FTEC 5520 (Week 3)

Agenda - CyberSecurity Basics and Cryptography Week 3

1. Basics of Cryptography
2. Example of Cryptanalysis
3. Block and Stream Cipher
4. Hash Functions
5. Digital Signature

Basics of Cryptography

Encryption and Hashing

Channel Encryption

Network channel encryption
WiFi encryption
SSL/TLS encryption
Secure Email

Machine Encryption and Hashing

Disk encryption
Password protection

Cryptography and Data Security

Cryptography and Steganography

CRYPTOGRAPHY

Art of hiding information

Core logical component in information security technologies

A branch of mathematics based on the transformation of information to achieve certain goals

- confidentiality
- integrity
- authenticity
- non-repudiation

STEGANOGRAPHY

Art of hiding communications

- Conceal the existence of the message

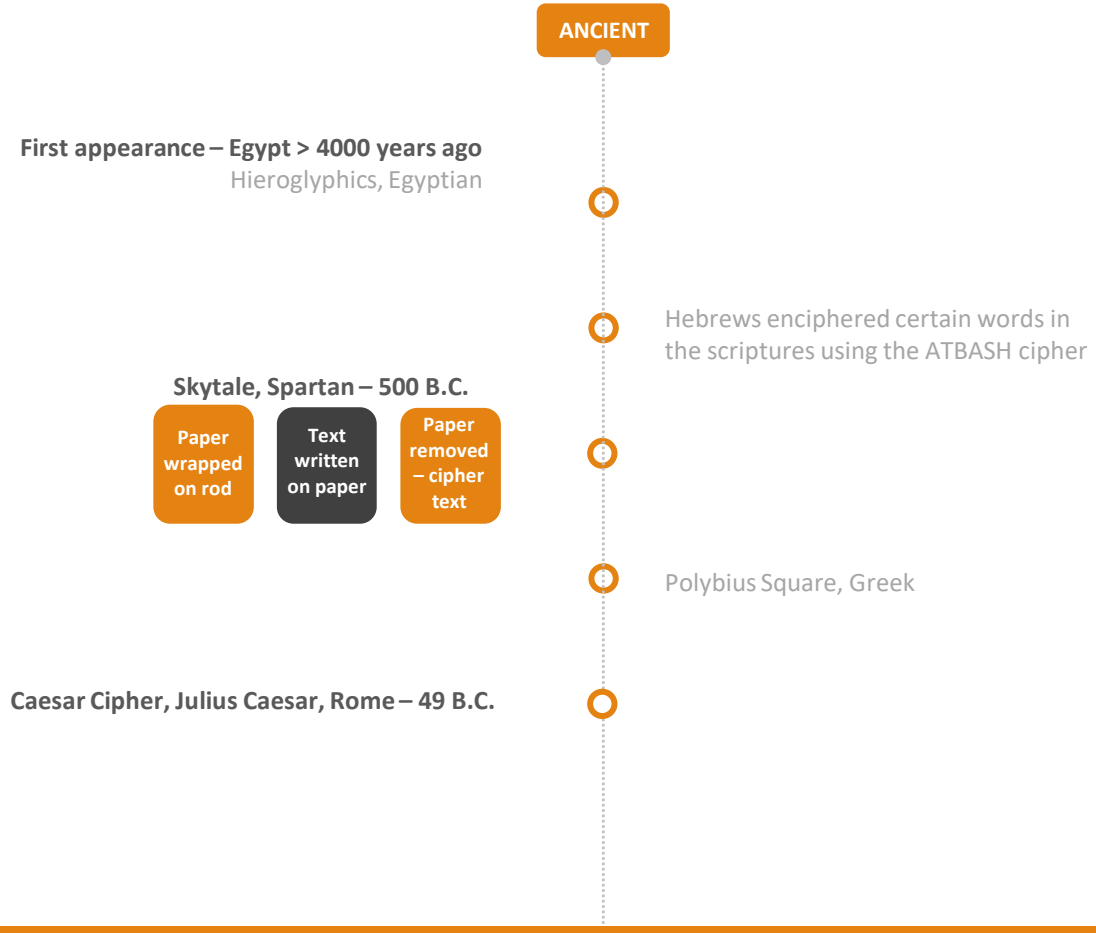
Information is hidden in carriers

- picture files, audio files, slack disk space, network signals, etc.
- convert channel

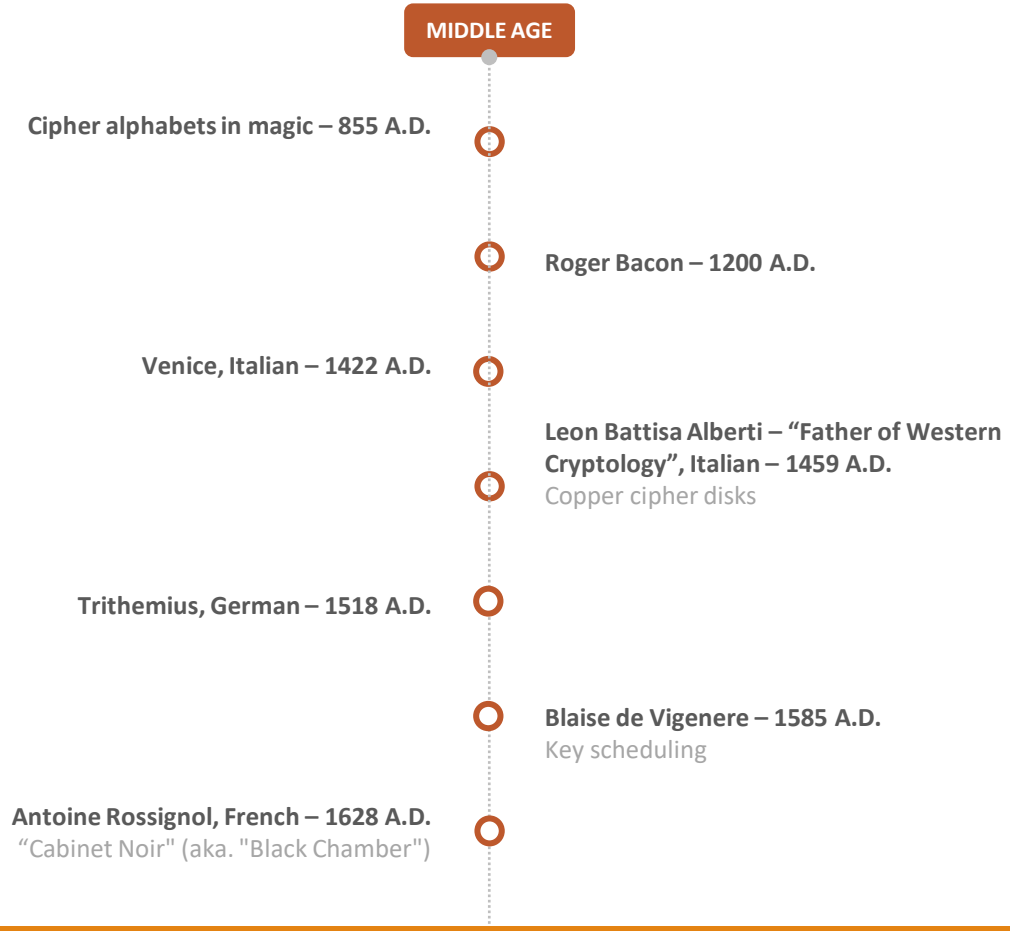
Usually, the carriers have a large information redundancy

- require much larger overhead than cryptography

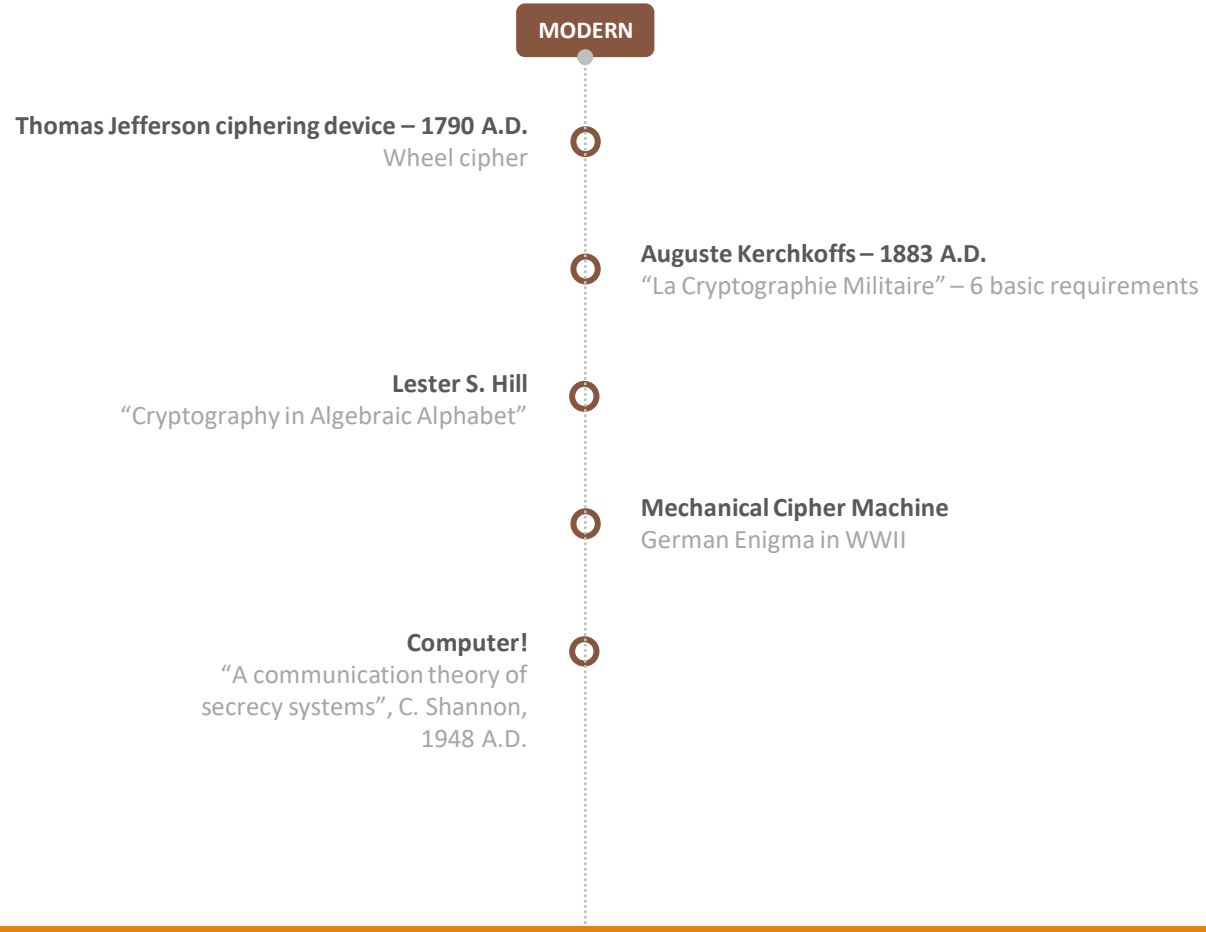
History – Ancient



History – Middle Age



History – Modern



Encryption and Decryption

The strength of the encryption is dependent on two basic items:

- The nature of the mathematical algorithm
- The size of the keys involved

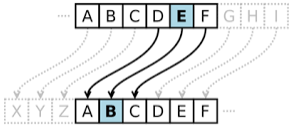
Provided that the encryption algorithm is a good one, that it is implemented correctly, and that there are no cracks in key management that allow the secret key to be divulged, the only way to crack an encrypted message is to try all possible values of the key until one works. This brute-force approach will take time and money.

Caesar Cipher (ROT3)

Examples:

- Caesar Cipher (ROT3)
  - `PLAIN:  ABCDEFGHIJKLMNOPQRSTUVWXYZ`
  - `ROT3:  DEFGHIJKLMNOPQRSTUVWXYZABC`

- Rotation 13 (ROT13)
  - It accommodated by some mail programs (e.g. some versions of Netscape).
  - `PLAIN:  ABCDEFGHIJKLMNOPQRSTUVWXYZ`
  - `ROT13:  NOPQRSTUVWXYZABCDEFGHIJKLM`



Caesar Cipher relatively simple, but substitution ciphers can be very powerful (e.g. one-time pad)

Cryptanalysis: Brute Force

Caesar cipher has **25** possible keys (keyspace = 25).

Information required for this attack:

- 1. Knowledge of the encryption algorithm used.
- 2. The number of possible keys.
- 3. The language of the plaintext.

The primary obstacle is item 2.

Substitution Ciphers

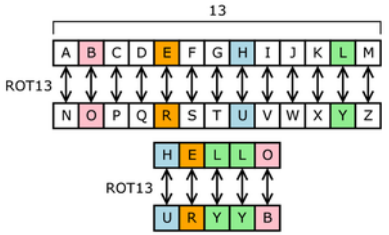
Simple substitution cipher:

- a = p, b = m, c = f, ...

Break via letter frequency analysis

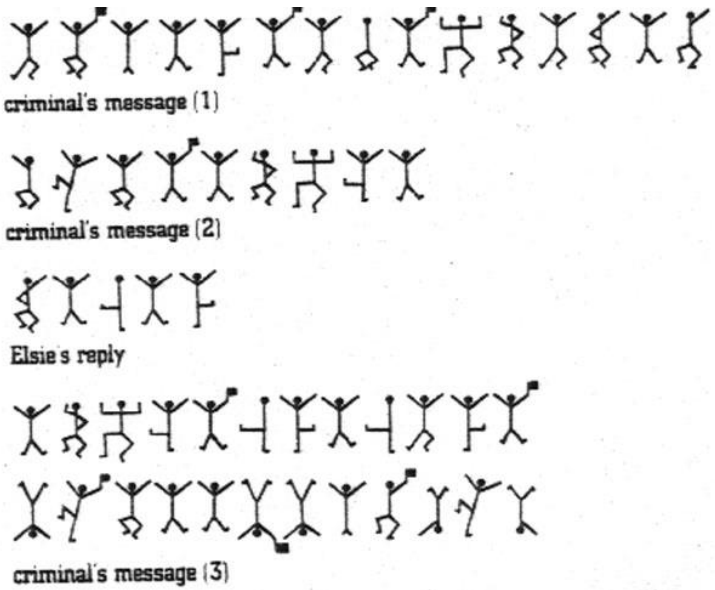
Polyalphabetic substitution cipher

- 1. a = p, b = m, c = f, ...
- 2. a = l, b = t, c = a, ...
- 3. a = f, b = x, c = p, ...



Break by decomposing into individual alphabets, then solve as simple substitution

The Dancing Men (Sherlock Holmes)



In Sherlock Holmes' story "The Adventure of the Dancing Men", a man reports that his wife, Elsie, became upset when she received several notes with figures of dancing men on them. Holmes went about deciphering the code.

He knew that E is the most common letter in the English language and that there was a high probability that the name "Elsie" appeared somewhere in one of the messages.

Using this information, can you decipher the messages she received and the one message she sent? You may also want to find out what the other most common letters used in the alphabet are in order to help you

Anything Wrong 

Study this paragraph and all things in it. What is vitally wrong with it? Actually, nothing in it is wrong, but you must admit that it is most unusual. Don't just zip through it quickly but study it scrupulously. With luck you should spot what is so particular about it and all words found in it. Can you say what it is? Tax your brains and try again. Don't miss a word or a symbol. It isn't all that difficult.

Language Redundancy and Cryptanalysis

Human languages are **redundant**

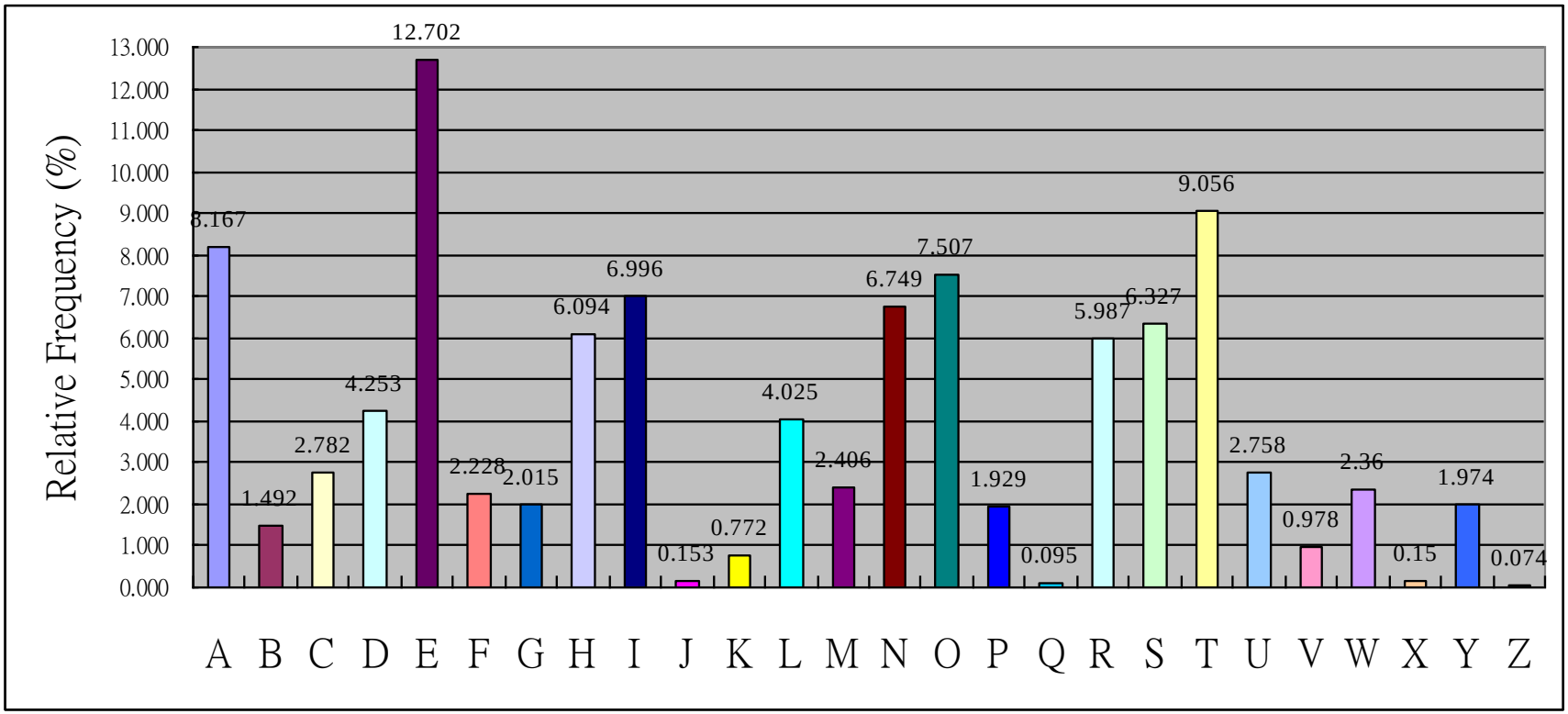
Letters are not equally commonly used

In English **e** is by far the most common letter then T,R,N,I,O,A,S

Other letters are fairly rare Z,J,K,Q,X

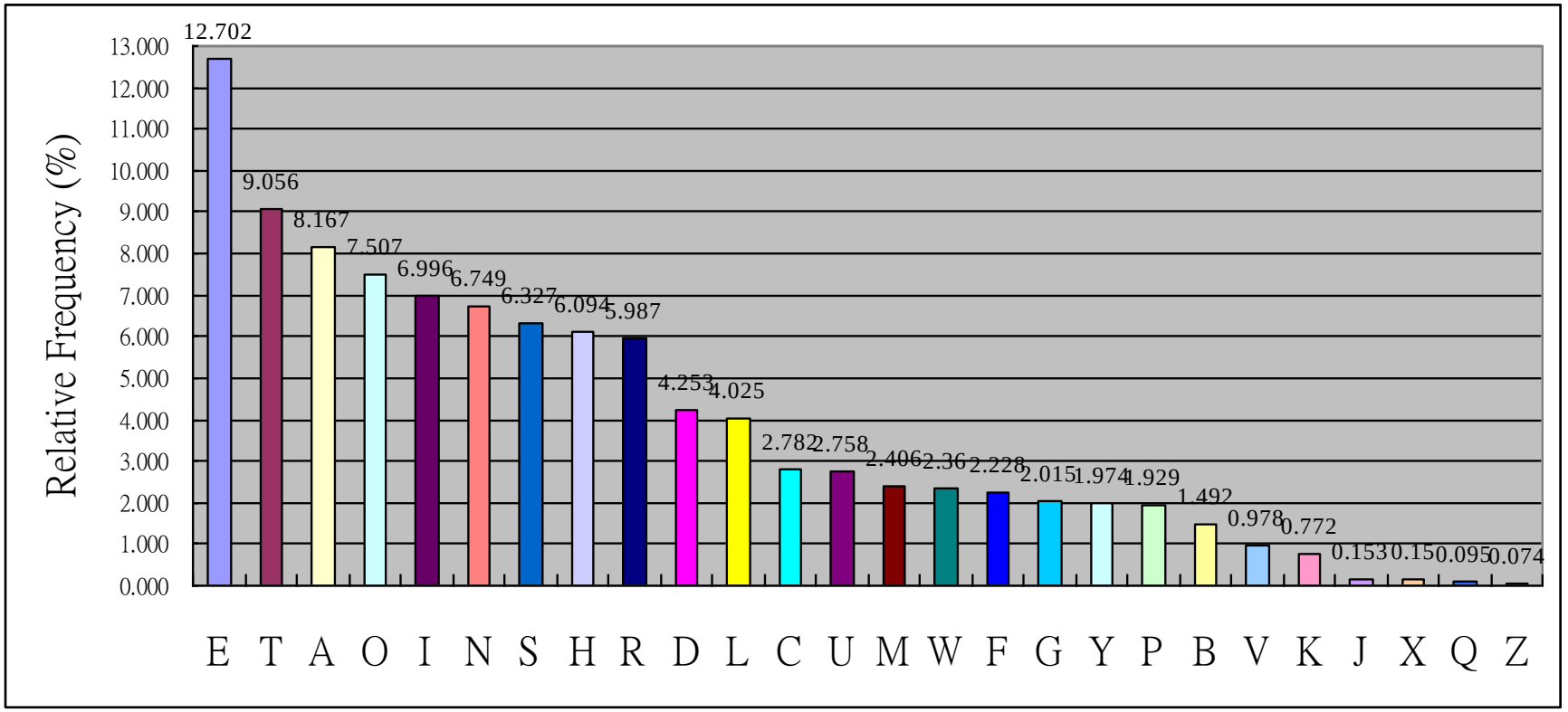
Have tables of single, double & triple letter frequencies

English Letter Frequencies



This graph is based on counts done at ADFA in the late 1980's, and used to develop the tables published in Seberry & Pieprzyk

English Letter Frequencies (cont.)



Transposition Ciphers

Instead of substituting ciphertext characters for plaintext, the transposition cipher rearranges the plaintext characters, e.g. **Rail Fence Cipher**:

```
sell entire portfolio now and buy gold
```

Write each character on alternate lines, like this:

```
Sletrprflooadugl
Elnieotoinwnbyod

Sletrprflooadugl + Elnieotoinwnbyod

= SLETRPRFLOOADUGLELNIEOTOINWNBYOD
```

Goal is diffusion rather than confusion

Transposition and Rotors

More challenging: transposition into rows and columns numbered by a key:

```
Key:          3 7 5 8 1 4 2 6
Plaintext:    s e l l e n t i
              r e p o r t f o
              l i o n o w a n
              d b u y g o l d
Ciphertext:
   EROGTFALSRLDNTWOLPOUIONDEEIBLONY
```

Will still yield to cryptanalysis, because it retains the letter frequency characteristics of the plaintext, but additional stages of encryption makes it much harder -- hence multiple rounds.

German “Enigma” cipher machines used transposition by mechanical rotors. Breaking this system helped create the first computers.

Cipher Machines

Famous rotor machines

- US: Converter M-209
- UK: TYPEX
- Japan: Red, Purple
- Germany: Enigma

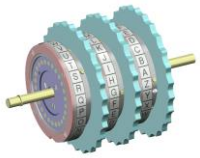
Many books on Enigma

- Kahn, Seizing the Enigma
- Levin, Ultra Goes to War
- Welchman, The Hut Six Story
- Winterbothm, The Ultra Secret

Various kludges made to try to improve security — none worked

Enigmas were sold to friendly nations after the war

Improved rotor machines were used into the 70's and 80's



Shannon's Thoughts

Practical ciphers such as DES are explicitly designed on the basis of Shannon's principles of confusion and diffusion

Confusion

- The enciphering process should be such that the ciphertext statistics depend on the plaintext statistics in a manner too complicated to be exploited by an attacker

Diffusion

- Each bit of the input key should influence many bits of the ciphertext

Running Key Cipher

aka Book Cipher

Key

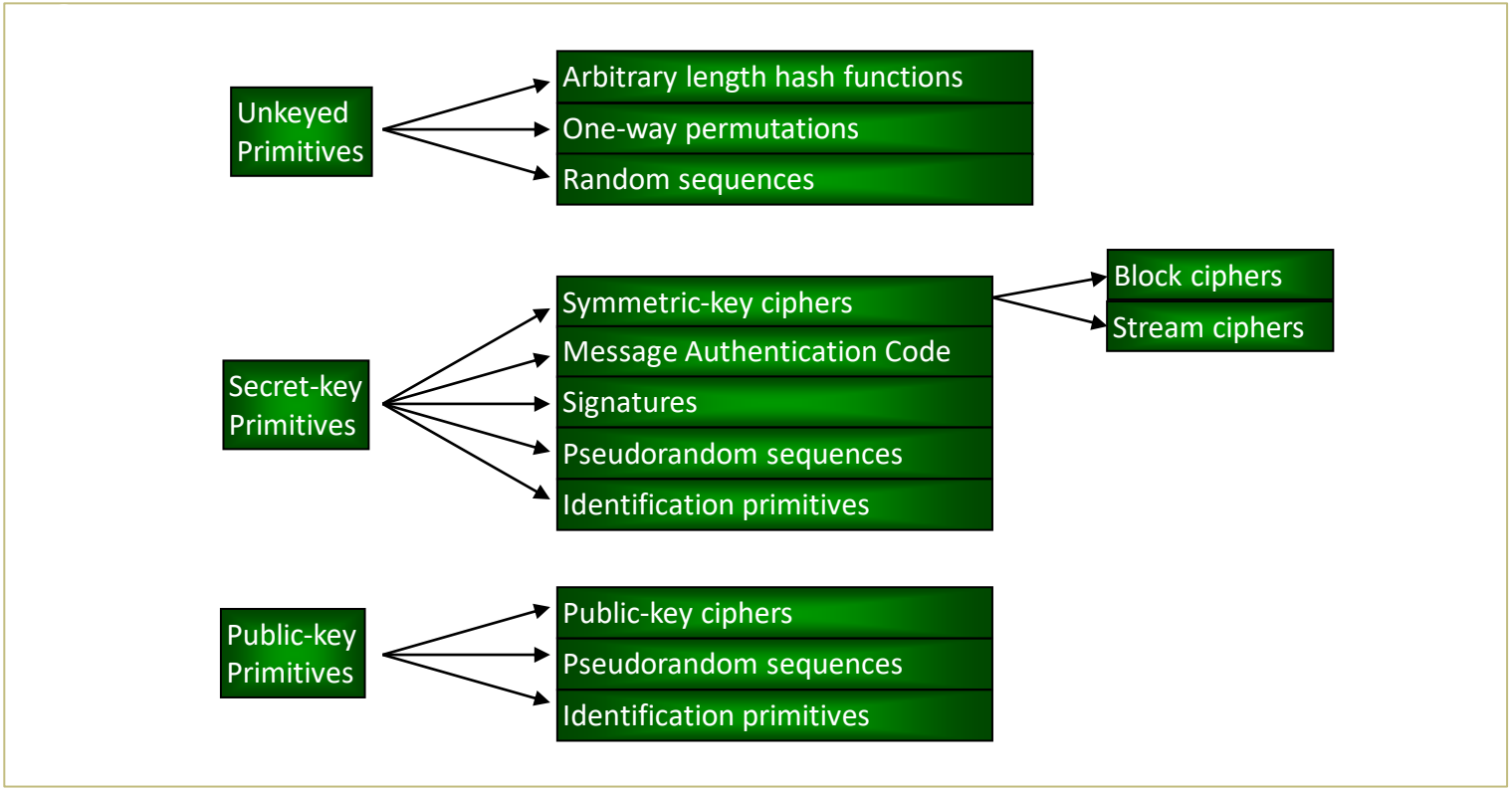
- certain paragraphs from a book with page, line and column number

e.g.

- paragraph: “gilderoy lockhart came slowly into view …”
- plain: “This is a test”
- key: “gild er o yloc”
- cipher: “zptv mj o rpgv”

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
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Taxonomy of Cryptographic Techniques



Types of Cryptography

Symmetric / Secret Key
- Fast
- Serious problem in key management

Asymmetric / Public Key
- Slow
- Minor problem in key management

Hash / One-way transformation

Encryption Cryptographic Systems

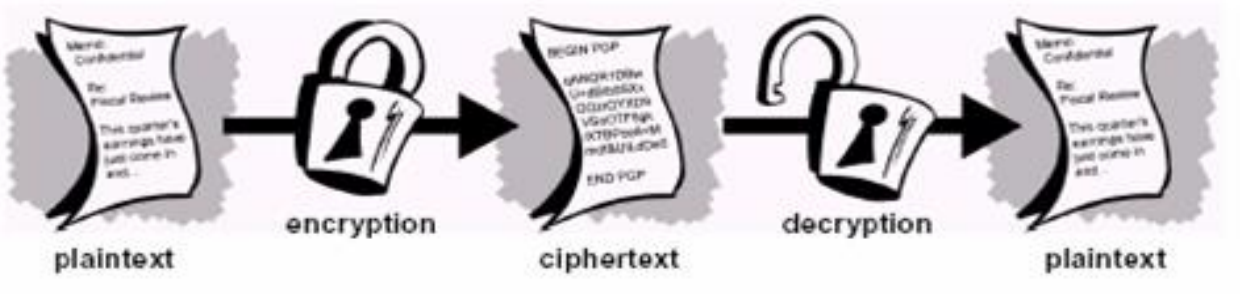
Ciphers type

- Block ciphers
- Stream ciphers

Encryption Scheme

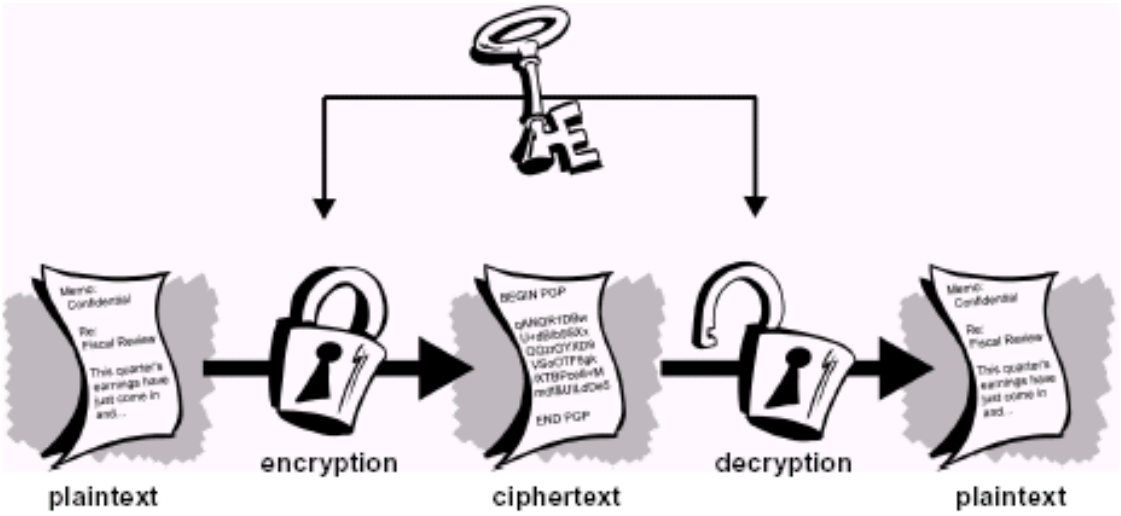
- Symmetric key system (secret key)
  - DES, IDEA
  - AES
- Asymmetric key system (public key)
  - RSA, DSA
- hybrid key system

Simplified Explanation of Cryptography



ABC ?.9 ABC

Symmetric (DES, 3DES, RC4, IDEA, AES)

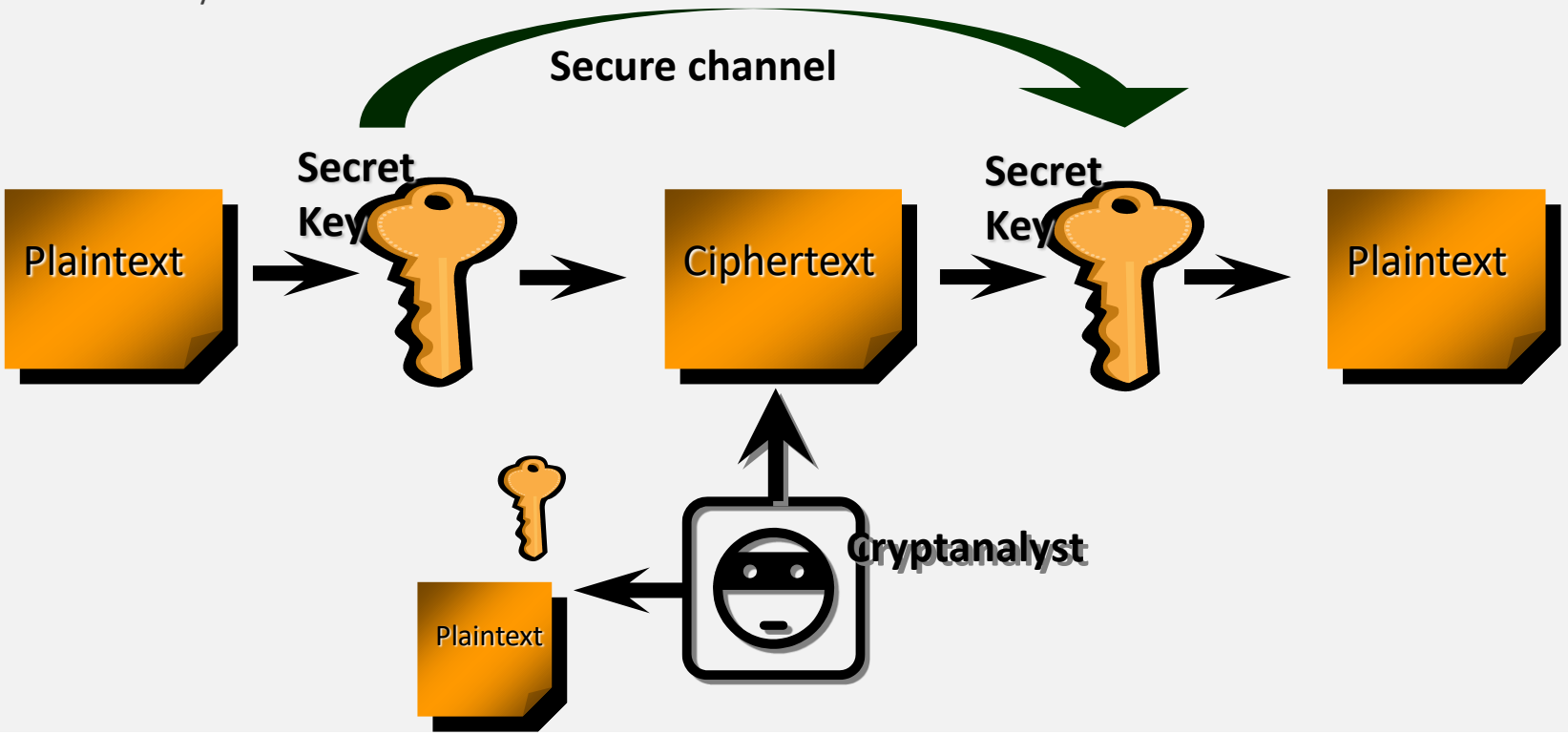


Secret-key Algorithms

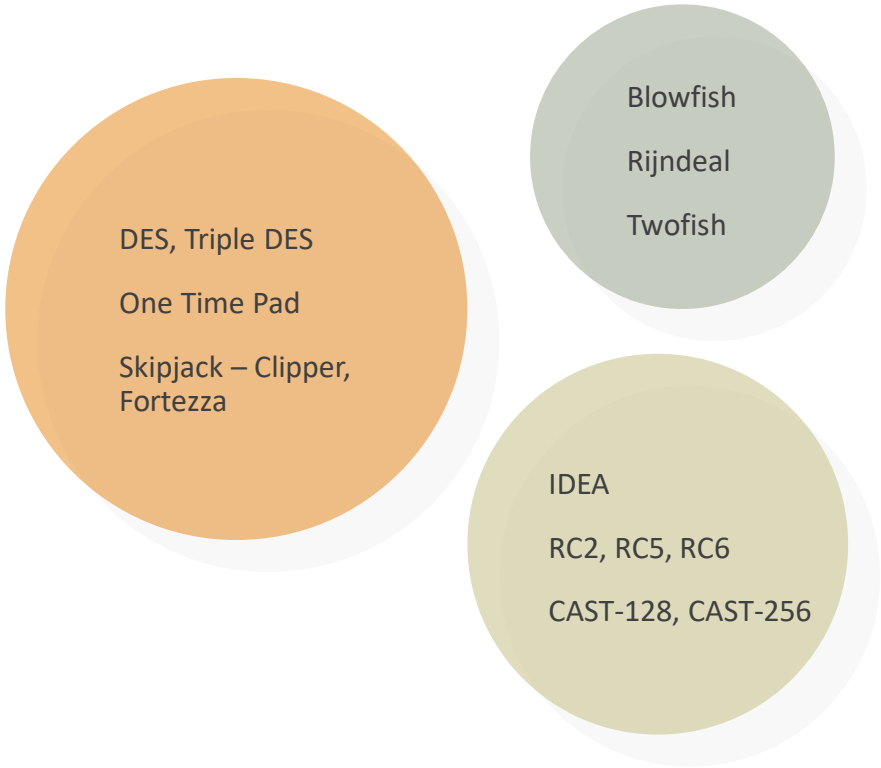
aka private key/single key/symmetric key algorithms

Same key used at encryption & decryption

- shared key

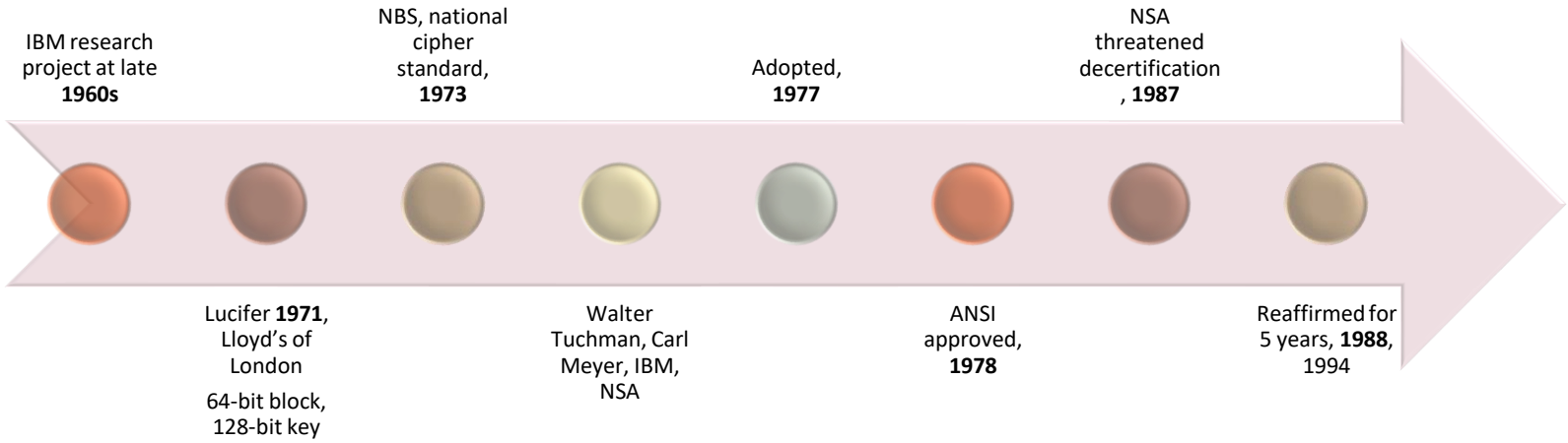


Examples of Secret-key Algorithms



Crypton, Deal, DFC, E2, Frog, HPC, Loki97, Magenta, Mars, Safer+, Serpent, …

Data Encryption Standard (DES) History



DES Algorithm

64-bit message block

56-bit key & 8 bit parity bits

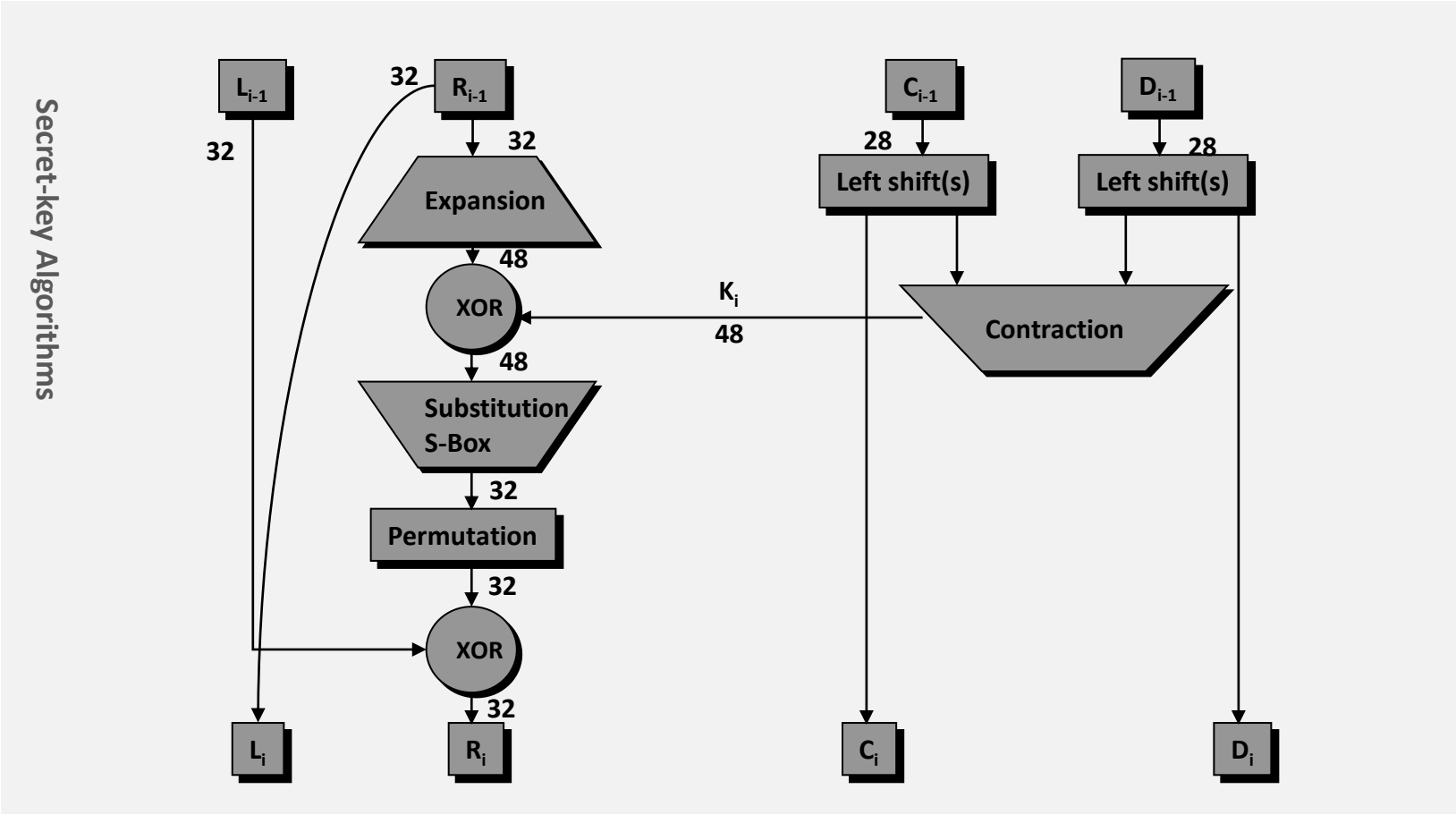
16 rounds of simple operations

- Permutation
- Transposition
- Substitution, 8 S-Boxes
- Diffusion, confusion
- Non-linearity

Decryption: Reverse the process

Software/hardware implementation

DES Algorithm Description



DES Algorithm Properties

Algorithms is publicly published

Design criteria of algorithms is NOT published

Brute force attack is the only possible way

- Some weak keys
- Some weakness in the S-boxes

Trying all possible keys is not economically justifiable (at that time)

DES Algorithm Cracked?!

Diffie & Hellman, **1977**

- Postulated parallel machine with 1 million encryption devices
- Each device, 1000 keys/second
- 10 hours break time
- US$20m

Wiener, **1993**

- Pipelined techniques, 5670 key search chips, 50M keys/second
- 35 hours
- US$100K
- 6 times faster at 1997

DES Algorithm Cracked?! (cont'd)

RSA DES secret-key challenge

- http://www.rsasecurity.com/rsalabs/node.asp?id=2100
- 1997
- Rocke Verser, distributed cracking program
- 18 Feb, 1997 started, 96 days
- ¼ of key space searched

Feb 1998 – 41 days

July 1998 – 56 hours

Jan 1999

- "Deep Crack", Electronic Frontier Foundation (EFF)
- 10,000 PCs
- 22 hours 15 minutes!

DES Applications

Federal agencies/department

- Unclassified sensitive or high value data
- Stored in medium vulnerable to theft

Non-Federal governments

- Use encouraged
  - Unclassified sensitive or high value data
  - Communicated or stored
- Government contractors
- Financial institutions
  - Exportable EFT & financial data

NIST recommends the use of DES applications other than extreme sensitivity

RC4 (Rivest Cipher)

Developed by Ron Rivest, 1987

Synchronous Stream Cipher

- Variable key size

10 times faster than DES

Widely used in modern applications

RC5, RC6

RC5 designed in 1994, RC6 2003

Block Cipher

- variable block size
- variable key size
- variable number of rounds

2040-bit max

RC6 is one of the 5 finalists in AES

RC5 Algorithm Cracked?!

RC5-64 challenge

http://www.rsasecurity.com/rsalabs/node.asp?id=2100

26 September 2002

- distributed.net
- 331,252 voluntary PCs
- 4 years!

Triple DES (3DES)

DES is not secure enough now

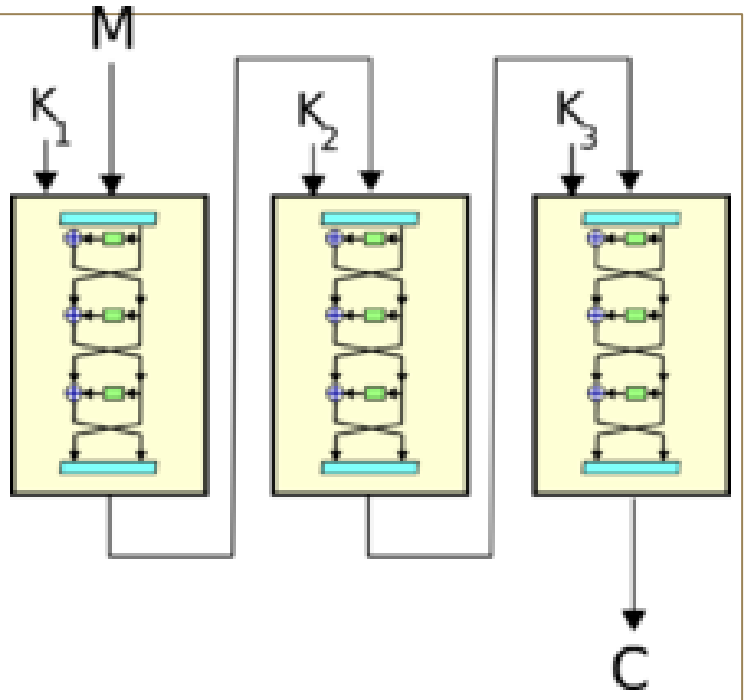
- Temporary replacement

Why NOT double DES?

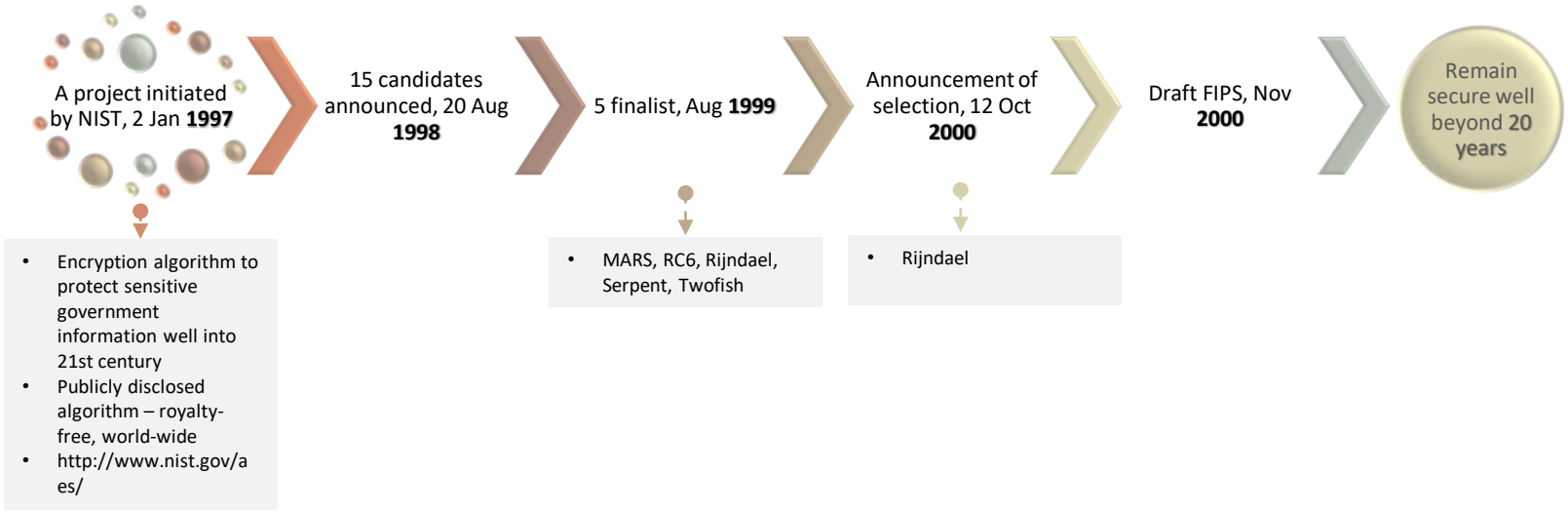
- Reported key length is 112 bits, but …
- Same work factor as DES
  ➔ No more secrecy

3DES: C := Ek1(Dk2(Ek3(M)))

- Why NOT E(E(E(M)))?
- 112 bits!
- No successful attacks reported
- Suitable for “Top Secret” data
- What if k1 = k2 = k3?



Advanced Encryption Standard (AES) History



AES Winner: Rijndael

Developed by 2 Belgium researchers

- Dr. Joan Daemen, Dr. Vincent Rijmen

Properties

- 128-bit message blocks
- Block Cipher – 128, 192, 256 bit key size
- 10, 12, 14 rounds
- small & fast
- ‘Round’ has 4 byte-oriented transformations
  - S-box substitution
  - Shifting array rows by different offsets
  - Mixing data within each column
  - Adding a ‘round key’ the state

AES: Rijndael (cont'd)

WHY selected?

- VERY good performance in both hardware & software
- Various computing environments, feedback or non-feedback modes
- Low memory requirement
- Instruction-level parallelism

Applications of Secret Key Algorithm

Data Storage

- Prevent disclosure

Telecommunication

- Prevent disclosure
  - Transmitting data
- Message authentication
  - Detect malicious insertion
  - Detect malicious deletion
  - Detect malicious modifications
  - Detect replays…
- Digital signature
  - Source verification

One Time Pad

Unbreakable by exhaustive search

(Truly) Random key same length as the message

- Perfect secrecy
- Random ciphertext with no statistical relationship with plaintext

Only used ONCE

Very high overhead – not practical for bulk data

XOR Calculations

Key & message both streams of bits

- 8-bit text character
- Each key bit **XOR** with corresponding message bit
- e.g. – One Time Pad –

```
plain:      01010101
key:        11001110
cipher:     10011011
```

Input		Output
0	0	0
0	1	1
1	0	1
1	1	0

One-time Pad (1917)

```
Message   s   e   c   r   e   t
         18   5   3  17   5  19
OTP +    15   8   1  12  19   5
          7  13   4   3  24  24
          g   m   d   c   x   x
```

OTP is unbreakable provided

- Pad is never reused (VENONA)
- Unpredictable random numbers are used (physical sources, e.g. radioactive decay)

One-time Pad Limits

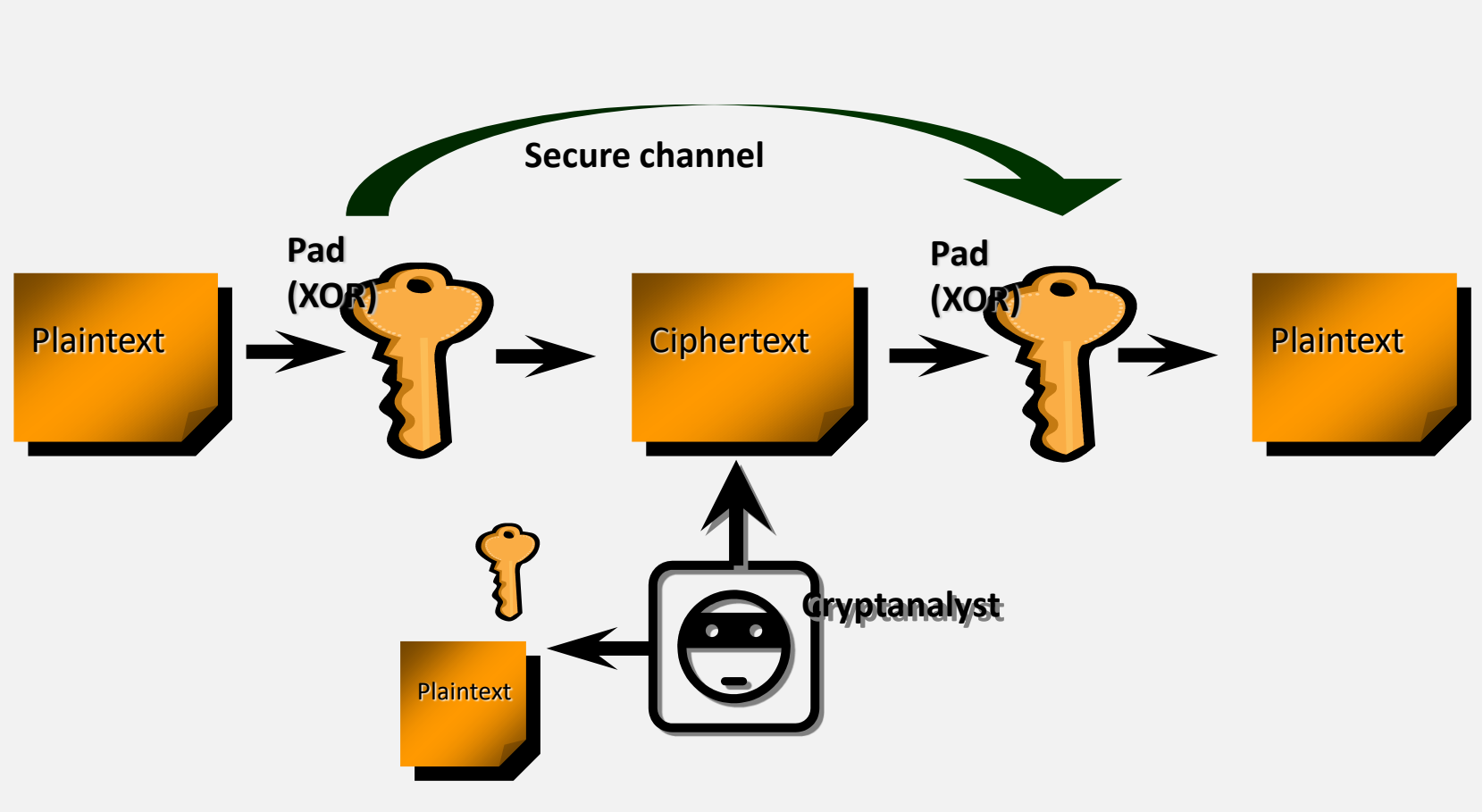
Not suitable for large scale commercial or military encryption:

1. key size problem = key as large as the total volume of encrypted information,
2. constant demand for new, truly random keys,
3. both sender and receiver have to hold and defend identical copies of enormous keys.

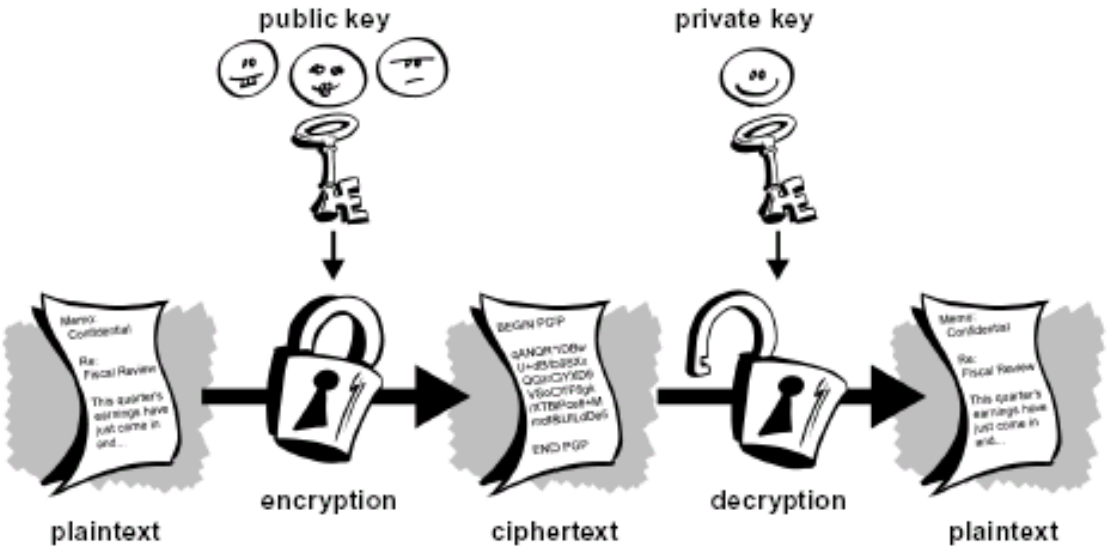
Pseudo- randomness on computers:

- Must seed, then generate, depends on algorithm.
- Seed with mouse movements, noise, static.
- Cannot duplicate/coordinate if truly random.

One Time Pad (cont'd)



Asymmetric (RSA, Diffie-Hellman, Elliptic Curve, ElGamal)

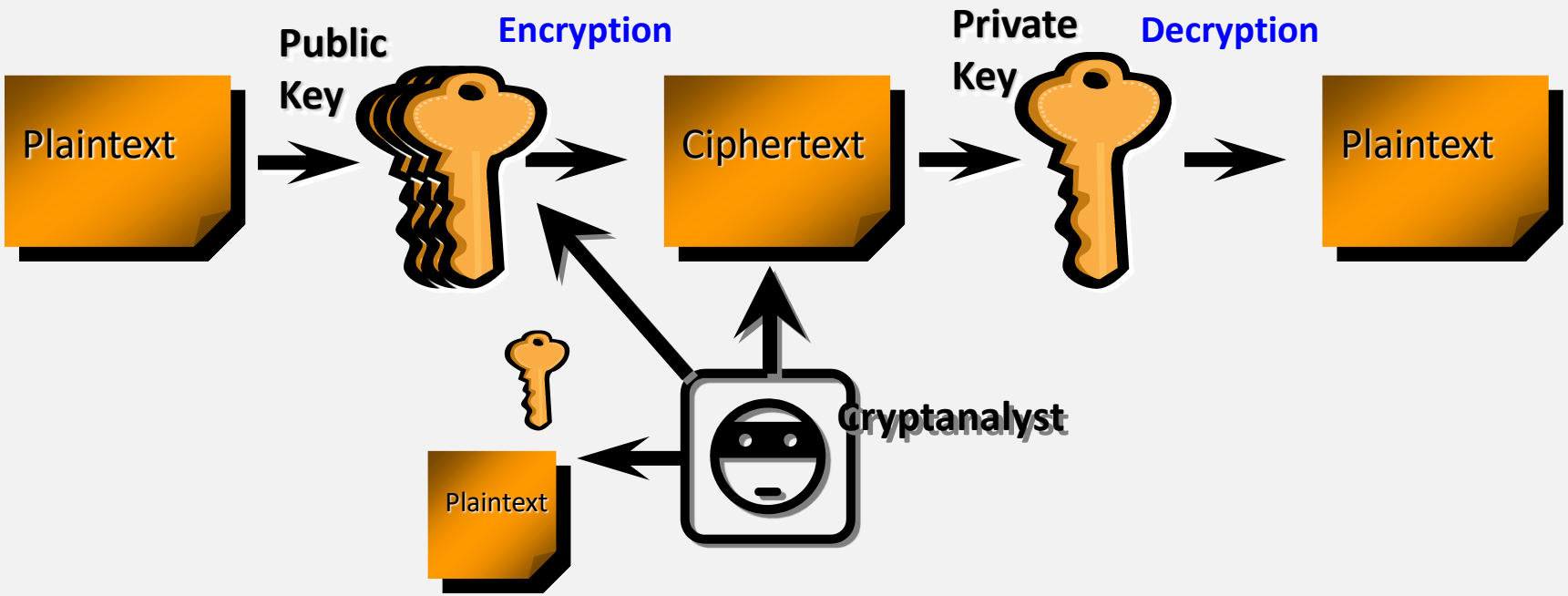


Public-key Algorithms

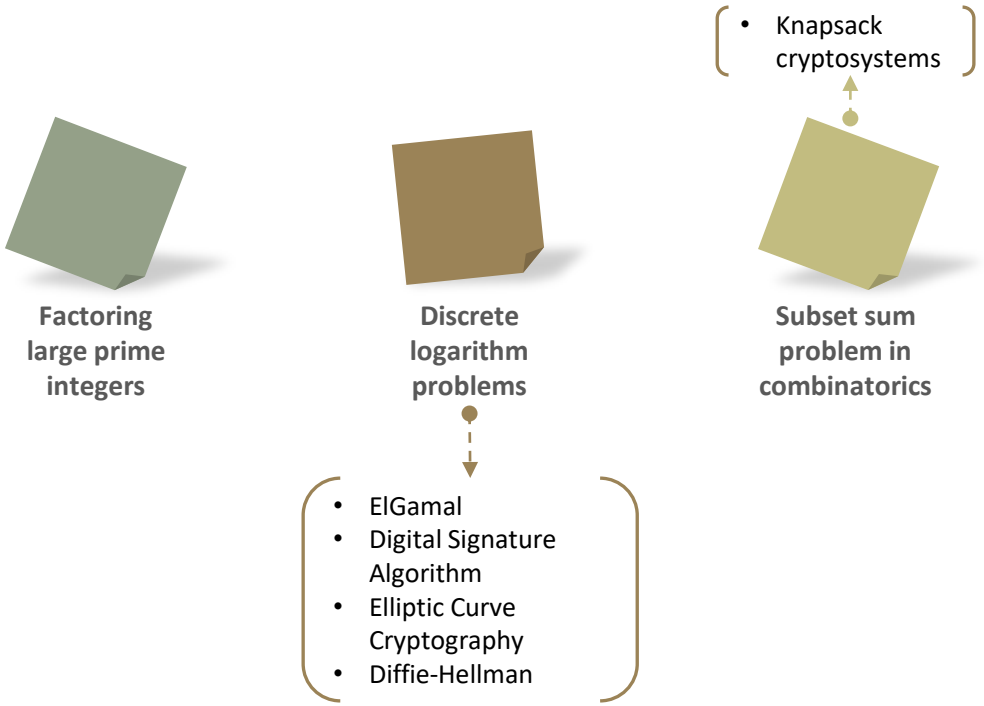
aka Asymmetric-key algorithms

Different keys and algorithms at encryption and decryption

- Dual keys



Examples of Public-key Algorithms



Diffie-Hellman

First published public-key algorithm

Key exchange algorithm

- Key distribution

Discrete logarithm problem

Problems

- No authentication
- Vulnerable to Man-In-the-Middle attack

Can be enhanced to have authentication

RSA

Developed by Rivest, Adi Shamir, Len Adleman, 1977

Block Cipher

Encryption and Digital Signatures

Factoring large prime integers problem

Widely adopted in various applications, standards and PKI

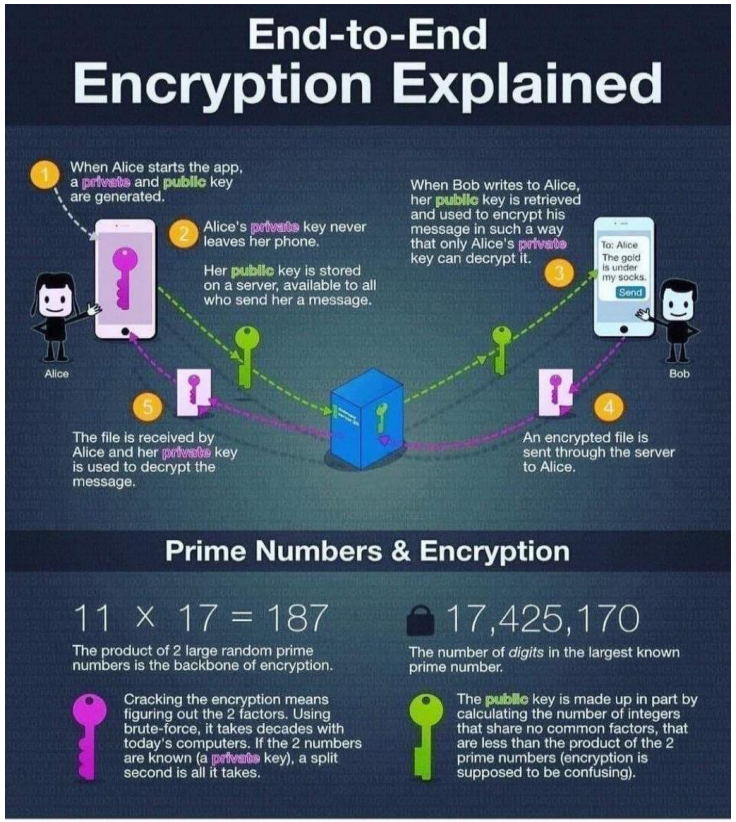
RSA Operation

Key generation

- Generate prime numbers – p, q
- Calculate n = pq and $\phi(n) = (p-1)(q-1)$
- Select e, $\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
- Calculate $d = e^{-1} \bmod \phi(n)$
- Public key := (e,n)
- Private Key := (d,p,q)

Encryption: C = Me(mod n), M < n

Decryption: M = Cd(mod n)



RSA Cracked?!

Key requirement

- p and q should differ in length by only a few digit
- Both (p-1) and (q-1) should contain a large prime factor
- gcd(p-1,q-1) should be small

RSA Factoring Challenge

RSA-155 challenge completed, 1999

- General Number Field Sieve
- 8000 MIPS year
- 7.4-month duration
- http://www.rsasecurity.com/rsalabs/node.asp?id=2098

RSA Factoring Challenge

- RSA-160 factored
- RSA-200 factored
- RSA-576 factored
  3 December 2003
- RSA-640 factored (LATEST)
  2 November 2005
  5 calendar months

Security of RSA

Unlike DES, you can choose different sizes of the numbers p,q,n for your purpose.

Commonly used 512, 1024, 2048-bit long n

Year	# of decimal digits factored	How many times harder to factor a 512-bit number
1983	71	> 20 million
1988	90	250,000
1989	100	30,000
1993	120	500
1994	129	100

Security of RSA

# of bits	Mips-years required to factor
512	30,000
768	$2*10^8$
1024	$3*10^{11}$
1280	$1*10^{14}$
2048	$3*10^{20}$

How long is 10^{20} years?

ElGamal

Similar security to RSA

Enhancement of Diffie-Hellman

- Encryption
- Digital Signatures

Discrete logarithm problem

Cons

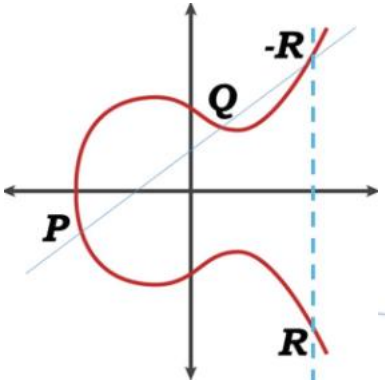
- Need for randomness
- Slower speed, esp. signing
- Message expansion

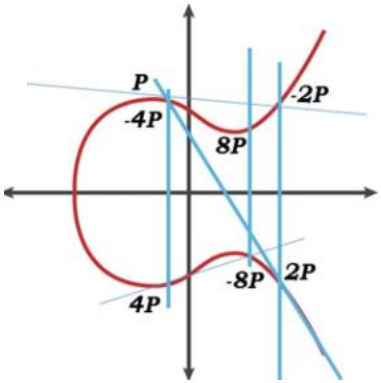
First public-key Cryptography suitable for encryption & digital signatures without patents

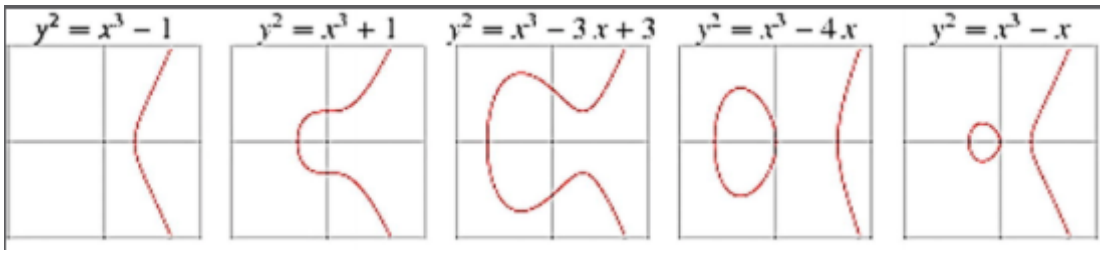
Elliptic Curve Cryptography (ECC)

ECC

- Proposed by Neal Koblitz & Victor Miller, 1985
- Operating on elliptic curves over a finite field
- Compared to RSA, ECC offers similar secrecy with shorter key length
- Provides Key Exchange Protocol (ECDH)
- Elliptic curve discrete logarithm problem (ECDLP)
- Mostly used in Digital Signatures, ECDSA







From “Beginning Blockchain”

Elliptic Curve Cryptography (ECC)

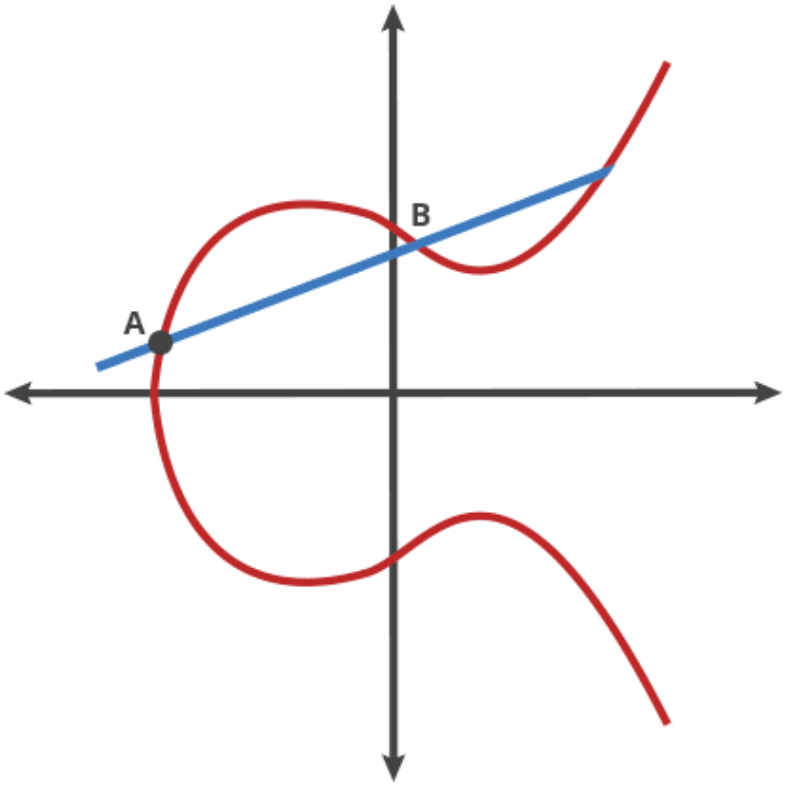
An elliptic curve cryptosystem can be defined by picking

- a prime number as a maximum,
- a curve equation, and
- a public point A on the curve

Repeat this process n times (this process represents a dot product.)

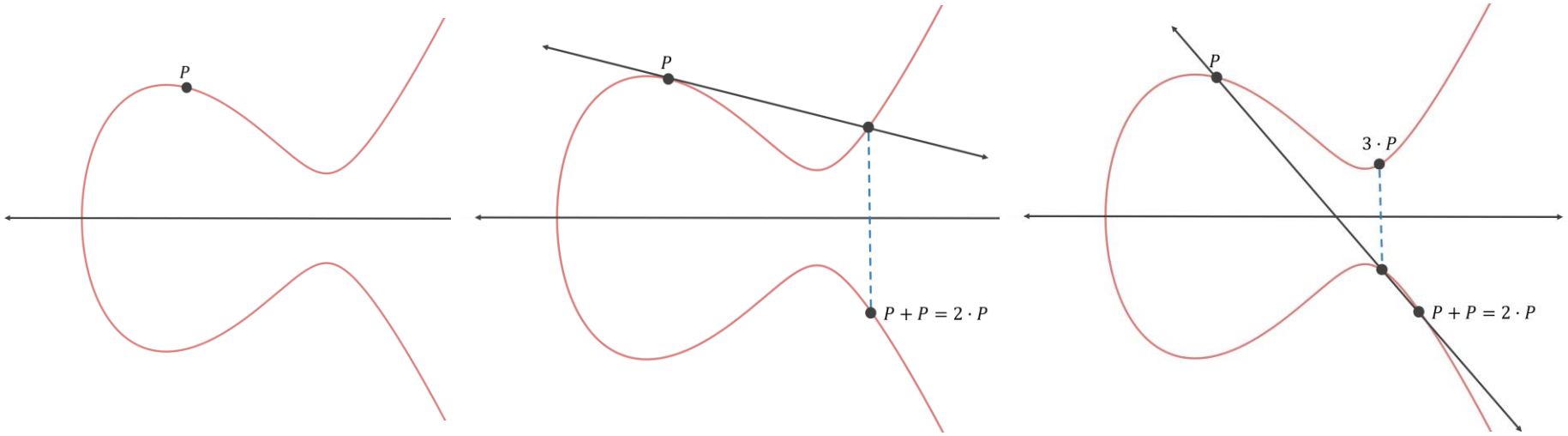
It turns out that if you have two points, an initial point 'bounced' with itself n times to arrive at a final point, finding out n when you only know the final point and the first point is hard.

In an elliptic curve cryptosystem, a private key is a number **n**, and a public key is **the public point A** 'bounced' with itself n times



https://dzone.com/articles/signing-and-verifying-ethereum-signatures?fromrel=true

Elliptic Curve Cryptography (ECC)



secp256k1 curve:
x-coordinate:
55066263022277343669578718895168534326250603453777594175500187360389116729240
y-coordinate:
32670510020758816978083085130507043184471273380659243275938904335757337482424

Elliptic Curve Cryptography (ECC)

Let's say I compute X=x•P,

- where x is a random 256-bit integer.
- The result will be some point on the curve.

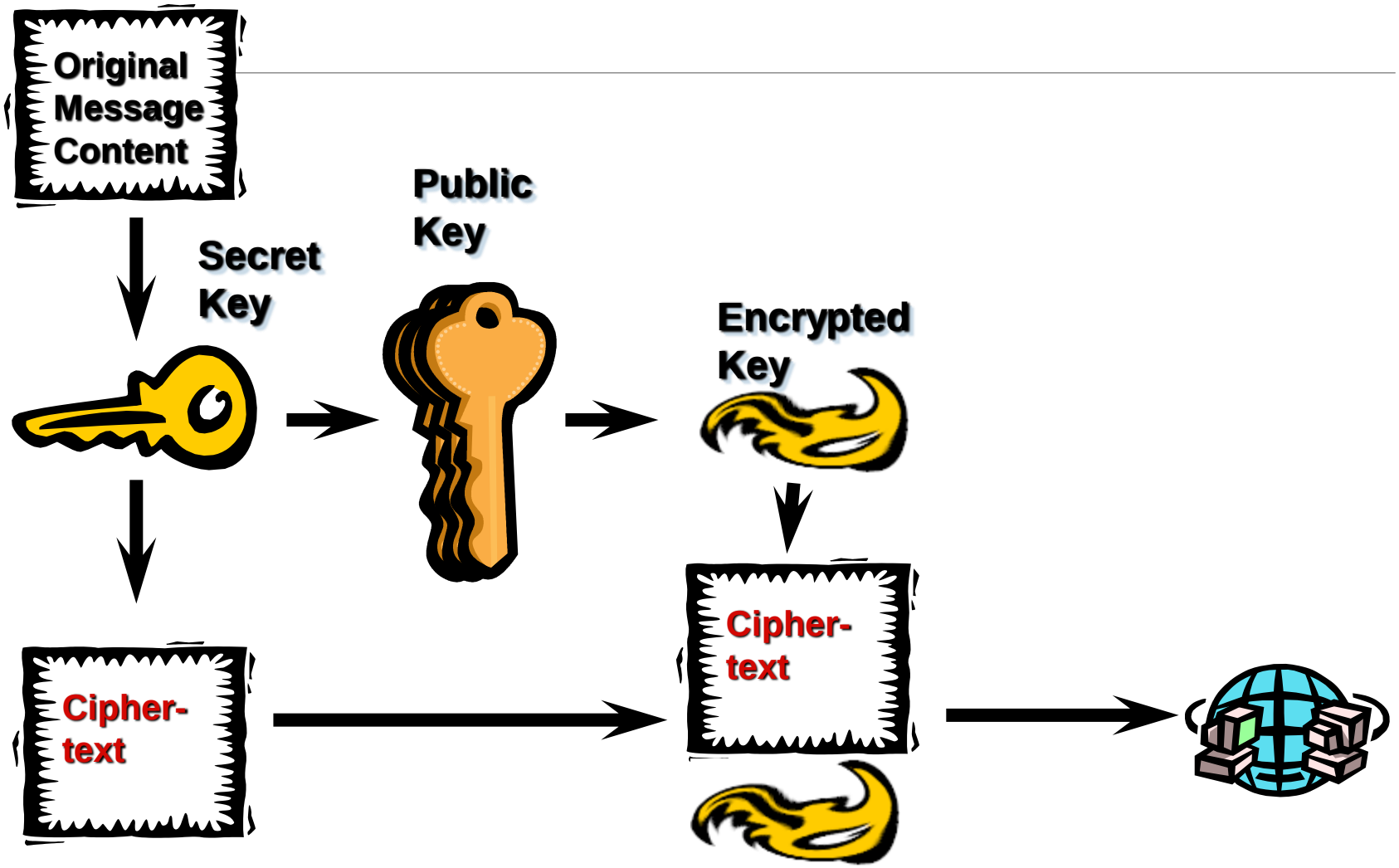
Could you determine how many times I added P to itself to get the point X on the curve?

Let's assume that you know what P is and you know what curve I was using.

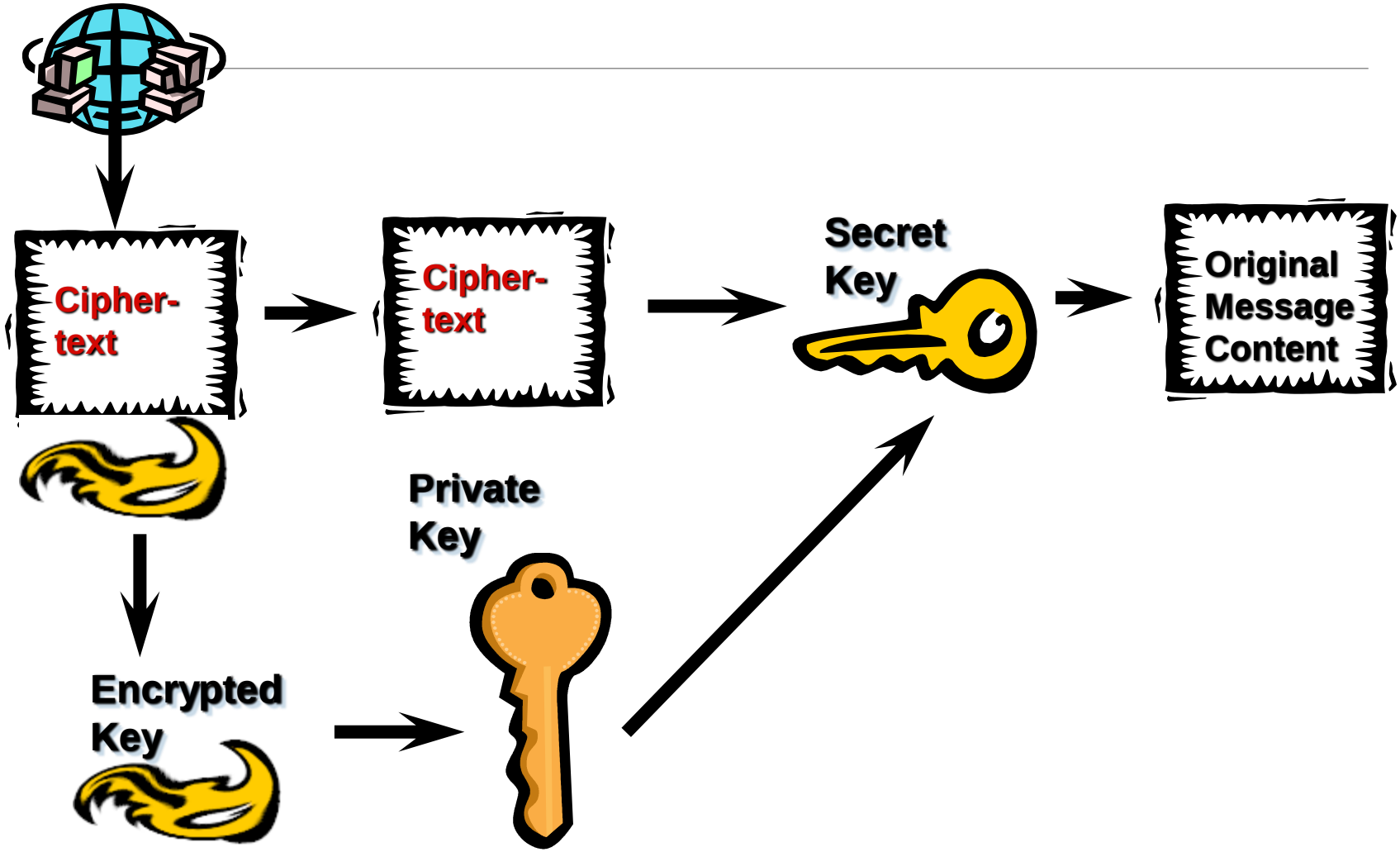
It turns out that is not feasible for you to figure out x, even if you had a super computer.

There is no known algorithm for determining x, so your only option is to keep adding P to itself until you get X or keep subtracting P from X until you get P.

Secure envelope approach -- encryption



Secure Envelope Approach – Decryption



ECC Advantages

By far, best secrecy over bit length

Bit savings

- Computation power
- Bandwidth
- Storage
- Less error prone

Shorter digital signatures & certificates

Ideal for very small hardware implementation

- smart card

Encryption and digital signatures stages separable to simplify export

ECC is Favorable, When …

Computation power limited

Integrated circuit space limited

ROM/RAM limited

Bandwidth limited

Intensive signing, verifying, authenticating required

Fast speed needed

ECC & RSA: Key size

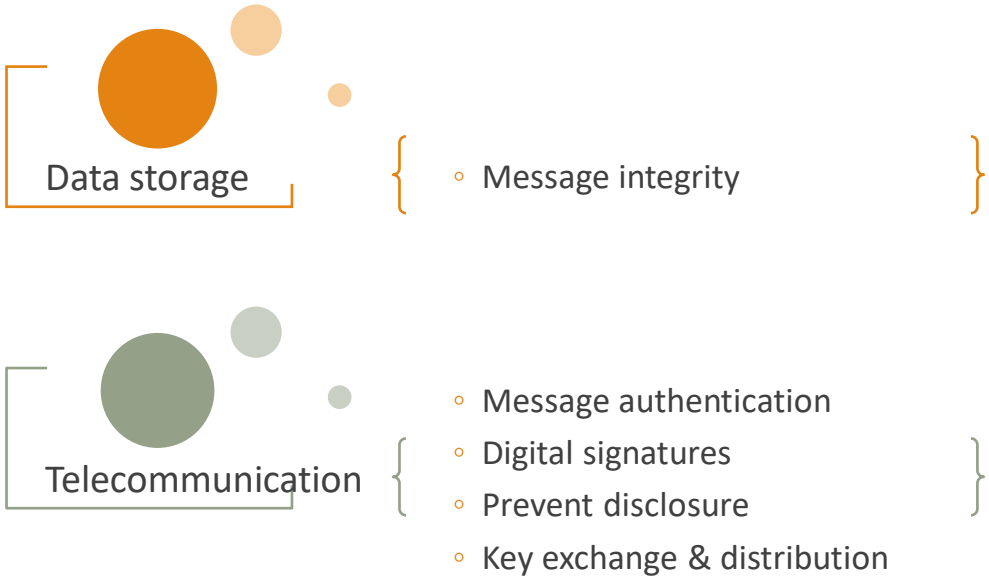
Time to break in MIPS year	RSA key size	ECC key size	RSA/ECC key size ratio
10^4	512	106	5:1
10^8	768	132	6:1
10^{11}	1024	160	7:1
10^{20}	2048	210	10:1
10^{78}	21000	600	35:1

ECC & RSA: Performance

	ECDSA or ECES over GF(q), 160bit	RSA with 1024-bit n, e=216+1 and CRT	Discrete logarithm systems with 1024 bit prime
decryption	**120**	**17**	**480**
encryption	**60**	**384**	**240**
signing	**60**	**384**	**240**
verification	**120**	**17**	**480**

Elliptic Curve Cryptosystems, Technical Note, *M.J.B. Robshaw & Yiqun Lisa Yin*, RSA Laboratories

Applications of Public-key Algorithms



Public-key Algorithms Standards

Public-key Cryptography Standard (PKCS)
- Developed by RSA Security

Digital Signatures
- FIPS 180-1, 186-1
- ISO/IEC 9796, 10118, 14888

IEEE P1363 (public-key cryptography)
- Covers main public key techniques
- RSA, ECC, ElGamal, Diffie-Hellman

ANSI
- RSA: ANSI X9.30, X9.31, X9.44
- ECC: ANSI X9.F.1, X9.62, X9.63
- DH: ANSI X9.42

Secret key v.s. Public key

Features	Secret Key	Public Key
Number of keys	single key shared by parties	pair of keys for each party
Types of keys	key is secret	public & private
Protection	disclosure & modification	private key: disclosure & modification public key: modification
Relative speed	faster	slower
Performance	more efficient	less efficient
Key length	fixed key length	variable key length
Application	files and communication channels	key distribution, authentication
Scalability	Bad	Good

Key Sizes and Algorithms

Conventional vs public-key vs ECC key sizes

Conventional	Public-key	ECC
(40 bits)	—	—
56 bits	400 bits)	—
64 bits	512 bits	—
80 bits	768 bits	—
90 bits	1024 bits	160 bits
112 bits	1792 bits	195 bits
120 bits	2048 bits	210 bits
128 bits	2304 bits	256 bits

Block and Stream Cipher

Block Ciphers vs. Stream Ciphers

Stream

- Not suitable for software implementation
  - time consuming manipulation of bits
- Easier to analyze mathematically
- Single error can damage only a single bit of data
- Application: T-1 link between 2 computers

Block

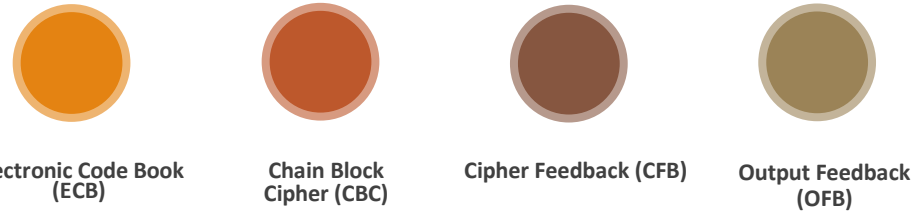
- Easy to implement in software
- General in use and algorithms are more strong
- Single error can damage a block's worth of data
- Application: data on computer desk

Encryption Modes of Operation

Block Cipher, in particular

Encrypt a chain of message blocks

4 modes



Electronic Codebook (ECB)

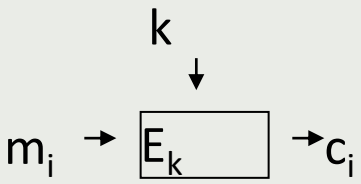
Secure transmission of single values

NOT for lengthy message

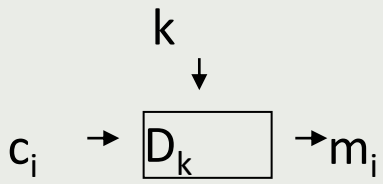
- Subject to cryptanalyst

Same plaintext block encrypts to same ciphertext block

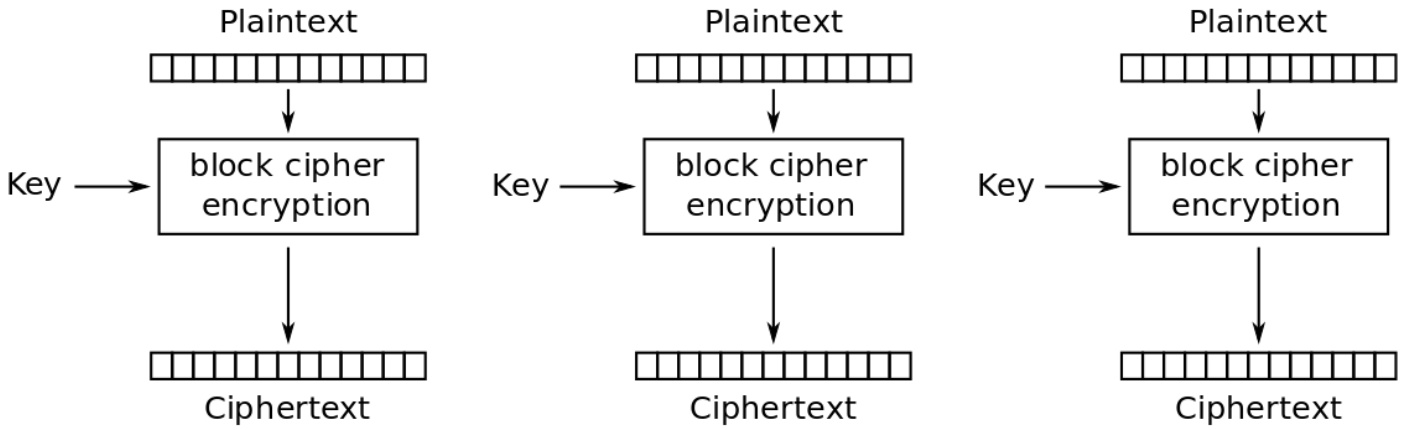
Encryption



Decryption



Electronic Codebook (ECB)



Electronic Codebook (ECB) mode encryption

Cipher Block Chaining (CBC)

General-purpose block-oriented transmission

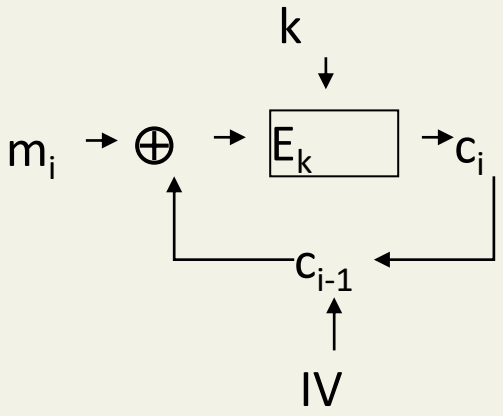
Random/Constant Initialization Vector (IV)

Encryption of lengthy messages

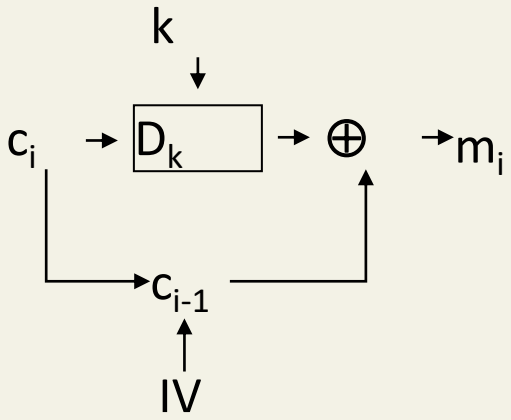
Authentication

Convert Block Cipher to Stream Cipher

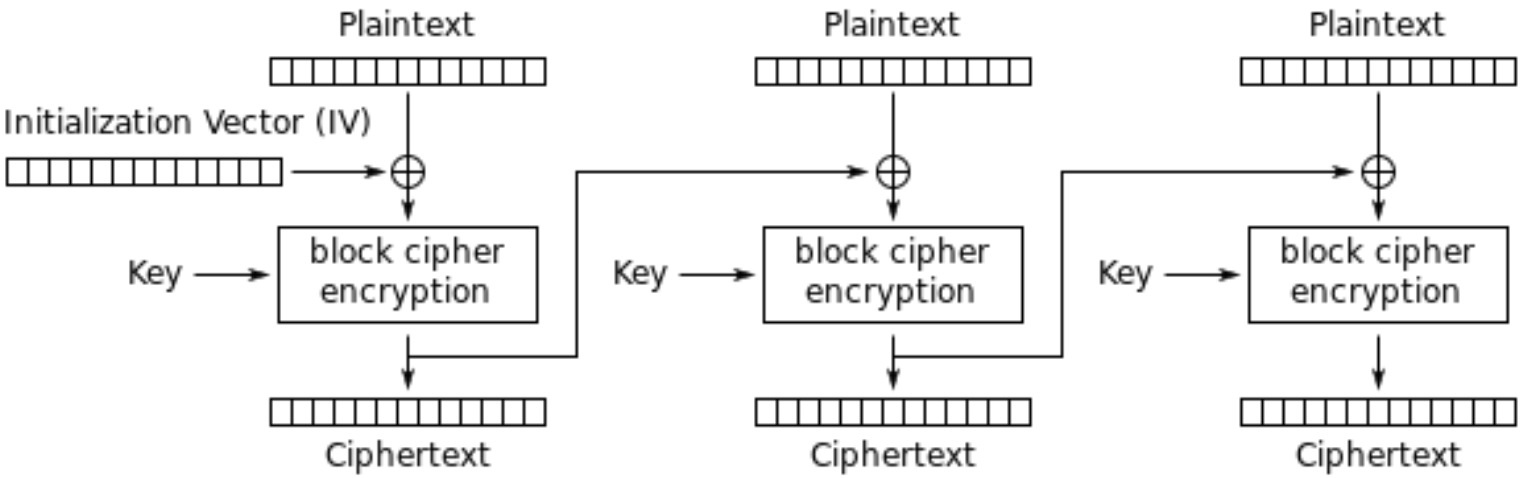
Encryption



Decryption

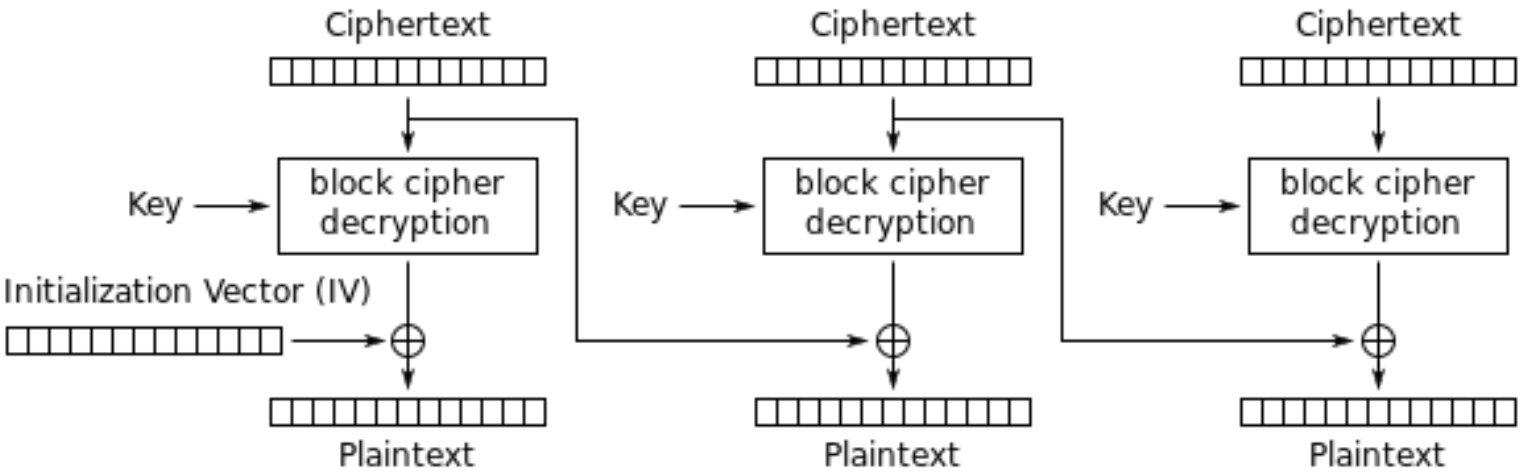


Cipher Block Chaining (CBC)



Cipher Block Chaining (CBC) mode encryption

Cipher Block Chaining (CBC)



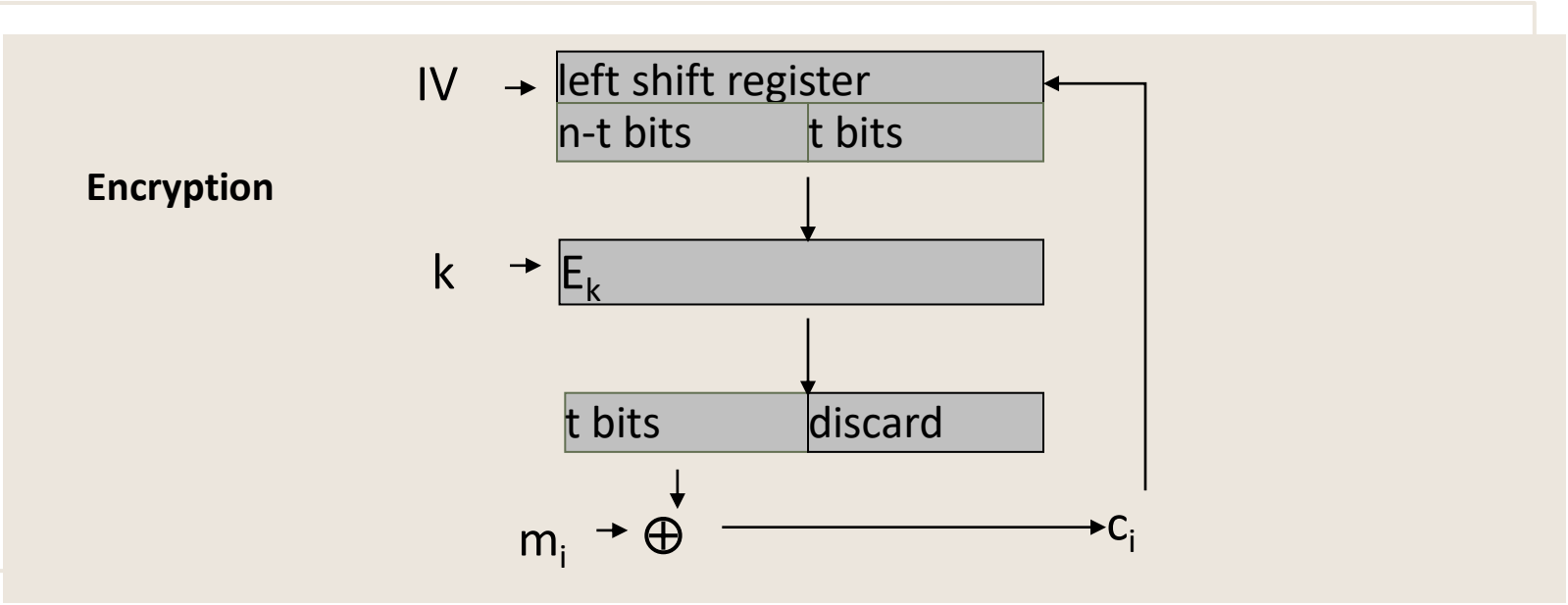
Cipher Block Chaining (CBC) mode decryption

Cipher Feedback (CFB)

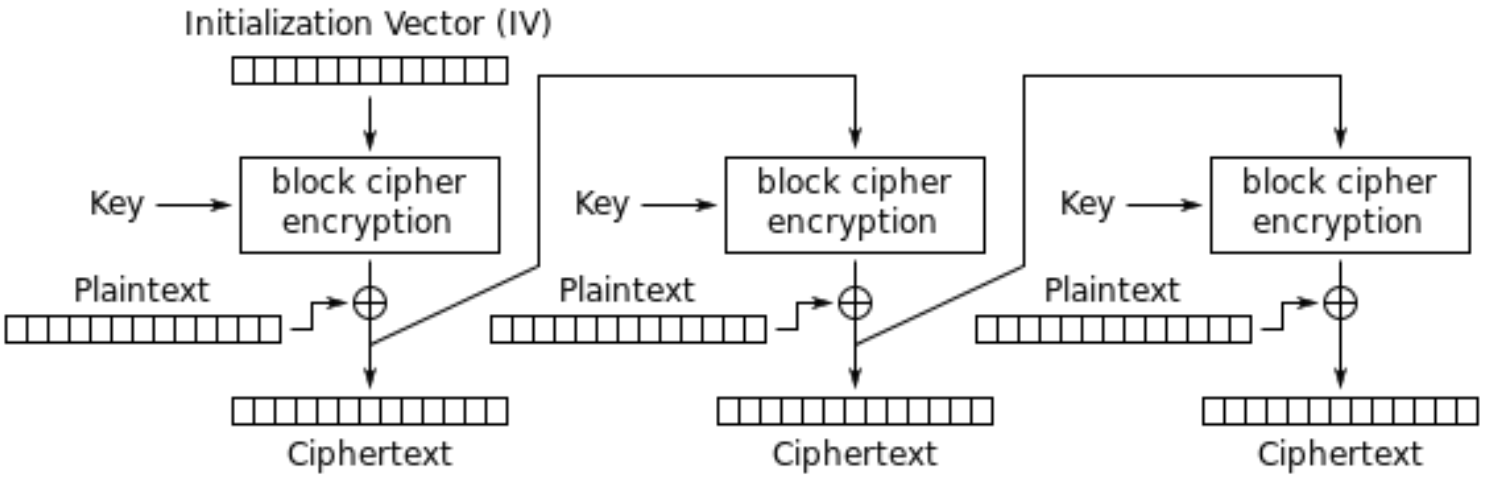
Another way to convert block cipher into (Self-Synchronous) Stream Cipher

Bit errors in transmission propagate into a number of positions

Authentication

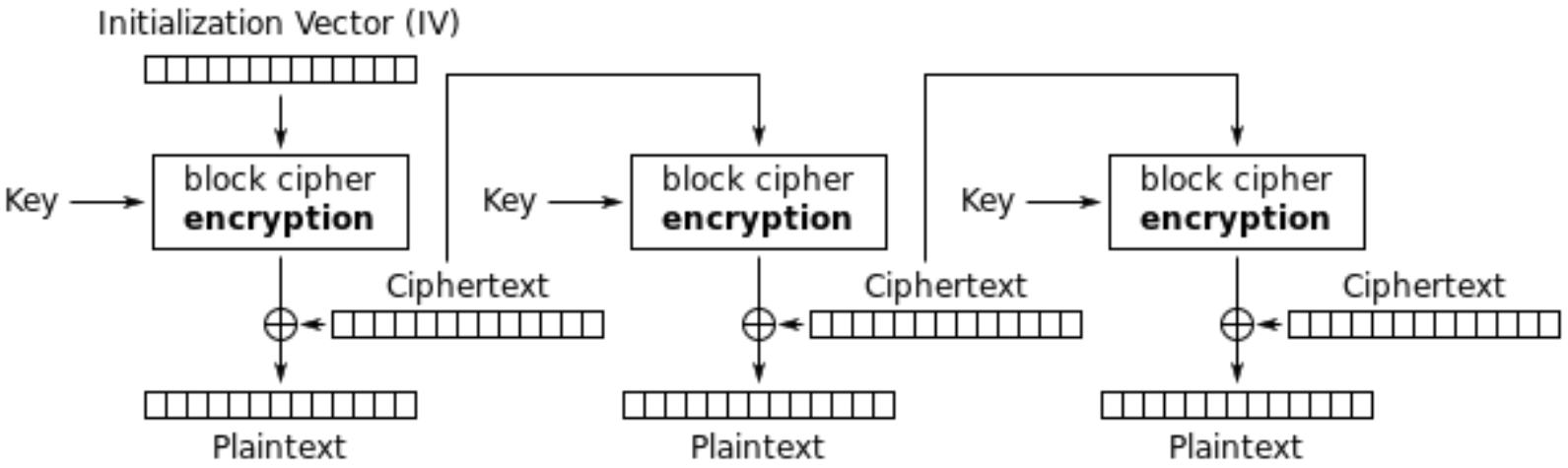


Cipher Feedback (CFB)



Cipher Feedback (CFB) mode encryption

Cipher Feedback (CFB)



Cipher Feedback (CFB) mode decryption

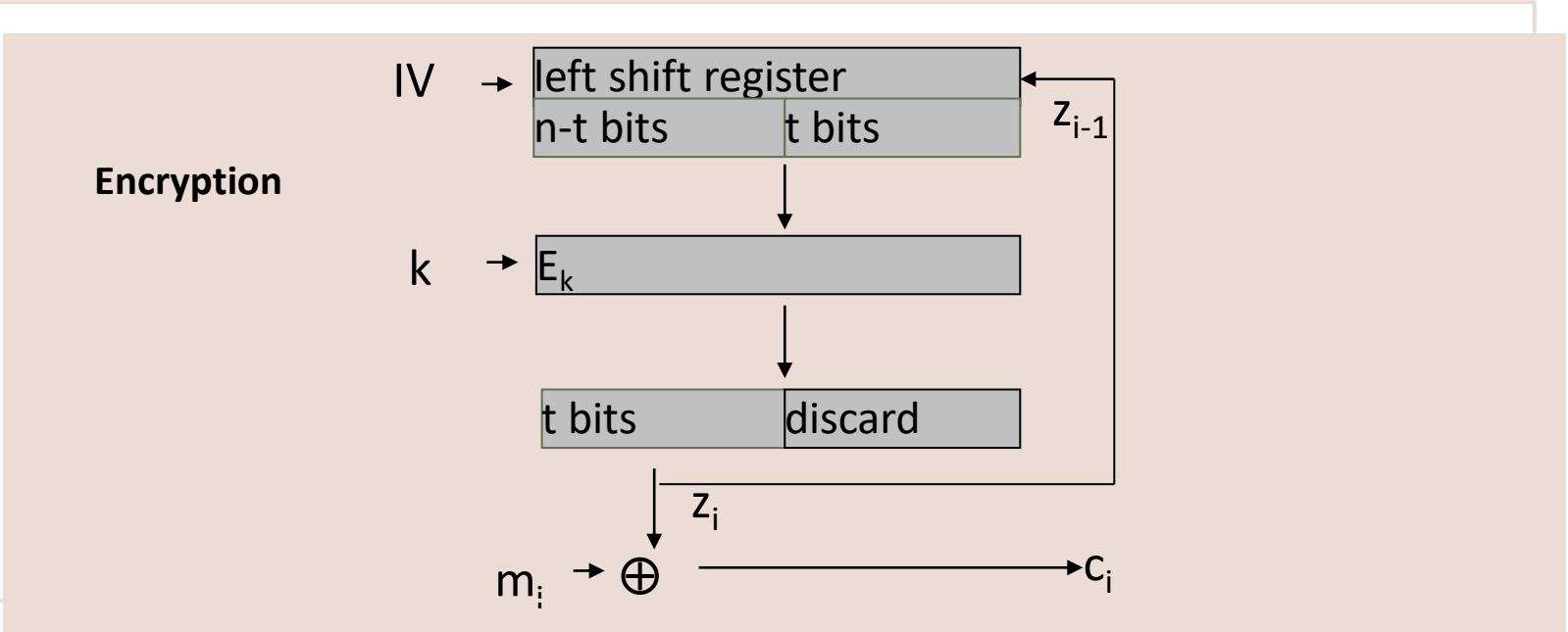
Output Feedback (OFB)

Another way to convert block cipher into (Syn-Synchronous) Stream Cipher

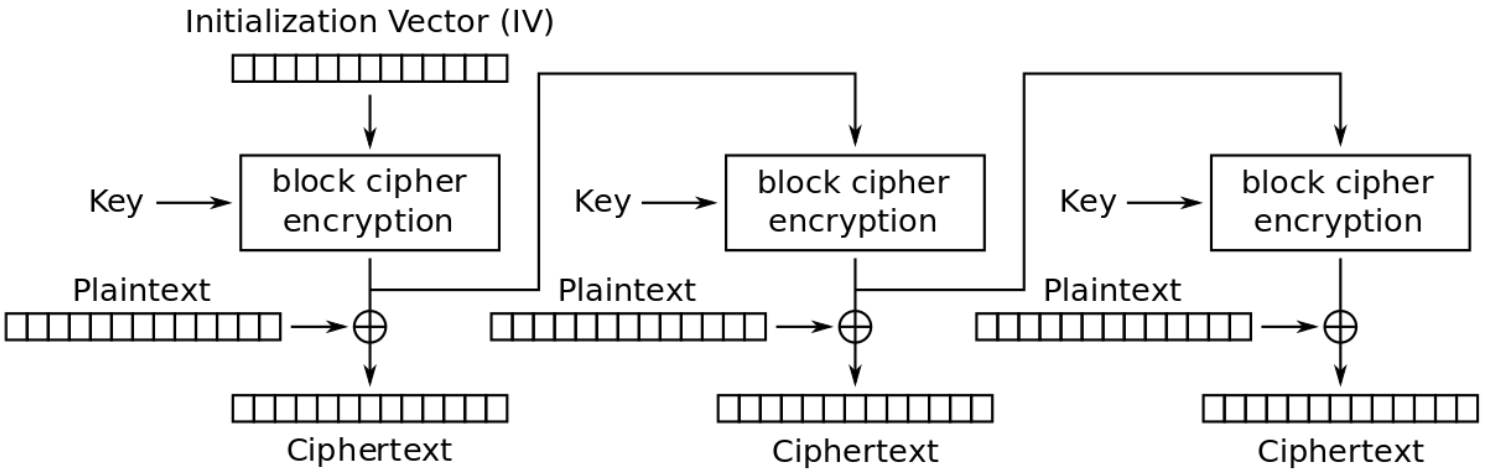
Bit errors in transmission do NOT propagate

Stream-oriented transmission over noisy channel

More vulnerable to message stream modification attack

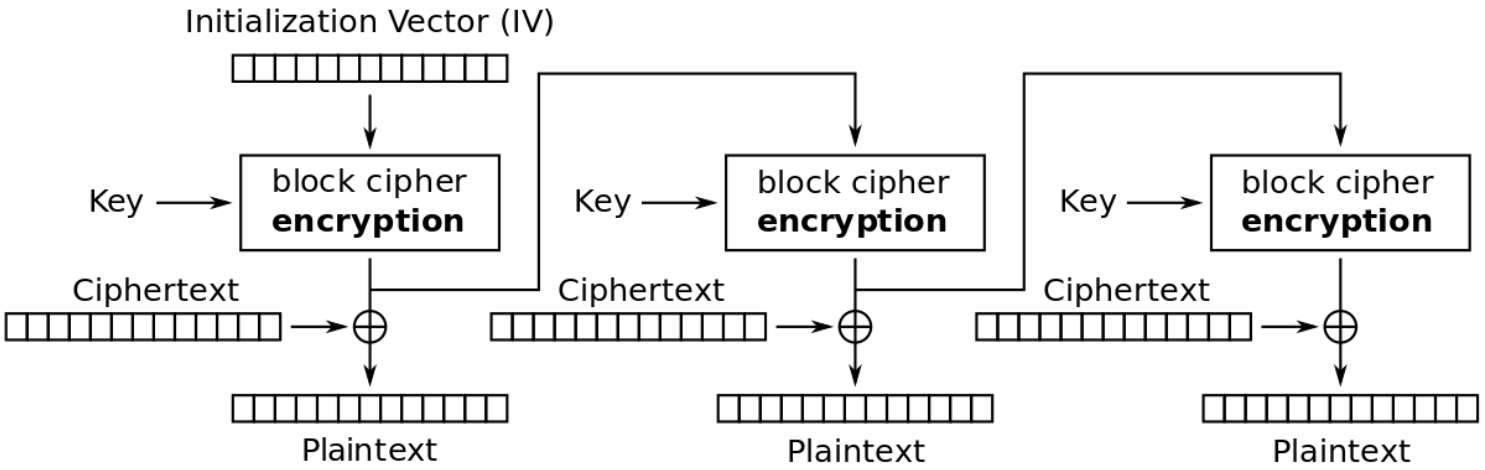


Output Feedback (OFB)



Output Feedback (OFB) mode encryption

Output Feedback (OFB)



Output Feedback (OFB) mode decryption

Hash and Digital Signature

Hashing

Hash functions, is to produce message digest

- Computationally infeasible to find a message which hashes to the same digests as a given message
- Computationally infeasible to find any two strings which hash to the same value

Message digest

- Fixed size result of hashing a message and is smaller than the full message

Digital signature

- Electronic signature of a digital message

Hash Functions

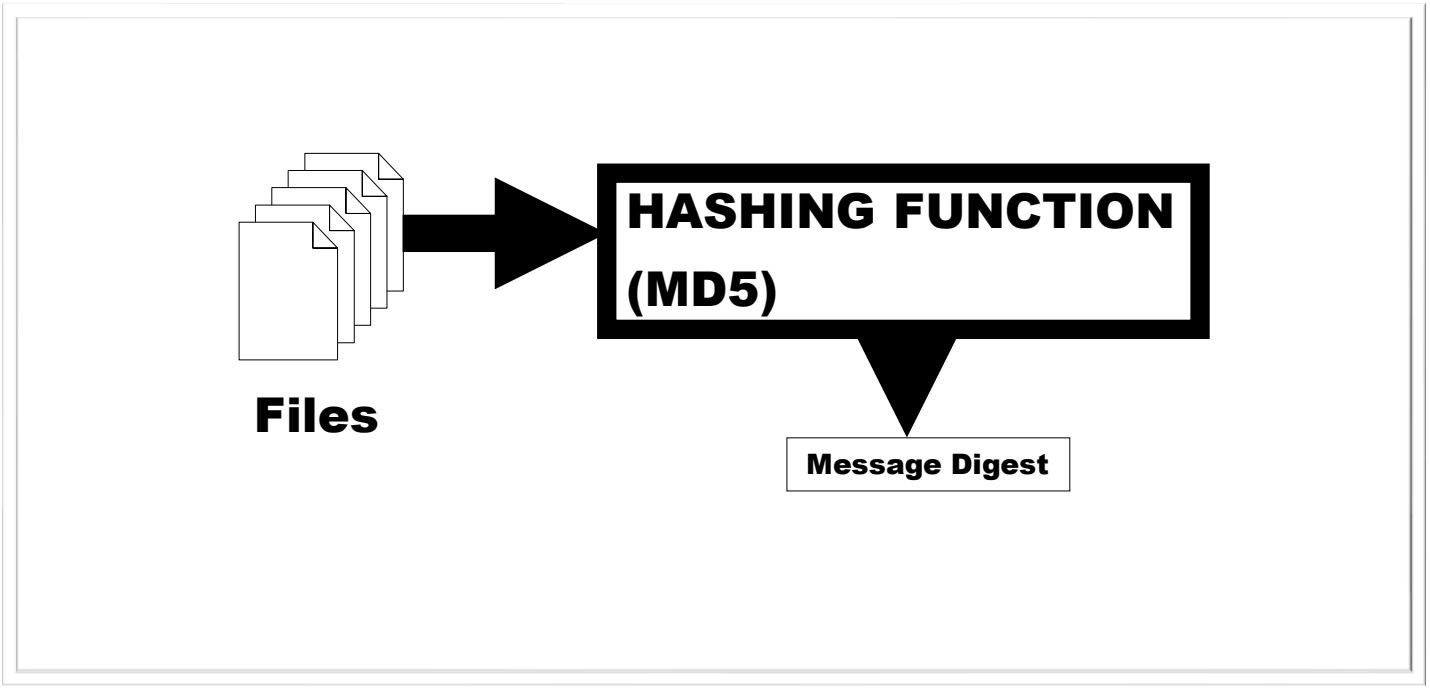
Mathematical transformation that maps arbitrary length string to a fixed-size result

Requirements – “Collision Free”

- WEAK
  Computationally infeasible to find a message which hashes to the same digests as a given message
- STRONG
  Computationally infeasible to find any two strings which hash to the same value

Birthday Attack

Hash (MD4, MD5, SHA-1, SHA-256)



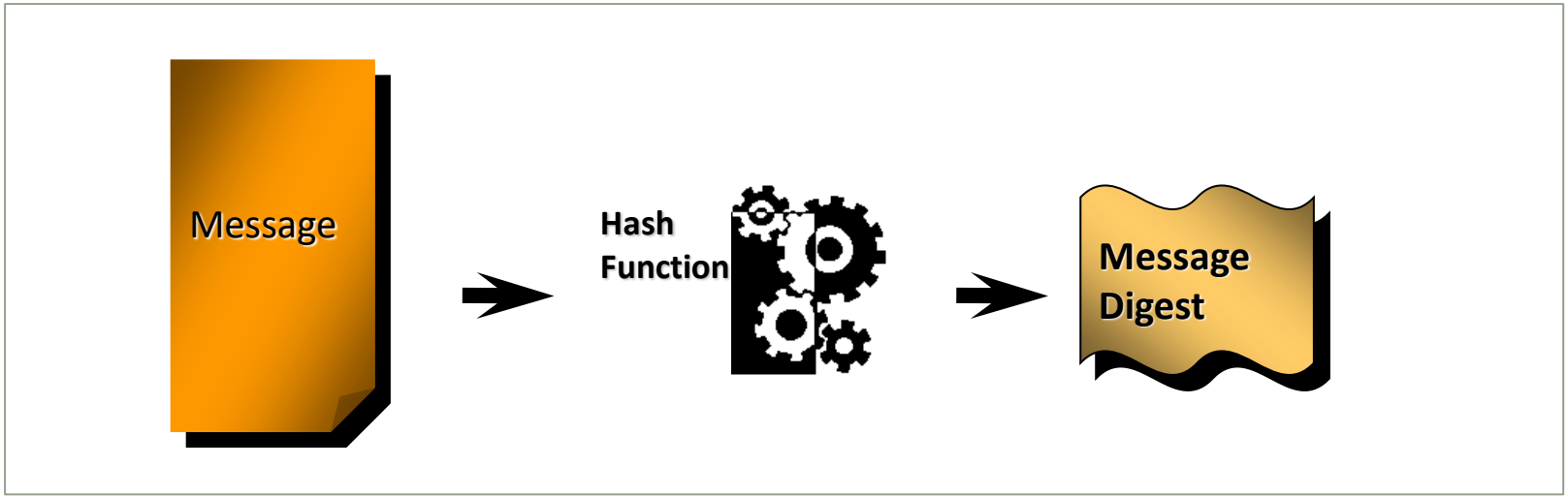
Message Digests

Fixed size result of hashing a message and is smaller than the full message

aka “Digital Fingerprint”

Example:

- MD2, MD4, MD5
- SHA, SHA-1
- RIPEMD-160



MD2, MD4, MD5

MD → Message Digest

Developed by Ron Rivest

MD5 …

- 512-bit message blocks
- 128-bit message digest
- 4 rounds, each 16 steps

Simple, compact and fast

Favor little-endian architecture

SHA, SHA-1, SHA-256, SHA-512

SHA → Secure Hash Algorithm. Firstly developed by NIST, 1993

Simple, compact and fast

- A little bit slower than MD5

Favor big-endian architecture

More resistant to brute-force attack

The algorithm for SHA-256 is specified in Federal Information Processing Standard (FIPS) 180-4

SHA-1	5 32-bit words 512-bit message blocks 160-bit message digest 4 rounds, each 20 steps
SHA-256 is similar to SHA-1	8 32-bit words => 256 bit hash message => 64 characters
SHA-512 is similar to SHA-1	8 64-bit words

MD5, SHA-1 cracked?

SHA-1, Reported in August 2004

- Research team of Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu
- mostly from Shandong University in China
- collisions in 2**69 (compared to 2**80)

MD5 and SHA-1 is considered insecure, but there are no immediate threats

Move on to SHA-256, SHA-348, SHA-512

MD5, SHA-1 cracked?

References

- Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD
  - http://www.infosec.sdu.edu.cn/paper/md4-ripemd-attck.pdf&e=10384
- Certificate Collision
  - http://www.win.tue.nl/~bdeweger/CollidingCertificates/
- New releases of SHA
  - http://csrc.nist.gov/CryptoToolkit/tkhash.html

Hash Demo

https://andersbrownworth.com/blockchain/

Digital Signatures

Electronic signature of a digital message

To achieve non-repudiation

Prevent senders from denying they have sent messages

- Authenticate the contents at time of signature
- Verify the author and the time of signature
- Can be resolved by third parties, to settle disputes

Requirements

Receiver must be able to validate sender's signature

Signature must not be forgeable computationally

Sender of a signed message must not be able to repudiate it later

Easy to generate and verify

Digital signature cannot be constant

- A function of the entire document to sign

Digital Signature (cont'd)

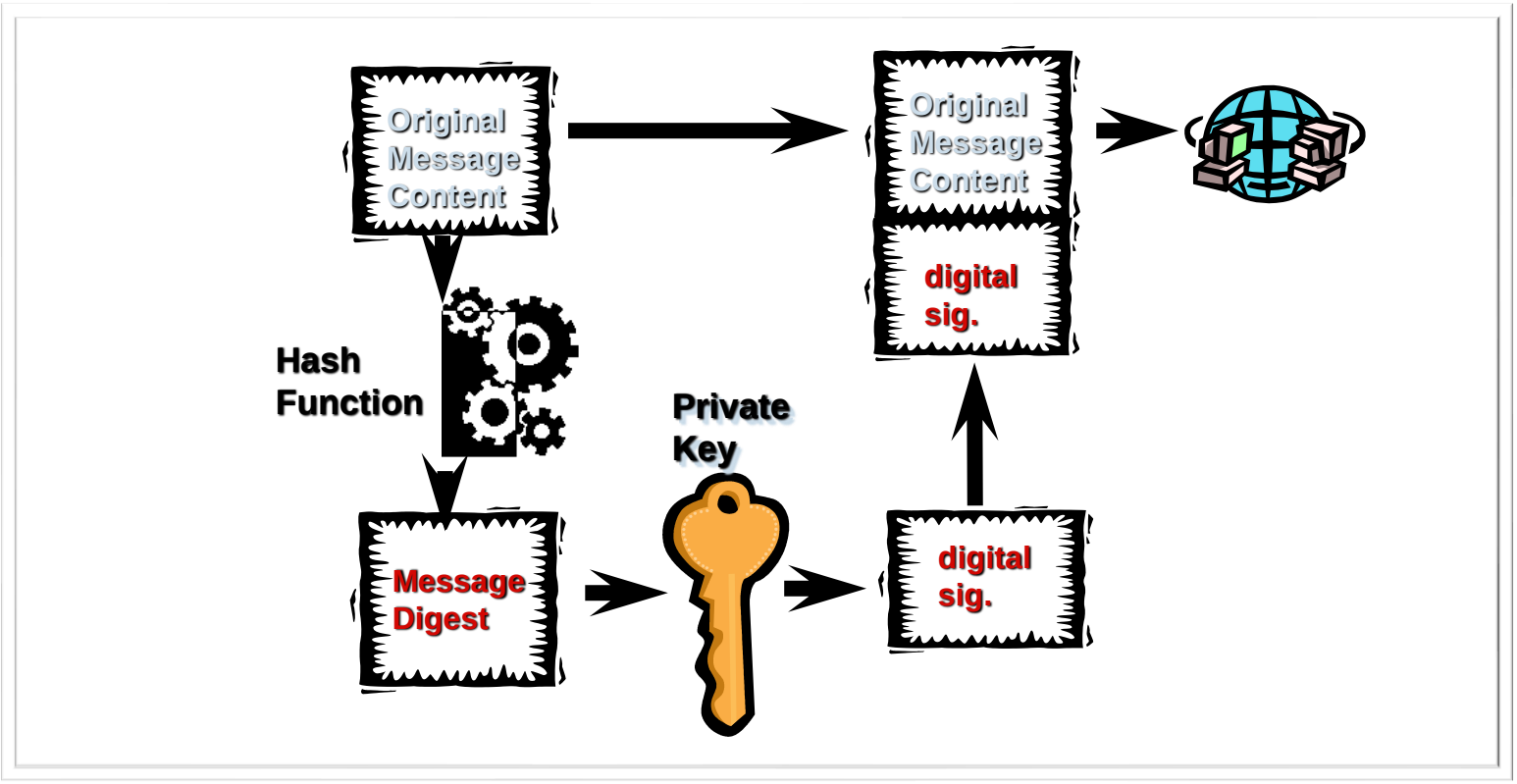
True signature

- Signed messages are forwarded directly from signer to recipient

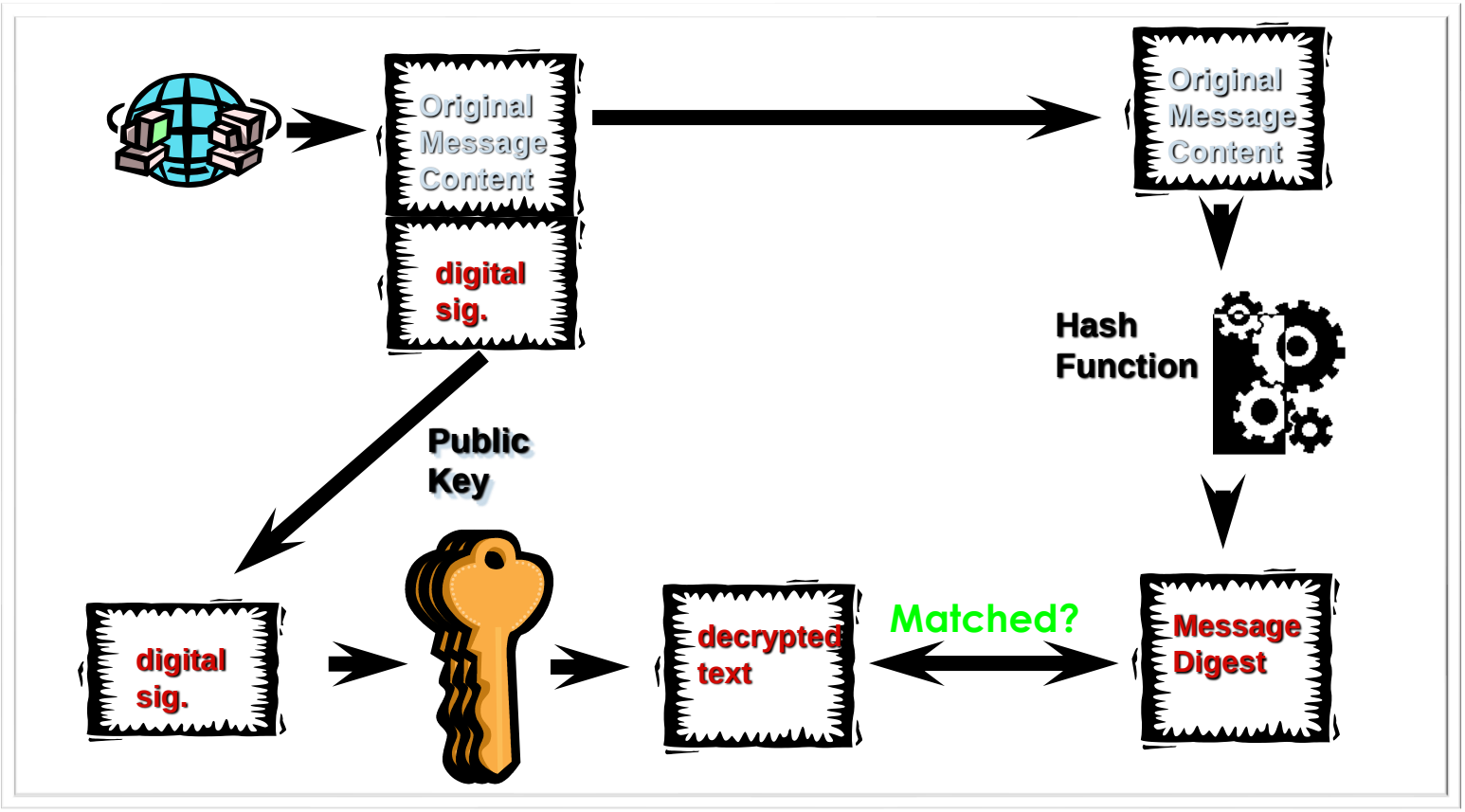
Arbitrated signature

- A witness validates a signature and transmits the message on behalf of the senders
- Settle future disputes

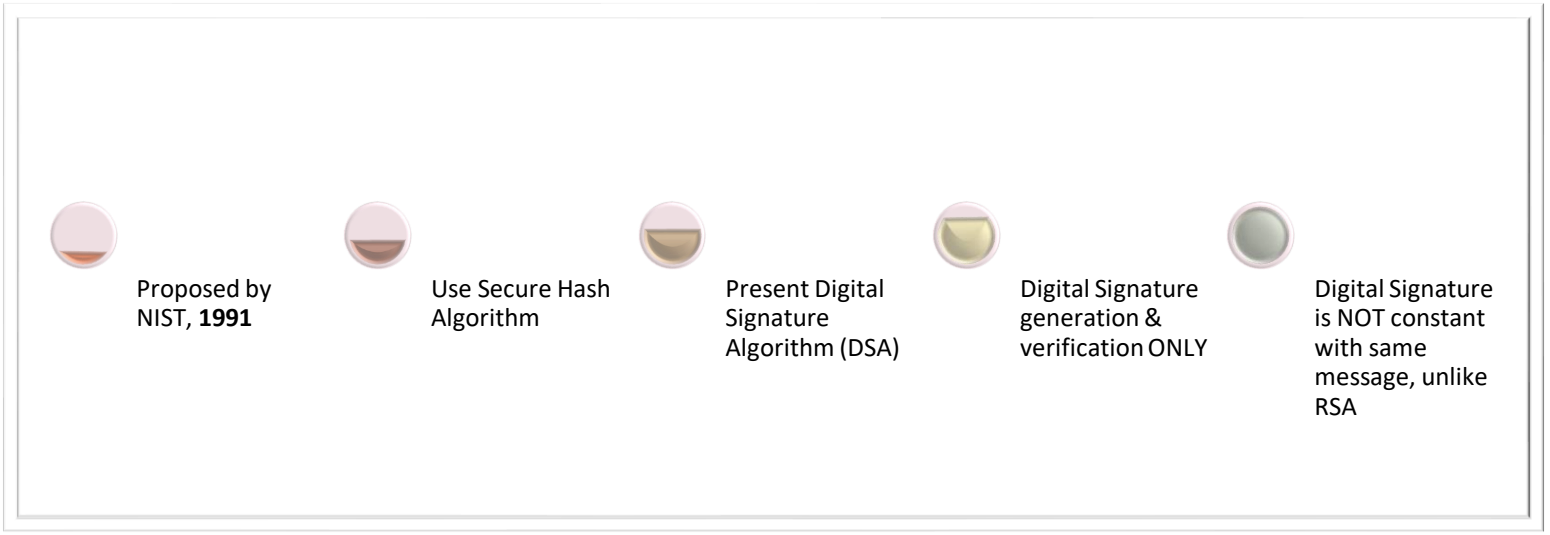
Digital Signature Generation Description



Digital Signature Verification Description



Digital Signature Standard (DSS)



Digital Signature Algorithm (DSA)

Based on ElGamal and Schnorr

FIPS-186, **1993**

1024-bit key

Faster signature generation

Discrete logarithm problem

Example:
Alice and Bob publicly agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).
Alice chooses a secret integer $a = 4$, then sends Bob $A = g^a \bmod p$

$A = 5^4 \bmod 23 = 4$

Bob chooses a secret integer $b = 3$, then sends Alice $B = g^b \bmod p$

$B = 5^3 \bmod 23 = 10$

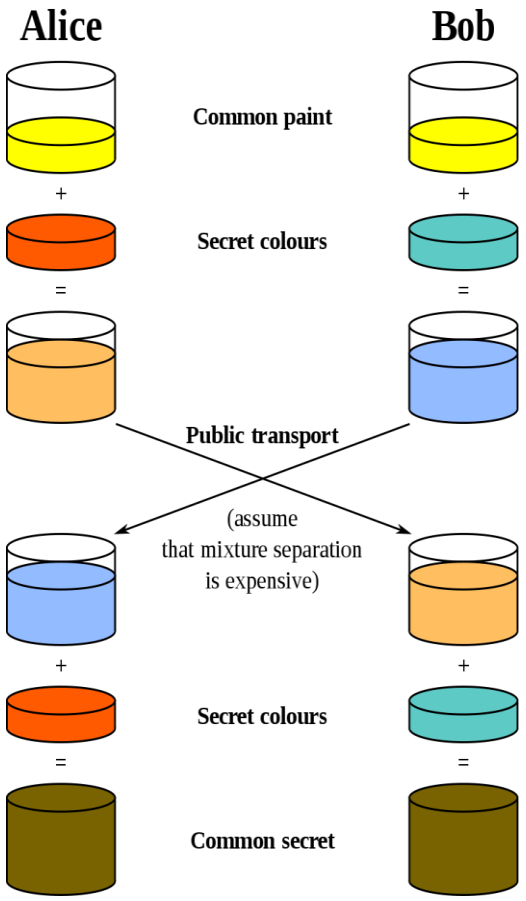
Alice computes $s = B^a \bmod p$

$s = 10^4 \bmod 23 = 18$

Bob computes $s = A^b \bmod p$

$s = 4^3 \bmod 23 = 18$

Alice and Bob now share a secret (the number 18).

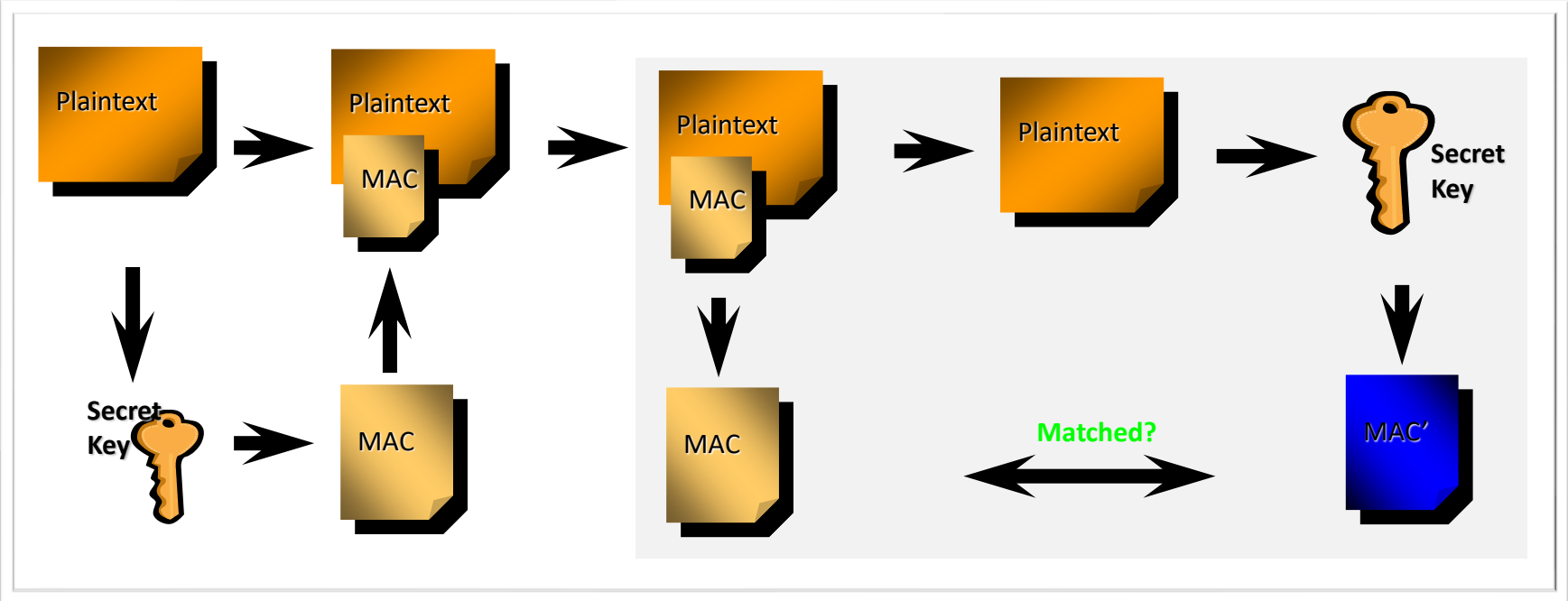


Message Authentication Code (MAC)

Secret-key approach, ANSI X9.9

Cons

- Receiver could forge message
- Sender could deny message



HMAC

HMAC as either keyed-hash message authentication code or hash-based message authentication code. MAC using hash functions as core component.

HMAC can provide digital signatures using a shared secret instead of public key encryption.

Background

Hash functions is mostly exportable

Hash functions code are publicly accessible

Hash functions normally operate faster

Objectives

Use available hash functions

Allow easy replacement of embedded hash functions

Preserve original performance of hash function

Use and handle key in a simple way

Have well understood cryptographic analysis of the strength of authentication mechanism

HMAC

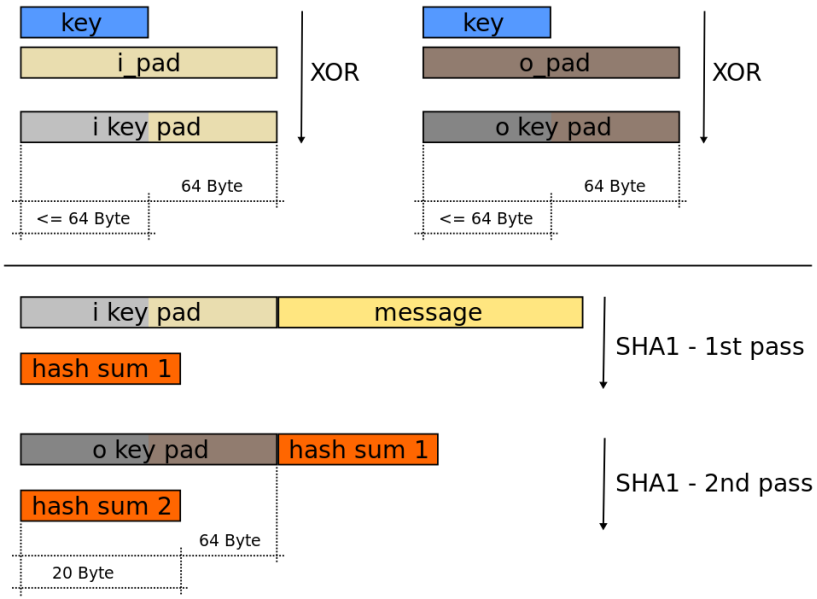
Definition stated in RFC 2104

$$\mathrm{HMAC}(K, m) = \mathrm{H}\left(\left(K' \oplus opad\right) \parallel \mathrm{H}\left(\left(K' \oplus ipad\right) \parallel m\right)\right)$$

$$K' = \begin{cases} \mathrm{H}(K) & K \text{ is larger than block size} \\ K & \text{otherwise} \end{cases}$$

where

- H is a cryptographic hash function
- m is the message to be authenticated
- K is the secret key
- K' is a block-sized key derived from the secret key, K; either by padding to the right with 0s up to the block size, or by hashing down to less than or equal to the block size first and then padding to the right with zeros
- || denotes concatenation
- ⊕ denotes bitwise exclusive or (XOR)
- opad is the block-sized outer padding, consisting of repeated bytes valued 0x5c
- ipad is the block-sized inner padding, consisting of repeated bytes valued 0x36



Trust the not-trust

An added encryption layer is implemented to ensure additional privacy and security, allowing the shares to be distributed amongst a network or group that are unknown to the secret owner.

Protects private information from organized attacks; even if each shareholder were to collude to recreate the original secret, they wouldn't be able to learn anything about that secret, as the original secret is encrypted.

Any way to proves knowledge of a secret without revealing the secret?

- Zero-knowledge Proof
- Shamir's Secret Sharing
- Ring Signatures
- Homomorphic encryption
- Pedersen Commitment

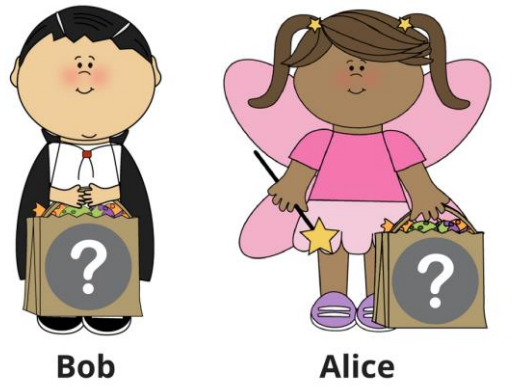
Zero-knowledge Proof

Zero Knowledge Protocol (or Zero Knowledge Password Proof, ZKP) is a way of doing authentication where no passwords are exchanged, which means they cannot be stolen.

ZKP allows you proving that you know some secret (or many secrets) to somebody at the other “end” of communication without actually revealing it.

Zero-knowledge Proof

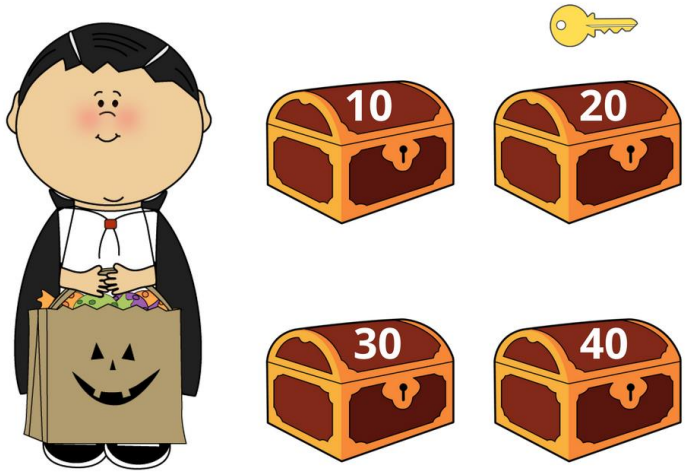
Alice and Bob would like to know if they received the same amount of candy, without disclosing their number of chocolates because they don't want to share.



Zero-knowledge Proof

Bob gets 4 lockable boxes and puts a label in each that says 10, 20, 30 or 40 (chocolate bars).

Then Bob throws away all the keys except for the key to the box that corresponds to the number of chocolate bars he's got (let's say he has 20 chocolate bars) and leaves.



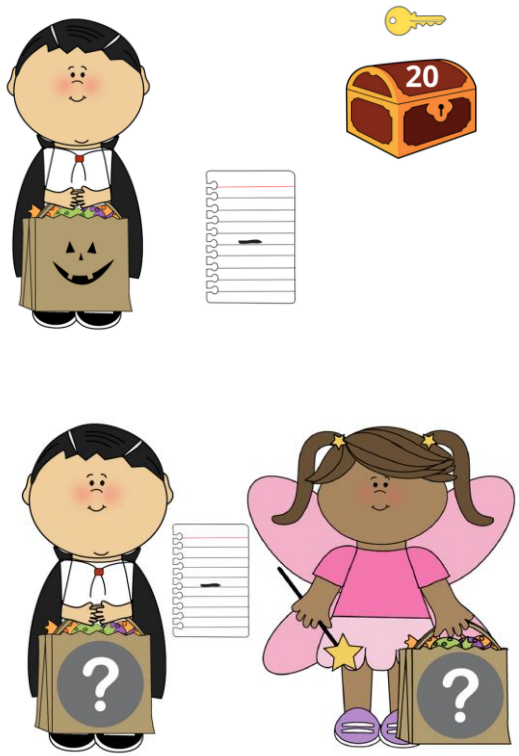
Zero-knowledge Proof

Then she slips the "+" piece through a slot into the box with the number that corresponds to the number of candies she's got (let's say she has 30 candy bars) and slips the pieces of paper with "-" on them into the rest of the boxes and also leaves.

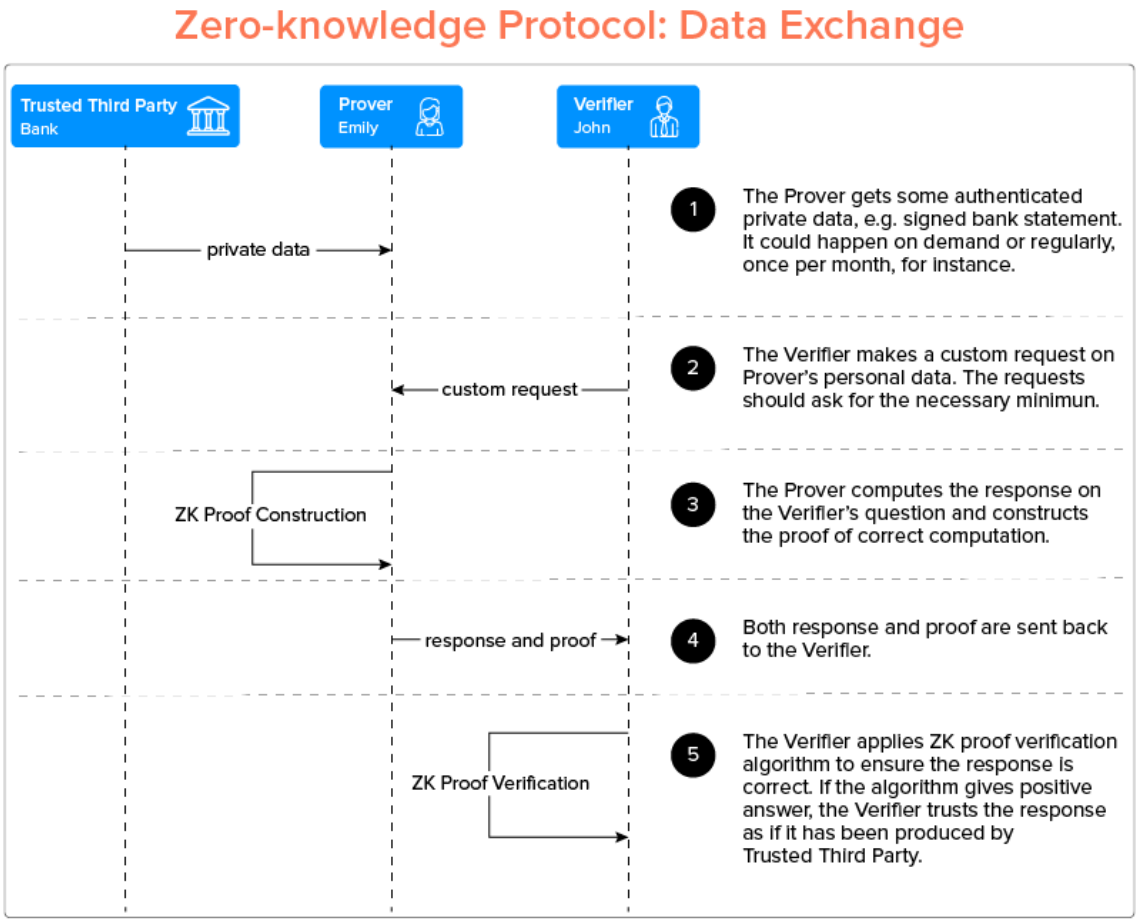
Zero-knowledge Proof

If it is a “plus”, Alice has the same number of chocolate bars in her bag. If the slip of paper says “-”, it means that they have a different amount of candy (but still will not share with each other).

Alice also returns and sees that Bob has a piece of paper with a “minus” on it. So he has a different amount of candy. But both Alice and Bob still don’t know how many chocolate bars each of them has. They only know that they don’t have the same amount.



Zero-knowledge Protocol



Zero Knowledge Proofs and Blockchain

ZCash is an open-source and permissionless blockchain platform that offers the functionality to keep transactions ‘transparent’ and ‘shielded’ as per the requirement.

ING is a Netherlands based bank that has introduced its own zero-knowledge blockchain. However, they have modified their zero-knowledge system to make it a zero-knowledge knowledge range proof to lower down the need for computational power.

Zcoin, The company uses Zerocoin protocol, which is based on zero-knowledge proof, to enhance security and anonymity in the transaction process. However, what makes it distinct from other projects working on this concept is that it offers scalability too.

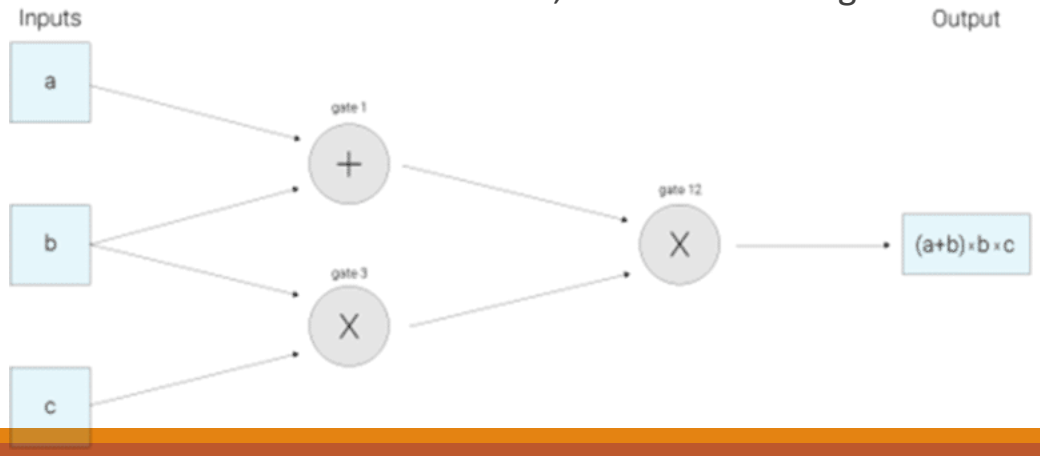
Non-interactive zero-knowledge proof

Non-interactive zero-knowledge proofs— also known as NIZK, zk-SNARK, zk-STARK are zero-knowledge proofs that require no interaction between the prover and verifier.

The first widespread application of zk-SNARKs was in the Zerocash blockchain protocol, where zero-knowledge cryptography provides the computational backbone, by facilitating mathematical proofs that one party has possession of certain information without revealing what that information is

zk-STARK (zero-knowledge Scalable Transparent ARgument of Knowledge)

“Zero-knowledge” proofs allow one party (the prover) to prove to another (the verifier) that a statement is true, without revealing any information beyond the validity of the statement itself. For example, given the hash of a random number, the prover could convince the verifier that there indeed exists a number with this hash value, without revealing what it is.



Multi-party Digital Signatures

Multiple-party Digital Signatures or Multi-Signatures are used for signing by multiple parties

Privacy-preserving cryptographic technique

Keyless sharing and storing of private data

Shamir's Secret Sharing algorithm that used for sharing of private information in distributed environment

Secret sharing works by splitting private information into smaller pieces — or shares — and then distributing those shares amongst a group or network.

Shamir's Secret Sharing Algorithm

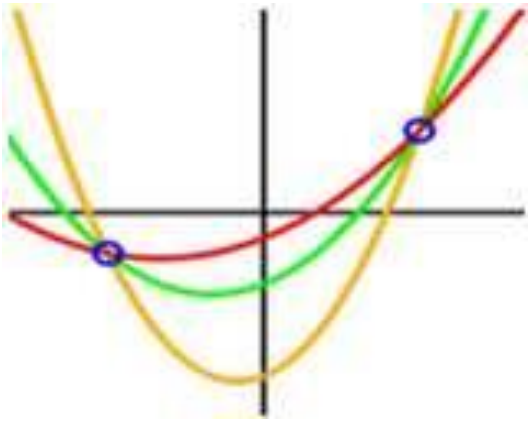
One of the challenges of distributing shares is that they can often be lost or compromised.

Shamir's Secret Sharing scheme is an algorithm that was first proposed in 1979 by the renowned Israeli cryptographer Adi Shamir.

This means that, instead of requiring all shares to reconstruct the original secret, Shamir's scheme requires a minimum number of shares — this minimum is referred to as the threshold.

The threshold needs to be met in order to reconstruct the secret.

- At least K of a set of N users need to participate to create a valid signature
- If there is anything less than the threshold, the secret cannot be reconstructed, thus making Shamir's Secret Sharing secure against an adversary — a malicious attacker
- Each person is given the X,Y coordinates of a point on the curve
- A shared signature can be pieced together from each individual signature



Ring Signatures

In cryptography, a ring signature is a type of digital signature that can be performed by any member of a group of users that each have keys.

Therefore, a message signed with a ring signature is endorsed by someone in a particular group of people.

One of the security properties of a ring signature is that it should be computationally infeasible to determine which of the group members' keys was used to produce the signature.

1 real public key with a number of decoy public key

The method was initially created by Ron Rivest, Adi Shamir, and Yael Tauman in 2001, and in their paper they proposed the White house leak dilemma

Stealth Addresses

Transactions are pseudonymous, meaning a public address is linked to an individual, but that individual's identity is unknown to the participants or the public.

A public key might be tracked to a person's identity (through an IP address, for example), leading to all of the transactions that have used that key being tracked throughout the blockchain.

Stealth addresses were proposed by Peter Todd in 2014. Also used in Monero Stealth Address scheme.

The stealth address' mechanism uses a combination of various public and private keys that are dynamic and for one-time use only. With a stealth address, you ask payers to generate a unique address in such a way that you (using some additional data which is attached to the transaction) can deduce the corresponding private key.

Stealth addresses protect the privacy of receivers of cryptocurrencies by requiring the sender to use a random, one-time address for every transaction. As a result, numerous transactions done with the same recipient are not able to be linked since each transaction has a one-time address.

Able to confirm the actual recipient of the content based on Stealth Address together with tag.

As a result, this anonymity makes it easier to obscure financial transactions from "legitimate scrutiny."

Homomorphic encryption

In an ideal world, homomorphic encryption has a multitude of practical, real-world applications—everything from electronic voting systems to analyzing medical data to enabling private queries in search engines

Homomorphic encryption helps to protect the integrity of your data by allowing others to manipulate its encrypted form while no one (aside from you as the private key holder) can understand or access its decrypted values

Homomorphic encryption is when arithmetic operations function on encrypted data

- Additive Homomorphism: A+B=C, E(A)+E(B)=E(C)
- Multiplicative Homomorphism: A*B=C, E(A)*E(B)=E(C)

Fully homomorphic encryption is slow and inefficient

Pedersen Commitment

The Pedersen commitment allows us to commit to a message, but not actually reveal it until some time in the future.

We can also use the Pedersen commitment to add commitment values (and thus implement partial homomorphic encryption)

```
Secret value:   160021850381606860961606826218608521138591291344 9
p= 835633126610936170404251455173303609923742171077
q= 167126625322187234080850291034660721984748434215 5
g= 853879156962728674489578584402624767640115848839
h= 312816528258481903392461143384659987330939968504

Msg1: 70
Msg2: 70

c1,r1: 931432606187308599362760090383550998212817597419 , 133154063878955883975175775621859189924022928785 7
c2,r2: 377312755894089489173415862148929013782547633796 , 106229632588669033453493136803351658193427515322 4

We can now multiply c1 and c2, which is same as adding Msg1 and Msg2

Commitment of adding (Msg1+Msg2):       943416350824651526788742276019896656762684796624

Result of verifying c1:         True
Result of verifying c2:         True
Result of verify Msg+Msg2:      True
```

Advanced Cryptographic Solutions Summary

Methods	Usage
Multi-Signatures	A multisignature is a digital signature that requires multiple private keys to generate a valid digital signature, allowing multiple parties to approve a transaction.
Zero-Knowledge Proofs	A zero-knowledge proof (ZKP) is designed for the situation where you know a secret and want to prove that you know it to someone without giving it away
Stealth Addresses	A special type of address on the blockchain designed to conceal the recipient of a transaction.
Ring Signatures	Designed to conceal the sender of a transaction by generating a valid digital signature
Homomorphic Encryption	Allows arithmetic operations to be performed on encrypted data
Pederson Commitments	Commits to the value of something without revealing the value