

# Hashing Attacks and Applications

Dimitris Mitropoulos  
[dimitro@di.uoa.gr](mailto:dimitro@di.uoa.gr)

# Hash Function

**Κρυπτογραφική Συνάρτηση Κατακερματισμού:** μια μαθηματική συνάρτηση που έχοντας ως είσοδο μια αυθαίρετου μεγέθους ομάδα δεδομένων, δίνει σαν έξοδο μια καθορισμένου μεγέθους συμβολοσειρά (string). Η έξοδος (που συνήθως αποκαλείται “σύνοψη” — digest) δεν μπορεί να χρησιμοποιηθεί για να παραχθεί η αρχική είσοδος.

$$H : \{0, 1\}^* \rightarrow \{0, 1\}^v$$

# Hash Function

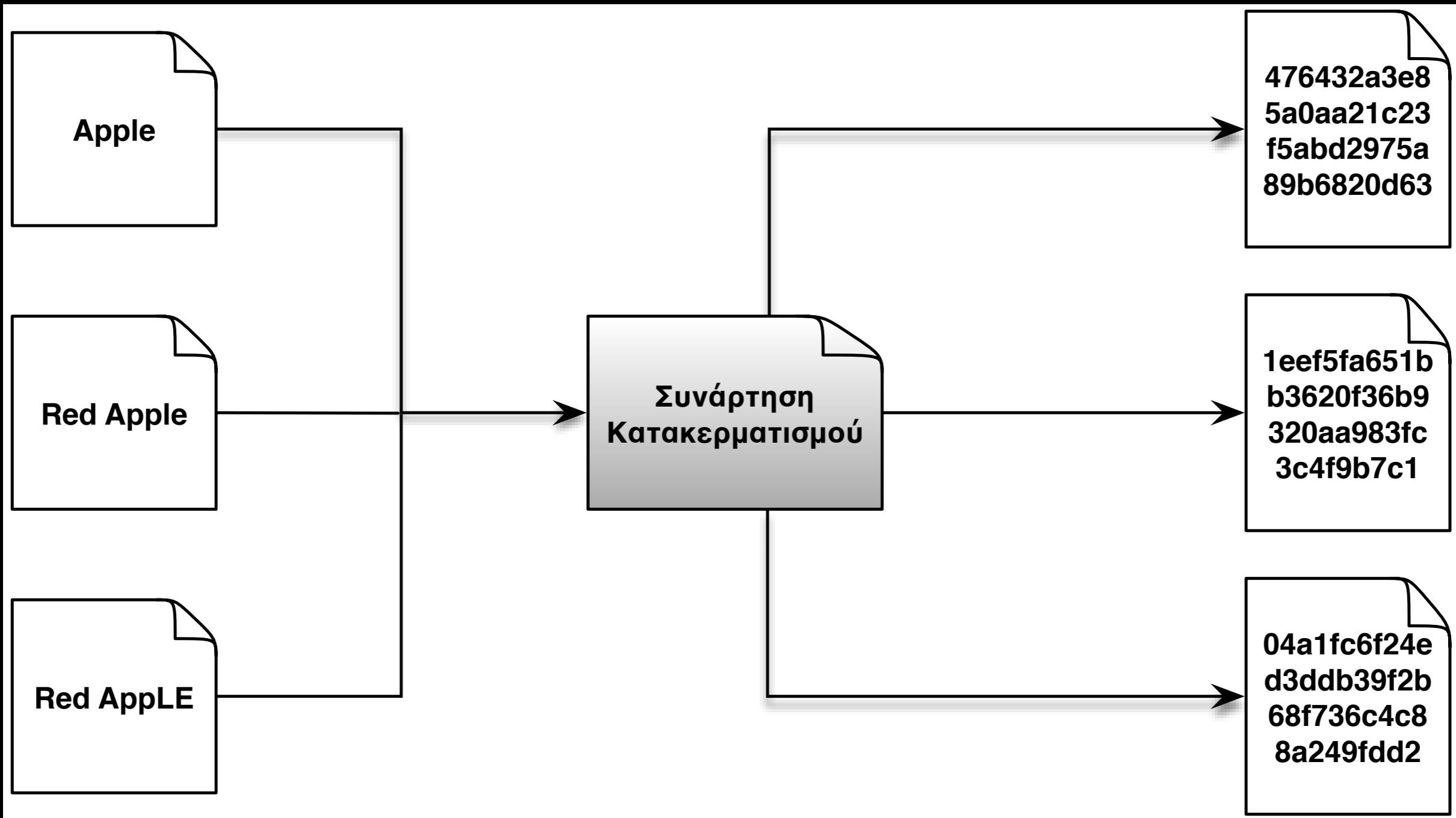
## Ιδιότητες

Ιδιότητες μιας **Ιδανικής** κρυπτογραφικής συνάρτησης κατακερματισμού:

1. Είναι εύκολο να υπολογιστεί η σύνοψη για οποιαδήποτε είσοδο.
2. Δεν είναι εφικτό να βρεις την είσοδο από την σύνοψη.
3. Δεν είναι εφικτό να τροποποιήσεις την είσοδο χωρίς να τροποποιηθεί η σύνοψη.
4. Δεν είναι εφικτό να βρεθούν δύο διαφορετικές είσοδοι που δίνουν την ίδια σύνοψη.

# Hash Function

## The Avalanche Effect



# Hash Function

## Εφαρμογές

- Μια αποδοτική συνάρτηση κατακερματισμού μπορεί να συμβάλει στην εξασφάλιση της **ακεραιότητας** των δεδομένων.
- Μπορεί να χρησιμοποιηθεί για να **δεσμεύσει** οντότητες που συσχετίζονται βάση συγκεκριμένων δεδομένων (π.χ. ένα ηλεκτρονικό συμβόλαιο).
- Η έξοδος της μπορεί να θεωρηθεί ως το **δακτυλικό αποτύπωμα** των δεδομένων εισόδου.
- Η αποθήκευση μιας “σύνοψης” μπορεί να συμβάλει στην εξασφάλιση της **ιδιωτικότητας** των χρηστών μιας εφαρμογής (λ.χ. passwords).

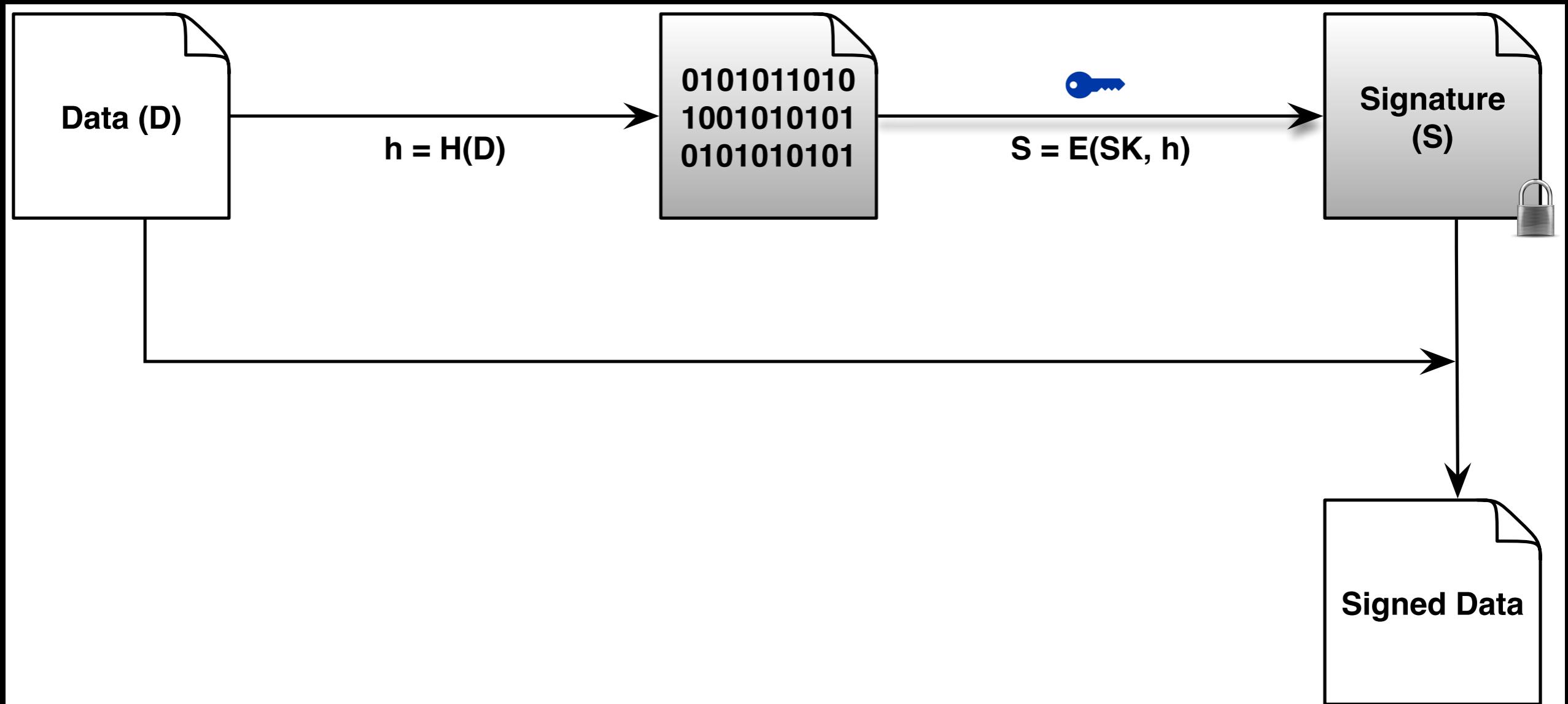
# Committing (Δέσμευση)

“Σφραγισμένες” προσφορές δημοπρασιών (bids):  
Υποβολή μυστικών και ανεξάρτητων προσφορών.

- Ο προσφέρων δημοσιοποιεί την τιμή  $H$ (bid).
- Μετά την δημοσιοποίηση, όλοι οι πλειοδότες ανακοινώνουν τις προσφορές τους.

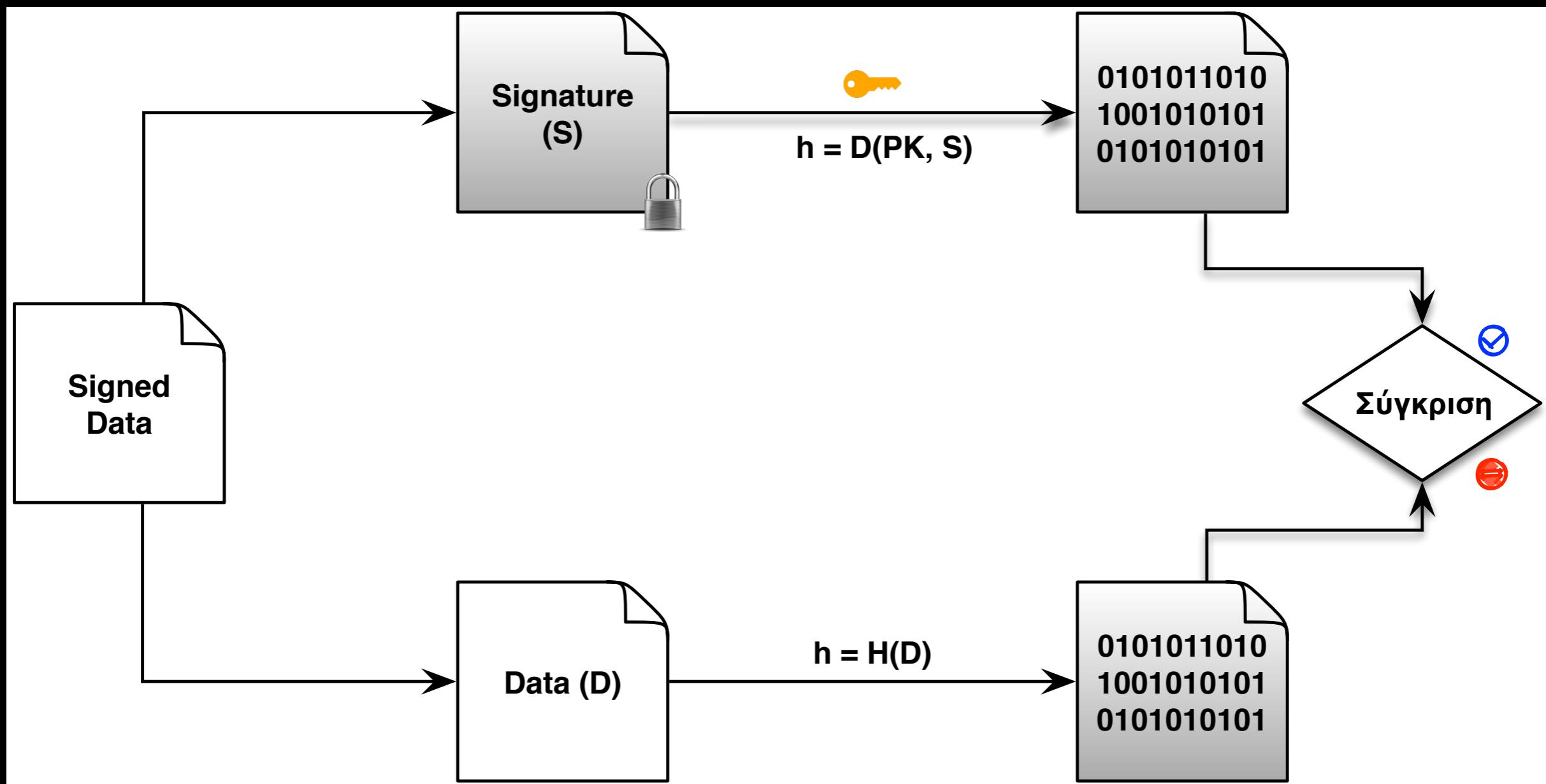
# Ψηφιακή Υπογραφή

## Sign



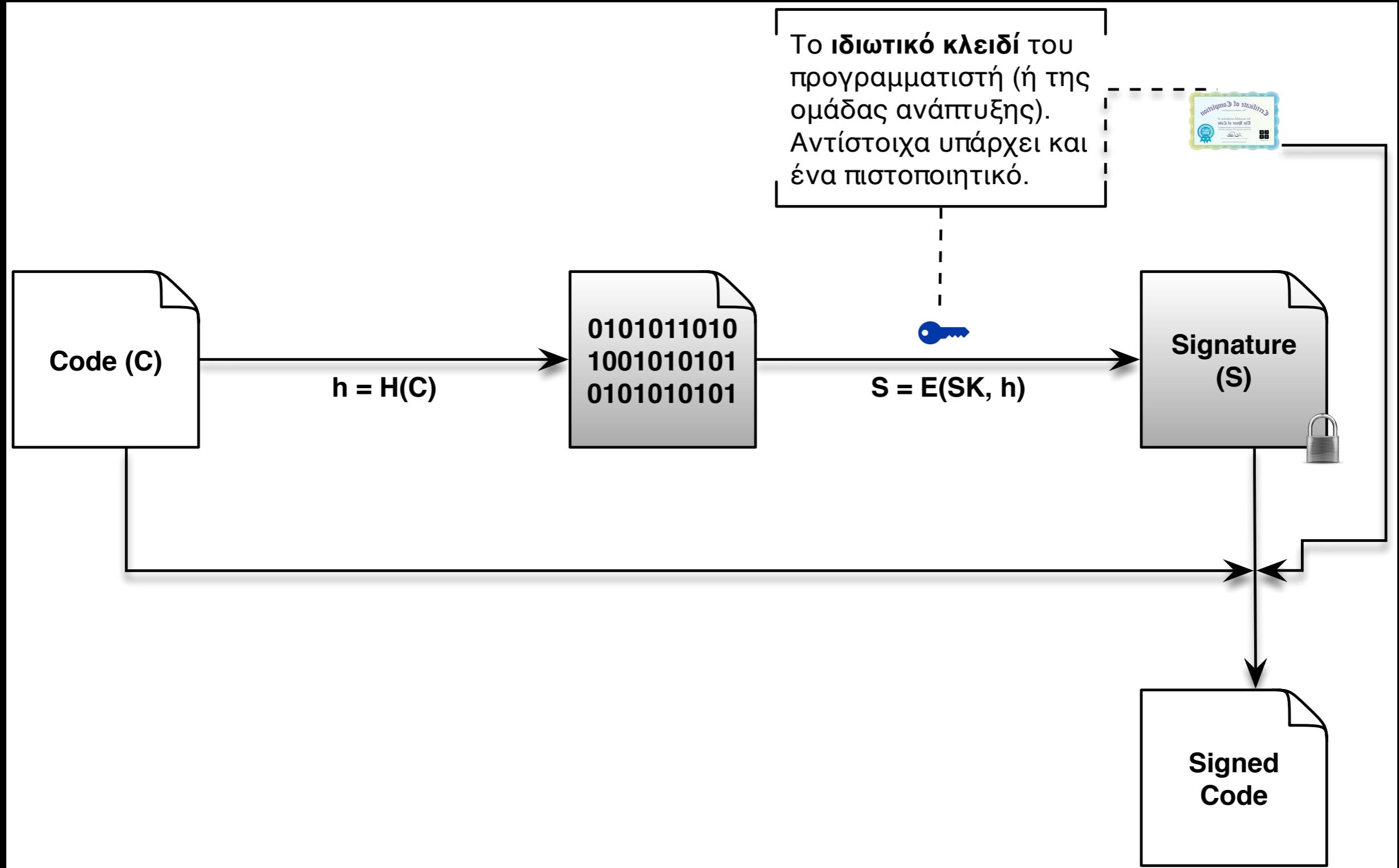
# Ψηφιακή Υπογραφή

## Verify



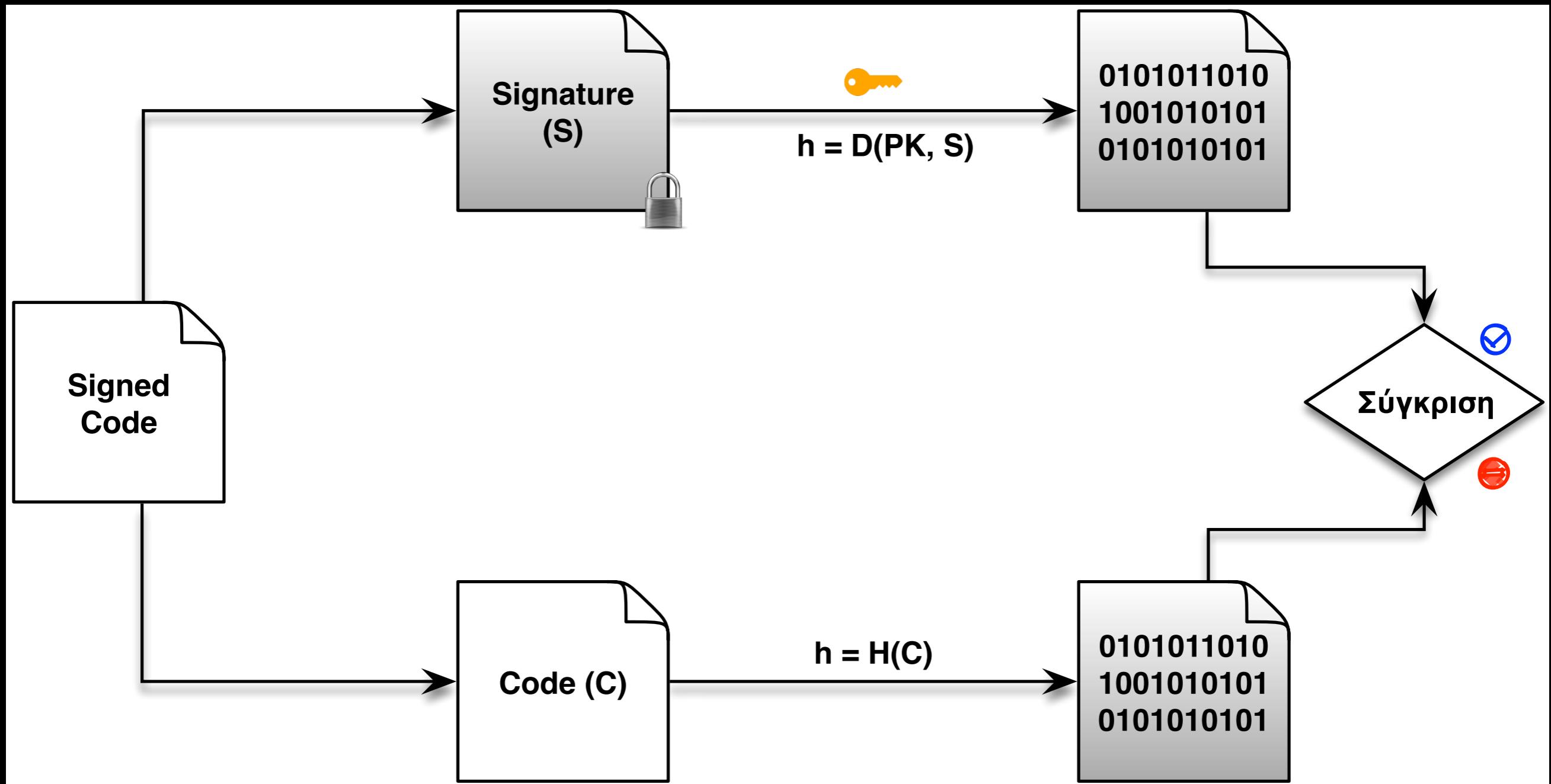
# Code Signing

## Sign



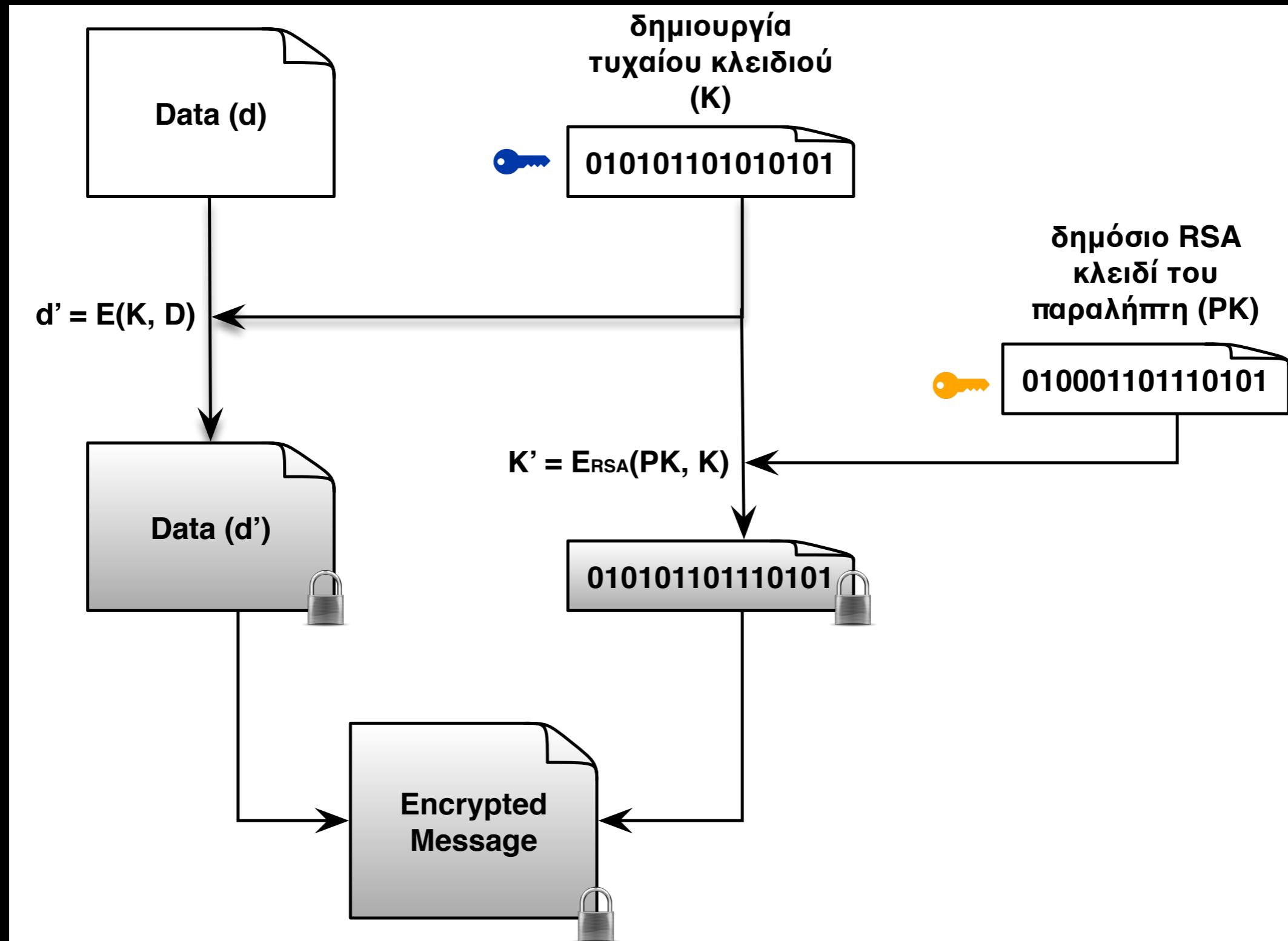
# Code Signing

## Verify



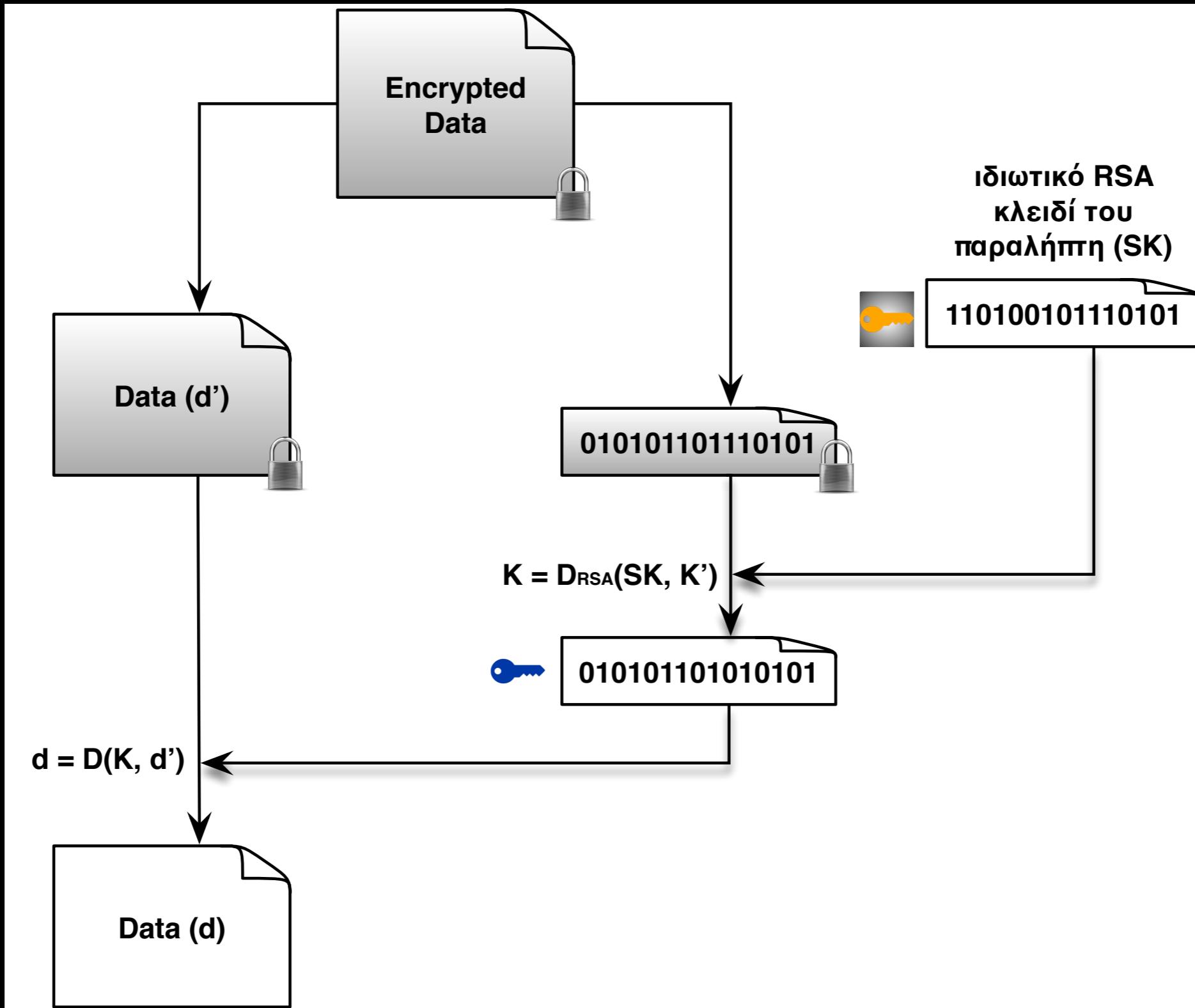
# Pretty Good Privacy

## Encryption



# Pretty Good Privacy

## Decryption



# Hash Function

## Επιθέσεις

- **First Pre-image Attack:** Δεδομένης μιας τιμής  $h = H(m)$ , ο επιτιθέμενος θέλει να βρει την τιμή  $m$ .
- **Second Pre-image Attack:** Δεδομένης μιας εισόδου  $m_1$ , ο αντίπαλος δοκιμάζει να βρει μια δεύτερη  $m_2$  (διάφορη της  $m_1$ ) ώστε να ισχύει η ισότητα:  $H(m_1) = H(m_2)$ .
- **Collision Attack:** Ο επιτιθέμενος προσπαθεί να βρει δυο διαφορετικές εισόδους  $m_1, m_2$  για τις οποίες να ισχύει η ισότητα:  $H(m_1) = H(m_2)$ .

# Hash Function

## First Pre-image Attack

Υποθέτουμε πως σε μια web εφαρμογή αποθηκεύονται στη βάση τα username και τα passwords με την μορφή:

$\{username, H(password)\}$

αντί της:

$\{username, password\}.$

Σε περίπτωση που η βάση κλαπεί, δεδομένου ενός  $h = H(m)$ , ο επιτιθέμενος θα προσπαθήσει να βρει το  $m$  (ή έστω ένα  $m'$  που θα παράξει το  $h$ ).

# Hash Function

## Second Pre-image Attack

Σε μια τέτοια επίθεση ο χρήστης γνωρίζει περισσότερα  $(H(m), m)$ . Υποθέτουμε πως έχουμε την συνάρτηση κατακερματισμού:

$$H(m) = m^d \bmod p^*q$$

Όπου  $p$  και  $q$  είναι δυο (μεγάλοι) πρώτοι αριθμοί και  $d$  ένα γνωστός ακέραιος. Είναι εύκολο να βρούμε ένα  $m' = m^*p^*q + m$  έτσι ώστε:

$$\begin{aligned} H(m') &= H(m^*p^*q + m) = \\ &(m^*p^*q + m)^d \bmod p^*q = \\ &m^d \bmod p^*q \end{aligned}$$

Quiz: Για ποιόν αλγόριθμο μιλάμε εαν η επίθεση είναι first pre-image;  
Quiz 2: Τι θα συνέβαινε εαν πετυχαίναμε κάτι τέτοιο στην περίπτωση  $H(\text{code})$ ;

# The Birthday Paradox

Πόσα άτομα θα πρέπει να βρεθούν σε ένα δωμάτιο ώστε η πιθανότητα να έχουν δυο από αυτά την ίδια μέρα γενέθλια, να είναι 50%;

# The Birthday Paradox

## Εξήγηση

$$p(\text{different}) = 1 \cdot \left(1 - \frac{1}{365}\right) \cdot \left(1 - \frac{2}{365}\right) \cdots \left(1 - \frac{22}{365}\right)$$

Όταν το  $x$  είναι κοντά στο 0 ισχύει:

$$e^x \approx 1 + x$$

κατά συνέπεια:

$$1 - \frac{1}{365} \approx e^{-1/365}$$

# The Birthday Paradox

## Εξήγηση (2)

$$p(\text{different}) \approx 1 \cdot e^{-1/365} \cdot e^{-2/365} \cdots e^{-22/365}$$

$$\Leftrightarrow p(\text{different}) \approx e^{(-1-2-3\cdots-22)/365}$$

$$\Leftrightarrow p(\text{different}) \approx e^{-(1+2+\cdots+22)/365}$$

εαν προσθέσουμε από 1 έως  $n$  έχουμε  $n(n+1)/2$ , έτσι:

$$p(\text{different}) \approx e^{-((23 \cdot 22)/(2 \cdot 365))} = .499998$$

βάζοντας  $n$  αντί για 23:

$$p(\text{different}) \approx e^{-(n^2/(2 \cdot 365))}$$

# The Birthday Paradox

## Εξήγηση (3)

Γενικότερα: έχοντας  $n$  ανθρώπους και  $T$  αντικείμενα:

$$p(\text{different}) \approx e^{-(n^2/2 \cdot T)}$$

υποθέτουμε πως θέλουμε πιθανότητα 50% και λύνουμε ως προς  $n$ :

$$p(\text{different}) \approx e^{-(n^2/2 \cdot T)}$$

$$<=> 1 - p(\text{match}) \approx e^{-(n^2/2 \cdot T)}$$

$$<=> 1 - .5 \approx e^{-(n^2/2 \cdot T)}$$

$$<=> -2\ln(.5) \cdot T \approx n^2$$

$$<=> n \approx 1.177\sqrt{T}$$

Θέλουμε Hash functions με μεγάλο  $T$  !

# Collision Attack

Χρησιμοποιώντας το B.P.

1. Τυχαία δειγματοληψία:  $x, H(x)$
2. Αποθήκευση η ζευγαριών σε ένα table.
3. Ταξινόμηση με βάση το  $H(x)$ .
4. Linear pass για entries που έχουν ίσα  $h$ .

# Collision Attack

## Χρησιμοποιώντας το B.P. (2)

- Η Mallory θέλει να ξεγελάσει τον Bob ώστε αυτός να υπογράψει μια **διαφορετική** σύμβαση από αυτή που συμφώνησαν.
- Η Mallory φτιάχνει μια κανονική σύμβαση **m** και μια “δόλια” **m'**.
- Έπειτα βρίσκει τρόπους να αλλάξει τη σύμβαση χωρίς να αλλάζει το νόημά της (προσθέτει κενά, αφαιρεί κόμματα, κ.α.) και έτσι παράγει πολλές **παραλλαγές** του m.
- Στη συνέχεια κάνει κάτι αντίστοιχο για την m'.
- Μετά, για όλες τις παραλλαγές παράγει τα αντίστοιχα digests χρησιμοποιώντας μια  $H(x)$ .
- Τέλος, ψάχνει να βρει περιπτώσεις όπου:  $H(m) = H(m')$ .

Quiz: είναι η ίδια περίπτωση με τα γενέθλια;

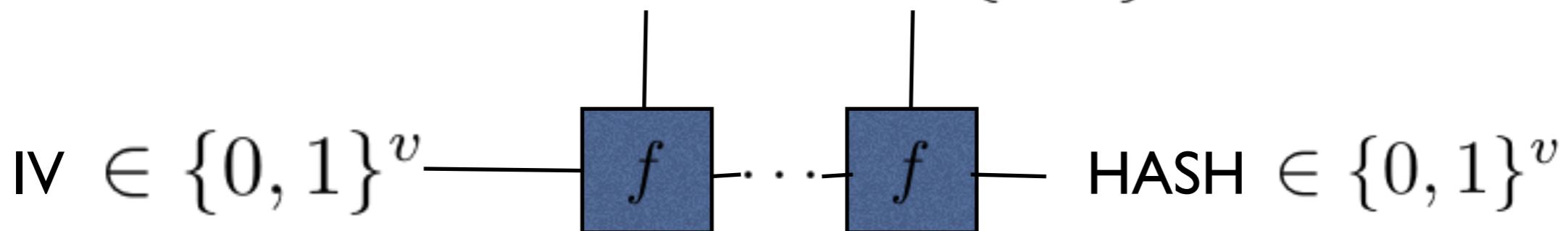
# Merkle Damgård Design

Υποθέτουμε πως έχουμε μια καλή one-way “compression” συνάρτηση (με αποδοτικό avalanche effect):

$$f : \{0, 1\}^v \times \{0, 1\}^b \rightarrow \{0, 1\}^v$$

Θέλουμε να επεκτείνουμε την είσοδο με το IV:

$$m_1, \dots, m_N \in \{0, 1\}^b$$



Μετά το τελευταίο compression γίνεται padding.

# Merkle Damgård Design

## Padding (?)

Εάν η είσοδος είναι:

HashInput

Kai το compression function είναι 8 bytes, είναι  
θεμιτό να κάνουμε padding με μηδενικά;

HashInput **0000000**

# Merkle Damgård Design

## Padding

Προτιμάμε για:

HashInput

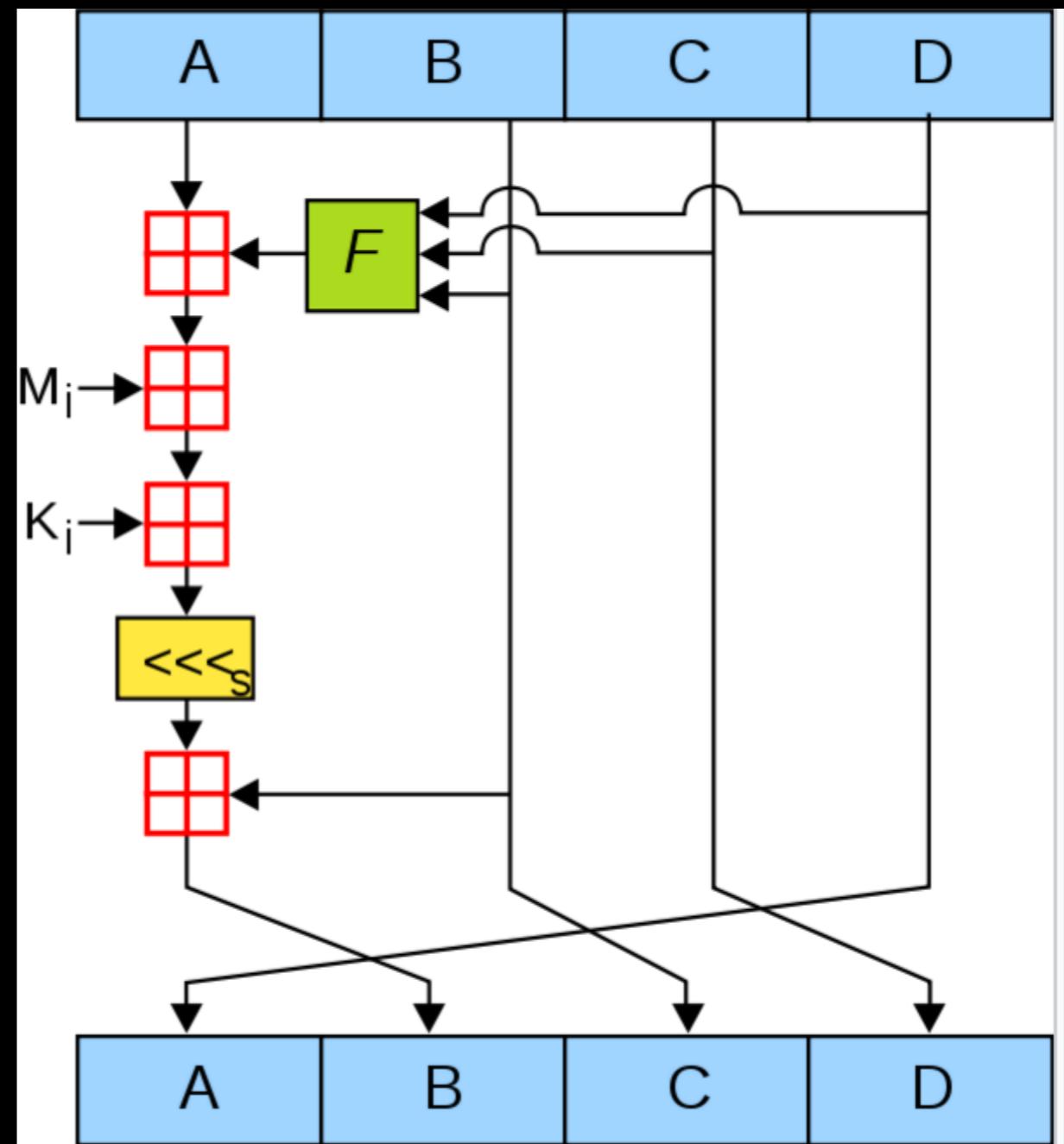
Να έχουμε:

HashInput **1000000 00000009**

Το μήκος της εισόδου

# MD5

- 128 bit έξοδος.
- Η είσοδος “σπάει” σε blocks των 512 bits (16 λέξεις των 32 bit).
- Padding μέχρι το μήκος να διαιρείται με το 512.
- Χρησιμοποίηση: addition modulo  $2^{32}$ , rotation.



Πηγή εικόνας: <https://en.wikipedia.org/wiki/MD5>

# MD5

## Επιθέσεις

- Ευάλωτος σε collision attacks (2004).
- Πλέον μπορεί να βρεθεί collision σε λιγότερο από μερικά **δευτερόλεπτα**.
- Ο ιός **Flame** εκμεταλλεύταν collisions για να δείχνει πως είναι επίσημο προϊόν της Microsoft.





# SHA

## Secure Hash Algorithms

- Ακολουθούν και αυτοί το Merkle Damgård Design.
- Αποτελούν NIST (National Institute of Standards and Technology) standards.
- Ο SHA-1 αντικατέστησε τον SHA-0 που ήταν ευάλωτος σε collision attacks από πολύ νωρίς (1998). Ο SHA-1 προσέθετε ένα rotation ακόμα. Και οι δύο παράγουν έξοδο 160 bits.
- Πρώτα collision attacks στον SHA-1 το 2005.
- Υπάρχουν επίσης οι SHA-2 και ο SHA-3 (224, 256, 384 ή 512 bits έξοδος).

πριν μερικές  
εβδομάδες...

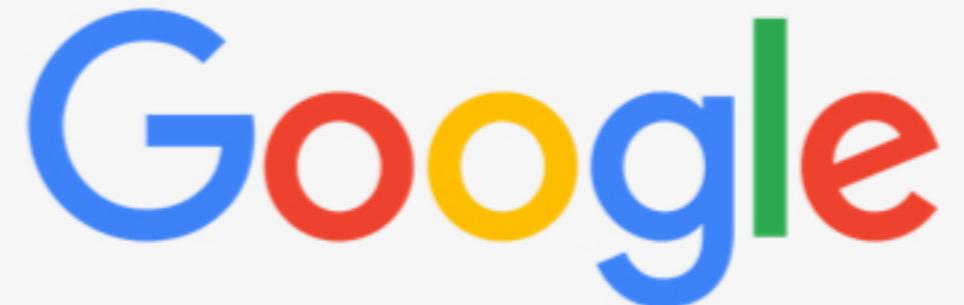
# SHAttered

The first concrete collision attack against SHA-1

*<https://shattered.io>*



Marc Stevens  
Pierre Karpman



Elie Bursztein  
Ange Albertini  
Yarik Markov

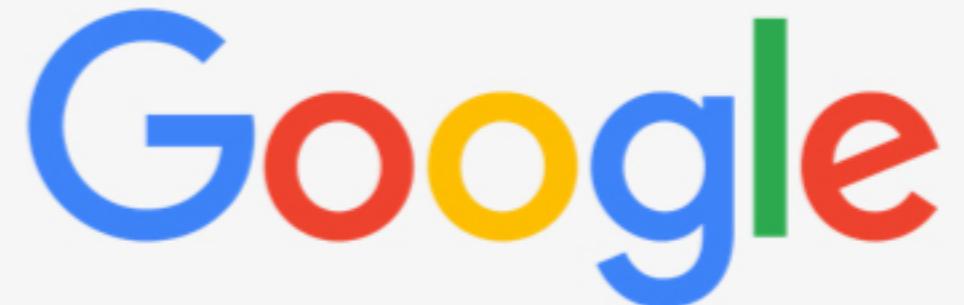
# SHAttered

The first concrete collision attack against SHA-1

*<https://shattered.io>*



Marc Stevens  
Pierre Karpman



Elie Bursztein  
Ange Albertini  
Yarik Markov

# SHA-1 Collisions

Επηρεάζουν τον τρόπο λειτουργίας του Git, του SVN, του BitTorrent και πολλών άλλων εφαρμογών.

# MACs

## Message Authentication Codes

- Χρησιμοποιούνται ευρέως για την διαφύλαξη της **ακεραιότητας** (δεν μας απασχολεί η εμπιστευτικότητα) των δεδομένων (OS files, banner ads κ.α.).
- Συνδυασμός συμμετρικής κρυπτογράφησης και συναρτήσεων κατακερματισμού:

$$\mathcal{H}_k : \{0, 1\}^* \rightarrow \{0, 1\}^n$$

# MACs

## Ορισμός



**MAC:** ένα ζεύγος αλγορίθμων ( $S, V$ ) που ορίζονται από 3 σύνολα ( $K, M, T$ ):

1.  $S(k, m)$  παράγει ένα  $T$
2.  $V(k, m, t)$  δίνει ως έξοδο 'yes' ή 'no'.

**Ορθότητα:**  $V(k, m, S(k, m)) = \text{'yes'}$

# MACs

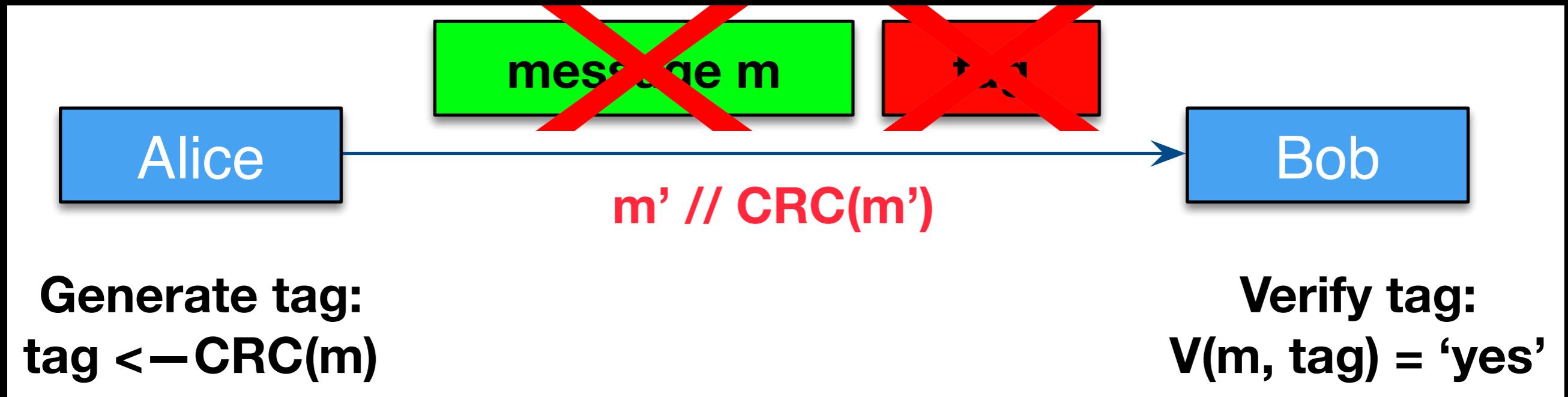
Χρειάζεται το key;



(CRC: Cyclic Redundancy Check)

# MACs

## Xwroíç to key



# Secure MACs

- Τι μπορεί να κάνει ο αντίπαλός;

## Chosen Message Attacks:

για  $m_1, m_2, \dots, m_q$ ,

ο αντίπαλος λαμβάνει  $t_i \leftarrow S(k, m_1)$

- Τι θέλει να κάνει;

## Existential Forgery:

Να δημιουργήσει νέα έγκυρα ζεύγη message/tag ( $m, t$ ).

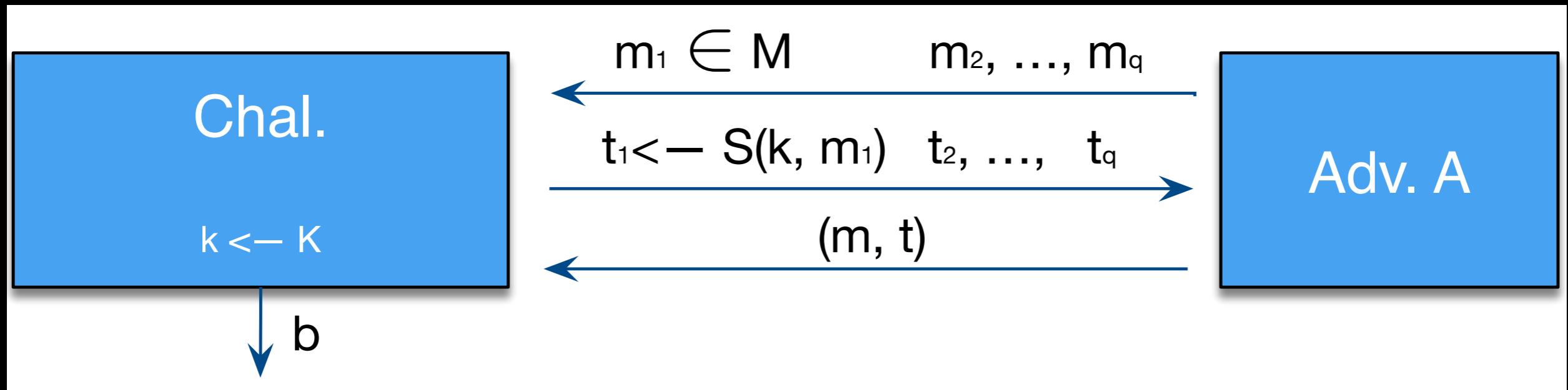
$(m, t) \notin \{(m_1, t_1), (m_2, t_2), \dots, (m_q, t_q)\}$

- 
1. Ο αντίπαλος **δεν** θα πρέπει να μπορεί να δημιουργήσει ένα τέτοιο έγκυρο ζεύγος.
  2. Δεδομένου ενός  $(m, t)$  ο αντίπαλος **δεν** θα πρέπει να μπορεί να παράξει ένα  $(m, t')$  για  $t \neq t'$ .

# Secure MACs

## Probability Game

Για έναν MAC  $I = (S, V)$  έχουμε το παρακάτω MAC Game:



$b = 1$  εαν  $V(k, m, t) = \text{'yes'}$  και  $(m, t) \notin \{(m_1, t_1), \dots, (m_q, t_q)\}$   
 $b = 0$  σε διαφορετική περίπτωση.

Ένας  $I = (S, V)$  είναι **ασφαλής** για όλους τους αντιπάλους εαν η πιθανότητα το  $b = 1$  είναι ελάχιστη.

# MACs

Προστασία των system files

$F_1$

$F_2$

...

$F_n$

$t_1 = S(k, F_1)$

$t_2 = S(k, F_2)$

$t_n = S(k, F_n)$

το  $k$  προέρχεται  
από το **password**  
που δίνει ο χρήστης

# Bιβλιογραφία

Stephane Manuel. Classification and Generation of Disturbance Vectors for Collision Attacks against SHA-1. [Online] Available: <http://eprint.iacr.org/2008/469.pdf>.

Halevi S., Krawczyk H. Strengthening Digital Signatures Via Randomized Hashing. *Lecture Notes in Computer Science*, pp. 41-59. 2006.

Mihir Bellare, Tadayoshi Kohno: Hash Function Balance and Its Impact on Birthday Attacks. *EUROCRYPT 2004*: pages 401–418.

Al-Kuwari, S., Davenport, J. H. and Bradford, R. J., 2010. Cryptographic hash functions: recent design trends and security notions. In: *Short Paper Proceedings of 6th China International Conference on Information Security and Cryptology (Inscrypt '10)*. Science Press of China, pp. 133-150.

Apple Code Signing Guide [Online]. Available: <https://developer.apple.com/library/content/documentation/Security/Conceptual/CodeSigningGuide/Introduction/Introduction.html>.

Xiaoyun Wang & Hongbo Yu. How to Break MD5 and Other Hash Functions. *Advances in Cryptology – Lecture Notes in Computer Science*. pp. 19–35.

McElroy, Damien; Williams, Christopher. "Flame: World's Most Complex Computer Virus Exposed". The Daily Telegraph. 28 May 2012 [Online] Available: <http://www.telegraph.co.uk/news/worldnews/middleeast/iran/9295938/Flame-worlds-most-complex-computer-virus-exposed.html>.

Florent Chabaud, Antoine Joux. Differential Collisions in SHA-0. In *CRYPTO '98*

Vincent Rijmen and Elisabeth Oswald. Update on SHA-1. Cryptology ePrint Archive: Report 2005/010 [Online]. Available: <http://eprint.iacr.org/2005/010>.

The Keyed-Hash Message Authentication Code (HMAC). National Institute of Standards and Technology. [Online]. Available: <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf>.