Smart Contracts and Blockchain Technology

Lecture 6. Cryptographic underpinnnings

Christian Ewerhart

University of Zurich

Fall 2022

Introduction and overview

Last lecture

- Equilibrium in the blockchain game
- Selfish mining

This lecture: Cryptographic underpinnings

- Binary and hexadecimal numbers
- Hash functions
- Private and public keys
- Finite fields
- Discrete logarithm problem

Cryptographic underpinnings (1)

Basics on binary and hexadecimal numbers

A **binary number** is a technical representation of a number based on powers of 2 (rather than 10 which is used for decimal numbers).

Each **binary digit** is taken from the set $\{\texttt{0},\texttt{1}\}$.

Example: `0b10100011` $= 1 \cdot 2^7 + 1 \cdot 2^5 + 1 \cdot 2^1 + 1 \cdot 2^0 = 163$, where the **prefix** `0b` indicates a binary number.

Bits are the units of information. For instance, a die with six faces bears information

$$I = \log_2(6) = \frac{\ln 6}{\ln 2} = 2.585 \text{ bit},$$

i.e., somewhat less than three bits.

Cryptographic underpinnings (2)

Basics on binary and hexadecimal numbers (continued)

A **hexadecimal number** is, in complete analogy, a technical representation of a number based on powers of 16.

Each **hexadecimal digit** is taken from the set $\{\texttt{0},\texttt{1},\texttt{2},\ldots,\texttt{9},\texttt{a},\ldots,\texttt{f}\}$.

Example: $\texttt{0xc03} = 12 \cdot 16^2 + 0 \cdot 16^1 + 3 \cdot 16^0 = 3075$, where the **prefix** `0x` indicates a hexadecimal number.

Cryptographic underpinnings (3)

Basics on binary and hexadecimal numbers (continued)

Hexadecimal numbers are useful because they are both compact and easily transformed into binary numbers:

- `0x0`=`0b0000`,
- `0x1`=`0b0001`,
- `0x2`=`0b0010`,
- ...
- `0xf`=`0b1111`.

Two hexadecimal numbers correspond to **one byte** (8 bits), which has been, in particular, the word length of the first commerically successful microprocessors (e.g., Intel 8080, Motorola 6800).

Cryptographic underpinnings (4)

Cryptography is a branch of mathematics used extensively in computer security.

The term means "secret writing" in Greek, and refers originally to the encryption of messages.

However, cryptographic methods are also used:

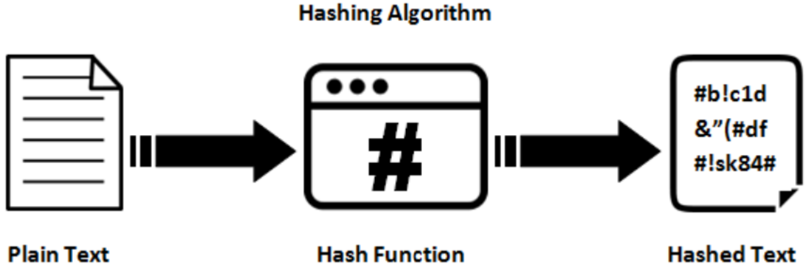
- to identify, and prove the integrity of, data (with a **digital fingerprint**, also known as hash),
- to prove the authenticity of a message (e.g., with a **digital signature**).

These methods are critical, in particular, to the operation of blockchain systems and smart contract applications.

Cryptographic underpinnings (5)

Digital fingerprints

Definition. *A* **hash function** *maps data of arbitrary size to data of fixed size.*



Cryptographic underpinnings (6)

Hash functions on bit strings

Denote by $\mathcal{A} = \{0, 1\}$ the **alphabet** consisting of the set of binary values, and by

$$\mathcal{A}^* = \{\,\text{``''},\ \text{``0''},\ \text{``1''},\ \text{``00''},\ \text{``01''},\ \text{``10''},\ \text{``11''},\ \text{``000''},\ \text{``001''},\ \dots\,\}$$

the set of **words** over $\mathcal{A}$. Then, a hash function is any mapping

$$\psi : \mathcal{A}^* \to \mathcal{A}^N = \underbrace{\mathcal{A} \times \ldots \times \mathcal{A}}_{N \text{ times}}, \tag{1}$$

where $N \geq 1$ is the length of the hash.

Cryptographic underpinnings (7)

Use cases of digital fingerprints

Data integrity. Comparing hash values can determine whether any changes have been made to a given data set.

Proof of work. Winning the block reward requires finding a hash with given properties.

Password verification. To prevent password theft, only a hash of a user password is communicated to the backend system.

Cryptographic underpinnings (8)

Check bits

For a word $w \in \mathcal{A}^*$, denote by

$$\psi(w) = \begin{cases} 0 & \text{if the number of 1's in } w \text{ is even} \\ 1 & \text{if the number of 1's in } w \text{ is odd} \end{cases} \tag{2}$$

the **parity bit**.

For example, $\psi(\text{“1010111”}) = 1$.

Examples

- American Standard Code for Information Interchange (**ASCII**): 7 bits for the code plus one parity bit, corresponding to one byte (8 bits)

Cryptographic underpinnings (9)

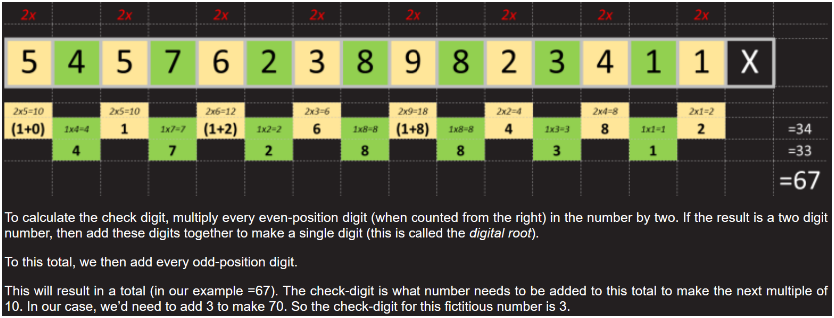
USASCII code chart

b7 b6 b5 →					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4	b3	b2	b1	Row ↓ / Column →	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	–	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	—	o	DEL

The **eigth bit** was used to check if the data was correct, e.g., on a punched tape for printer data.

Cryptographic underpinnings (10)

The Luhn Algorithm for credit card numbers



Also, payment operators use simple methods for payment card identification:

$$\psi(\text{“4362 3245 2314 0012”}) = \text{“**** **** **** *012”}$$

Cryptographic underpinnings (11)

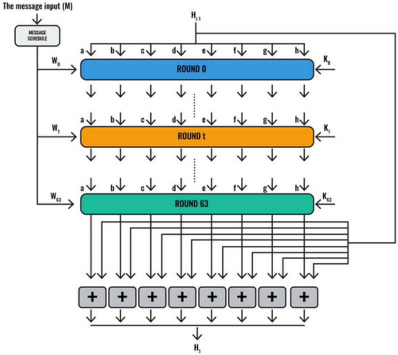
Properties of hash functions used in blockchain systems

- **Determinism.** A given input message always produces the same hash output.
- **Verifiability.** Computing the hash of a message is efficient (linear complexity).
- **Noncorrelation.** A small change in the message (e.g., a 1-bit change) should change the hash output so extensively that it cannot be correlated to the hash of the original message.
- **Irreversability.** Computing the message from its hash is infeasible, equivalent to a brute-force search through all possible messages.
- **Collision protection.** It should be infeasible to calculate two different messages that produce the same hash output.

Cryptographic underpinnings (12)

Secure Hash Algorithm

The **SHA-256** falls into the class SHA-2, which was created by the United States National Security Agency (NSA) as a successor to SHA-1 in 2001.[1]



[1] Irrespective of the size of input data, the hash will always consist of **256 bits**.

Cryptographic underpinnings (13)

Private keys

Definition. *A* **private key** *is an element of a finite set of feasible keys,* $\pi \in \Pi = \{\pi_1, \ldots, \pi_N\}$*, where* $N = \#\Pi$ *denotes the cardinality of the set of feasible keys.*

Cryptographic underpinnings (14)

Examples of private keys

PIN for payment card (4 digits):
$\Pi = \{\text{“0000”}, \text{“0001”}, \cdots, \text{“9999”}\}$, with $N = 10^4 = 10'000$.

Phone PIN (between 4 and 6 digits):
$\Pi = \{\text{“0000”}, \cdots, \text{“9999”}\} \cup \{\text{“00000”}, \cdots, \text{“99999”}\} \cup \{\text{“000000”}, \cdots, \text{“999999”}\}$, with $N = 10^4 + 10^5 + 10^6 = 1'110'000$.

Alphanumeric password (e.g., between 8 and 12 symbols, at least one upper-case and one-lower case letter, at least one symbol): “#qwerty123”, with $N \approx 10^{24}$.

Cryptographic underpinnings (15)

Risks of private keys

Possession of a private key is the root of user control. Therefore, holders of private keys are exposed to the following risks:

- **Private key loss.** If a private key is lost, it cannot be recovered and control may be lost forever. Therefore, a private key must be backed up and protected from accidental loss.
- **Private key compromise.** A private key must remain secret at all times. Revealing it to a third party is equivalent to sharing control (then, it may not be feasible to disentangle who authenticated a transaction).
    - The key has been revealed to an unauthorized party (stolen key)
    - The key *may* have been revealed to an unauthorized party (uncertainty)
    - An attacker has identified the private key by guesswork and/or trial-and-error techniques.

Cryptographic underpinnings (16)

Most common passwords

Rank	2021
1	123456
2	123456789
3	12345
4	qwerty
5	password
6	12345678
7	111111
8	123123
9	1234567890
10	1234567

Source: nordpass.com.

Cryptographic underpinnings (17)

Generation of a private key

- **Offline** and **not chosen by a human**
- **Secure source of randomness**, e.g.:
  - Mouse-wiggling
  - Cosmic radiation noise from microphone channel
  - Quantum random generator
- Pseudo random number generator must be **cryptographically secure** (CSPRNG)[2]

[2] CSPRNG requirements fall into two groups: (i) they pass statistical randomness tests, (2) they hold up well under serious attack, even when part of their initial or running state becomes available to an attacker.

Cryptographic underpinnings (18)

Ethereum private keys

A private key in the Ethereum system is a binary number with 256 digits (corresponding to a hexadecimal number with 64 digits).

Example:

`0xe3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855`

$N = 16^{64} = 2^{256} \approx 1.1579 \cdot 10^{77}$ (the estimated number of atoms in the visible universe is 10^{80}).

Cryptographic underpinnings (19)

Public keys

The public key is generated from the private key using a function that is easy to compute but practically irreversible. Such functions are called **one-way functions** or **trapdoor functions**.

Examples:

- Exponentiation in a finite field (discrete logarithm problem)
- Scalar multiplication on elliptic curves

Note: Also hash functions are one-way functions. However, in contrast to the functions considered here, they take data of arbitrary size and lack the mathematical properties (homomorphy) needed to run digital signature protocols.

Cryptographic underpinnings (20)

Finite fields

Finite fields are finite mathematical structures that are the basis for cryptographic proofs. An important example are **residue class fields with prime characteristic**.[3]

Example. The residue class field $\mathbb{F}_7$ consists of

- the set of residue classes $\{0, 1, 2, 3, 4, 5, 6\}$
- the operation of addition modulo 7, e.g.,

$$2 + 6 \equiv 1 \bmod 7 \tag{3}$$

- the operation of multiplication modulo 7, e.g.,

$$5 \cdot 4 \equiv 6 \bmod 7. \tag{4}$$

[3] For any prime power p^m, there exists precisely one finite field. Residue class fields (where $m = 1$) are of an elementary nature.

Cryptographic underpinnings (21)

Primitive roots

Example. In the residue class field $\mathbb{F}_7$, consider the powers of 3:

$$\begin{aligned} 3^0 &\equiv 1 \bmod 7 \\ 3^1 &\equiv 3 \bmod 7 \\ 3^2 &\equiv 2 \bmod 7 \\ 3^3 &\equiv 6 \bmod 7 \\ 3^4 &\equiv 4 \bmod 7 \\ 3^5 &\equiv 5 \bmod 7 \end{aligned}$$

Note that each nonzero residue class corresponds to some power of 3 (this property makes 3 a **primitive root** or **generator** of the multiplicative group $\mathbb{F}_7^* = \mathbb{F}_7 \backslash \{0\}$).

Cryptographic underpinnings (22)

Table of primitive roots

Analogously, one defines $\mathbb{F}_p$ for any prime number $p \in \{2, 3, 5, 7, 11, 13, 17, 19, 23, \ldots\}$.

p	Primitivwurzeln modulo p
2	1
3	2
5	2,3
7	3,5
11	2,6,7,8
13	2,6,7,11
17	3,5,6,7,10,11,12,14
19	2,3,10,13,14,15
23	5,7,10,11,14,15,17,19,20,21

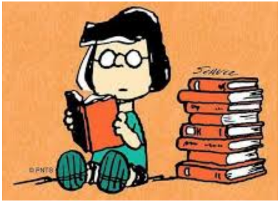
Cryptographic underpinnings (23)

The discrete logarithm problem

The **discrete logarithm problem** is to find, for a given residue class r, the exponent m such that $g^m \equiv r \bmod p$.

For large primes p, this can be a very difficult problem.

This chapter is loosely based on Antonopoulos and Wood (2018, Ch. 4).



Cryptographic underpinnings (25)

References

Antonopoulos, A.M., Wood, G. (2018), *Mastering Ethereum: Building Smart Contracts and Dapps*, O'Reilly Media.