THE UNIVERSITY of NEW ORLEANS

CSCI 4621/5621 Intro to CyberSecurity

10: ESSENTIAL CRYPTOGRAPHIC TOOLS

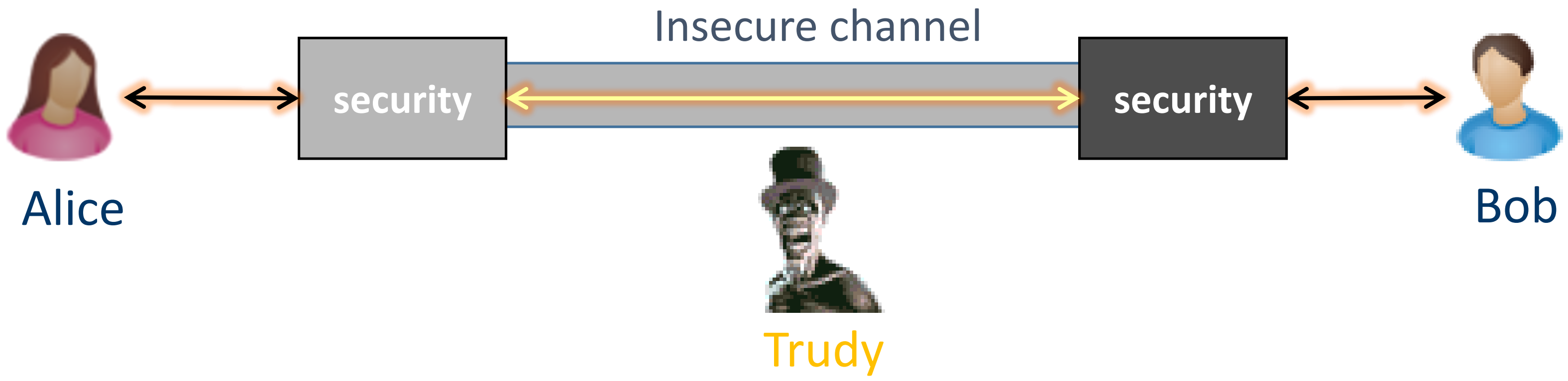
Vassil Roussev

vassil@cs.uno.edu

READING: [Oorschot [ch2]]

The world of Alice & Bob

- Alice & Bob want to communicate ***securely*** over an insecure channel, where:
- Trudy (intruder) may interfere ***arbitrarily*** with the communication.



Security requirements

- Confidentiality:
  - » *only sender, intended receiver should be able to **understand** message contents*
    - sender encrypts message
    - receiver decrypts message
- Authentication:
  - » *sender, receiver want to confirm identity of each other*
- Message integrity:
  - » *sender, receiver want to ensure message **not altered** (in transit, or afterwards) **without detection***

Who might Alice & Bob be?

- Real-life people
- Web browser/server for electronic transactions
- Online banking client/server
- DNS servers
- Routers exchanging routing table updates

- Generally, *any two processes communicating over the network*

ADVERSARIAL MODEL

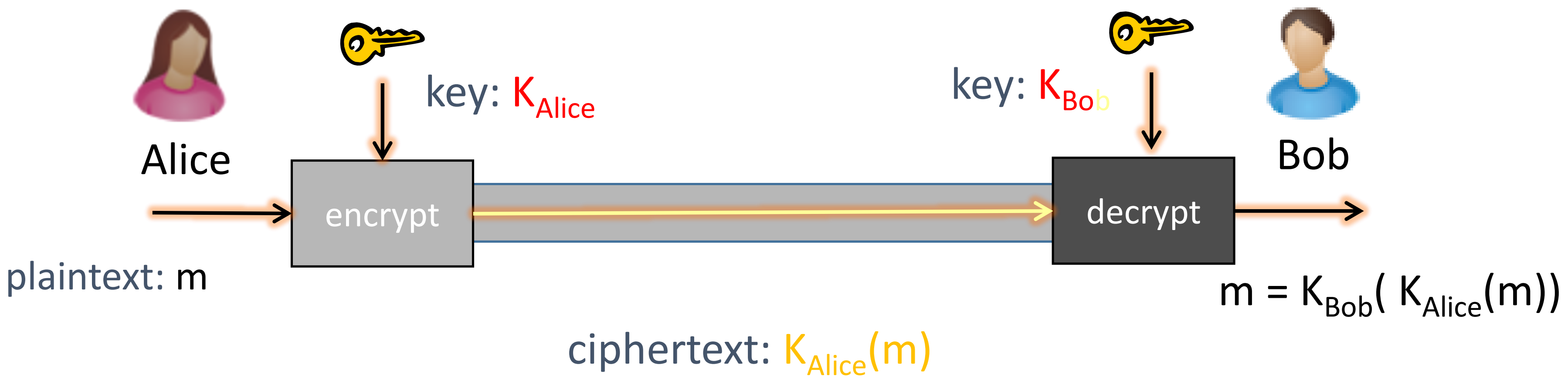
Q: What can Trudy do?

A: Arbitrarily manipulate the message exchange

- *eavesdrop*
  → intercept all messages
- *fake/replay*
  → send spoofed messages into connection, or replay old messages
- *impersonate*
  → can fake (spoof) source address in packet (or any field in packet)
- *hijack*
  → take over ongoing connection by removing sender or receiver, inserting himself in place
- *deny service*
  → prevent service from being used by others (e.g., by overloading resources)

CRYPTO BUILDING BLOCKS

Basic crypto terms



SIMPLE ENCRYPTION SCHEME

Substitution cipher: substituting one thing for another

» *monoalphabetic cipher: substitute one letter for another*

```
plaintext:   abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

E.g.:

plaintext: **bob. i love you. alice**

ciphertext: **nkn. s gktc wky. mgsbc**

Key → the mapping from the set of 26 letters to the set of 26 letters

Cryptanalysis

- Ciphertext-only attack:
  - *Trudy has ciphertext that she can analyze*
- Two approaches
  - *Brute force*
    - Search through all keys:
    - Must be able to differentiate resulting plaintext from gibberish
  - *Statistical analysis*

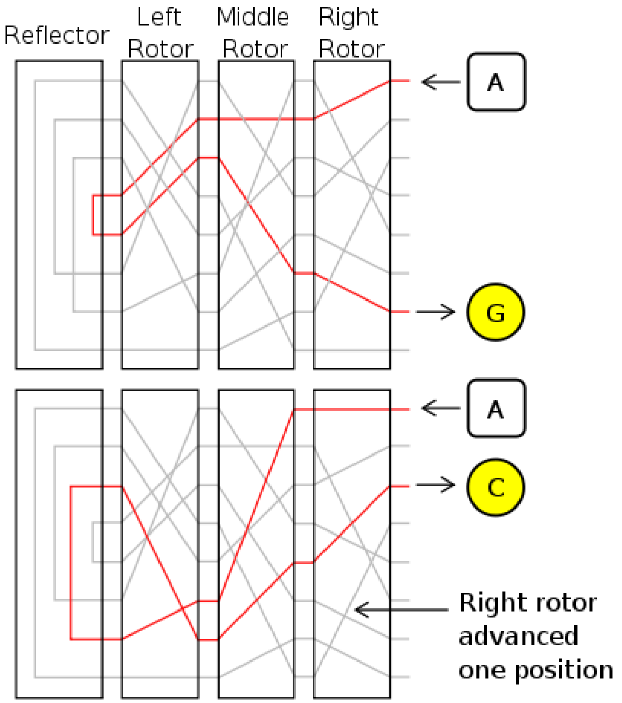
- Known-plaintext attack
  - *Trudy has some plaintext corresponding to some ciphertext*
  - *e.g., in monoalphabetic cipher, Trudy determines pairings for*

    a l i c e b o

- Chosen-plaintext attack:
  - *Trudy can get the ciphertext for some chosen plaintext*

Viginere cipher

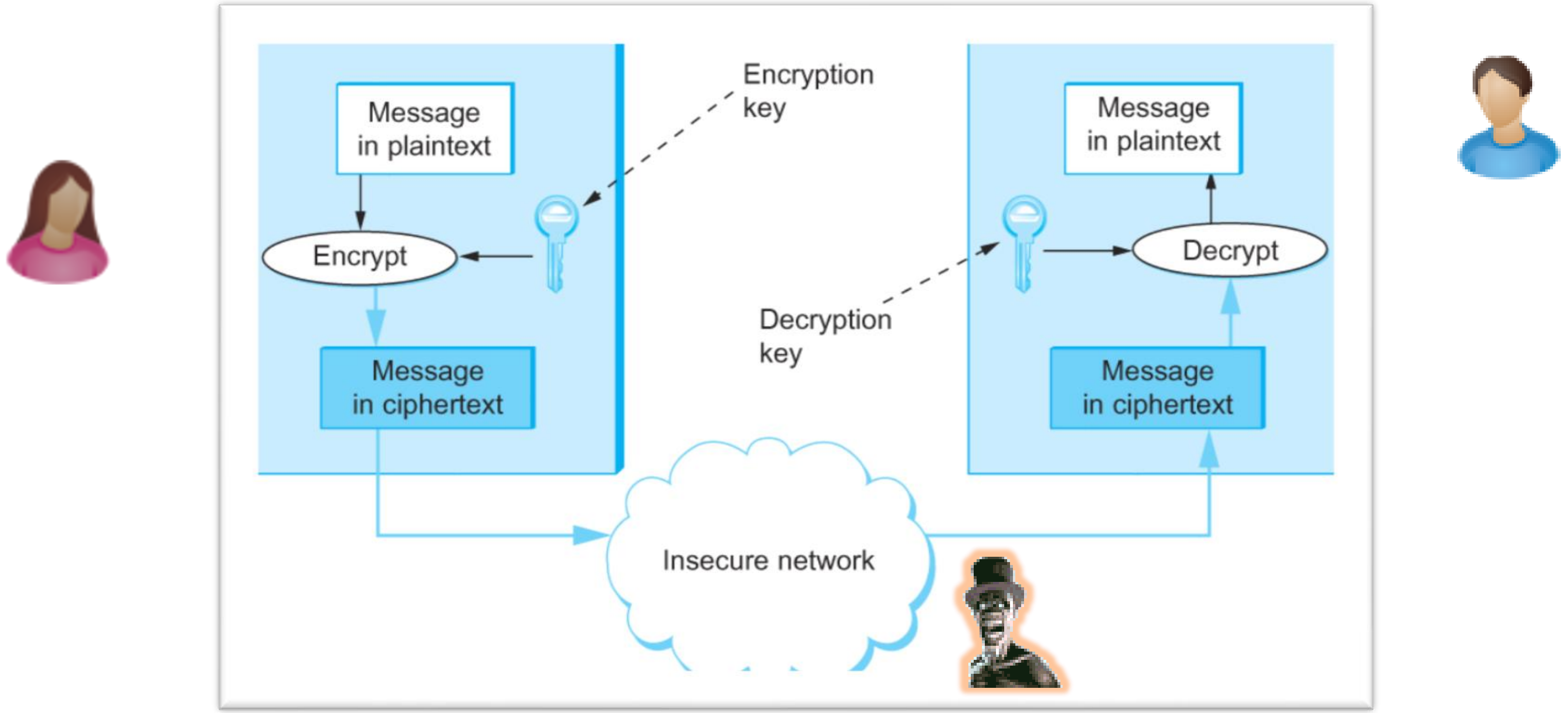
- Invented by Bellaso in 1553,
  - *misattributed to Viginere in the 19th century*
- Idea:
  - *string together multiple substitution ciphers*
  - *first broken in 1863 by Kasiski*
- A version used in the Enigma machines
  - *rotor-based implementation*
  - *+ plugboard ➔ letter swaps*



Modern approach to cryptography

- Security based solely on key secrecy
  - *Algorithm is known to everyone (standard)*
  - *Only keys are secret*
- Symmetric/private key cryptography
  - *Uses one (shared) private key*
- Asymmetric/public key cryptography
  - *Uses of a pair of <public, private> keys*
- Cryptographic hash functions
  - *Generate message digests*
  - *Used to verify integrity of transmission*

SYMMETRIC KEY CRYPTOGRAPHY

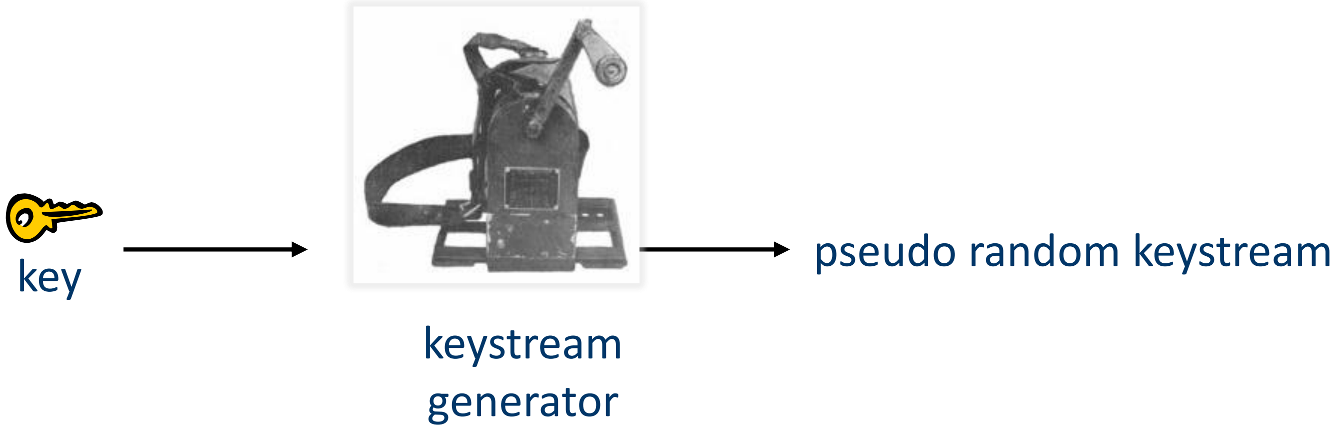


- Confidentiality is based on having a secret shared key
  - *Secure key exchange is critical!*

SYMMETRIC CIPHERS

- Stream ciphers
  - *Encrypt one bit at time*
- Block ciphers
  - *Break plaintext message in equal-size blocks*
  - *Encrypt each block as a unit*

STREAM CIPHERS



$m(i)$ = i^{th} bit of message

$ks(i)$ = i^{th} bit of keystream

$c(i)$ = i^{th} bit of ciphertext

- Combine each bit of keystream with bit of plaintext to get bit of ciphertext:

  $c(i) = ks(i) \oplus m(i)$ → encryption

- Combine each bit of keystream with bit of ciphertext to get bit of plaintext

  $m(i) = ks(i) \oplus c(i)$ → decryption

RC4/ARC4 STREAM CIPHER

- RC4 is a popular stream cipher
  - *Extensively analyzed and considered insecure*
    - Has known biases
  - *Key can be from 1 to 256 bytes*
  - *Used in WEP for 802.11*
  - *Can be used in SSL → now deprecated*

Block ciphers

- Message to be encrypted is processed in blocks of k bits (e.g., 64-bit blocks).
- 1-to-1 mapping is used to map k-bit block of plaintext to k-bit block of ciphertext

Example for k=3:

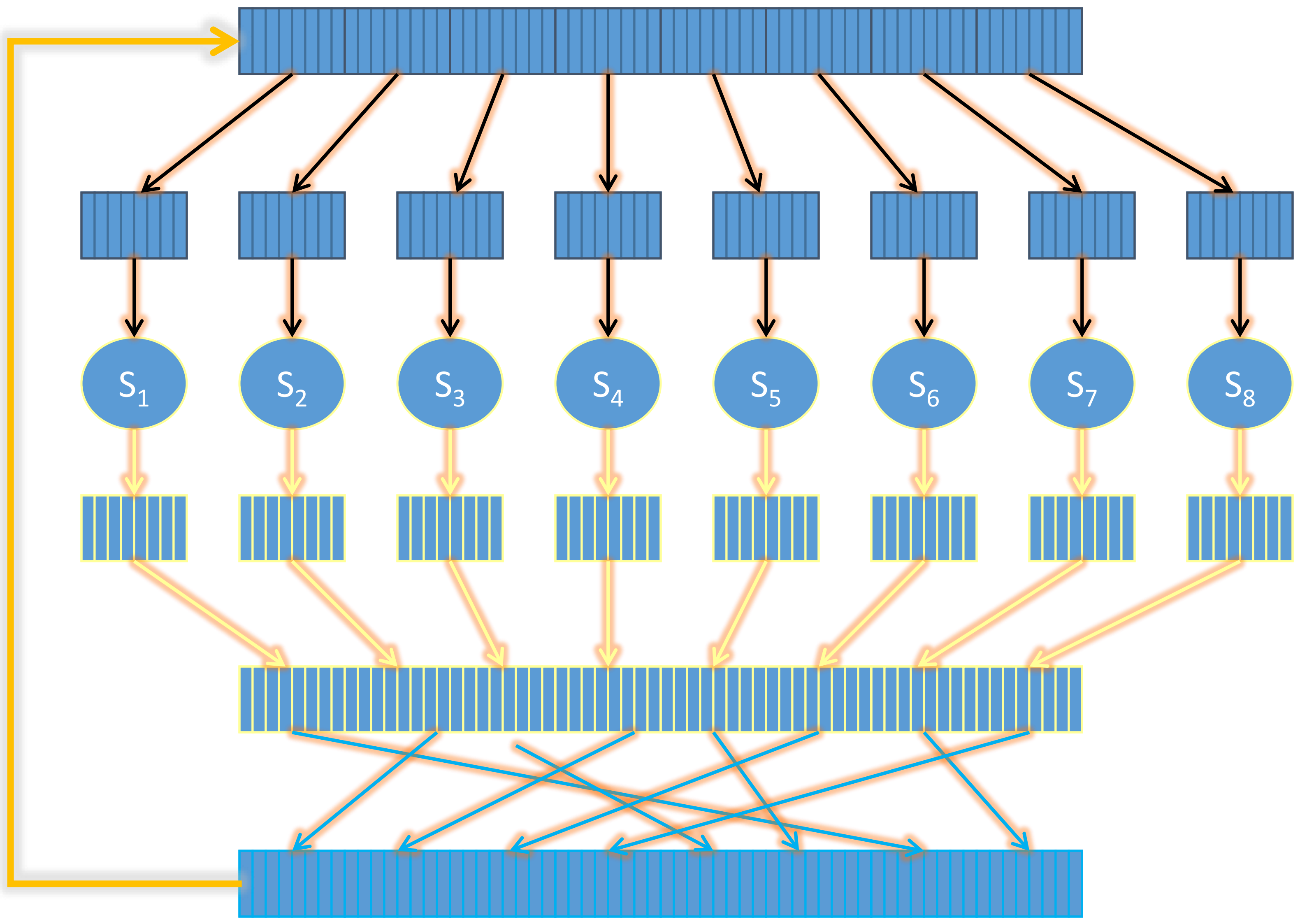
input	output	input	output
000	110	100	011
001	111	101	010
010	101	110	000
011	100	111	001

Q: What is the ciphertext for `010110001111`?

BLOCK CIPHERS

- How many possible mappings are there for k=3?
  - *How many 3-bit inputs?*
  - *How many permutations of the 3-bit inputs?*
  - *Answer: 40,320 ➔ not very many!*
- In general, $2^k!$ mappings
  - ***HUGE*** *for k=64*
- Problem:
  - *Table approach requires table with 2^{64} entries, each entry with 64 bits*
- Solution:
  - *Use a function that simulates a randomly permuted table*

GENERIC 64-BIT BLOCK CIPHER

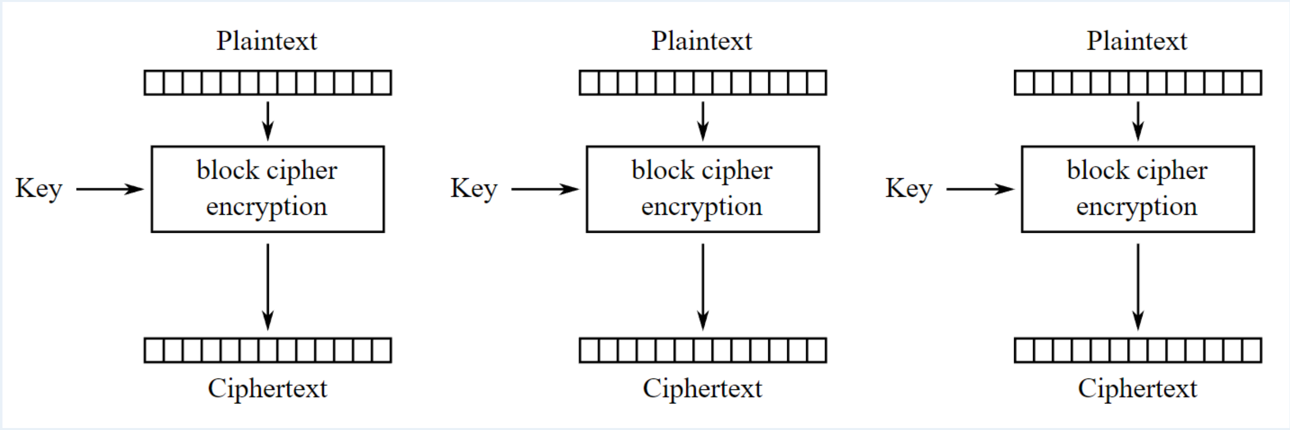


WHY ROUNDS IN PROTOTYPE?

- If only a single round, then one bit of input affects at most 8 bits of output.
- In 2nd round, the 8 affected bits get scattered and inputted into multiple substitution boxes.
- How many rounds?
  - *How many times do you need to shuffle cards?*
  - *Eventually, each additional round becomes less efficient as **n** increases*

ENCRYPTING A LARGE MESSAGE: ECB

- ECB: electronic code-book
  - *split each message into (64-bit) blocks*
  - *encrypt each one with the key*
  - *transmit resulting stream*
- Problems?
  - *same plaintext → same ciphertext*
- Solution?
  - *add random bits to each block*
- Drawbacks?
  - *inefficient*



[CREDIT: Wikipedia]

Cipher Block Chaining (CBC) mode

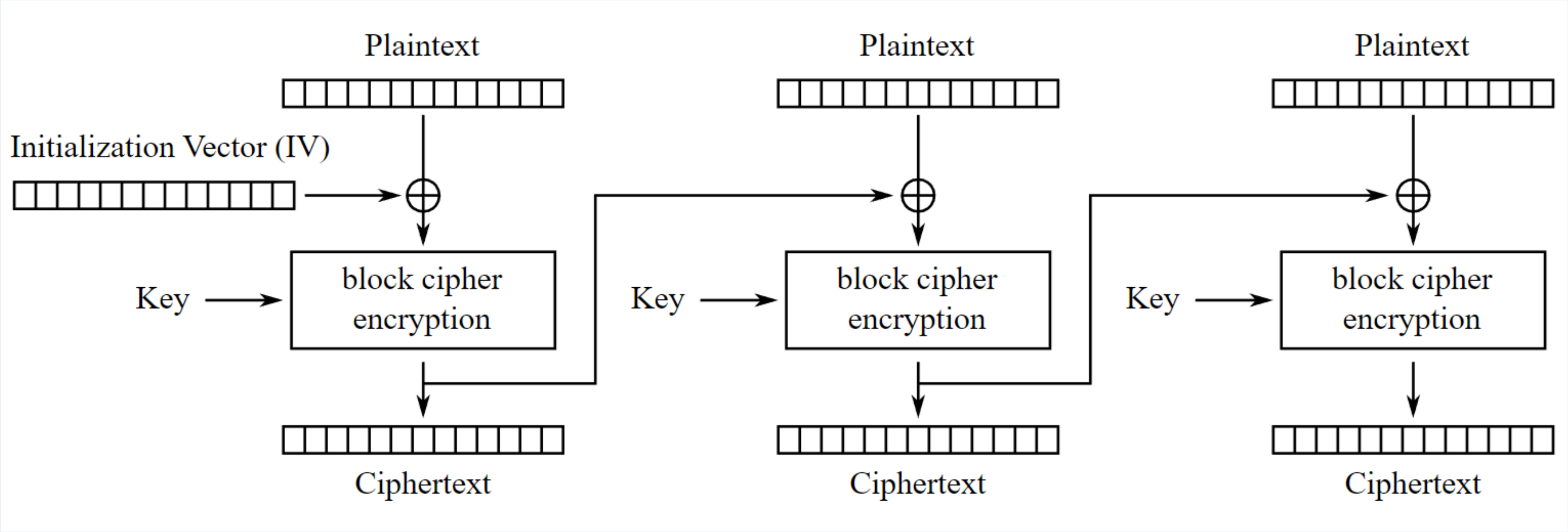
- CBC generates its own random numbers
  - *Have encryption of current block depend on result of previous block*

    $c(i) = K_S(\, m(i) \oplus c(i-1)\,)$

    $m(i) = K_S(\, c(i)) \oplus c(i-1)$

- How do we encrypt first block?
  - *Initialization vector (IV): random block = c(0)*
  - *IV does not have to be secret*
- Change IV for each message (or session)
  - *Guarantees that even if the same message is sent repeatedly, the ciphertext will be completely different each time*

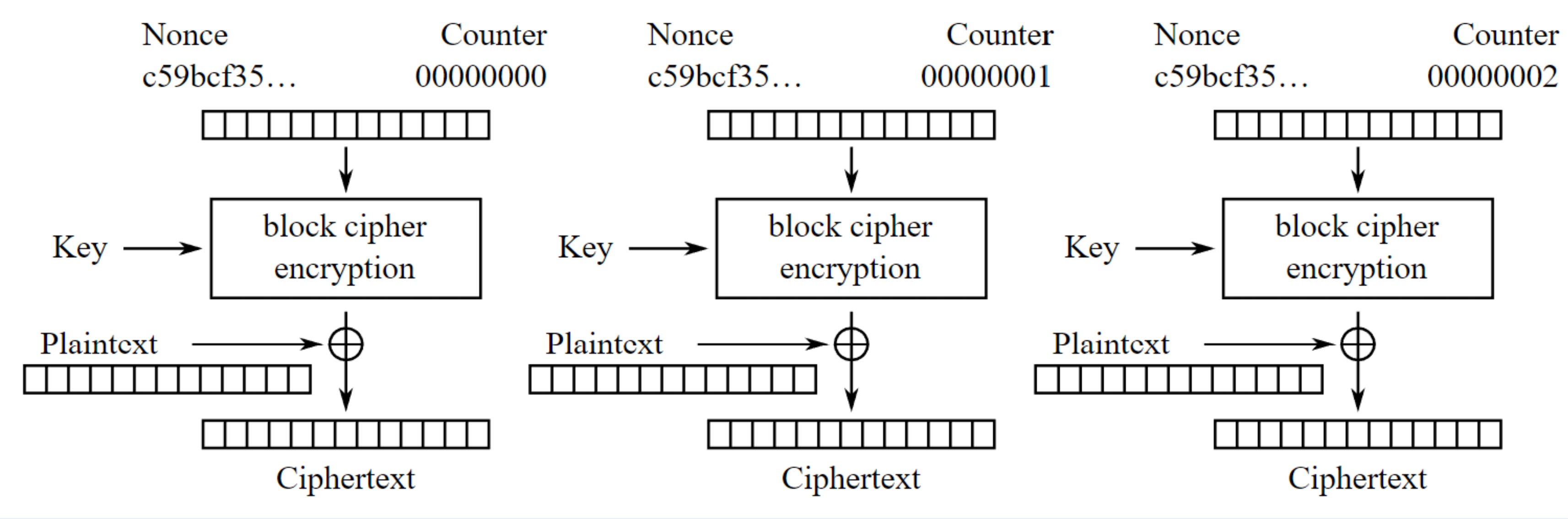
CIPHER BLOCK CHAINING



[CREDIT: Wikipedia]

- Each ciphertext block is dependent on **all** plaintext blocks processed up to that point

COUNTER MODE (CTR)



[CREDIT: Wikipedia]

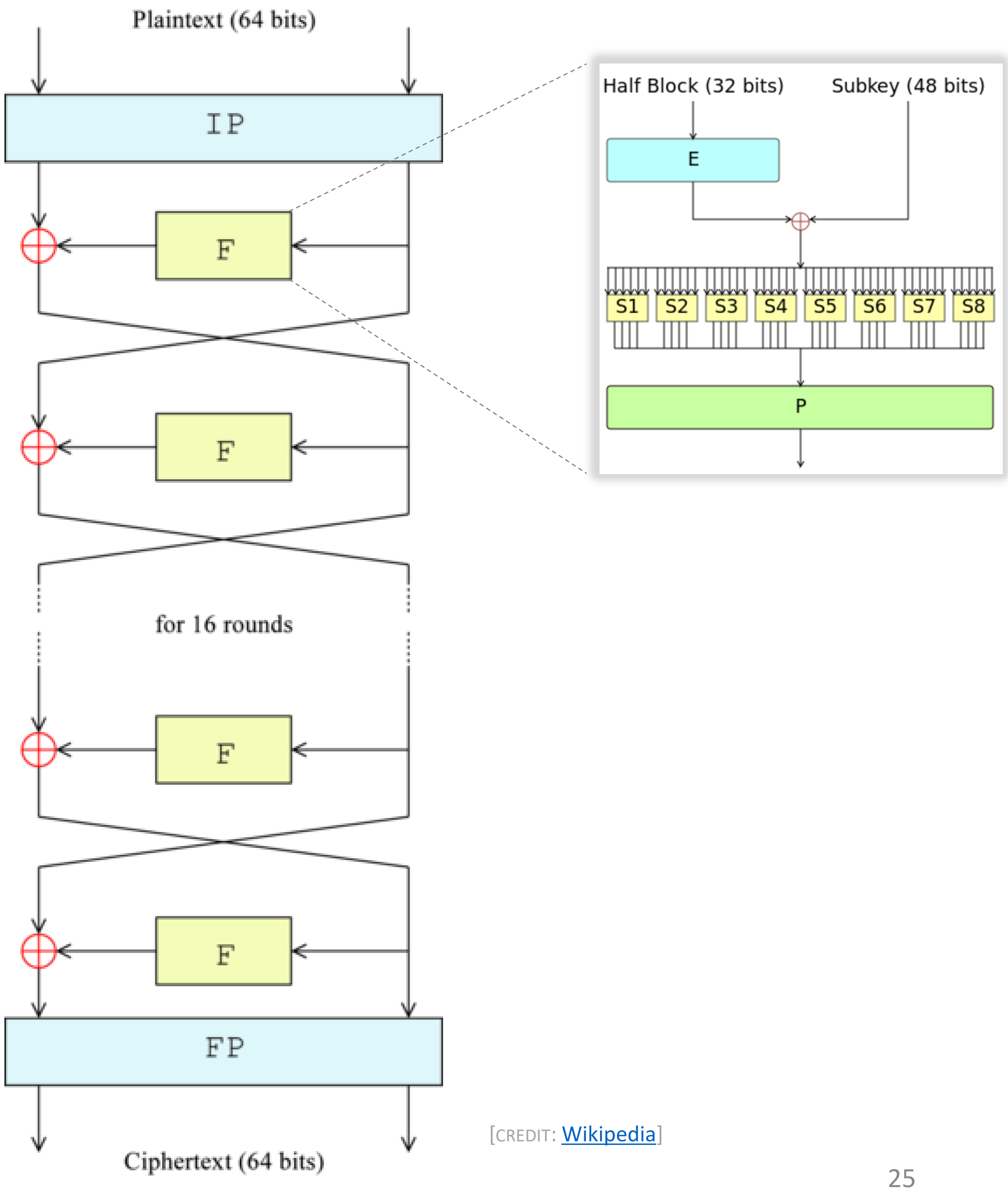
- Recommended over CBC

SYMMETRIC (OR PRIVATE) KEY CRYPTO: DES

- DES: Data Encryption Standard
  - *US encryption standard 56-bit symmetric key, 64-bit plaintext input*
  - *Block cipher with cipher block chaining*
- How secure is DES?
  - *DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in less than a day*
  - *No known good analytic attack*
- Making DES more secure:
  - *3DES: encrypt 3 times with 3 different keys*
    - (encrypt, decrypt, encrypt)
- No good reason to use DES today → use **AES**

DES OPERATION

- Initial permutation
- 16 identical "rounds" of function application, each using different 48 bits of key
- Final permutation



[CREDIT: Wikipedia]

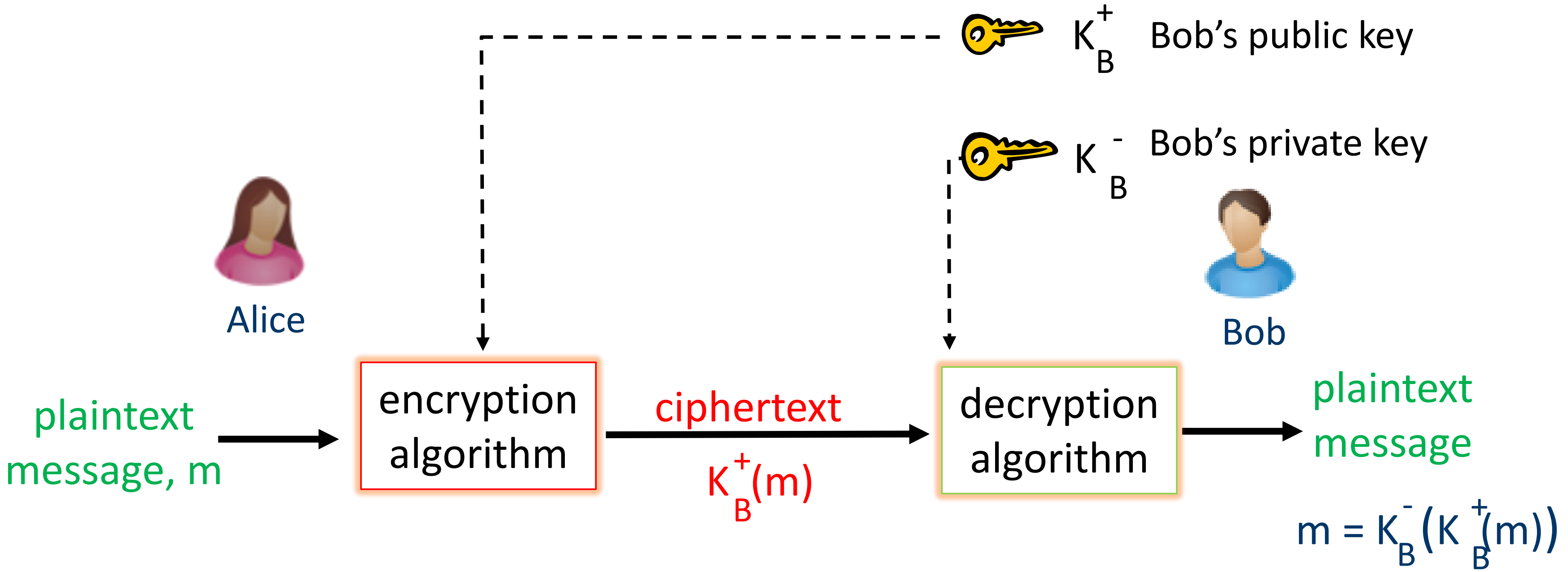
AES: Advanced Encryption Standard

- Current (Nov 2001) symmetric-key NIST standard, replacing DES
- Processes data in 128 bit blocks
  - *128, 192, or 256 bit keys*
- Brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

Asymmetric (public-key) cryptography

- Motivation:
  - *private-key cryptography relies in a shared secret key*
  - *Q: How do Alice & Bob agree on key in first place?*
- Public-key cryptography
  - *relies on a pair of keys* $<K_{public}, K_{private}>$
  - *essential property*
    - ciphertext produced with one of the keys can only be decrypted with the other

PUBLIC-KEY CRYPTOGRAPHY (RSA/DSA/ECC)



NIST-RECOMMENDED KEY LENGTHS

Symmetric-key security strength	RSA modulus	DH		ECC
		modulus	private key	
112 (triple-DES)	2048	2048	224	224-255
128 (AES)	3072	3072	256	256-383

[CREDIT: Oorschot]

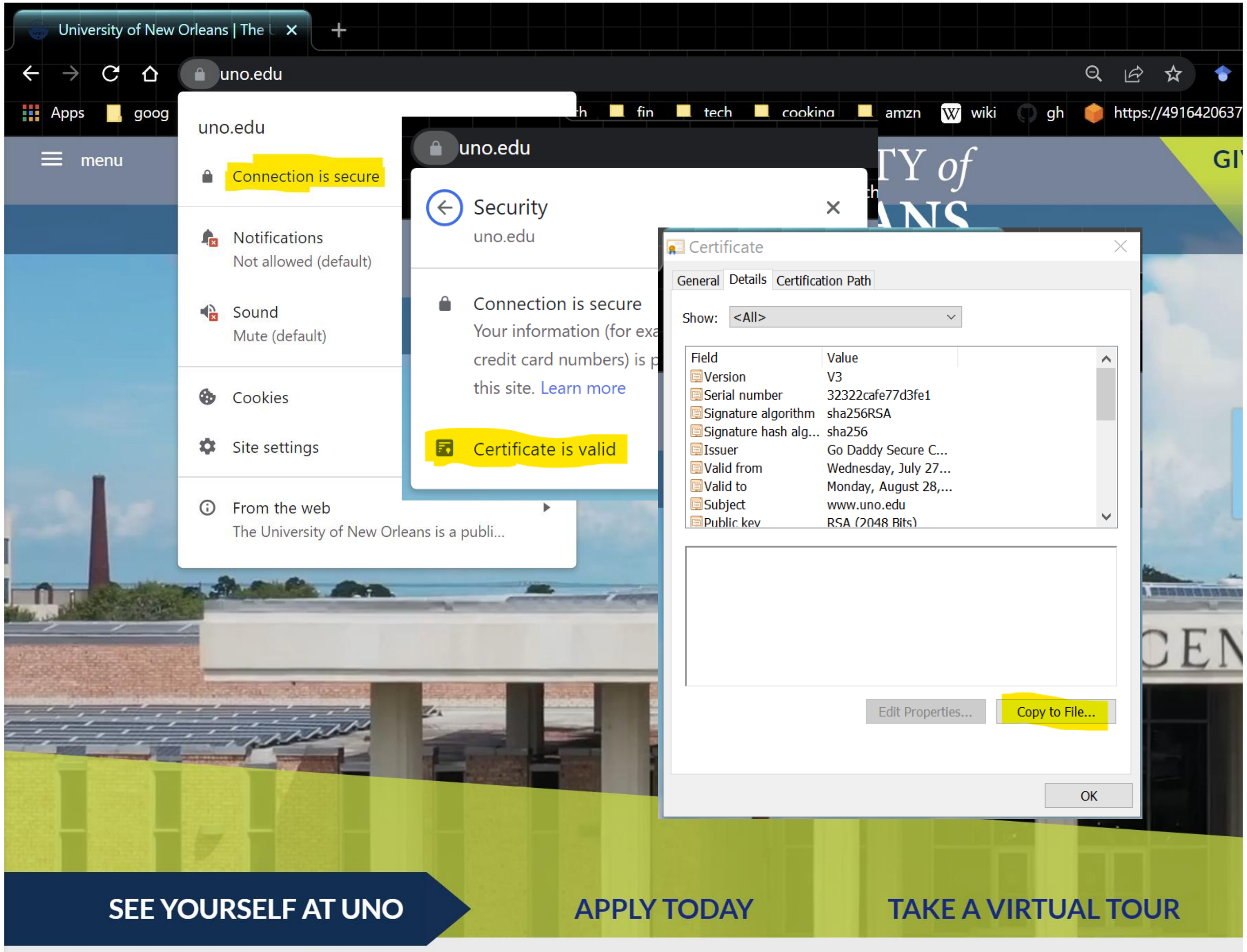
CONFIDENTIALITY & AUTHENTICATION

- Private-key crypto
  - *both confidentiality and authentication rely on the secrecy of the shared key*
- Public-key crypto
  - *these are accomplished in two steps and are (usually) combined to achieve both*
- Signed message
  - *encrypted with the sender's private key*
  - ➔ *authenticates the sender (non-repudiation)*
- Secret message
  - *encrypted with the receiver's public key*
  - ➔ *ensures the only the receiver can decrypt it (w/ private key)*
- Signed + secret
  - *first sign then encrypt*

Session key exchange

- Exponentiation is computationally intensive
  - *symmetric ciphers are much faster than RSA*
- Idea:
  - *combine public/private key systems*
  - *use public key crypto to exchange session key* $\boldsymbol{K_S}$ *(shared secret)*
  - *switch to private key encryption using* $\boldsymbol{K_S}$
- **Q:** How do we learn public keys in a trustworthy way?
  - *certificates*
  - *issues by trusted third parties – certification authorities*

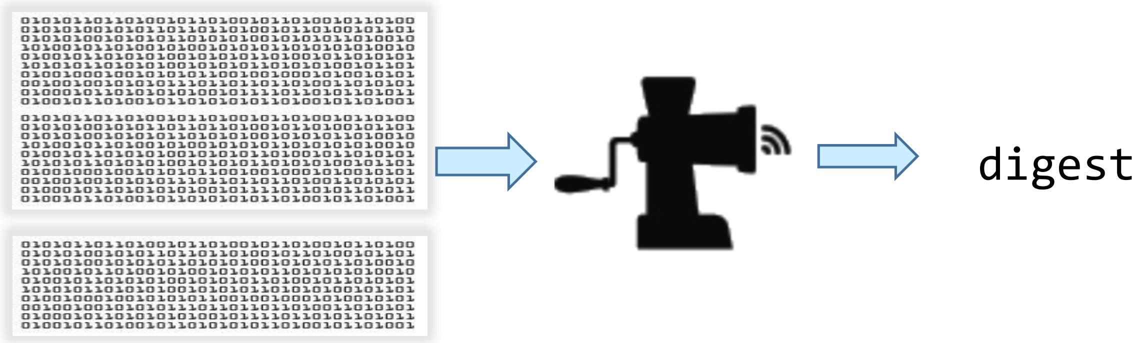
PKI & X.509 BY EXAMPLE



MESSAGE INTEGRITY VERIFICATION

- Allows communicating parties to verify that received messages are authentic. I.e.:
  - *content of message has not been altered*
  - *source of message is who/what you think it is*
  - *message has not been replayed*
  - *sequence of messages is maintained*

MESSAGE DIGEST FUNCTION



- Function *H*( ) that takes as input an arbitrary length message and outputs a **fixed-length** string → ***message signature***
  - » *it is a many-to-1 function*
  - » *often called a* ***cryptographic hash function***
- Desirable properties:
  - » *easy to calculate*
  - » *irreversibility: cannot determine* ***m*** *from* ***H*****(*****m*****)**
  - » *collision resistance:*
    - computationally difficult to produce ***m*** and ***m'*** such that ***H***(***m***) = ***H***(***m'***)
  - » *seemingly random output*

STANDARD CRYPTOGRAPHIC HASH FUNCTIONS

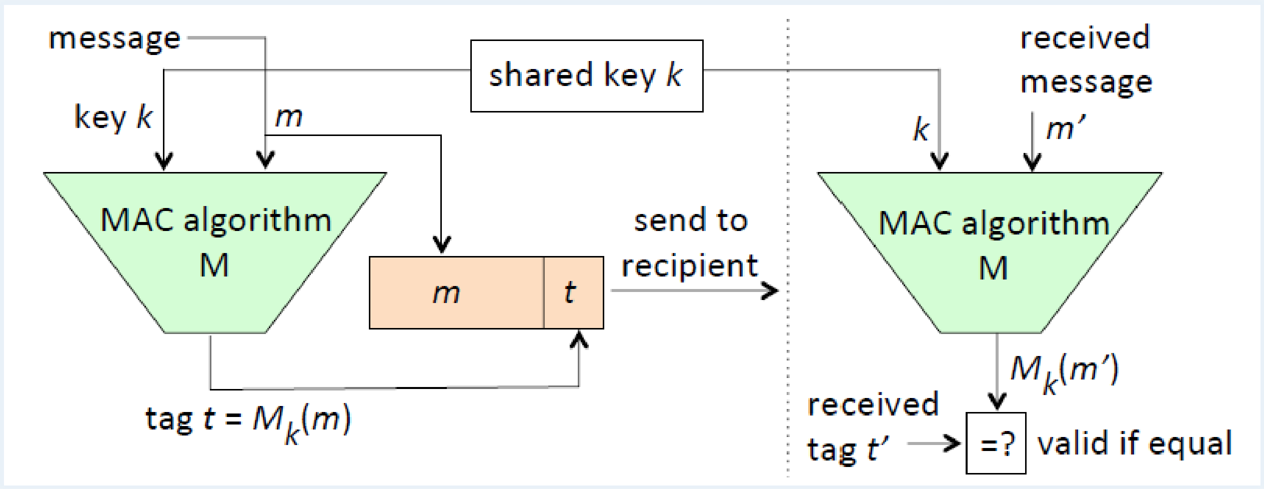
- MD5 hash function widely used (RFC 1321)
  - » *computes 128-bit message digest in a 4-step process*
  - » *broken → **avoid** for security applications*
- SHA-1 is also used
  - » *US standard [NIST, FIPS PUB 180-1]*
  - » *160-bit/256/ message digest*
  - » *considered broken*
- SHA-2
  - » *224/256/384/512-bit output*
- SHA-3 (wiki)
  - » *the result of a NIST competition*

HMAC

- ***Hash-based message authentication code***

  aka ***keyed-hash message authentication code***

- Idea:
  - *add a shared secret to the message*



[CREDIT: Oorschot]