HYPERLEDGER

EB 20/21	Enterprise Blockchain Technologies	Number:	2
Module I - Introduction		Issue Date:	
Background: Cryptography & Security		Due Date:	

Preliminary Notes

This class provides background about cryptography and security. This laboratory is based on several sources [2, 3, 5].

1 Cryptography and Security

The goal of Cryptography is to allow communication between people or computers in a way that others cannot understand that communication. Whereas having a single dialect could serve that purpose, it is not scalable nor safe.

Cryptography is much older than computers, dated from as early as 400 BC, where Spartans used a cipher device called *scytale* for military communication [4]. A *cipher* is an algorithm that is used to obfuscate information (or to encrypt information). Information is *encrypted* using a key as the parameter of the cipher. This allows the algorithm to be public. Historical ciphers are, for example, the Caesar cipher[1], the substitution cipher[2], the Vigener cipher[3].

Let us define notation to refer to ciphers and messages and their keys.

Definition 1.1. Cryptogram C

$C = E_k(M)$, where C is the cryptogram (or encrypted message), E_k is the key used for the encryption, and M is the original message.

Definition 1.2. Decrypted Message M

$M = E_k^{-1}(C)$, where M is the decrypted message, K^{-1} is the inverse key, and C is the cryptogram.

In the context of cryptography, often, techniques are separated into symmetric and asymmetric cryptography.

1.1 Symmetric Cryptography

In *symmetric cryptography*, a single key is used to cipher and decipher the message ($K = K^{-1}$). Figure 1 illustrates communication between Alice and Bob.

There are some problems in practice: the party generating the key must deliver it in a confidential way. The party receiving the key must assure the key received is the key intended to be sent, and it should be able to confirm it is the most recent. Moreover, there needs to be a key, for each pair of parties.

Typical examples of symmetric cryptography include the TEA algorithm, the Data Encryption Standard (DES), Triple DES, Twofish, and the AES.

[1] https://cryptii.com/pipes/caesar-cipher
[2] https://planetcalc.com/7984/
[3] https://www.dcode.fr/vigenere-cipher

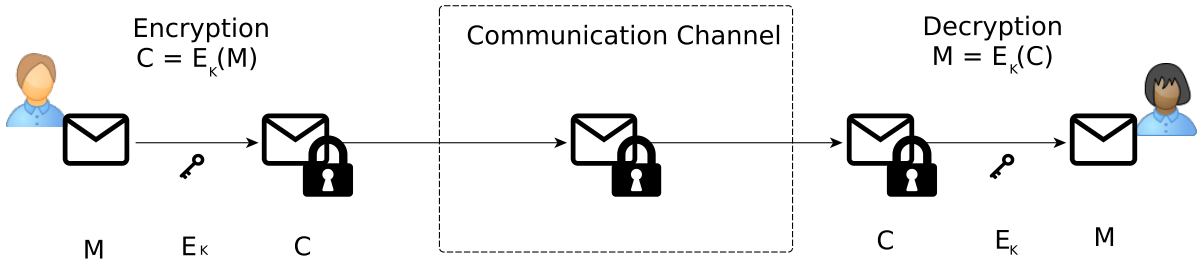

Figure 1: Symmetric cryptography. Alice sends a message to Bob. $K = K^{-1}$

1.2 Asymmetric Cryptography

In *asymmetric cryptography*, there are two keys: one is used for cipher, and another to decipher. Normally, we call *public key* to the key that ciphers (K_u), and *private key* to the key that deciphers (K_r). We remark that it is also possible to cipher with the private key and decipher with the public key. Figure 2 illustrates this process.

Contrarily to symmetric key cryptography, there is no need to assure the confidentiality of the public key. The party that wants to communicate needs to publish the public key (assuring its integrity and authenticity). There is no need to distribute a pair of keys per pair of parties.

Many ciphers, such as *RSA*, are not considered *totally safe*, as an attacker can discover the key. However, RSA is considered *practically safe*, given that there is a need for a substantial computational investment to find the key.

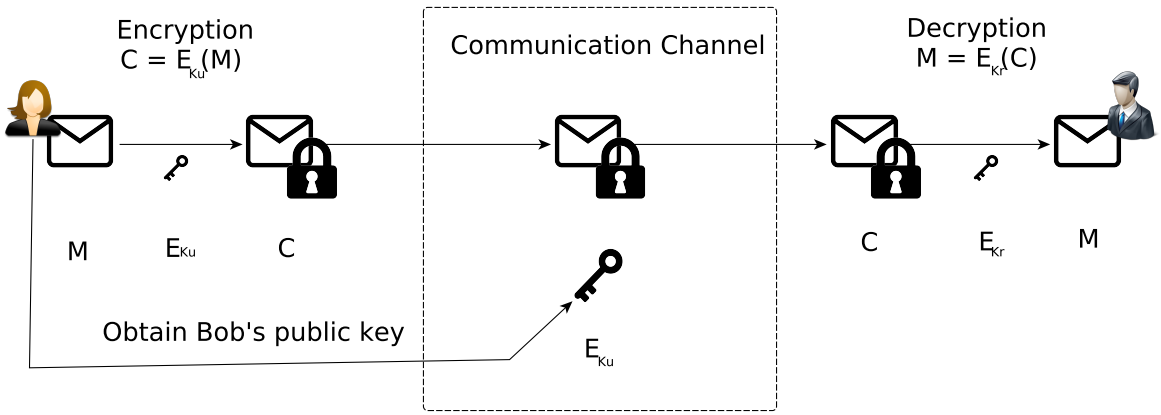

Figure 2: Asymmetric cryptography. Alice obtains Bob's public key, and uses it to encrypt and send a message.

Typical examples of symmetric cryptography include elliptic curve algorithms, Diffie Hellman, and RSA.

1.3 Digital Signatures

The purpose of digital signatures is to assure *integrity, authenticity, and non-repudiation* of information (e.g., messages, documents, transactions). Integrity regards preventing changes from the document. Authenticity allows us to uniquely identify the subject that signed the document. Non-repudiation enforces accountability for the content created. In other words, the entity signing a document cannot deny the signature.

A signature of a piece of information (also called digest) is made through a hash function [3]. Typically, hash functions are very efficient. A hash function receives a text and returns a sequence of bits of fixed length such that it follows three properties:

Definition 1.3. Pre-image resistance: It is computationally infeasible to find any input M that hashes to a given output D. In other words, given M, it is computationally infeasible to find M such that $H(M) = D$.

This corresponds to saying that a hash function cannot be inverted. In other words, given a hash, it is difficult to calculate its original value.

Definition 1.4. Second Pre-image resistance: It is computationally infeasible to find any second input which has the same output as any specified input, i.e., given $H(M)$, it is difficult to calculate $M' \neq M$ such that $H(M') = H(M)$.

This corresponds to saying that it is difficult to find a different input that produces the same hash.

Definition 1.5. Collision Resistance: It is computationally infeasible to find any two distinct inputs M, M', that hash to the same output D, i.e., such that $H(M) = H(M') = D$.

Collision resistance implies second pre-image resistance but does not guarantee pre-image resistance. Noninvertible hash functions aim to create a unique digest. It is then desirable that hash functions are collision resistance. Otherwise an attacker could try to find a different input that matches the same digest. Famous examples are the MD5 hash function (deprecated), MAC, HMAC, SHA256 [1].

Figure 3 depicts a simplified process underlying the creation of a secure message, ready to be transmitted. Alice creates a document and calculates its digest using a hash function. After that, she encrypts the digest using a cipher—finally, the document to be sent consists of the original message and an encrypted digest. The validation process is shown by Figure 4: Bob retrieves the combination message and encrypted hash of the message. Alice, Bob decrypts the digest of the message with the public key of the sender, created by Alice (authenticity and non-repudiation). After that, Bob recalculates the digest from M and compares it with the sent digest. If they are the same, it means that the information did not change - and hence, we assure integrity.

However, this solution may not work in all cases (see Exercise 2).

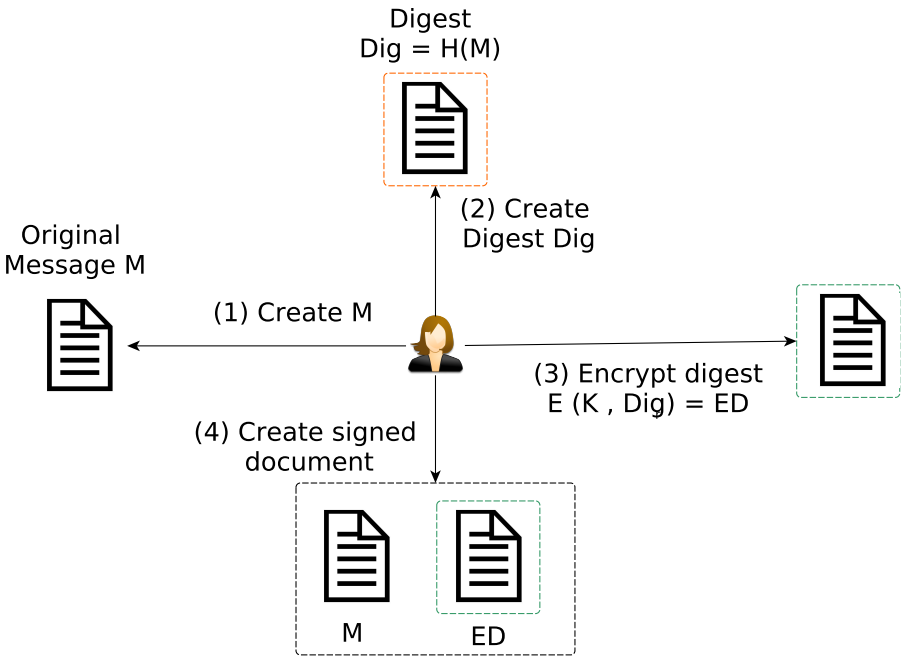

Figure 3: Simple digital signature process

1.4 Authentication, authorization, and accounting

Authentication aims to provide an answer to the question, “who is the client?”, or the subject that issues an access control request. It is concerned to identify the subject that asks permission to realize a particular action on a certain object in a specific context. *Authorization* provides an answer to, “what is the client allowed to do?”. The *Accounting*, on its turn, answers to the question, ”what did the client do?”. These fundamental security building blocks enable networks to have effective and dynamic security.

Authentication can be performed in several ways:

- *Shared-key*: A username is associated with a password.
- *One-time password*: a token is used to generate an access code valid for one access only. This process has to be synced with a token server (PIP).
- *Digital certificate*: emitted and signed by an authority. This certificate contains information about the entity it refers to. It contains a private-public key pair, used for encrypting and decrypting the same message.
- *Biometric credential*: based on what the client is, such as a fingerprint.

The authorization phase takes place after authentication. It verifies if a subject can perform an action on an object, based on access control policies. Accountability tracks all actions realized over a system: every access control request and corresponding decision. This allows to associated performed actions on objects to subjects.

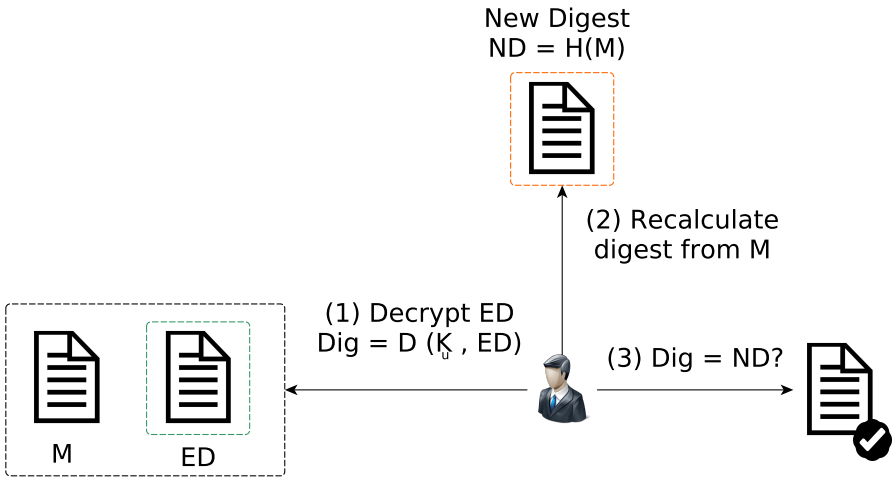

Figure 4: Simple digital signature process

1.5 Asymetric Key Cryptography: RSA

RSA (Rivest–Shamir–Adleman) is one of the first public-key algorithms used for secure data transmission. RSA ciphers per block of dimension k, such that $2^k < N$.

One can divide RSA's excution into several steps:

1. Key Generation
2. Encryption
3. Decryption

The first step of RSA consists in key generation. Two keys are considered, the public key $K_u = (N, e)$ and the private key, $K_r = (N, d)$. We choose two prime numbers p and q, such that $N = p.q$, and $Z = (p-1).(q-1)$. More concretely:

- Choose primes p and q, and $N = p.q$
- Choose e such that the greatest common divisor between e and $(p-1).(q-1)$ is 1.
- Calculate d such that $e.d \mod (p-1).(q-1) = 1$. Restriction: $0 < d < (p-1).(q-1)$.
- Communicate the public key (N, e), and the private key (N, d).

After keys are generated, text can be ciphered following a specific enconding, e.g., ASCII[4] encoding, according to the following Figure 5:

[4] https://www.ascii-code.com/

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Figure 5: ASCII encoding

Therefore, number 65 represents character 'A'. The cryptogram can be calculated as follows: $C = M^e \mod N$. For decryption, a similar operation is performed, with the private key: $M = C^d \mod N$.

In practice, to encode a large number, one can define $B <= N$, such that the message to be encrypted is divided into blocks of B digits each one. The cryptogram is sent into blocks. The decryption phase is analogous: it decrypts the several blocks received into a single message.

For the attacker to discover K_r, he needs to know Z. To know Z, the attacker needs to discover p and q. As p and q are prime numbers, it is computationally expensive to discover both, such that $p.q = n$. At the key generation step, one should destroy p and q, so the public and private keys cannot be recalculated. Although RSA can be considered safe, the exponents need to be very large numbers; if not, attacks are possible [6].

2 Hands on Crypto

Exercise 1: How many combinations can the MD5, and SHA256-3 algorithms generate? What is the likelihood of a hash collision with MD5? And with SHA256-3? Is SHA256-3 safer than MD5?

Exercise 2: Refer to Figures 3 and 4. Is this approach of signing and validating a document secure?

Exercise 3: Fork the course repository [5]. Inspect the support code

Note: Code at university-course/support/Lab02/RSA.

Exercise 4: Calculate the RSA keys associated with primes $p = 7$ and $q = 29$

Hint: you may use the support code to create a test case that helps you answer the following exercises.

Exercise 5: Given that the public key $K_u = (N, e) = (143, 7)$ and the private key is $K_r = (N, d) = (143, 103)$, encrypt the following message: "P"

Exercise 6: Decrypt the criptogram 80, given that $p = 3$ and $q = 31$

References

[1] E. Conrad, S. Misenar, and J. Feldman. Domain 3: Security Engineering (Engineering and Management of Security). In *CISSP Study Guide*, pages 103–217. Elsevier, jan 2016.

[2] T. Lisboa. Initial Page · Distributed Systems, 2017.

[3] P. Rogaway and T. Shrimpton. Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. Technical report, 2004.

[4] G. Simmons. Cryptology - History of cryptology — Britannica.

[5] Tecnico Lisboa. Discrete Mathematics, 2020.

[6] M. J. Wiener. Cryptanalysis of Short RSA Secret Exponents. *IEEE Transactions on Information Theory*, 36(3):553–558, 1990.

[5] https://github.com/hyperledger-labs/university-course