CIS 4360
Secure Computer Systems

Cryptography

Professor Qiang Zeng

Spring 2017

TEMPLE
UNIVERSITY

Previous Class

- Attack Surface
  - Components reachable and exploitable by attackers
- Attack surface reduction
  - A practice that minimizes the attack surface
- Adversary model
  - To identify the motives and capabilities of the adversary
- Security engineering
  - Threat modeling -> Security requirement -> Development of security measures
- Threat modeling
  - STRIDE
  - Attack trees

Previous class…

What is the Attack Surface in terms of entering the SERC building?

The ID card reader system
Human (who monitor the gates)
All windows and doors
All kinds of service people who have keys to the doors
Garbage collection car

Outline

- Classical Cryptography
- Goals of Cryptography
- Building blocks of Cryptography
- Cryptographic Hash

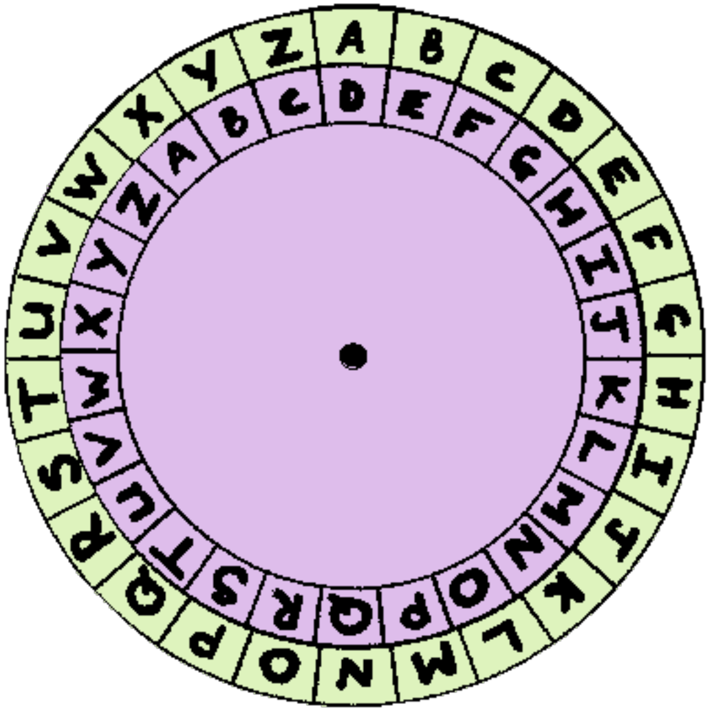
Cryptography

- Cryptography: from Greek
  - Crypto-: secret; -graphy: writing
  - The art of secret writing
- Cryptography: creates ciphers
- Cryptanalysis: break ciphers
- Cryptography + Cryptanalysis = Cryptology

The history of Cryptology is the arm race between Cryptography and Cryptanalysis

Caesar Cipher

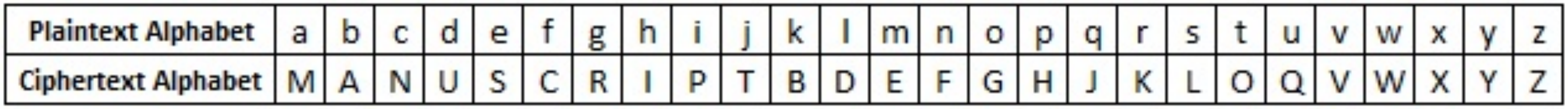
- Y = (X + k) % 26; Julius Caesar used k = 3
- k=13 was also popular, called ROT13
- How to decipher “KHOOR” with k = 3?


A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Caesar Cipher

- Given k=3, one can easily get “HELLO”; but what if you do not know k?
- Easy. There are only 26 different values for k
- Surprisingly, the Mafia godfather Provenzano used it until being arrested in 2006 and the police was able to decipher much crime evidence easily

Keyword Cipher



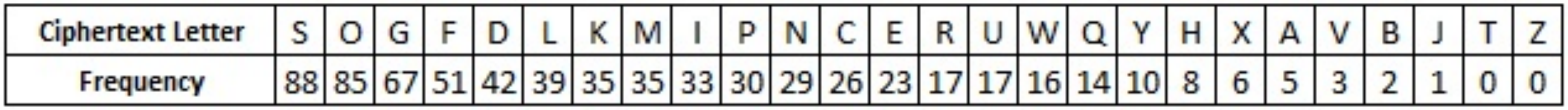
Plaintext Alphabet	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Ciphertext Alphabet	M	A	N	U	S	C	R	I	P	T	B	D	E	F	G	H	J	K	L	O	Q	V	W	X	Y	Z

GFS WMY OG LGDVS MF SFNKYHOSU ESLLMRS, PC WS BFGW POL DMFRQMRS, PL OG CPFU M UPCCSKSFO HDMPFOSXO GC OIS LMES DMFRQMRS DGFR SFGQRI OG CPDD GFS LISSO GK LG, MFU OISF WS NGQFO OIS GNNQKKSFNSL GC SMNI DSOOSK. WS NMDD OIS EGLO CKSJQSFODY GNNQKKPFR DSOOSK OIS 'CPKLO', OIS FSXO EGLO GNNQKKPFR DSOOSK OIS 'LSNGFU' OIS CGDDGWPFR EGLO GNNQKKPFR DSOOSK OIS 'OIPKU', MFU LG GF, QFOPD WS MNNGQFO CGK MDD OIS UPCCSKSFO DSOOSKL PF OIS HDMPFOSXO LMEHDS. OISF WS DGGB MO OIS NPHISK OSXO WS WMFO OG LGDVS MFU WS MDLG NDMLLPCY POL LYEAGDL. WS CPFU OIS EGLO GNNQKKPFR LYEAGD MFU NIMFRS PO OG OIS CGKE GC OIS 'CPKLO' DSOOSK GC OIS HDMPFOSXO LMEHDS, OIS FSXO EGLO NGEEGF LYEAGD PL NIMFRSU OG OIS CGKE GC OIS 'LSNGFU' DSOOSK, MFU OIS CGDDGWPFR EGLO NGEEGF LYEAGD PL NIMFRSU OG OIS CGKE GC OIS 'OIPKU' DSOOSK, MFU LG GF, QFOPD WS MNNGQFO CGK MDD LYEAGDL GC OIS NKYHOGRKME WS WMFO OG LGDVS.

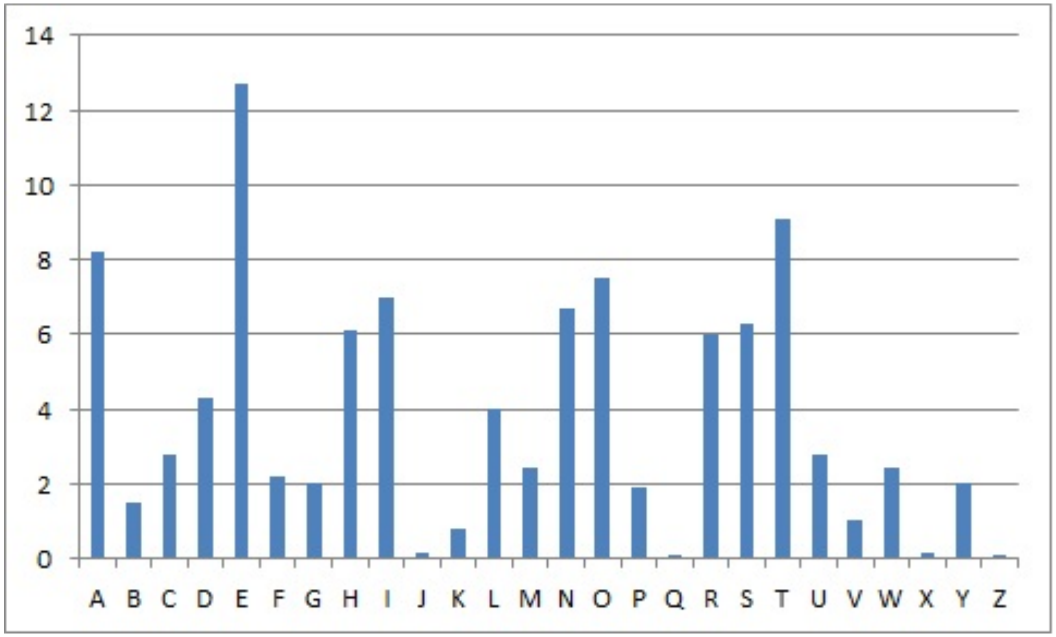
How to decipher the message without the key? There are a huge number of possible keys (26!) rather than 26

Frequency Analysis

The ordered occurrences of letters in the cipher text



Ciphertext Letter	S	O	G	F	D	L	K	M	I	P	N	C	E	R	U	W	Q	Y	H	X	A	V	B	J	T	Z
Frequency	88	85	67	51	42	39	35	35	33	30	29	26	23	17	17	16	14	10	8	6	5	3	2	1	0	0



So,
S -> E
O -> T
…
Another frequency-based strategy: the most frequent three-letter word “OIS” is probably “THE”
…

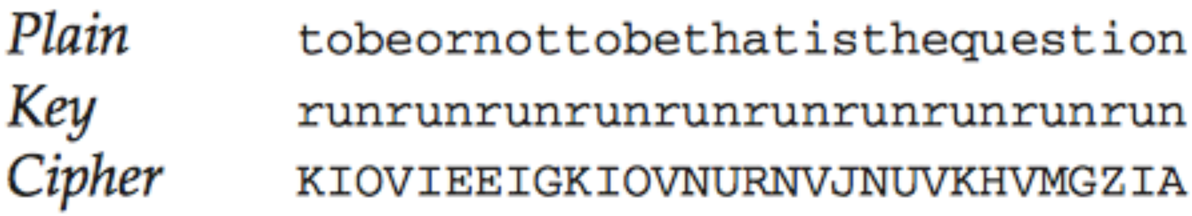
Frequency Analysis

“one way to solve an encrypted message, if we know its language, is to find a different plaintext of the same language long enough to fill one sheet or so, and then we count the occurrences of each letter. we call the most frequently occurring letter the 'first', the next most occurring letter the 'second' the following most occurring letter the 'third', and so on, until we account for all the different letters in the plaintext sample. then we look at the cipher text we want to solve and we also classify its symbols. we find the most occurring symbol and change it to the form of the 'first' letter of the plaintext sample, the next most common symbol is changed to the form of the 'second' letter, and the following most common symbol is changed to the form of the 'third' letter, and so on, until we account for all symbols of the cryptogram we want to solve.”

-- Al-Kindi, 850AD

Vigenere Cipher

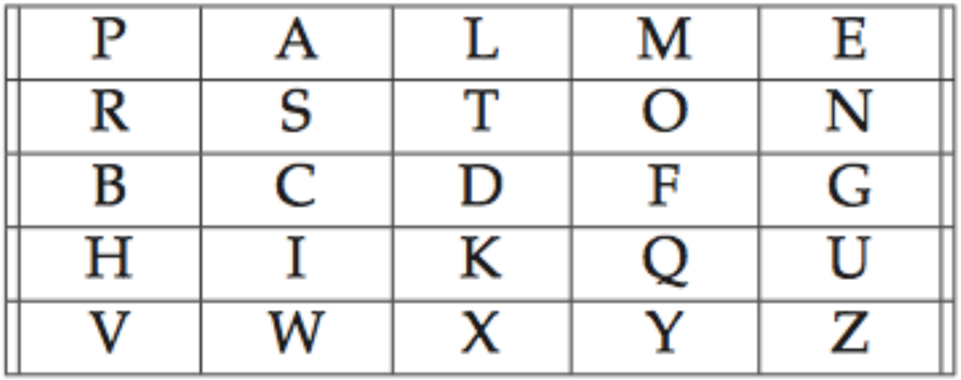
- A variant of Caesar Cipher
- Instead of using one key, it uses multiple keys, e.g.,
  - Use k1 for letters at the positions 1, 4, 7, 10, …
  - Use k2 for letters at the positions 2, 5, 8, 11, …
  - Use k3 for letters at the positions 3, 6, 9, 12, …



Plain	tobeornottobethatisthequestion
Key	runrunrunrunrunrunrunrunrunrun
Cipher	KIOVIEEIGKIOVNURNVJNUVKHVMGZIA

Playfair Cipher

- Instead of transforming letters one by one (stream cipher), it encrypts multiple letters each time (block cipher)
- In Playfair, two letters are encrypted per time
  - – Key: e.g., lo -> mt, rd -> tb, gr -> bn



P	A	L	M	E
R	S	T	O	N
B	C	D	F	G
H	I	K	Q	U
V	W	X	Y	Z

All classic ciphers are easy to break by some **frequency analysis**

Never ever use any home-made cryptography!

One-Time Pad

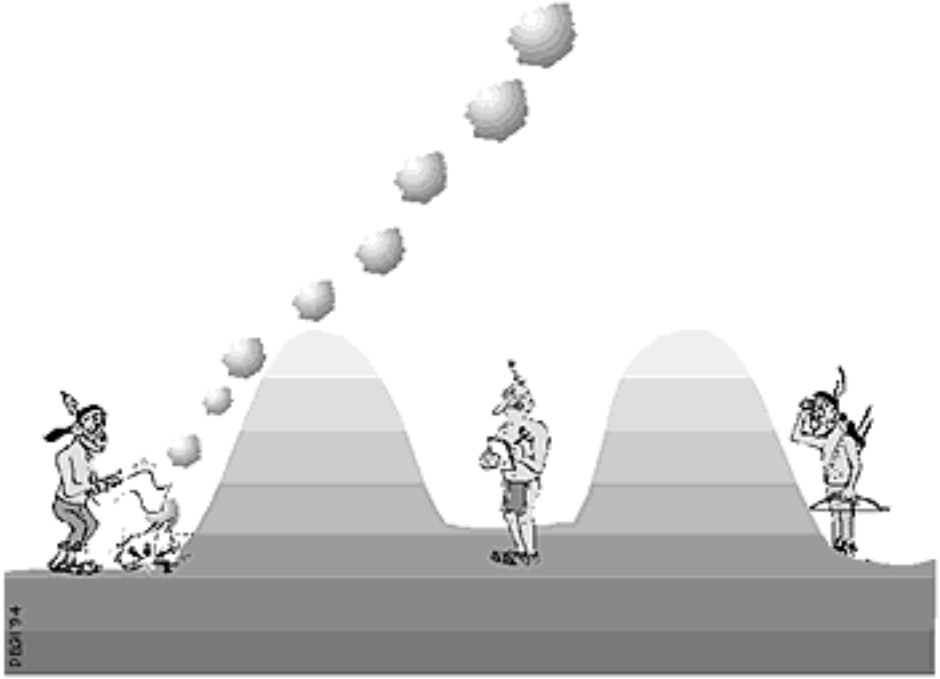
- *Is there any cipher unbreakable?*
- Yes, it is One-Time Pad, also called Vernam cipher; created by Gilbert Vernam during WWI
  - The key is at least as long as the plaintext

$$C_i = P_i \text{ XOR } K_i$$

  - The key is truly random
  - Each key is used only once
- Shannon proved that it offers perfect secrecy, as the adversary cannot get any useful information from the ciphertext
- Too expensive!

What do you need Cryptography for?

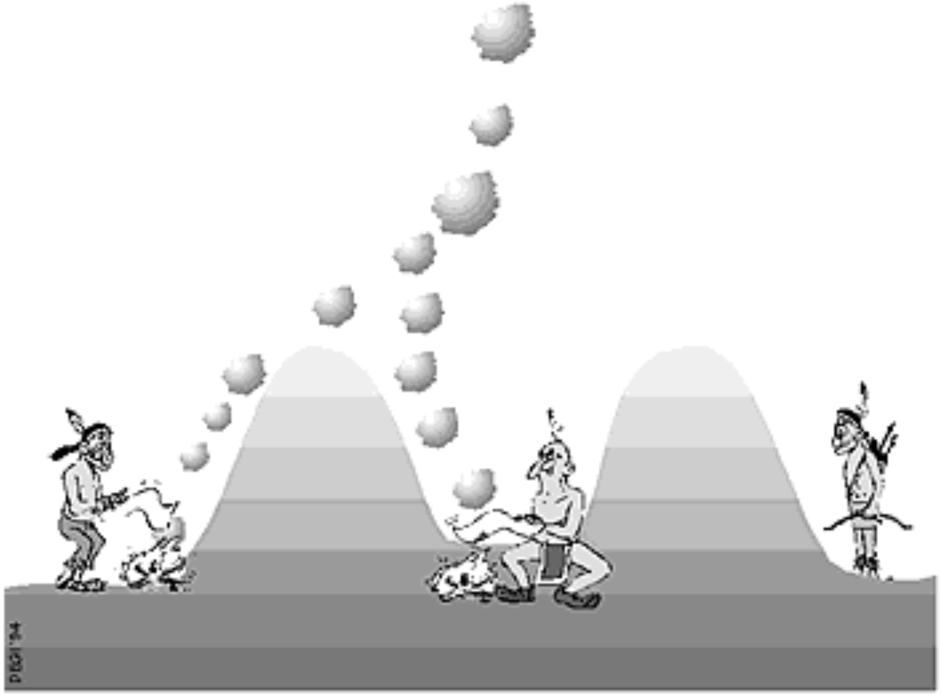
- Confidentiality
  - – I do not want the message to be read by my enemy



(The copyright of these graphs belong to Mark Vandenwauver)

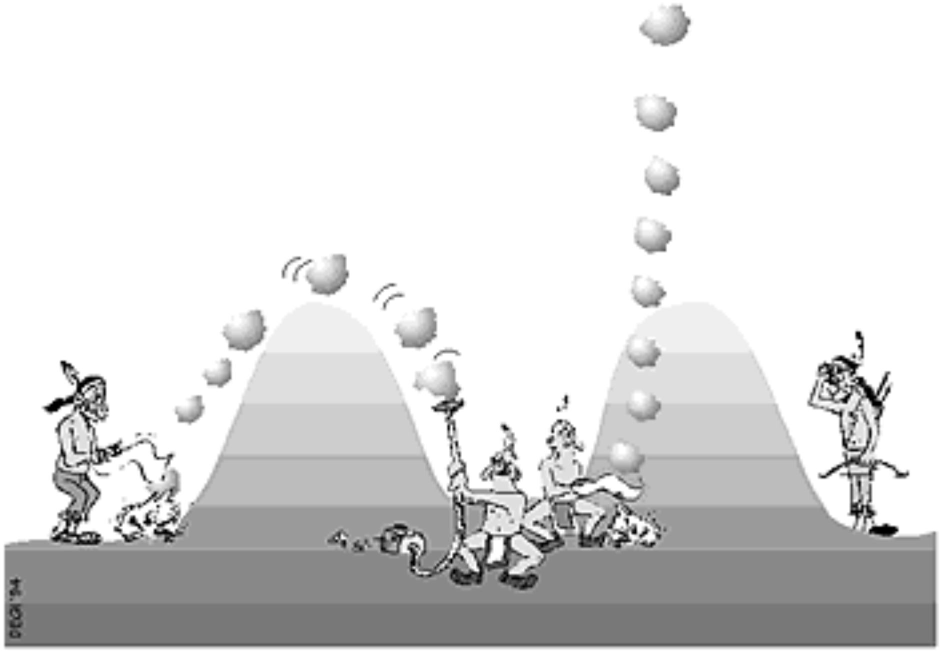
What do you need Cryptography for?

- Data Integrity
  - Has the message been manipulated?



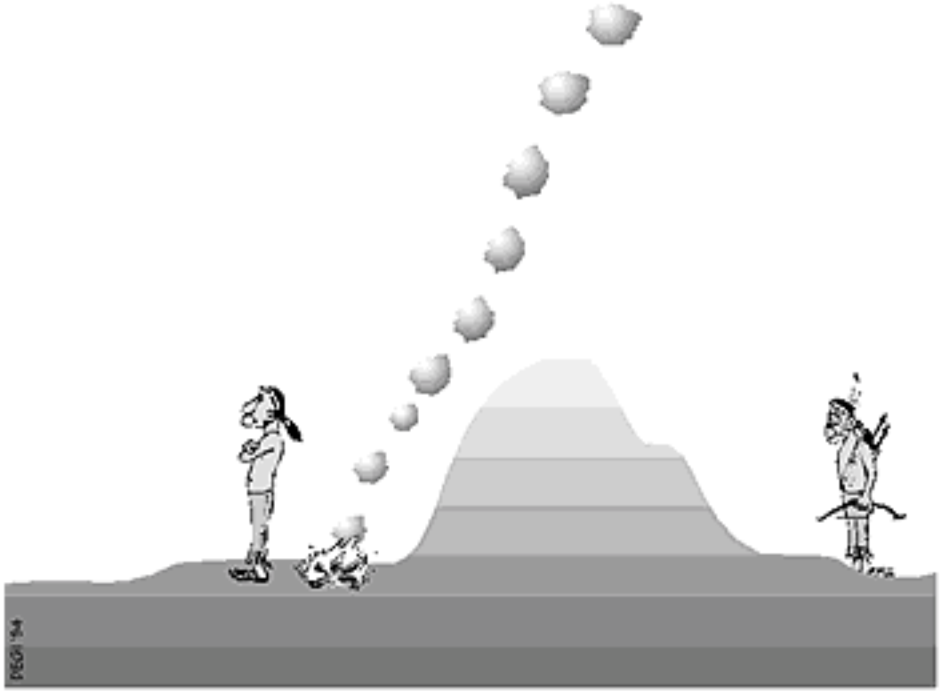
What do you need Cryptography for?

- Authentication
  - Am I talking with the right person?
  - Is this message really from the chief?



What do you need Cryptography for?

- Non-repudiation
  - I can prove you indeed sent/received the message



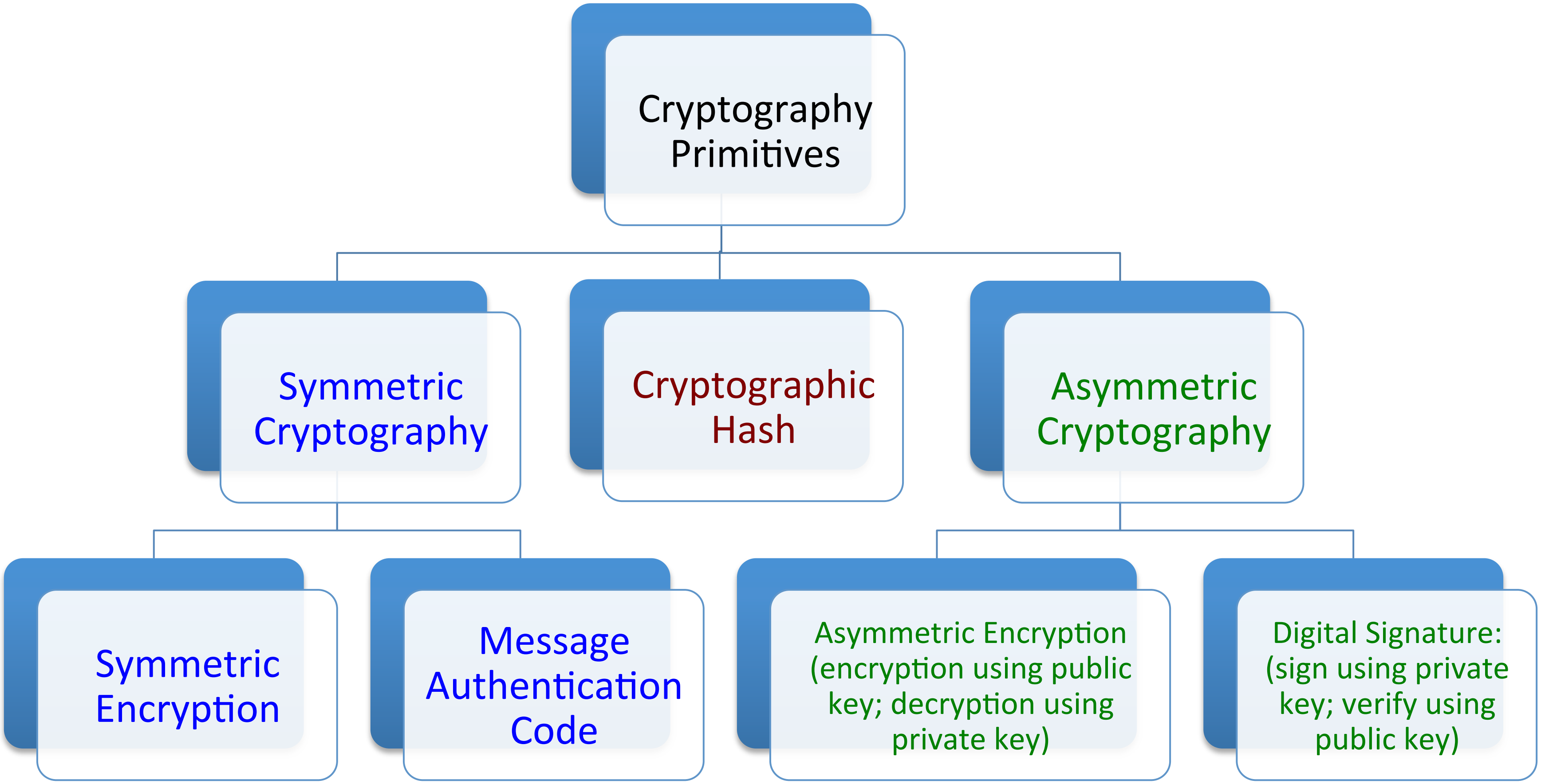
Cryptographic Goals

- Confidentiality
  - Information is unintelligible to attackers
- Data Integrity
  - Data manipulation by attackers can be detected
  - Data manipulation: insertion, deletion, and substitution
- Authentication
  - Entity authentication: impersonation can be detected
  - Data-origin authentication: fake messages can be detected
- Non-repudiation
  - You cannot deny

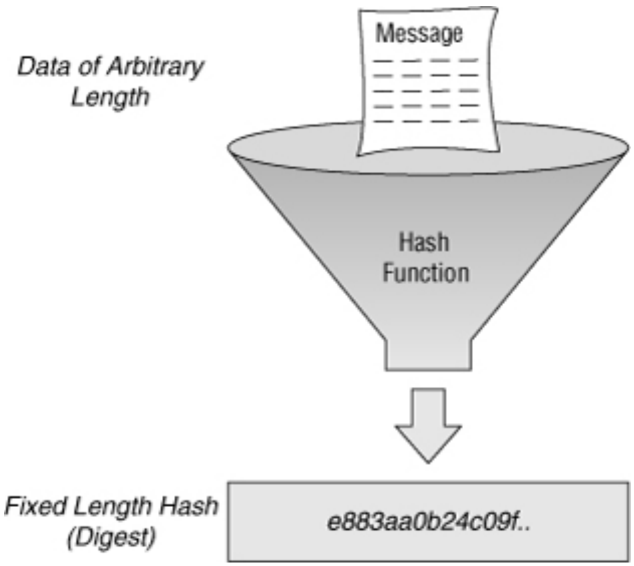
Cryptography Primitives (informal intro.)

- Encryption
  - Encryption: Plaintext -> Ciphertext; Decryption: Cipher -> Plain
  - Symmetric: encryption and decryption using the same key
  - Asymmetric: public key for encryption, private key for decryption
- Cryptographic Hash
  - An arbitrarily long input -> a short string of fixed size, e.g., 128bit
- Message Authentication Code
  - An arbitrarily long input + key -> a short string of fixed size
  - Created and verified using the same key
- Digital Signature
  - An arbitrarily long input + private key -> a short string
  - Created using the private key; verified using the public key
  - Thus, can be verified by the world

Another View of the Primitives

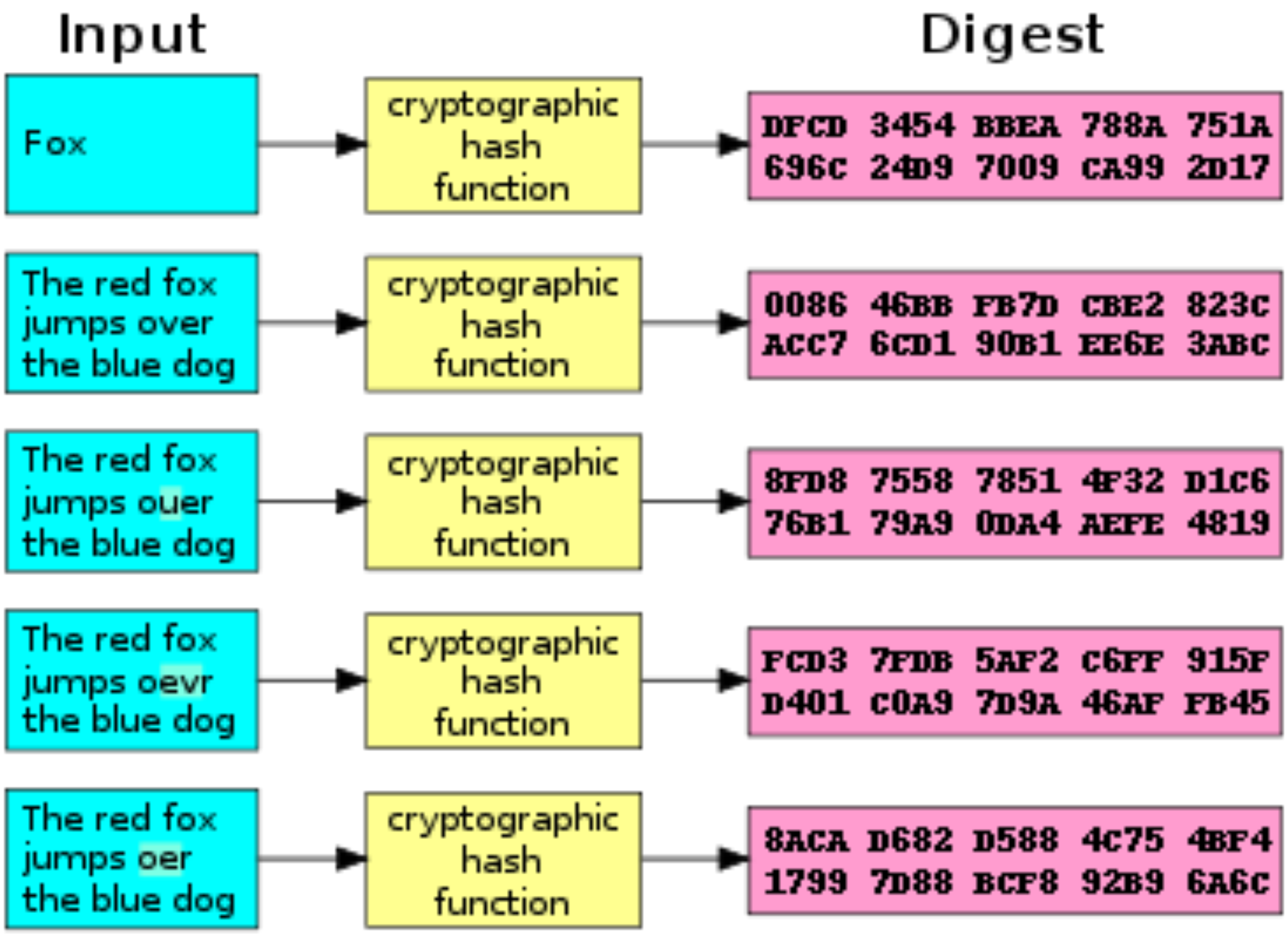

Cryptography Primitives
Symmetric Cryptography
Cryptographic Hash
Asymmetric Cryptography
Symmetric Encryption
Message Authentication Code
Asymmetric Encryption (encryption using public key; decryption using private key)
Digital Signature: (sign using private key; verify using public key)

Cryptographic Hash



- H(m) = h
- h is also called the message digest)
- Properties
  - Pre-image resistance: Given a hash value h it is computationally difficult to find the m, such that H(m) = h. (Functions of this property is called one-way functions)
    - It implies that you cannot infer m from h
  - Second pre-image resistance: Given m_1 it is computationally difficult to find another message m_2 such that $H(m_1) = H(m_2)$
    - It implies that you cannot forge a message that has a target hash value
    - It also implies that if you modify the message, its hash will definitely change
  - Collision resistance: It is computationally difficult to find any pair of messages m_1 and m_2 such that $H(m_1) = H(m_2)$
    - It implies that even you have trillions of files, you are ensured that they do not happen to have the same hash values

Example



Input

Digest

Fox → cryptographic hash function → DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17

The red fox jumps over the blue dog → cryptographic hash function → 0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC

The red fox jumps ouer the blue dog → cryptographic hash function → 8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819

The red fox jumps oevr the blue dog → cryptographic hash function → FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45

The red fox jumps oer the blue dog → cryptographic hash function → 8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

Applications of Cryptographic Hash

- File/message integrity
  - Given h = H(m), if the integrity of h is assured, h can be used to verify whether m has been modified
  - E.g., in bittorrent file transmission, the digest value of the file *F* is stored in the small file .torrent; after you download the file *F* through P2P (insecure channel), the digest of the downloaded file is generated and compared against the one in the .torrent
  - *In the example about, you trust the .torrent file and the digest value inside. What if h can also be attacked? (Writing assignment)*

Applications of Cryptographic Hash

- Password storage and verification
  - Instead of storing the password as plaintext, store its hash value h
  - Salt is critical to resist rainbow attack, which pre-computes a large table of the mappings between passwords and hashes
- To store a password
  - Generate a long random salt using a CSPRNG (Cryptographically Secure Pseudo Random Number Generator)
  - Prepend the salt to the password and hash it with a **standard** slow password hashing function (e.g., Argon2, bcrypt, scrypt, or PBKDF2)

$$h = H(m \, || \, salt)$$

  - Save both the salt and the hash in the user's database record
- To validate a password
  - Retrieve the user's salt and hash from the database
  - Prepend the salt to the given password and hash it using the same function
  - Compare the hash of the given password with the hash from the database
  - If they match, the password is correct. Otherwise, the password is incorrect

Cryptographic Hash Function Standards

- MD5 (Message Digest 5): 128-bit hash
  - Broken since 2004 by Xiaoyun Wang
  - Collisions can be constructed in seconds on a laptop
- SHA-1 (Secure Hash Algorithm 1): 160-bit hash
  - Considered insecure
  - Practically broken since 2005 by Xiaoyun Wang
  - Your browsers will refuse SHA-1 certificates by 2017

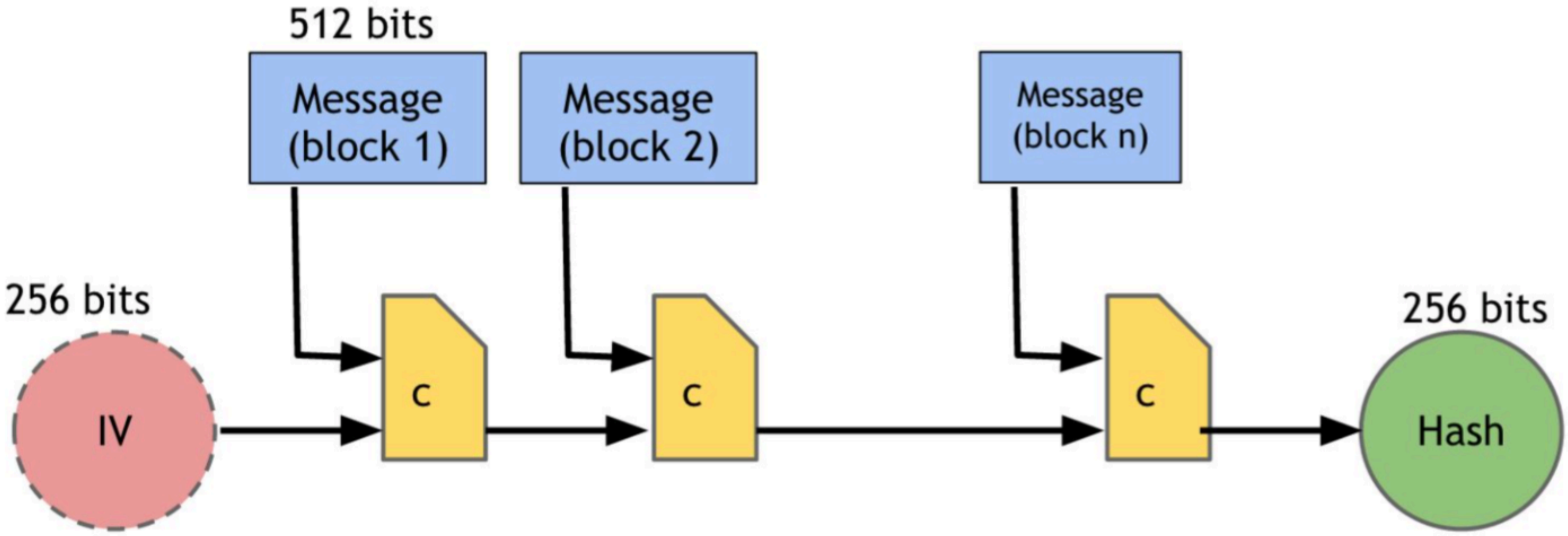
Do not use them in your security products!

Cryptographic Hash Function Standards

- SHA-2 family: 224, 256, 384 or 512 bits
- SHA-3 family (Keccak): output can be arbitrary size
  - NIST started a competition for SHA-3 in 2007
  - NIST's original concern was that SHA-2 would soon be broken, although in fact SHA-2 is still fine
  - Anyway, Keccak won it and became SHA-3 in 2012

Merkle-Damgard Construction for Hash

- SHA-256 as an example: breaking the input up into a series of equal-sized blocks, and operating on them in sequence
- SHA-3 uses a different structure: Sponge



Summary

- Classical Cryptography
  - Frequency analysis
  - Never use home-made cryptography
- Goals of Cryptography
  - Confidentiality, data integrity, authentication, non-repudiation
- Building blocks of Cryptography
  - Cryptographic hash, encryption, MAC, digital signature
- Cryptographic Hash
  - Password storage, verifying data integrity
  - Do not use MD5 and SHA-1

Writing Assignments

- By sending m and Hash(m), can the receiver verify the integrity of m, i.e., to verify whether m has been manipulated during the transmission? If not, what should you do?
- We have covered two of the most important applications of cryptographic hash, can you find another two or three? Hints: hash chain, proof-of-work, file identifiers