^*$ , define the matrix  $\mathbf{A}_{\text{XOF},id} \in \mathbb{Z}_q^{n \times m}$  as:

$$\mathbf{A}_{\mathrm{XOF},id} := \mathrm{pack}_{u,n,m} \big( \mathrm{XOF}(\langle u \rangle_{16} \, \langle n \rangle_{16} \, \langle m \rangle_{16} \, id)_{0,\dots,unm-1} \big) \in \mathbb{Z}_q^{n \times m}, \tag{4.1}$$

where  $\operatorname{pack}_{u,n,m}\colon\{0,1\}^{unm}\to\mathbb{Z}_q^{n\times m}$  constructs its output matrix from its input string in row-major order, using u bits per entry. That is, for all  $i=0,\ldots,n-1$  and  $j=0,\ldots,m-1$ , the distinguished representative of the (i,j)th entry  $a_{i,j}\in\mathbb{Z}_q$  of  $\mathbf{A}=\operatorname{pack}_{u,n,m}(w)$  has binary representation

$$\langle \bar{a}_{i,j} \rangle_u = w_{u(im+j),\dots,u(im+j)+(u-1)}$$
.

#### 4.2 Concrete Parameters

The implemented functions are (unsalted)  $H:=H_{\mathbf{A},e}$  and (salted)  $\tilde{H}:=\tilde{H}_{\mathbf{A},e}$ , where:

- the modulus  $q = 2^{64}$ , so  $u = \log q = 64 = 2^6$ ;
- the dimension  $n=8=2^3$ , so the output length is  $\ell=n\log q=512=2^9$ ;
- the input length  $m = 1024 = 2^{10}$ , so the block length is  $b = m \ell = 512 = 2^9$ ;
- the representation length (of the hash input length) is  $e = 128 = 2^7$ ;
- the extendable-output function is XOF = SHAKE-256;
- the matrix is  $A := A_{XOF,id}$  where id is the ASCII representation of Algorand.

### References

- [Ajt96] M. Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.
- [Dam89] I. Damgård. A design principle for hash functions. In CRYPTO, pages 416–427. 1989.
- [HK06] S. Halevi and H. Krawczyk. Strengthening digital signatures via randomized hashing. In *CRYPTO*, pages 41–59. 2006.
- [IN89] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *J. Cryptology*, 9(4):199–216, 1996. Preliminary version in FOCS 1989.
- [Mer89] R. C. Merkle. A certified digital signature. In CRYPTO, pages 218–238. 1989.