Πιστοποίηση ταυτότητας

Νικόλαος Ε. Κολοκοτρώνης
Επίκουρος Καθηγητής

Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Πανεπιστήμιο Πελοποννήσου
Email: nkolok@uop.gr
Web: http://www.uop.gr/~nkolok/

ΑΣΦΑΛΕΙΑ ΣΥΣΤΗΜΑΤΩΝ

Περιεχόμενα

- Απλή πιστοποίηση
- Βιομετρικά συστήματα
- Σύστημα Kerberos
- Διαλογικά σχήματα

Πιστοποίηση χρηστών

- fundamental security building block
  - basis of access control & user accountability
- is the process of verifying an identity claimed by or for a system entity
- has two steps:
  - identification - specify identifier
  - verification - bind entity (person) and identifier
- distinct from message authentication

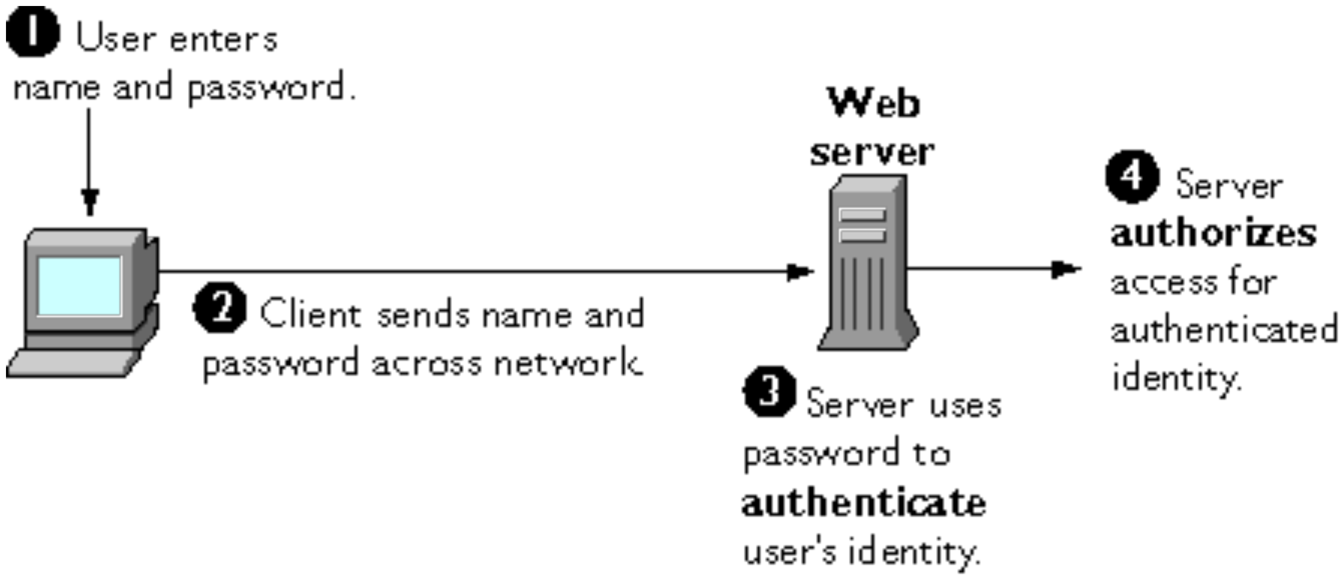
Πιστοποίηση χρηστών: τρόποι

- four means of authenticating user's identity
- based one something the individual
  - knows - e.g. password, PIN
  - possesses - e.g. key, token, smartcard
  - is (static biometrics) - e.g. fingerprint, retina
  - does (dynamic biometrics) - e.g. voice, sign
- can use alone or combined
  - all can provide user authentication
  - all have issues

Πιστοποίηση: με συνθηματικό

- widely used user authentication method
  - user provides name/login and password
  - system compares password with that saved for specified login
- authenticates ID of user logging and
  - that the user is authorized to access system
  - determines the user’s privileges
  - is used in discretionary access control

Πιστοποίηση: με συνθηματικό



Πιστοποίηση: με συνθηματικό

- Είναι ασθενής πιστοποίηση
- Ο Β περιμένει από τον Α συγκεκριμένο κωδικό, για να πειστεί ότι είναι αυτός
- Οι κωδικοί κρατούνται κρυπτογραφημένοι σε κάποιο αρχείο (συνάρτηση σύνοψης μίας κατεύθυνσης)
  - Ακόμα κι αν κάποιος εισβολέας αποκτήσει πρόσβαση στο αρχείο, δεν μπορεί να ανακαλύψει τους κωδικούς

Πιστοποίηση: με συνθηματικό

Offline dictionary attack

- Determined hackers can frequently bypass access controls and gain access to the system's password file

Password guess against one user

- The attacker attempts to gain knowledge about the account holder and system password policies and uses that knowledge to guess password

Exploiting multiple password use

- Attacks can become much more effective or damaging if different network devices share the same or a similar password for a given user

Popular password attack

- Attack is to use a popular password and try it against a wide range of user IDs

Πιστοποίηση: με συνθηματικό

Workstation hijacking

- The attacker waits until a logged-in workstation is unattended

Specific account attack

- The attacker targets a specific account and submits password guesses until the correct password is discovered

Electronic monitoring

- If a password is communicated across a network to log on to a remote system, it is vulnerable to eavesdropping

Exploiting user mistakes

- Attackers are often successful in obtaining passwords via social engineering tactics that trick a user into revealing a password (see the next video)

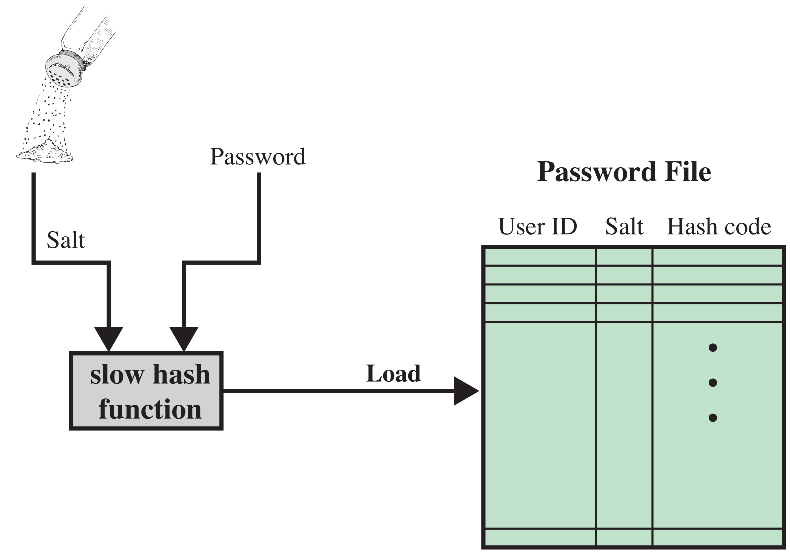
Social engineering attack

Πιστοποίηση: με συνθηματικό

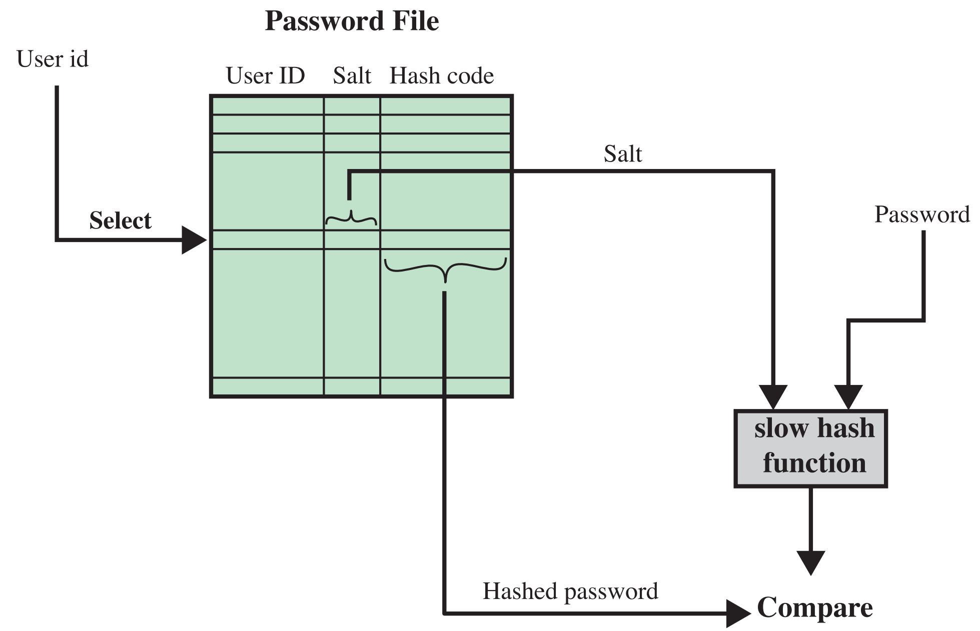
- Countermeasures
  - stop unauthorized access to password file
  - intrusion detection measures
  - account lockout mechanisms
  - policies against using common passwords but rather hard to guess passwords
  - training & enforcement of policies
  - automatic workstation logout
  - encrypted network links

Πιστοποίηση: με συνθηματικό

‘slow hash function’
= crypt(3)
in Unix systems



(a) Loading a new password



(b) Verifying a password

Πιστοποίηση: με συνθηματικό

- UNIX crypt(3) Implementation: original scheme has
  - 8 character password form 56-bit key
  - 12-bit salt used to modify DES encryption into a one-way hash function
  - 0 value repeatedly encrypted 25 times
  - output translated to 11 character sequence
- now regarded as woefully insecure
  - e.g. supercomputer, 50 million tests, 80 min
- sometimes still used for compatibility

Πιστοποίηση: με συνθηματικό

- New implementations have stronger hash/salt variants
- many systems (incl. Linux, Solaris, and FreeBSD) use MD5
  - with 48-bit salt
  - password length is unlimited
  - is hashed with 1000 times inner loop
  - produces 128-bit hash
- OpenBSD uses Blowfish block cipher based hash algorithm called Bcrypt
  - uses 128-bit salt to create 192-bit hash value

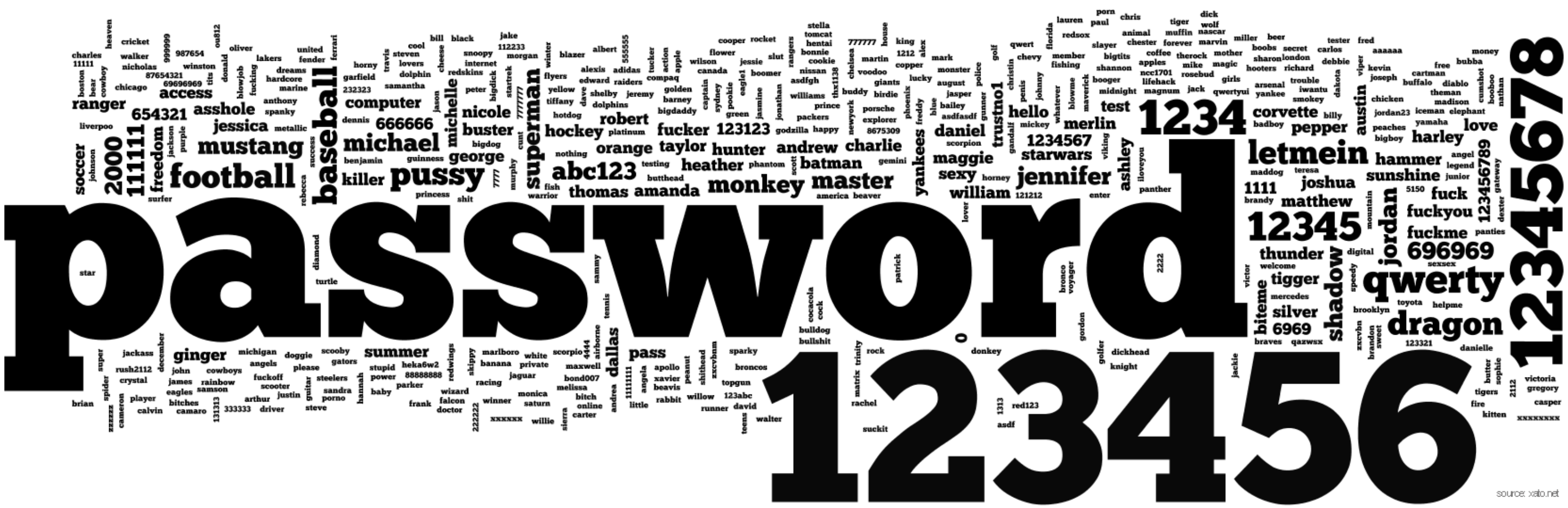
Χρήση συνθηματικών: επιθέσεις

- dictionary attacks
  - try each word in large dictionary against hash in password file
  - RFC 1750, X9.17
- rainbow table attacks
  - precompute tables of hash values for all salts
  - a mammoth table of hash values
  - e.g. 1.4GB table cracks 99.9% of alphanumeric Windows passwords in 13.8 secs
  - not feasible if larger salt values used

Χρήση συνθηματικών: επιλογή

- users may pick short passwords
  - e.g. 3% were 3 chars or less, easily guessed
  - system can reject choices that are too short
- users may pick guessable passwords
  - so crackers use lists of likely passwords
  - e.g. one study of 13.797 encrypted passwords guessed nearly 1/4 of them (next slides)
  - would take about 1 hour on fastest systems to compute all variants, and only need 1 break!

Χρήση συνθηματικών: επιλογή



Χρήση συνθηματικών: επιλογή

- Observed password lengths

Length	Number	Fraction of Total
1	55	.004
2	87	.006
3	212	.02
4	449	.03
5	1260	.09
6	3035	.22
7	2917	.21
8	5772	.42
Total	13787	1.0

Χρήση συνθηματικών: επιλογή

Passwords cracked from a sample set of 13,797 accounts

Type of Password	Search Size	Number of Matches	Percentage of Passwords Matched	Cost/Benefit Ratio[a]
User/account name	130	368	2.7%	2.830
Character sequences	866	22	0.2%	0.025
Numbers	427	9	0.1%	0.021
Chinese	392	56	0.4%	0.143
Place names	628	82	0.6%	0.131
Common names	2239	548	4.0%	0.245
Female names	4280	161	1.2%	0.038
Male names	2866	140	1.0%	0.049
Uncommon names	4955	130	0.9%	0.026
Myths & legends	1246	66	0.5%	0.053
Shakespearean	473	11	0.1%	0.023
Sports terms	238	32	0.2%	0.134
Science fiction	691	59	0.4%	0.085
Movies and actors	99	12	0.1%	0.121
Cartoons	92	9	0.1%	0.098
Famous people	290	55	0.4%	0.190
Phrases and patterns	933	253	1.8%	0.271
Surnames	33	9	0.1%	0.273
Biology	58	1	0.0%	0.017
System dictionary	19683	1027	7.4%	0.052
Machine names	9018	132	1.0%	0.015
Mnemonics	14	2	0.0%	0.143
King James bible	7525	83	0.6%	0.011
Miscellaneous words	3212	54	0.4%	0.017
Yiddish words	56	0	0.0%	0.000
Asteroids	2407	19	0.1%	0.007
TOTAL	62727	3340	24.2%	0.053

Χρήση συνθηματικών: επιλογή

- clearly have problems with passwords
- goal to eliminate guessable passwords
- whilst still easy for user to remember
- techniques:
  - user education
  - computer-generated passwords
  - reactive password checking
  - proactive password checking

Χρήση συνθηματικών: επιλογή

- The goal is to eliminate guessable passwords
- while allowing the user to select a password that is memorable

User education

- Users are told the importance of hard-to-guess passwords and are provided with guidelines for selecting strong passwords

Computer generated passwords

- Computer-generated password schemes have a history of poor acceptance by users
- Users have difficulty remembering them

Reactive password checking

- A strategy where the system periodically runs own password cracker to find any guessable passwords

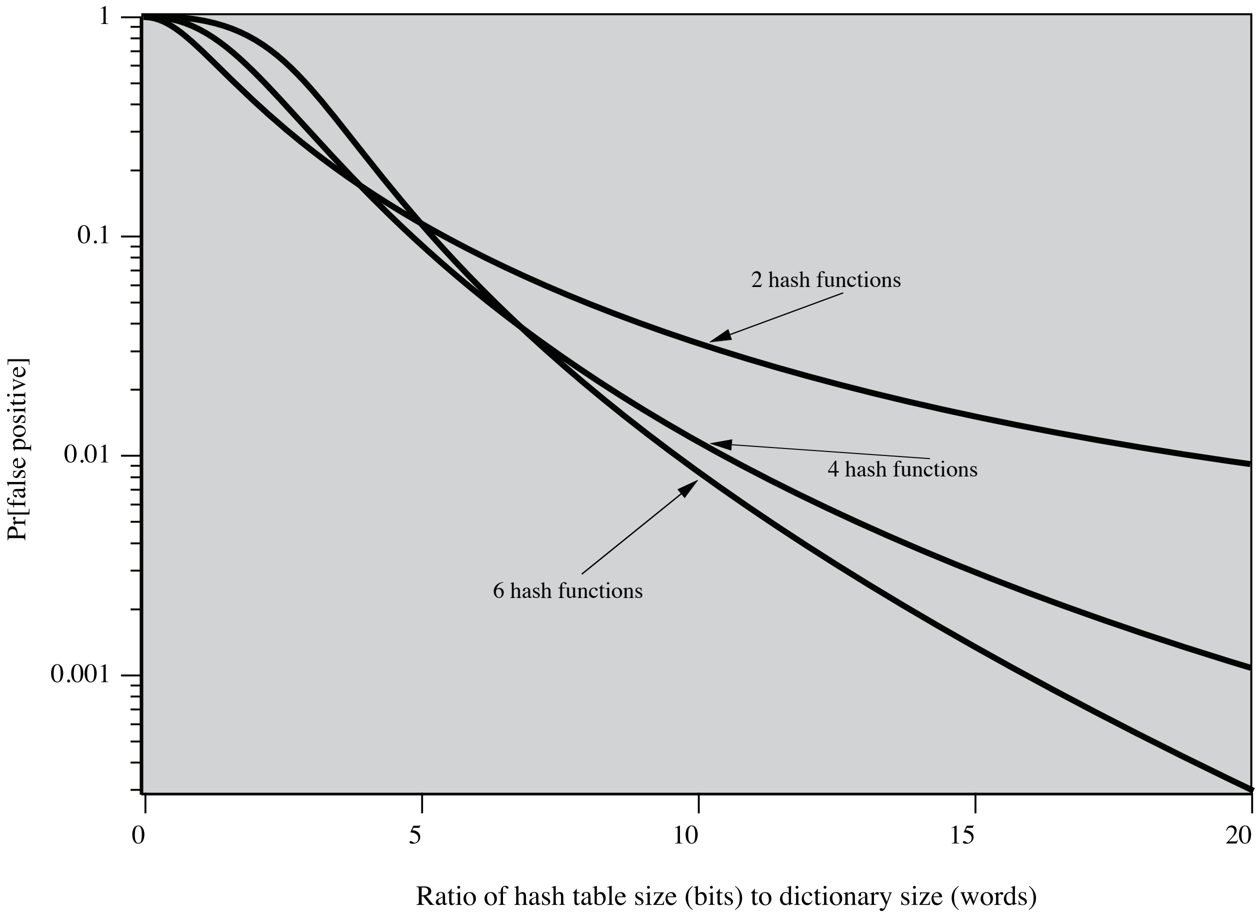
Proactive password checking

- User selects his own password, but also
- The system checks to see if it is allowable
- If not, it rejects it

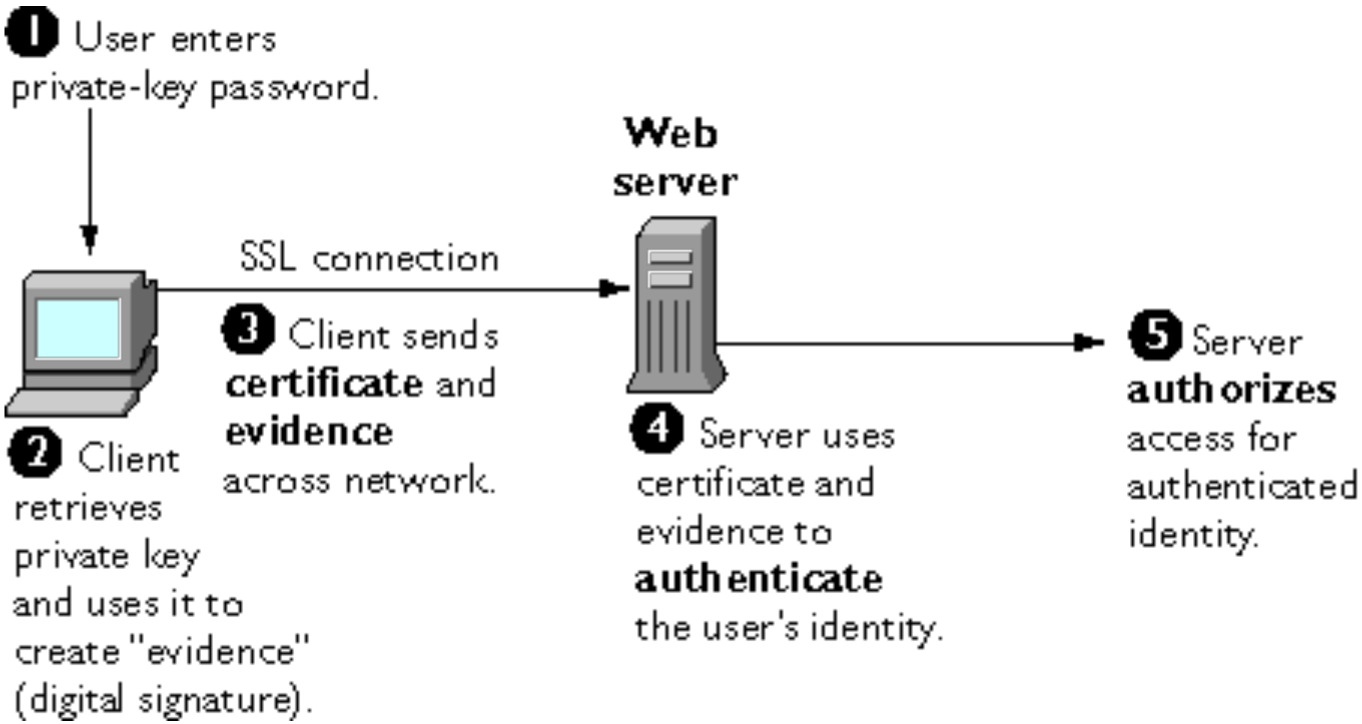
Χρήση συνθηματικών: έλεγχος

- rule enforcement plus user advice, e.g.
  - 8+ chars, upper/lower/numeric/punctuation
  - may not suffice
- password cracker
  - time and space issues
- Markov Model
  - generates guessable passwords
  - hence reject any password it might generate
- Bloom Filter
  - use to build table based on dictionary using hashes
  - check desired password against this table

Απόδοση φίλτρου Bloom



Πιστοποίηση: με πιστοποιητικό



Σύστημα Kerberos

Kerberos

- Key distribution and user authentication service
  - It was developed at MIT
  - Relies exclusively on symmetric encryption
- Has a centralized authentication server to authenticate users to servers and servers to users

Two versions are in use

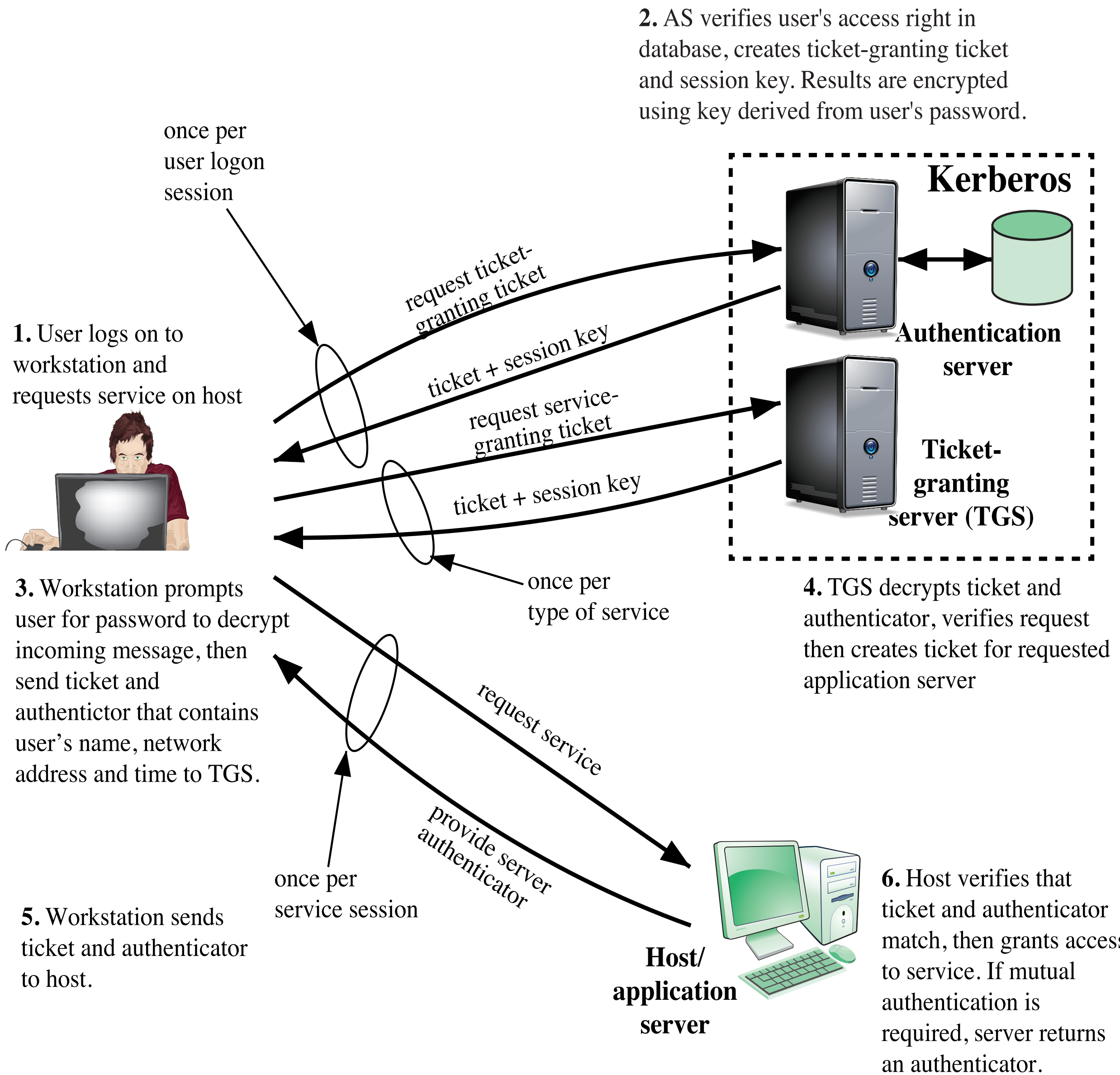
- V4 implementations exist though this version is being phased out
- V5 corrects some of the security deficiencies of V4 and has been issued as a proposed Internet Standard (RFC 4120)

Kerberos version 4

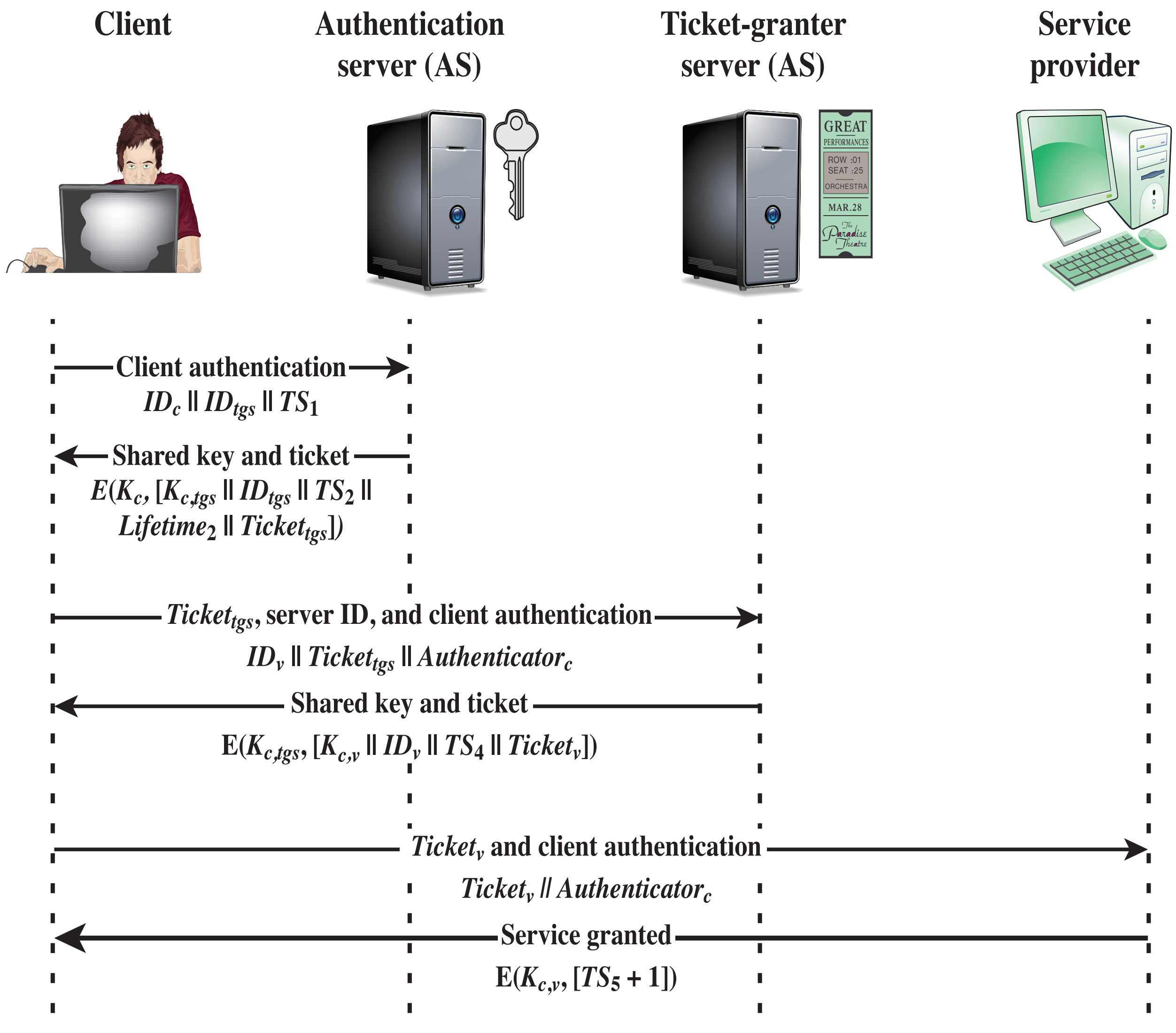
- A basic third-party authentication scheme
- Authentication Server (AS)
  - Users initially negotiate with AS to identify self
  - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)
- Ticket Granting Server (TGS)

  Users subsequently request access to other services from TGS on basis of users TGT
- Complex protocol using DES

Kerberos: Επισκόπηση



Kerberos 4: Μηνύματα



Kerberos 4: Μηνύματα

(1) C → AS $ID_c \parallel ID_{tgs} \parallel TS_1$

(2) AS → C $E(K_c, [K_{c,tgs} \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgs}])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \parallel ID_C \parallel AD_C \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2])$$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) C → TGS $ID_v \parallel Ticket_{tgs} \parallel Authenticator_c$

(4) TGS → C $E(K_{c,tgs}, [K_{c,v} \parallel ID_v \parallel TS_4 \parallel Ticket_v])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \parallel ID_C \parallel AD_C \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2])$$

$$Ticket_v = E(K_v, [K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \parallel AD_C \parallel TS_3])$$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) C → V $Ticket_v \parallel Authenticator_c$

(6) V → C $E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)

$$Ticket_v = E(K_v, [K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \parallel AD_C \parallel TS_5])$$

(c) Client/Server Authentication Exchange to obtain service

Kerberos 4: Φιλοσοφία

Message (1)	Client requests ticket-granting ticket.
ID_C	Tells AS identity of user from this client.
ID_{tgs}	Tells AS that user requests access to TGS.
TS_1	Allows AS to verify that client's clock is synchronized with that of AS.
Message (2)	AS returns ticket-granting ticket.
K_c	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2).
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key.
ID_{tgs}	Confirms that this ticket is for the TGS.
TS_2	Informs client of time this ticket was issued.
$Lifetime_2$	Informs client of the lifetime of this ticket.
$Ticket_{tgs}$	Ticket to be used by client to access TGS.

(a) Authentication Service Exchange

Kerberos 4: Φιλοσοφία

Message (3)	Client requests service-granting ticket.
ID_V	Tells TGS that user requests access to server V.
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket .
Message (4)	TGS returns service-granting ticket.
$K_{c,tgs}$	Key shared only by C and TGS protects contents of message (4).
$K_{c,v}$	Copy of session key accessible to client created by TGS to permit secure exchange between client and server without requiring them to share a permanent key.
ID_V	Confirms that this ticket is for server V.
TS_4	Informs client of time this ticket was issued.
$Ticket_V$	Ticket to be used by client to access server V.
$Ticket_{tgs}$	Reusable so that user does not have to reenter password.
K_{tgs}	Ticket is encrypted with key known only to AS and TGS, to prevent Tampering.
$K_{c,tgs}$	Copy of session key accessible to TGS used to decrypt authenticator, thereby authenticating ticket.
ID_C	Indicates the rightful owner of this ticket.
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket.
ID_{tgs}	Assures server that it has decrypted ticket properly.
TS_2	Informs TGS of time this ticket was issued.
$Lifetime_2$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures TGS that the ticket presenter is the same as the client for whom the ticket was issued has very short lifetime to prevent replay.
$K_{c,tgs}$	Authenticator is encrypted with key known only to client and TGS, to prevent tampering.
ID_C	Must match ID in ticket to authenticate ticket.
AD_C	Must match address in ticket to authenticate ticket.
TS_3	Informs TGS of time this authenticator was generated.

(b) Ticket-Granting Service Exchange

Kerberos 4: Φιλοσοφία

Message (5)	Client requests service.
$Ticket_V$	Assures server that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.
Message (6)	Optional authentication of server to client.
$K_{c,v}$	Assures C that this message is from V.
$TS_5 + 1$	Assures C that this is not a replay of an old reply.
$Ticket_v$	Reusable so that client does not need to request a new ticket from TGS for each access to the same server.
K_v	Ticket is encrypted with key known only to TGS and server, to prevent Tampering.
$K_{c,v}$	Copy of session key accessible to client; used to decrypt authenticator, thereby authenticating ticket.
ID_C	Indicates the rightful owner of this ticket.
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket.
ID_V	Assures server that it has decrypted ticket properly.
TS_4	Informs server of time this ticket was issued.
$Lifetime_4$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures server that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay.
$K_{c,v}$	Authenticator is encrypted with key known only to client and server, to prevent tampering.
ID_C	Must match ID in ticket to authenticate ticket.
AD_c	Must match address in ticket to authenticate ticket.
TS_5	Informs server of time this authenticator was generated.

(c) Client/Server Authentication Exchange

Kerberos realms

- A set of managed nodes that share the same Kerberos DB
- Kerberos DB is on a master computer (in a secure room)
- All changes to the DB must be made on the master computer
- Read-only copy of Kerberos DB also on other computers
- Accessing Kerberos DB needs the Kerberos master password

Kerberos system consists of

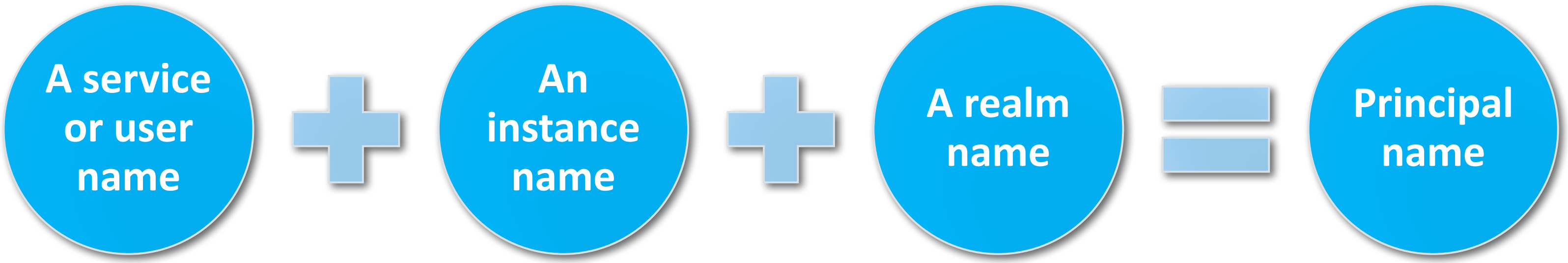
A Kerberos server

A number of clients

A number of application servers

Kerberos principal

- A service or user that is known to the Kerberos system
- Each Kerberos principal is identified by its principal name
- Principal names consist of three parts:



Διαφορές μεταξύ V4 & V5

ENVIRONMENTAL CONS

- Encryption system dependence
- Internet protocol dependence
- Message byte ordering
- Ticket lifetime
- Authentication forwarding
- Interrealm authentication

TECHNICAL DEFICIENCIES

- Double encryption
- PCBC encryption
- Session keys
- Password attacks

Kerberos 5: Μηνύματα

(1) **C → AS** *Options* || *IDc* || *Realmc* || *IDtgs* || *Times* || *Nonce*1

(2) **AS → C** *Realmc* || *IDC* || *Tickettgs* || E(*Kc*, [*Kc,tgs* || *Times* || *Nonce*1 || *Realmtgs* || *IDtgs*])

Tickettgs = E(*Ktgs*, [*Flags* || *Kc,tgs* || *Realmc* || *IDC* || *ADC* || *Times*])

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) **C → TGS** *Options* || *IDv* || *Times* || || *Nonce*2 || *Tickettgs* || *Authenticatorc*

(4) **TGS → C** *Realmc* || *IDC* || *Ticketv* || E(*Kc,tgs*, [*Kc,v* || *Times* || *Nonce*2 || *Realmv* || *IDv*])

Tickettgs = E(*Ktgs*, [*Flags* || *Kc,tgs* || *Realmc* || *IDC* || *ADC* || *Times*])

Ticketv = E(*Kv*, [*Flags* || *Kc,v* || *Realmc* || *IDC* || ADC || *Times*])

Authenticatorc = E(*Kc,tgs*, [*IDC* || *Realmc* || *TS*1])

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) **C → V** *Options* || $Ticket_v$ || $Authenticator_c$

(6) **V → C** $E_{K_{c,v}}$ [TS_2 || *Subkey* || *Seq#*]

Ticket*v* = E(*Kv*, [Flags || *Kc,v* || *Realmc* || *IDC* || *ADC* || *Times*])

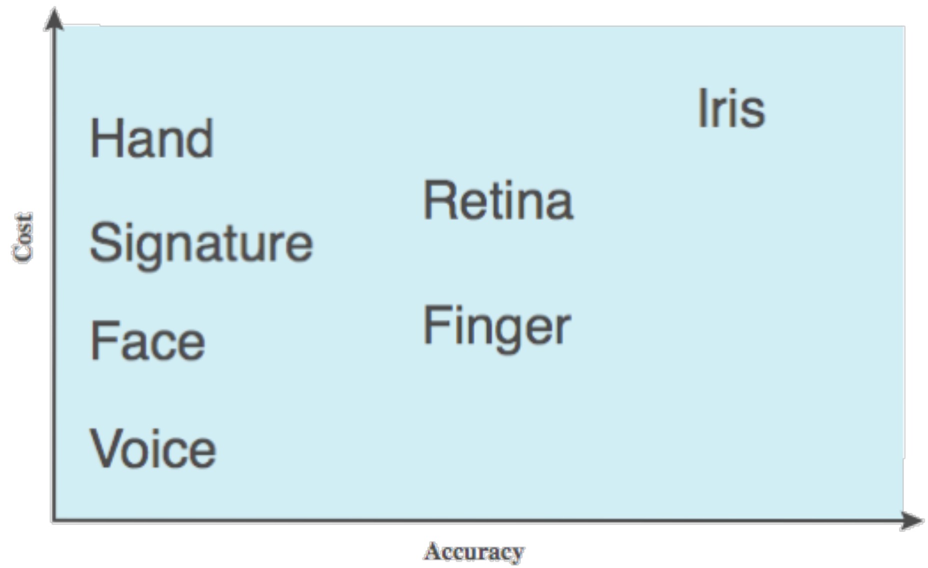
Authenticatorc = E(*Kc,v*, [*IDC* || *Realmc* || *TS*2 || *Subkey* || *Seq#*])

(c) Client/Server Authentication Exchange to obtain service

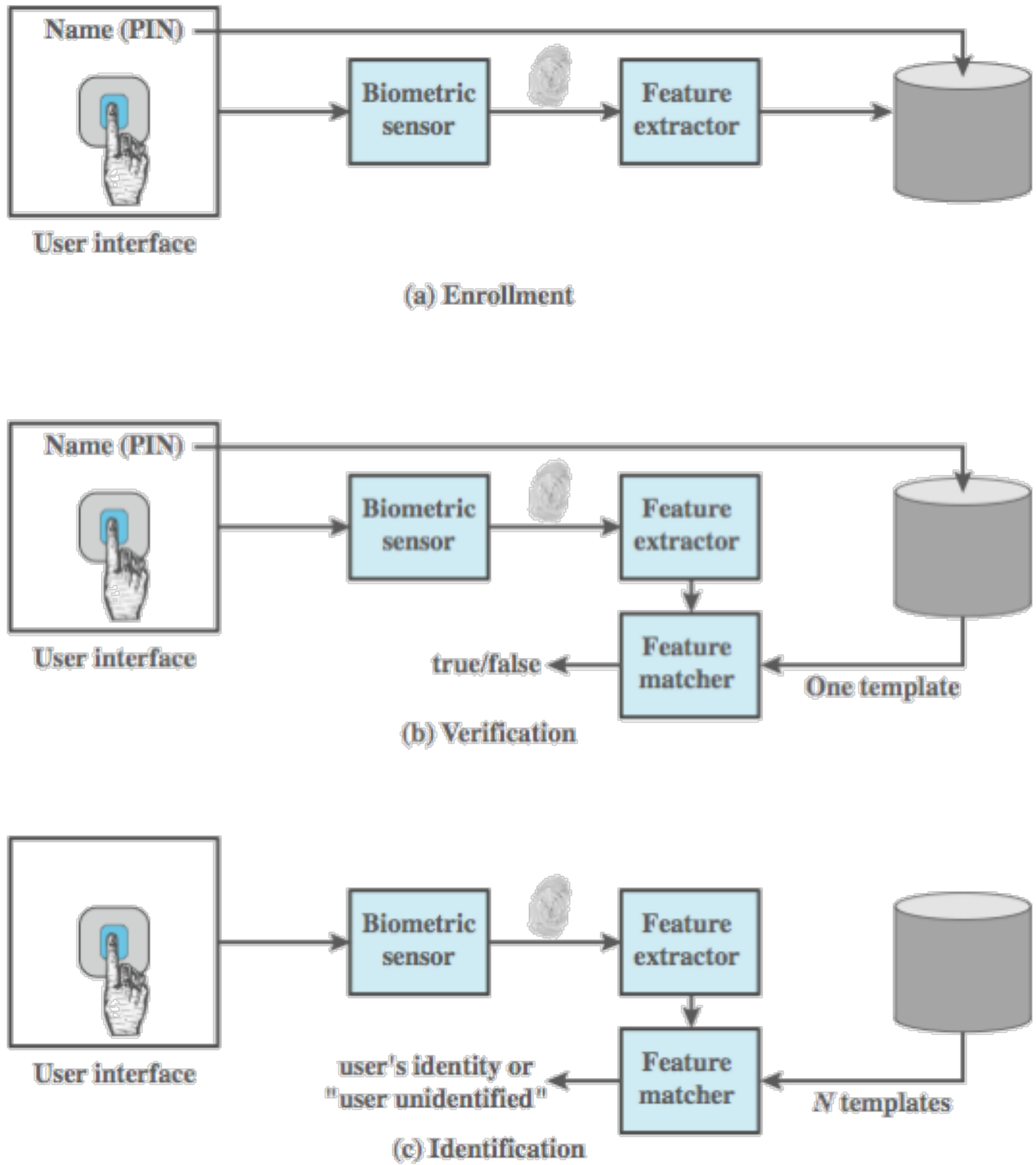
Βιομετρικά συστήματα

Πιστοποίηση: βιομετρική

- Authenticate user based on one of their physical characteristics

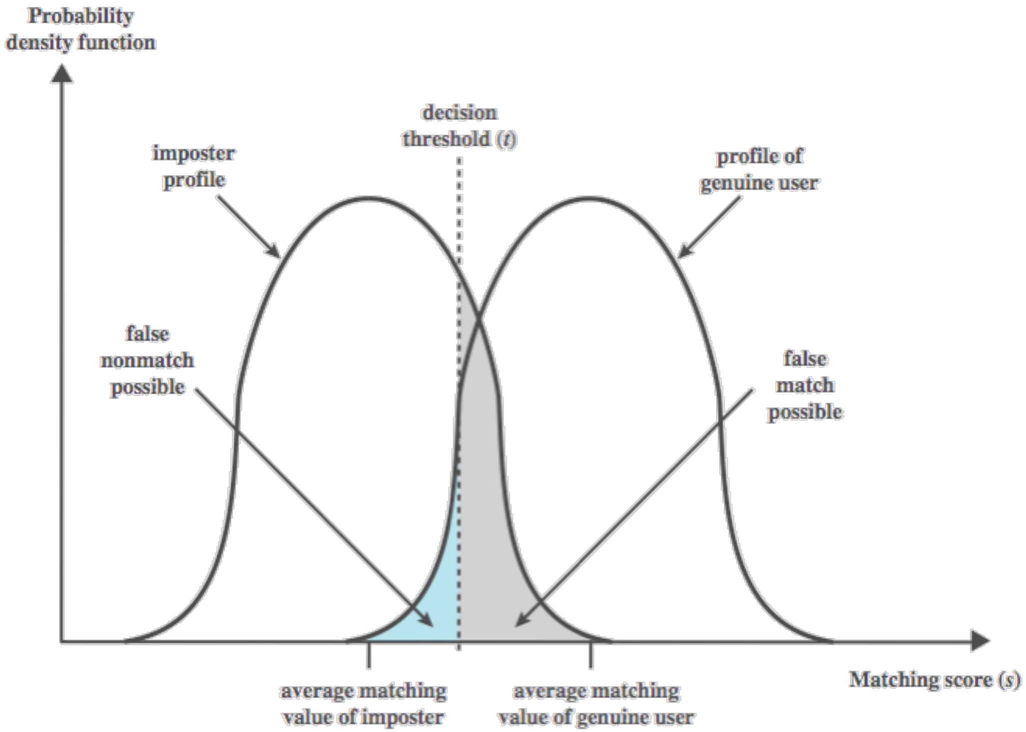


Λειτουργία βιομετρικού συστήματος



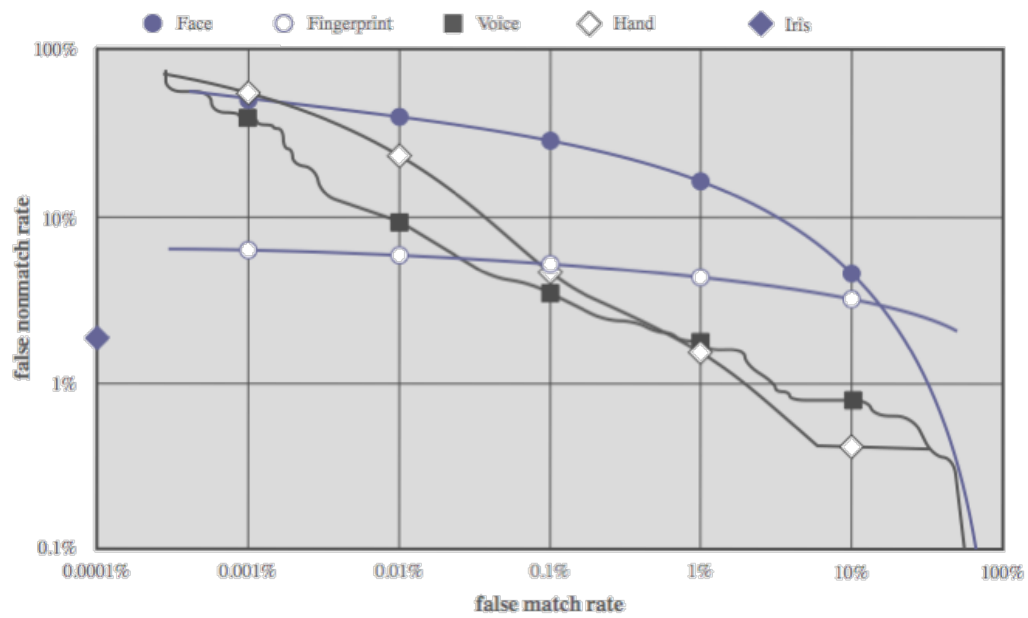
Ακρίβεια βιομετρικών μεθόδων

- never get identical templates
- problems of false match / false non-match



Ακρίβεια βιομετρικών μεθόδων

- can plot characteristic curve
- pick threshold balancing error rates



Διαλογικά σχήματα

Ισχυρή πιστοποίηση

- Οντότητες διαλογικού πρωτοκόλλου
  - Χρήστης Α (διεκδικητής - claimant)
  - Χρήστης Β (επαληθευτής - verifier)
- Ο χρήστης Α αποδεικνύει την ταυτότητά του στον Β μέσω κατοχής μυστικής γνώσης (χωρίς να την αποκαλύπτει)
  - Επιτυγχάνεται με την απάντηση σε μία «ερώτηση-πρόκληση» του Β
  - Η απάντηση εξαρτάται, από την ερώτηση και τη μυστική γνώση
- Πρωτόκολλα αυτά ονομάζονται μηδενικής γνώσης (ZK)

Ισχυρή πιστοποίηση

- Ένα πρωτόκολλο πιστοποίησης ταυτότητας πρέπει να έχει τα ακόλουθα χαρακτηριστικά:
  - Ο B να μη μπορεί να αποσπάσει πληροφορία που χρησιμοποίησε ο A για να πιστοποιηθεί, έτσι ώστε να προσποιηθεί σε κάποιον τρίτο C ότι είναι ο A (transferability)
  - Να μην υπάρχει η δυνατότητα σε κάποιον τρίτο C ξεγελάσει τον B ότι είναι ο A (impersonation)
  - Τα παραπάνω να ισχύουν για όσο μεγάλο πλήθος επαναλήψεων του πρωτοκόλλου μεταξύ των A, B, ακόμα κι αν ένας εισβολέας C έχει παρακολουθήσει όλες τις συνομιλίες των A, B

Ισχυρή πιστοποίηση

- Συνήθως τα πρωτόκολλα αυτά έχουν 3 μηνύματα
- Δέσμευση: ο A παράγει τυχαία μία βεβαίωση (witness)
  - Την οποία στέλνει στον B
- Πρόκληση: ο B κάνει μία ερώτηση-πρόκληση (challenge)
  - Μόνο κάποιος που κατέχει το μυστικό κλειδί του A πρέπει να μπορεί να απαντήσει σωστά σε όλες τις προκλήσεις
- Απόκριση: ο A στέλνει στον B την απάντηση (response)
  - Ο B πρέπει να επιβεβαιώσει την ορθότητα της απάντησης

Σχήμα Fiat–Shamir

- Έστω $N = pq$, όπου p,q είναι πολύ μεγάλοι πρώτοι αριθμοί
  - Το N τουλάχιστον 1024 bits
  - Το N παράγεται από έναν «διαιτητή», αποδεκτό από όλους
- Ο Α επιλέγει τυχαία $s < N$ και υπολογίζει υ τέτοιο ώστε:
  - $\upsilon = s^2 \pmod{N}$
- Κλειδιά
  - Δημόσιο κλειδί : (u, N)
  - Ιδιωτικό κλειδί : (s, p, q)

Σχήμα Fiat–Shamir

A 

B 

$x = r^2 \bmod N \xrightarrow{\quad x \quad}$

$\xleftarrow{\quad e \quad}$ $e = 0$ ή 1

$y = rs^e \bmod N \xrightarrow{\quad y \quad} y^2 = xu^e \bmod N$

$y^2 = r^2 s^{2e} \bmod N = (r^2 \bmod N)(s^2 \bmod N)^e \bmod N = xu^e \bmod N$

Σχήμα Fiat–Shamir

- Η παραπάνω διαδικασία επαναλαμβάνεται πολλές φορές
  - Επιλέγεται τυχαία βεβαίωση r και τυχαία πρόκληση e
- Ένας επιτιθέμενος που θέλει να προσποιηθεί ότι είναι ο A
  - Μπορεί να επιλέξει τυχαίο r
  - Να στείλει $x = r^2 / \upsilon$
  - Σε κάθε πρόκληση $e = 1$ να απαντά $y = r$ (κάτι που ο B θα το ανιχνεύει ως σωστή απάντηση)
  - Δεν θα μπορεί να απαντήσει σωστά για $e = 0$

Σχήμα Fiat–Shamir

- Μετά από *k* γύρους, η πιθανότητα λάθους (ο εισβολέας να εξαπατήσει τον Β) ισούται με $(1/2)^k$
  - Αν θεωρήσουμε ότι μπορεί να απαντήσει σωστά σε μία από τις δύο ερωτήσεις
- Δεν πρέπει να χρησιμοποιείται το ίδιο r
  - Ένας εισβολέας που παρακολουθεί τη συνομιλία μπορεί να πραγματοποιήσει ένα replay attack
  - Μαθαίνει τις απαντήσεις του Α για τις εκάστοτε ερωτήσεις του Β και να τις επαναλάβει

Σχήμα Fiat–Shamir

- Δυσκολία στην παραγοντοποίηση:
  - ένας αλγόριθμος που «σπάει» τον Fiat-Shamir είναι ισοδύναμος με έναν αλγόριθμο που παραγοντοποιεί τον N
- Τυχαιότητα:
  - Του r (όσον αφορά τη μηδενική γνώση)
  - Της ερώτησης (αυτό εμποδίζει τον Α στο να εξαπατήσει)

Προτεινόμενη βιβλιογραφία

- W. Stallings
  Cryptography and Network Security: Principles & Practice
  7th Ed., Prentice Hall, 2017
- W. Stallings and L. Brown
  Computer Security: Principles & Practice
  3rd Ed., Prentice Hall, 2015
- M. Bishop
  Computer Security: Art and Science
  Addison Wesley, 2003