Etablissement, gestion et certification des clés

A. Bonnecaze

Gestion des clés

Ensemble de processus comprenant

- La génération de clés
- La distribution de clés (secrète ou privée)
- Le stockage et l'archivage
- Remplacement/destruction de clés

Génération de clefs

- Sources physiques : pas pratique
- Générateur de bit pseudo-aléatoires (PRBG)
  - LFSR
  - À base de crypto asymétrique
  - À base de fonction de hachage
  - À base de crypto symétrique

PRBG

- Basé sur une fonction de hachage
  - germe s, $z_n = h(s+n \bmod p)$ $n = 0,1,2\ldots$
- Basé sur AES
  - $K' = AES_K(\text{temps universel})$
  - $R_n = AES_{K'}(n)$ $n = 0,1,2,\ldots$
  - Sécurité basée sur la clef secrète K
- Ce sont des générateurs utilisés dans la pratique

Architecture des clefs

- Hiérarchie des clefs
  - **Master keys** : elles ne sont pas protégées cryptographiquement. Distribuées manuellement et protégées physiquement (isolation électronique, porte blindée,…)
  - **Key-encrypting keys** : symétriques ou publiques, elles sont utilisées pour le transport ou le stockage d'autres clefs
  - **Data keys** : utilisées par l'utilisateur pour chiffrer et authentifier des données.

Objectifs cryptographiques

- Protocoles d'authentification
  - Fournir à une partie une assurance sur l'identité d'une autre partie (avec laquelle elle communique)
- Protocole d'établissement de clef
  - Établir un secret partagé entre les parties
- Protocole d'établissement de clef authentifiée
  - Établir un secret partagé entre parties dont les identités ont été ou peuvent être vérifiées
  - On dit aussi *distribution de clef authentifiée*

Distribution de clefs

- Distribution de clefs secrètes
  - La distribution doit assurer la confidentialité des clefs
  - Le service de distribution (s'il existe) doit être on-line
- Distribution de clefs publiques
  - La distribution doit assurer l'intégrité des clefs
  - Le service de distribution peut être off-line

Utilisation de clefs de session

- Une clef de session est éphémère. Par exemple le temps d'une connexion. Pourquoi?
  - Pour limiter les chiffrés qui peuvent faire l'objet d'une attaque
  - Pour limiter l'exposition en matière de période de temps et de quantité de données dans le cas de compromission de la clef
  - Eviter le stockage à long terme de clef : on utilise une clef nouvelle pour chaque besoin
  - Créer une indépendance entre sessions/applications

Distribution de clefs

- Si Alice et Bob veulent communiquer, comment vont-ils se mettre d'accord sur une clef secrète de session ?
- hypothèse : canal non sûr
- Bien sûr

  Alice et Bob pourraient se rencontrer en personne

  Alice pourrait envoyer la clef à Bob par courier sûr

Ces solutions ne sont pas acceptables dans le monde Internet

Distribution de clef sans serveur

- A et B partage une long-term clef k qui est utilisée pour établir de nouvelles clefs de session
- A choisit une clef et l'envoie à B en une passe :

$$A \rightarrow B : E_k(rA)$$

- Option : time-stamp, identité de B

Avec challenge-response

- On veut s'assurer que la clef est nouvelle (pas de rejeu) mais sans utiliser de time-stamp

  $A \leftarrow B : nB$

  $A \rightarrow B : E_k(rA, nB, B^*)$
- Le clef de session est rA

Avec challenge-response

- On veut que la clef soit fonction de A et B
- W = f(rA, rB)

  A ← B : nB

  A → B : E_k(rA, nA, nB, B*)

  A ← B : E_k(rB, nB, nA, A*)

nA et nB sont des nonces utilisées pour garantir la « fraîcheur » de la clef

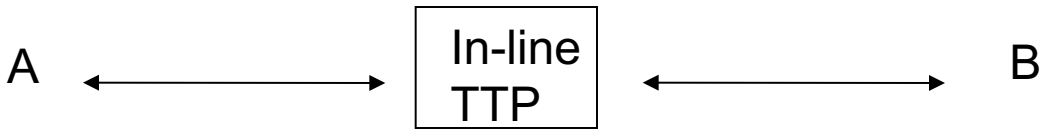
Sans clef partagée par A et B

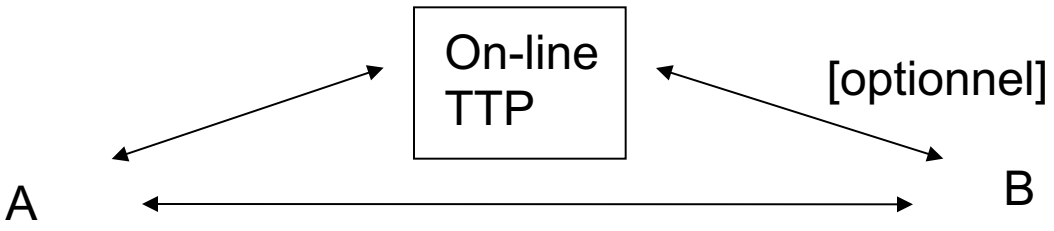
- A choisit k comme clef de session
- p est premier (publique) p>k
- A (resp B) choisit aléatoirement a (resp b)
- A → B : $k^a \bmod p$ (1)
- A ← B : $(k^a)^b \bmod p$ (2)
- A → B : $(k^{ab})^{a^{-1}} \bmod p$ (3)
- B calcule $b^{-1} \bmod p-1$ et obtient k

Distribution de clef avec serveur

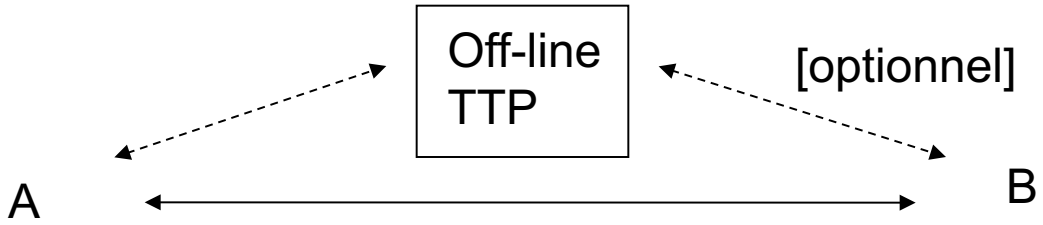
- Un centre de clefs de confiance (TTP) On-line permet d'établir une clef de session
  - Chaque utilisateur à une "long-term private/secret key" qu'il partage avec le centre
- Remarque : pour la distribution de clef publique
  - Alice et Bob n'ont pas besoin de TTP
  - Un centre de clefs peut être utilisé pour stocker et distribuer les clefs publiques mais les utilisateurs ne partagent pas de clef privée avec le centre

Interaction avec le TTP





TTP peut communiquer avec A et B pendant le déroulement protocole



TTP ne participe pas
En temps réel

Rôles des TTP

- Autorité de certification (CA)
  - Authenticité des clefs
- Serveur de nom
  - Gestion (unicité) des noms
- Autorité de registre
  - Responsable des autorisations des utilisateurs
- Générateur, établissement, gestion de clefs
  - Publiques/privées, secrètes, passwords,…
- BD de certificats
  - BD accessible en read access

Distribution de clefs avec TTP

- Alice et Bob partagent chacun une clef secrète avec le serveur de clefs (TTP)
  - KB = clef de Bob ; KA = clef d'Alice
  - Le seul but de ces clefs est de communiquer avec le TTP
  - Ces clefs sont gardées d'une manière sûre par le TTP
  - Si KA n'est pas compromise, elle permet d'authentifier Alice au TTP et vice-versa

Protocole simple (crypto sym)

- Alice → TTP: A, B
  - TTP génère aléatoirement la clef de session K
- TTP → Alice: $E_{KA}(K)$, $E_{KB}(K)$
  - Alice déchiffre sa partie pour obtenir K
- Alice → Bob: $E_{KB}(K)$
  - Bob déchiffre sa partie pour obtenir K
- Alice → Bob: $E_K(M)$
  - Alice peut envoyer un message chiffré...

Replay Attacks

- Iwan intercepte les communications chiffrées
- Iwan renvoie les messages à Bob

  Iwan → Bob: $E_{KB}(K)$
- Bob déchiffre sa partie pour obtenir K

  Iwan → Bob: $E_{K}(M)$
- Iwan peut aussi prétendre être Alice
- Quelles peuvent être les conséquences ?

Protocole de Needham-Shroeder

(crypto sym)

- Alice → TTP: A, B, nA
- TTP → Alice: $E_{KA}(K, nA, B, E_{KB}(K, A))$
- Alice → Bob: $E_{KB}(K, A)$
- Bob → Alice: $E_K(nB)$
- Alice → Bob: $E_K(nB+1)$
- Alice → Bob: $E_K(M)$ etc.

- nA, nB sont des nombres aléatoires (nonces)

Autre Attaque

- Supposons que Iwan a cassé K et qu'il a gardé en mémoire

  Alice → Bob: $E_{KB}(K, A)$

- Iwan peut alors se faire passer pour Alice et envoyer des messages à Bob

Denning/Sacco

- Utiliser un timestamp T pour éviter le rejeu
- Alice → TTP: A, B
- TTP → Alice: $E_{KA}(K, B, T, E_{KB}(K, A, T))$
- Alice → Bob: $E_{KB}(K, A, T)$
- Alice → Bob: $E_K(M)$ etc.

- Nouveaux problèmes ?

TTPs comme serveurs d'Authentification

- Les clefs KA et KB authentifient Alice & Bob au TTP et vice versa, si elles ne sont pas compromises
- Si le TTP reçoit un message chiffré par KA, il ne peut provenir que d'Alice
- Si Alice reçoit un message chiffré par KA, il ne peut provenir que de TTP
- TTPs peuvent être utilisés comme serveurs d'authentification

Needham-Schroeder (crypto asym)

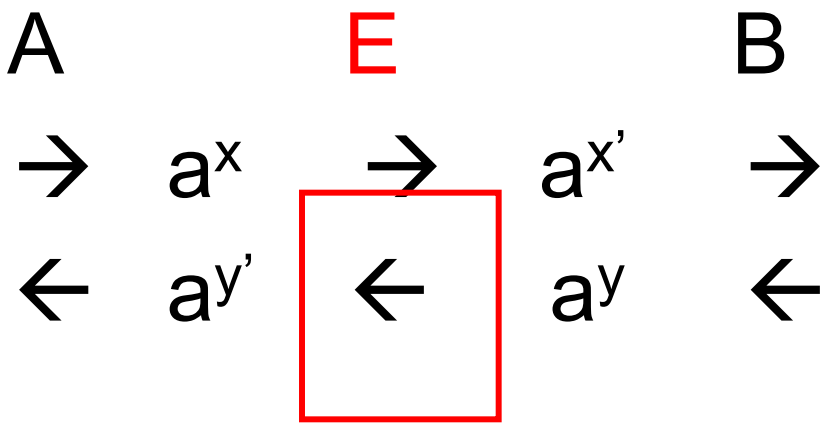
- Buts :
  - Authentification mutuelle
  - Echanger chacun un secret (k_1 et k_2)
  - Obtenir une clef de session $f(k_1,k_2)$
- $A \rightarrow B : E_{Bp}(k_1, A)$ (1)
- $A \leftarrow B : E_{Ap}(k_1, k_2)$ (2)
- $A \rightarrow B : E_{Bp}(k_2)$ (3)
- Comment éviter le chiffrement en (3)?

MTI (variante de DH)

- A et B veulent s'échanger une clef
- Clefs privées : A => a, B => b
- Clefs publiques : za = α^a, zb = α^b
- A→B : α^x mod p
- A←B : α^y mod p
- A calcule k = $(\alpha^y)^a zb^x$ mod p
- B calcule k = $(\alpha^x)^b za^y$ mod p

Sécurité des établissements de clefs (attaques)

- *Man in the middle*
- Sur le protocole de Diffie-Hellman



Utilisation des clefs

- Attention lors d'un chiffrement par logiciel si on utilise un SE multitache
- Lorsque le SE a une tache urgente à faire, il sauvegarde toutes les infos sur le disque
- La clef est donc sur le disque jusqu'à ce que l'ordinateur réécrive à cet endroit
- Un adversaire peut examiner le disque et retrouver la clef

Mise à jour des clefs

- A et B ont une clef k et une fonction f à sens unique communes
- A et B peuvent calculer f(k) et obtenir une nouvelle clef
- One time key

Stockage des clefs

- La clef de A peut être stockée
  - dans le cerveau de A
  - Dans une carte magnétique
  - Dans une clef USB
  - À moitié dans une carte et à moitié dans l'ordinateur
  - Chiffrée (par exemple avec AES) sur un disque
  - Dans une base de donnée centrale ou privée

Clef compromise

- Clef secrète
  - Changer la clef
- Clef privée
  - La clef publique ne doit plus être utilisée
  - Diffuser l'info à tous les serveurs du réseau
  - Essayer de savoir quand la clef a été compromise
  - L'utilisation de datation permet de différencier les messages suspects des légitimes
  - Utiliser une clef pour chaque application

Destruction de clefs

- Les vieilles clefs permettent de lire d'anciens messages chiffrés
- Elles doivent être détruites de manière sûre
- Sur papier : brûler ou utiliser une machine à déchiqueter de bonne qualité
- Sur mémoire
  - La puce doit être broyée
  - Réécrire plusieurs fois sur le disque (ou broyer!)
  - Programme d'effacement de disque
  - Effacer le contenu des fichiers temporaires

Gestion de clefs publiques

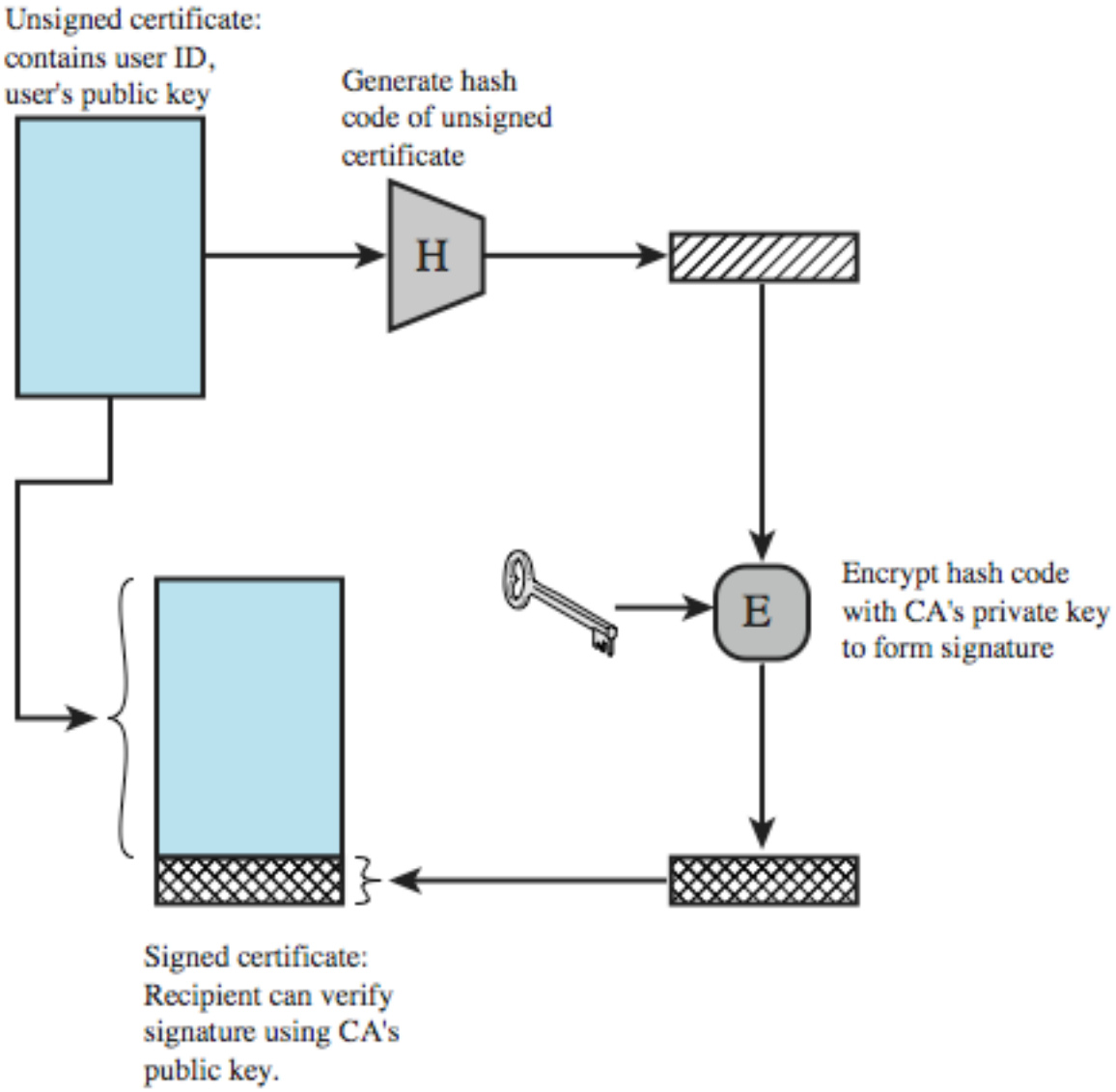
- Pour obtenir la clef publique de Bob, Alice peut
  - L'obtenir auprès de Bob
  - L'obtenir auprès d'une BD centrale
  - L'obtenir à partir de sa propre BD privée
  - L'obtenir avec un certificat à une BD TTP

PKI (Public Key Infrastructure)

fournit des garanties permettant de faire *a priori* confiance à un certificat signé par une **autorité de certification** grâce à un ensemble de **services**

- Enregistrement des utilisateurs
- Génération, renouvellement, révocation de certificats
- Publication de certificats
- Publication des listes de révocation

Certificats



Certificate:
Data:
Version: 3 (0x2)
Serial Number: 1 (0x1)
Signature Algorithm: md5WithRSAEncryption
Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
OU=Certification Services Division,
CN=Thawte Server CA/Email=server-certs@thawte.com
Validity Not Before: Aug 1 00:00:00 1996 GMT
Not After : Dec 31 23:59:59 2020 GMT
Subject: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
OU=Certification Services Division,
CN=Thawte Server CA/Email=server-certs@thawte.com
Subject Public Key Info:
Public Key Algorithm: rsaEncryption RSA Public Key: (1024 bit)
Modulus (1024 bit): 00:d3:a4:50:6e:c8:ff:56:6b:e6:cf:5d:b6:ea:0c: 68:75:47:a2:aa:c2:da:84:25:fc:a8:f4:47:51:da:
85:b5:20:74:94:86:1e:0f:75:c9:e9:08:61:f5:06: 6d:30:6e:15:19:02:e9:52:c0:62:db:4d:99:9e:e2: 6a:0c:44:38:cd:fe:be:e3:64:09:70:c5:fe:b1:6b:
29:b6:2f:49:c8:3b:d4:27:04:25:10:97:2f:e7:90:d:c0:28:42:99:d7:4c:43:de:c3:f5:21:6d:54:9f:d:c3:58:e1:c0:e4:d9:5b:b0:b8:dc:b4:7b:df:36:3a:c2:b5:66:22:1
2:d6:87:0d
Exponent: 65537 (0x10001)
X509v3 extensions: X509v3 Basic Constraints: critical
CA:TRUE
Signature Algorithm: md5WithRSAEncryption
07:fa:4c:69:5c:fb:95:cc:46:ee:85:83:4d:21:30:8e:ca:d9: a8:6f:49:1a:e6:da:51:e3:60:70:6c:84:61:11:a1:1a:c8:48: 3e:59:43:7d:4f:95:3d:a1:8b:b7:0b:62:98:7a:75:8a:dd:88:
4e:4e:9e:40:db:a8:cc:32:74:b9:6f:0d:c6:e3:b3:44:0b:d9: 8a:6f:9a:29:9b:99:18:28:3b:d1:e3:40:28:9a:5a:3c:d5:b5: e7:20:1b:8b:ca:a4:ab:8d:e9:51:d9:e2:4c:2c:59:a9:da:b9:
b2:75:1b:f6:42:f2:ef:c7:f2:18:f9:89:bc:a3:ff:8a:23:2e: 70:47

Gestion distribuée des clefs (PGP)

- Suppose qu'il n'existe pas d'AC en laquelle Alice et Bob ont confiance
  - Utiliser des parrains qui signent les clefs publiques de leurs amis : Bob présente sa clef publique à Alice avec les signatures de Bernard et Christine
  - Si Alice connaît Bernard ou Christine, elle acceptera la clef comme valide

Sécurité des SI

Authentification de l'utilisateur

Authentification de l'utilisateur

- Block de sécurité fondamental
  - contrôle d'accès & user accountability
- RFC 2828 : deux étapes :
  - identification - specifie l'identité
  - vérification – relie l'entité (la personne) et l'identité
- distinct de message authentication

Types d'authentification

- Il en existe quatre
- Basé sur quelque chose que l'individu
  - connait : password, PIN
  - possède : key, token, smartcard
  - est (static biometrics) : fingerprint, retine
  - fait (dynamic biometrics) : voix, signes
- Peut être utilisé seul ou combiné
- Tous peuvent fournir l'authentification
- Tous ont leurs défauts

Password Authentication

- Largement utilisé comme méthode d'authentification
  - L'utilisateur donne name/login et password
  - Le système compare le password avec celui qui devrait être utilisé pour le login spécifié
- Authentifie l'ID qui détermine
  - Si l'utilisateur est autorisé à accéder au système
  - Les droits de l'utilisateur

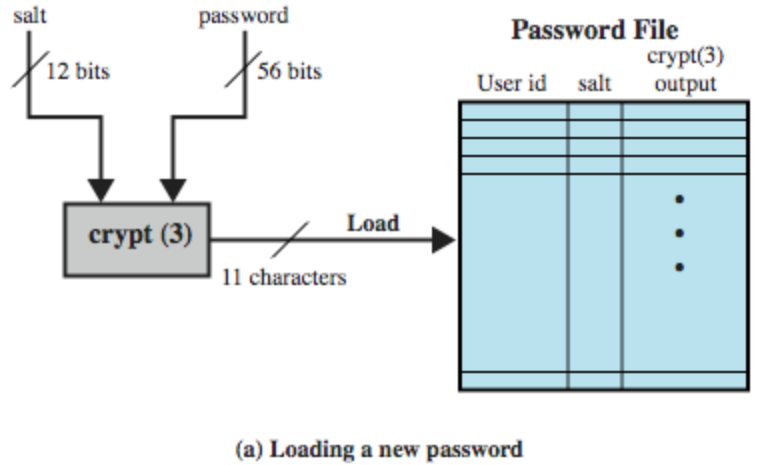
Vulnérabilité du Password

- offline dictionary attack
- specific account attack
- popular password attack
- password guessing against single user
- workstation hijacking
- exploiting user mistakes
- exploiting multiple password use
- electronic monitoring

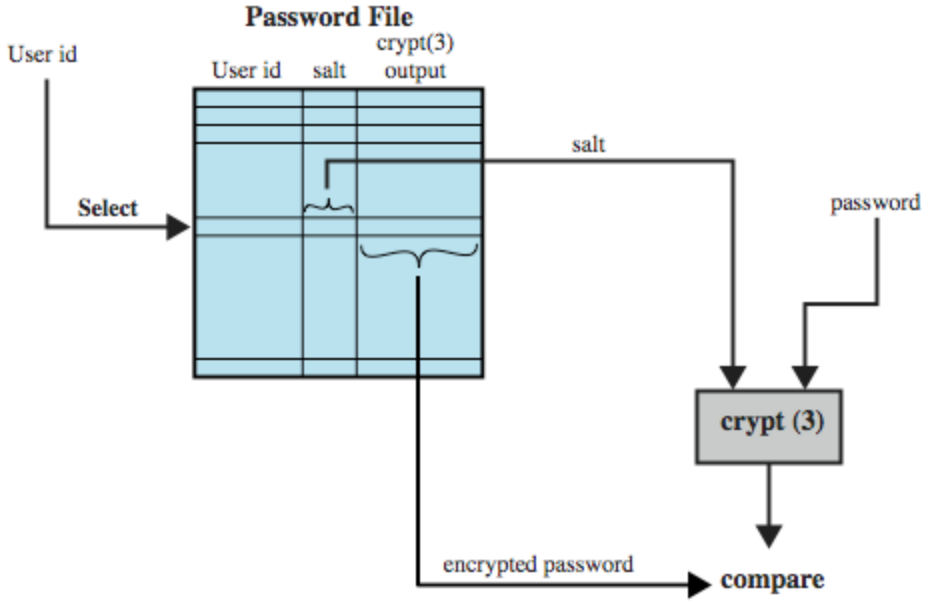
Solutions

- Interdire l'accès non autorisé au fichier password
- Mesures des détections d'intrusions
- Mécanisme de vérouillage après essais infructueux
- Reglement pour choisir des passwords solides
- workstation logout automatique
- Chiffrement des liens de communication

Utilisation de passwords hachés



(a) Loading a new password



(b) Verifying a password

UNIX Implémentation

- Schéma original
  - password de 8 caractères pour clé de 56bits
  - Sel de 12bits pour modifier chiffrement DES dans une fonction de hachage à sens unique
- Maintenant considéré comme non sûr
  - e.g. supercomputer, 50 million tests, 80 min
- Mais encore utilisé pour des pbs de compatibilité

Améliorations

- Il existe des variantes plus sûres
- Certains systèmes utilisent MD5
  - Avec sel de 48bits
  - Taille de password illimitée
  - Produit un haché de 128bits
- OpenBSD utilise Blowfish pour construire une fonction de hachage appelée Bcrypt
  - Sel de 128bits produit un haché de 192bits

Password Cracking

- Attaques par dictionnaire
  - essayer chacun des mots du dico avec ses variantes
  - Il faut posséder le fichier des passwords
- rainbow table attacks
  - Tables précalculées des hachés avec tous les sels
  - Beaucoup de hachés...
  - Ex : 1.4GB permet de cracker 99.9% des passwords alphanumeriques de Windows en 13.8 secs
  - Non utilisable avec des tailles de sel grands

Choix de Passwords

- Passwords trop courts
  - Purdue Uni (1992): 3% sur 7000 utilisateurs ont choisi une taille inférieure à 3 caracteres
  - Le système peut rejeter les passwords trop courts
- Passwords devinables
  - Crackers utilisent les listes de passwords connus
  - Représente en 1990 environ ¼ des passwords
  - Retrouve le password en moins d'une heure
  - Conséquences pour le système entier

Controle d'accès du fichier de passwords

- Accès reglementé au fichier permet d'éviter une attaque offline
  - Accès aux super utilisateurs
  - Utilise souvent un fichier shadow
- D'autres vulnerabilités
  - Exploitation de O/S bug
  - Accident de protection
  - Utilisateurs avec même password sur d'autres systèmes
  - Accès à des backups non protégés
  - Sniff de passwords sur réseaux non protégés

Meilleurs Passwords

- Difficile de deviner le password
- Mais mémorisable par l'utilisateur
- Techniques :
  - Education
  - Computer-generated passwords
  - Reactive password checking
  - Proactive password checking

Proactive Password Checking

- Exigences
  - 8+ chars, upper/lower/numeric/punctuation
  - Pas toujours suffisant
- Dictionnaire de mauvais passwords
  - Gourmand en temps et espace
- Markov Model
  - Génere des passwords “devinables”
  - Rejete tout password construit de cette manière
- Bloom Filter
  - Utilisé pour construire des tables de hachés
  - Rejete les passwords dont les hachés sont ds la table

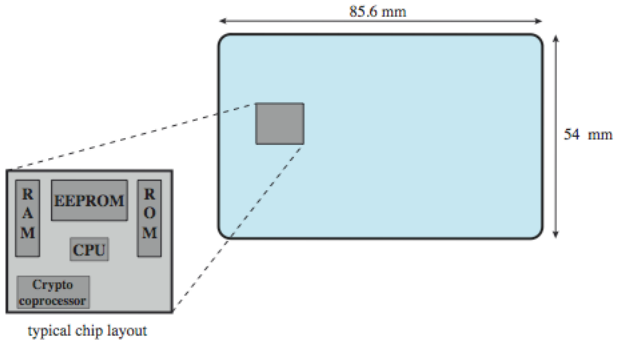
Token Authentication

- Objet que l'utilisateur possède et qui permet son authentification
  - Carte magnétique (carte bancaire)
  - Carte à mémoire (carte de télephone)
  - Smartcard

Carte à mémoire

- Stoque mais ne traite pas les données
- Carte magnetique (bon marché)
- Electronic memory card (plus évolué)
- Utilisée seule pour des accès physiques
- avec password/PIN pour accès PC
- Points faibles :
  - Besoin d'un lecteur spécial
  - Perte de la carte
  - Possible insatisfaction des utilisateurs

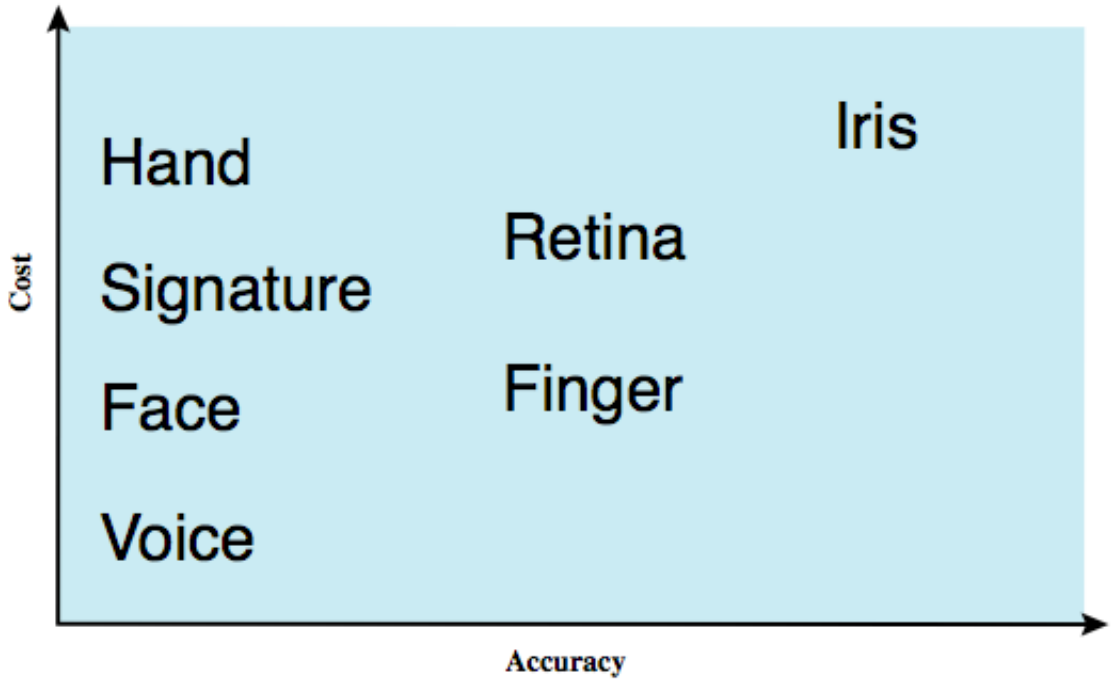
Smartcard



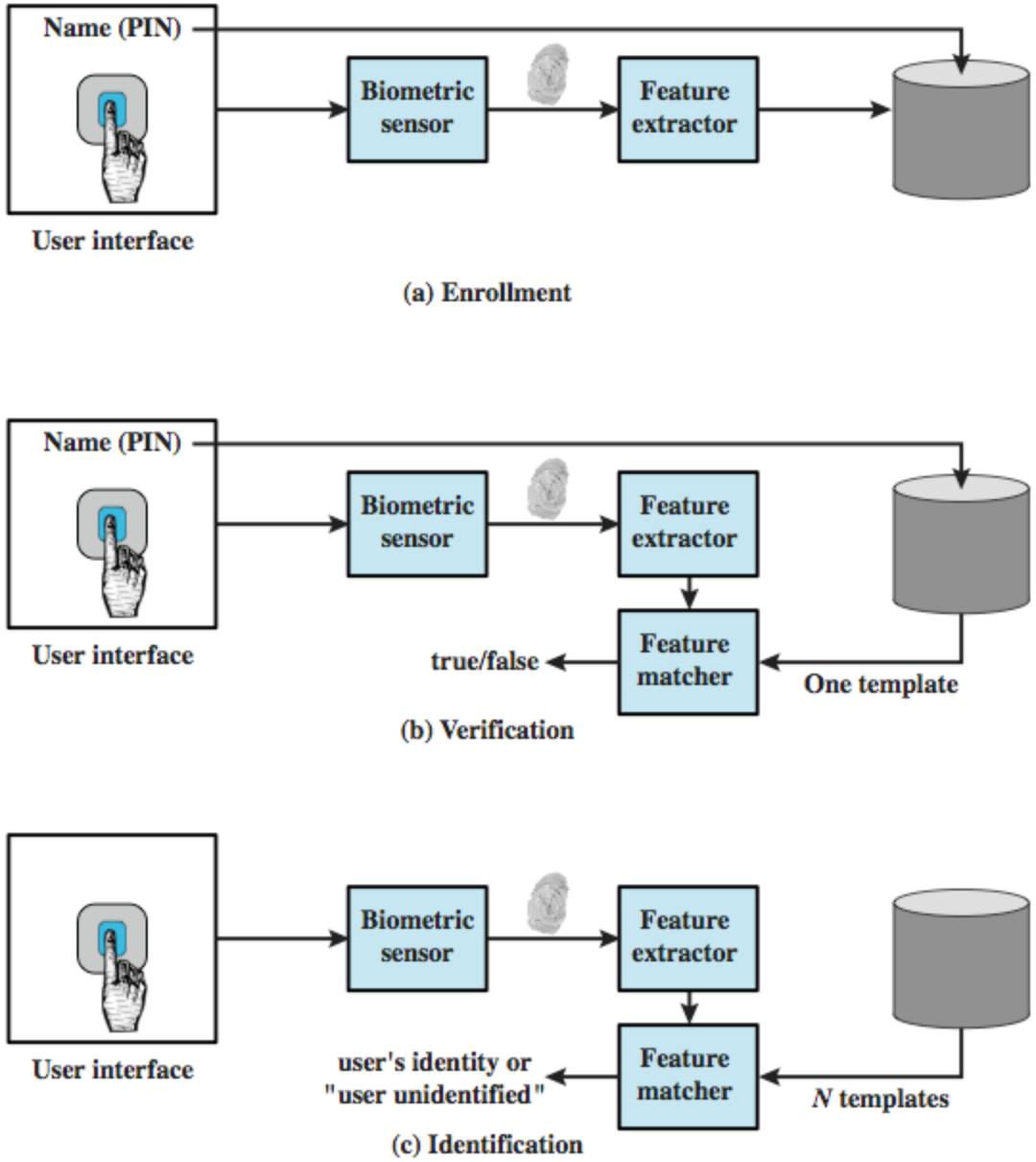
- Apparence d'une credit-card
- A son propre processeur, mémoire, I/O ports
  - Acces avec ou sans fil
  - Peut avoir un co-processor crypto
  - ROM, EEPROM, RAM memory
- Exécute un protocole pour authentifier reader/computer (static, dynamic, challenge)
- Clé USB : pas besoin de lecteur de carte

Authentification Biometrique

- Basé sur les caracteristiques physiques de l'utilisateur

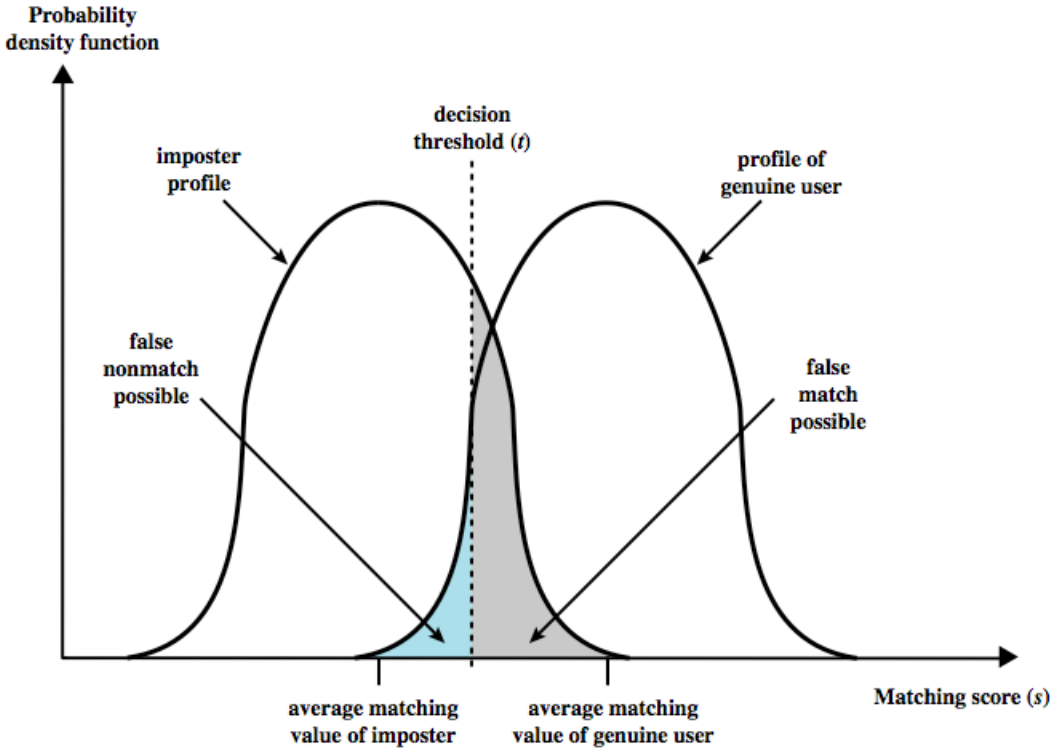


Operation of a Biometric System



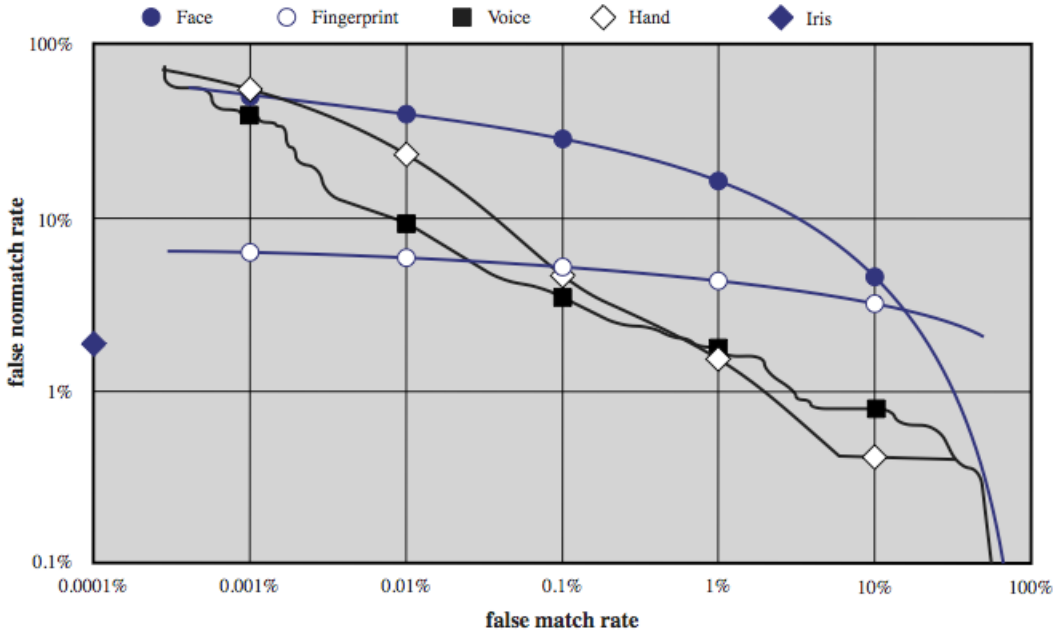
Fiabilité du Biometrique

- Comment transformer des caracteristiques en chiffres?
- Fausse acceptation ou Faux rejet ?



Fiabilité du Biometrique

- Courbe des caractéristiques suivant systèmes



Authentification à distance

- Plus complexe sur le réseau
  - problèmes d'espionnage, rejeu
- Utilise généralement challenge-response
  - *user* envoie son *identity*
  - *host* répond par un **nonce** r
  - *user* calcule et renvoie f(r,h(P))
  - *host* compare la valeur reçue avec celle qu'il calcule
- Protège contre certaines attaques

Attaques

- Client attacks (masquerade,...)
- Host attacks (sur le fichier de passwords)
- Eavesdropping
- Replay
- Trojan horse
- Denial-of-service

Kerberos

- Developpé au MIT pour le projet Athena (1988 pour vers. 4, 1994 pour vers. 5)
- Technique d'authentification avec TTP pour systèmes distribués
- Réalise une authentification centralisée
  - Permet l'accès à des services distribués sur le réseau (BD, File transfert, remote login,...)
  - Single sign-on

Kerberos overview

- Le Centre de Distribution de clés (KDC) contient une BD de
  - Clients et services
  - Clés secrètes (1 par client, 1 par service)
- Deux entités : SA (authentification) et TGS (permet l'accès au service demandé)
- N'utilise PAS de clé publique

Procédure d'authentification

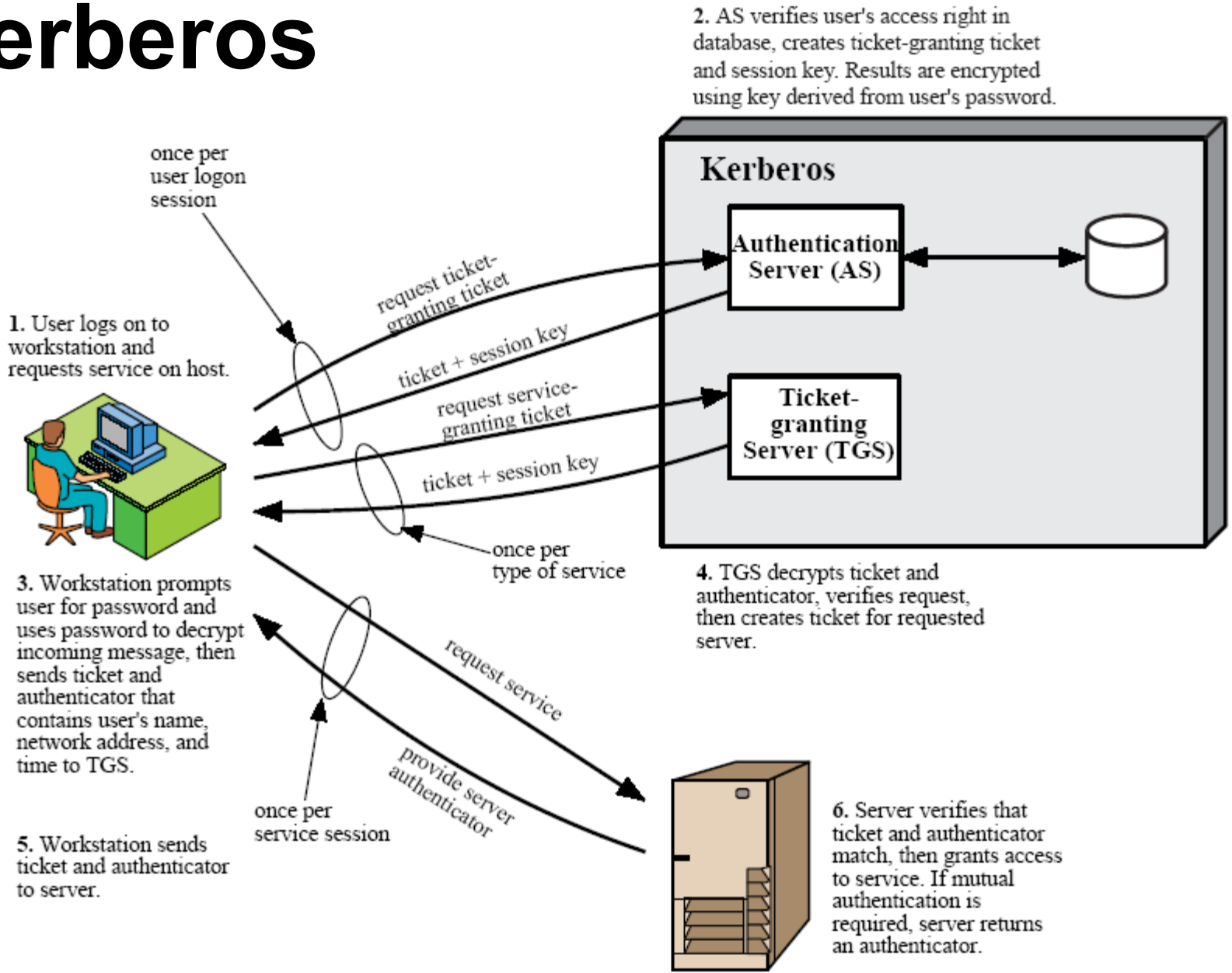
- Authentification initiale de l'utilisateur
  - L'utilisateur demande un jeton au SA
- Requête pour un service
  - L'utilisateur demande à TGS d'accéder à un service
  - TGS renvoie un jeton pour ce service
  - L'utilisateur envoie le jeton au serveur accompagné de la requête

Version 4 Dialogue d'Authentification

- Problèmes :
  - Lifetime associe avec le ticket-granting ticket
  - Si trop court → rentrer plusieurs fois le password
  - Si trop long → opportunité de rejeu

- Le danger est qu'un attaquant vole le ticket et l'utilise avant son expiration

Kerberos


2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.
Kerberos
Authentication Server (AS)
Ticket-granting Server (TGS)
once per user logon session
request ticket-granting ticket
ticket + session key
request service-granting ticket
ticket + session key
once per type of service
1. User logs on to workstation and requests service on host.
3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.
4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.
request service
provide server authenticator
once per service session
5. Workstation sends ticket and authenticator to server.
6. Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

Version 4 Dialogue d'Authentication

Authentication Service Exhange: obtenir un Ticket-Granting Ticket

(1) C → AS: IDc || IDtgs ||TS1

(2) AS → C: $E_{Kc}[K_{c,tgs}|| IDtgs || TS_2 || Lifetime_2 || Ticket_{tgs}]$

Ticket-Granting Service Echange: obtenir un Service-Granting Ticket

(3) C → TGS: IDv ||$Ticket_{tgs}$ ||$Authenticator_c$

(4) TGS → C: $E_{Kc,tgs}[K_{c,v}|| IDv || TS_4 || Ticket_v]$

Client/Server Authentication Exhange: accéder à un Service

(5) C → V: $Ticket_v$ || $Authenticator_c$

(6) V → C: EKc,v[TS5 +1]

Sécurité de kerberos

- Différences avec Needham-Schroeder?
- Avec le protocole de N-S, B n'a pas de moyen de savoir si la clé est nouvelle (ancienne clef compromise?)
- Le time-stamp et le lifetime permet d'éviter ce scénario

Résumé

- Introduction de l'authentification de l'utilisateur
  - par passwords
  - Par cartes
  - Par des moyens biométriques
- Authentification à distance