27-4-2017

Ο στόχος μας μέχρι τώρα ήταν να προστατεύσουμε το μήνυμα, δηλαδή ή μυστικότητα. Μια άλλη ιδιότητα που θέλουμε να διασφαλίσουμε είναι η αυθεντικότητα του μηνύματος και του αποστολέα, να γνωρίζουμε δηλαδή αν το μήνυμα είναι αλλοιωμένο ακεραιότητα) και να γνωρίζουμε την ταυτότητα του αποστολέα (ψηφιακή υπογραφή).

Συνάρτηση hash

h: F* -> G^s όπου F* είναι το σύνολο μηνυμάτων αυθαίρετου μήκους και G^s το σύνολο μηνυμάτων μήκους s.

Στην κρυπτογραφία ορίζουμε εμείς το s. Συνήθως επιλέγουμε s ίσο με 128 ή 160 bit. Η συνάρτηση hash δεν έιναι 1:1, αλλά το σύνολο F* είναι πολύ μεγαλύτερο από το G^s. Η μορφή της συνάρτησης hash που χρησιμοποιούμε στην κρυπτογραφία είναι η **one-way hash function:**

1. Δεδομένου y=h(x), είναι υπολογιστικά δύσκολο να βρω το x.
2. Ασθενής αντίσταση σε συγκρούσεις: Δεδομένων x, h(x), είναι υπολογιστικά δύσκολο να βρώ x≠x' : h(x)=h(x').
3. Ισχυρή αντίσταση σε συγκρούσεις: Είναι υπολογιστικά δύσκολο να βρεθούν x, x' με x≠x' : h(x)=h(x').

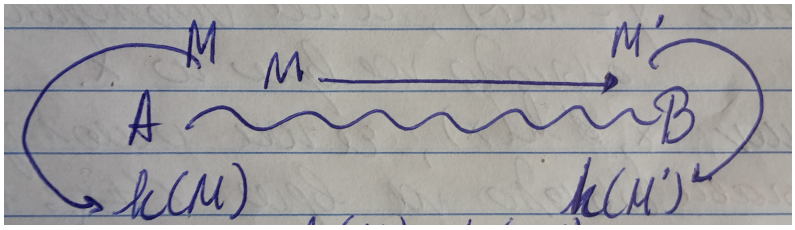
Δεν γνωρίζουμε αν υπάρχουν hash συναρτήσεις, αλλά ελπίζουμε να υπάρχουν. Στην κρυπτογραφία δεχόμαστε ότι κάτι ισχύει αν δεν μπορούμε να αποδείξουμε ότι δεν ισχύει. Έτσι μπορούμε να χρησιμοποιήσουμε hash συναρτήσεις ακόμα και αν δεν γνωρίζουμε ότι υπάρχουν.

Το παράδοξο των γενεθλίων – the birthday paradox

Έστω P(n) η πιθανότητα σε ένα σύνολο n ανθρώπων δύο από αυτούς να έχουν γενέθλια την ίδια μέρα. Η πιθανότητα αυτή είναι λίγο πάνω από 50% για μόλις 23 ανθρώπους, ενώ φτάνει στο 99% με 70. Ο υπολογισμός γίνεται από τον τύπο $P(n)=\frac{365!}{(365-n)!\,365^n}$

Γενικά, αν το έτος έχει *k* ημέρες, θα χρειαζόμασταν ένα σύνολο ανθρώπων της τάξης $\sqrt{k}$ για να έχουμε πιθανότητα τουλάχιστον ίση με 50%.

Αυθεντικότητα μηνύματος



Αν h(M)=h(M’) τότε το μήνυμα δεν έχει αλλοιωθεί.

Για να διασφαλίσουμε την ακεραιότητα του μηνύματος χρησιμοποιούμε τις παρακάτω μεθόδους: MAC – Message Authentication Code και MDC – Message Detection Code.

Αυθεντικότητα αποστολέα

Με τον όρο αυτό εννοούμε την ψηφιακή υπογραφή.
Συνάρτηση sign(M)->σ
συνάρτηση verify(M,σ) =1 αν σ=sign(M)
=0 αν σ≠sign(M)

Ιδιότητες:

Οι sign και verify είναι συναρτήσεις που χρειάζονται πολυωνυμικό χρόνο δεδομένων M, σ.
Είναι υπολογιστικά δύσκολο να βρω M≠M’ με sign(M)=sign(M’)

Η ψηφιακή υπογραφή υλοποιείται με 2 τρόπους: με παράρτημα και με ανάκτηση μηνύματος.

Με παράρτημα

Δεδομένου μηνύματος M:
αποστολέας:

- υπολογίζω m’=h(M)
- υπολογίζω σ=sign(m’)
- στέλνω (M, σ)

παραλήπτης:

- υπολογίζω h(M)
- verify(h(M), σ)

Με ανάκτηση μηνύματος

Δεδομένου μηνύματος M:
αποστολέας:

- υπολογίζω m’=R(M) όπου R λέγεται **συνάρτηση περίσσειας:** επιβάλλει συγκεκριμένη δομή στο μήνυμα, ώστε να μην μπορεί να αλλοιωθεί.
- στέλνω σ=sign(m’)

παραλήπτης:

- σ → m’’
- R^{-1}(m’’)=M

Η R είναι δημόσια γνωστή συνάρτηση, όπως και η hash. Η ασφάλεια επιτυγχάνεται από το συνδυασμό των sign, verify και της διαδικασίας σ → m’’, οι οποίες είναι προκαθορισμένες για το κάθε πρωτόκολλο.

RSA με ψηφιακή υπογραφή

1. Κάθε χρήστης δημιουργεί κλειδιά υπογραφών:

- διαλέγει 2 μεγάλους πρώτους, p,q.
- Υπολογίζει n=pq
- διαλέγει e: gcd(e, φ(n))=1, φ(n)=(p-1)(q-1)
- υπολογίζει d με ed=1(mod(φ(n)))
- P_A=(n,e)
- S_A=(d)

2. sign σ=sign(M)=M^dmodn

3. verify(M,σ) ελέγχω αν σ^emodn=M

ΘΕΛΕΙ ΕΠΕΞΗΓΗΣΗ:

Αν θέλουμε να προστατεύσουμε το μήνυμα, θα πρέπει να ακολουθήσουμε την παρακάτω διαδικασία: (RSA με παράρτημα και hash)

A: M → M^{dA}modn_A → $(M^{dA}\text{mod}n_A)^{eB}modn_B$

B: $((M^{dA}\text{mod}n_A)^{eB}\text{mod}n_B)^{dB}modn_B$ → $(M^{dA}\text{mod}n_A)^{eA}modn_A$

βλ. Λειτουργία RSA verify

Στην πράξη έχει σημασία αν το n_A είναι μεγαλύτερο από το n_B. Αν ισχύει, τότε υπάρχουν M_1, M_2 με $M_1 \neq M_2$ που κατά την κρυπτογράφηση δίνουν το ίδιο αποτέλεσμα, οπότε κατά την αποκρυπτογράφηση, ο παραλήπτης δεν θα γνωρίζειο ποιό είναι το πραγματικό μήνυμα. Το πρόβλημα αυτό αντιμετωπίζεται με χρήση hash και redundancy functions.