People Authentication I

Dr. Shlomo Kipnis
December 15, 2003

Authentication Objectives

- User identification (name, id, etc.)
- User validation (proof of identity)
- Resource identification (name, address, etc.)
- Resource validation (proof of identity)
- Access control (permission lists)
- Monitoring (online, offline)
- Auditing (logs, books)
- Other . . .

Authentication Considerations

- Accuracy Level
- Validation Time
- Processing Power
- Operating Costs
- System Reliability
- Ease of Forging
- Ease of Use
- Portability
- Transferability
- Revocability

Authentication Methods

1. Something the user knows:
   - Memorable Data – passwords, pass phrases, personal information
2. Something the user is/has:
   - Biometrics – finger prints, palm geometry, retina scan, iris scan, face recognition, signature characteristics, typing characteristics, DNA recognition
3. Something the user holds:
   - Tokens – ID card, paper card, key, smart card, crypto token, etc.

Passwords (I)

- Desirable properties:
  - Password should be easy to remember
  - Password should be hard to guess
- Problem:
  - Cannot achieve desirable two properties above with (current) humans

Passwords (II)

- Password lengths at some university (from a study in the early 1990's):

Length	Number	Percentage
1	55	0.4
2	87	0.6
3	212	2
4	449	3
5	1260	9
6	3035	22
7	2917	21
8	5772	42

Passwords (III)

- Common situation:
  - Short passwords (between 4 and 8 characters)
  - English words (few thousands)
  - Names (few hundreds)
  - Personal data (easy to obtain)
  - Combinations of above (easy to compute)
  - Passwords are written somewhere (easy to find)
  - Password in many systems (break one – break all)

Passwords (IV)

- Attacks on passwords:
  - Password guessing
  - Searching for passwords
  - Online dictionary attacks
  - Offline password cracking
  - Sniffing communication lines
  - Social engineering

Passwords (V)

- How/where passwords are Stored:
  - Not hidden and not access-protected (some PC's)
  - Hidden but not access-protected (some appliances)
  - Access-protected (Unix, NT, other OS)
  - Encrypted (but where is the key stored)
  - Hashed (Unix, NT, other OS)
  - Protected server (distributed systems)

Passwords (VI)

- Unix "salt" mechanism:
  - When user A opens an account, a password pw_A is defined, and a random 12-bit $salt_A$ is selected
  - The Unix password file stores both $salt_A$ and the value of $h(pw_A, salt_A)$ at the entry for user A
  - When a user attempts to log on as A and types some pw, the system takes $salt_A$ from A's entry in the file, computes $h(pw, salt_A)$, and compares the result to what is stored in the file. A match allows logging on.
  - This scheme makes online dictionary attacks infeasible (since each password can be hashed in $4096 = 2^{12}$ ways). This scheme is not resilient against offline password cracking.

Passwords (VII)

- Password breaking recipes:
  - Try default passwords used in standard system accounts
  - Exhaustively search all short passwords
  - Try words from online dictionaries
  - Collect and try data that is related to users (user names, family member names, birth dates, identification numbers, etc.)
  - Try common combinations of user data (e.g., reverse writing, adding digits at end of passwords, etc.)
  - Look for written passwords
  - Observe password typing patterns

Passwords (VIII)

- Password breaking recipes (continued):
  - Use a Trojan horse to steal users' passwords
  - Eavesdrop to communication lines
  - Get access to passwords files
  - Analyze the (hashed / encrypted) passwords file
  - Get from machine to machine with OS facilities and/or with known passwords
  - Pretend to be a legitimate user and ask the administrator to issue you a new password
  - Pretend to be a legitimate administrator and ask the user to disclose the password

Passwords (IX)

- System recommendations:
  - Educate users of importance of password security
  - Monitor user accounts for suspicious behavior
  - Lock account after a number of unsuccessful login attempts
  - Keep password file encrypted or hashed
  - Use password strengthening mechanisms (e.g. Unix salt)
  - Keep password files in secure locations (directories in the file system, special servers, etc.)
  - Request users to change passwords frequently
  - Run password cracking tests and disallow weak passwords
  - Use passwords only as one factor in authentication process

Passwords (X)

- Password selection recommendations:
  - Use combinations of letters, upper-case, lower-case, digits, other characters
  - Change passwords frequently
  - Use different passwords in different systems
  - Use random passwords (8-10 characters long)
  - Use readable passwords (16-20 characters)
  - Use pass-phrases (30-40 character sentences)

Passwords (XI)

- Password Administration Scenario:
  - Admin: Passwords must be changed every 90 days
  - User: Changes the password to the same password
  - Admin: Check that password is changed to a new one
  - User: Changes the password and changes again to the old one
  - Admin: Tracks last n passwords and checks that password is new
  - User: Changes the password $n+1$ times and returns to old one
  - Admin: Disallows more than one password change per day
  - User: Changes to the same password with $1, 2, 3, \ldots$ at the end
  - Admin: Disallows passwords that are "too similar" to old ones
  - User: Invents a random password and writes it down on paper

Passwords (XII)

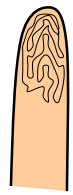
- Summary:
  - Accurate negative
  - Not accurate positive
  - Secret user data
  - Secret server data
  - Easy to break
  - Simple to operate
  - Portable
  - Transferable
  - Not easily revocable

Biometrics (I)

- Biometrics consists of checking physical, biological or physiological properties of a person
- Certain properties are highly unique to each person
- Need to select properties that are:
  - Easy to detect
  - Provide high levels of accuracy
- Need to maintain database of biometrics parameters

Biometrics (II)

- Finger Prints:
  - 2-D geometry
  - 3-D geometry
  - Liveliness checks (pulse, temperature, etc)
  - Capacitance / resistance checks
- Relatively accurate
- Needs maintenance
- Acceptance level is increasing



Biometrics (III)

- Palm Geometry:
  - 2-D geometry
  - Finger lengths
  - Finger widths
  - Gaps between fingers
- Good accuracy
- Low maintenance
- High acceptance

Biometrics (IV)

- Retina Scan:
  - 2-D map of blood vessels
  - Blood vessels are warmer than surrounding tissues
  - Detected by IR radiation
- Highly accurate
- Expensive (special equipment)
- Invasive (low acceptance)

Biometrics (V)

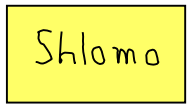
- Iris Scan:
  - 2-D map of iris texture
  - Detection by camera
- Relatively accurate
- Inexpensive
- Non-invasive (high acceptance)

Biometrics (VI)

- Face Recognition:
  - 2-D image
  - Bone-muscle model
  - Under-skin thermal radiation
- Detection by camera / IR
- Good accuracy
- Non-invasive

Biometrics (VII)

- Signature Characteristics:
  - 2-D image
  - Dynamic signature
  - Online test
  - Speed, pressure, angles, etc.
- Highly accurate
- Low maintenance
- High acceptance



Biometrics (VIII)

- Typing Characteristics:
  - Dynamic typing parameters
  - Speed, gaps, letter patterns
  - Online test
  - Could be tacit
- Highly accurate
- Low maintenance
- High acceptance

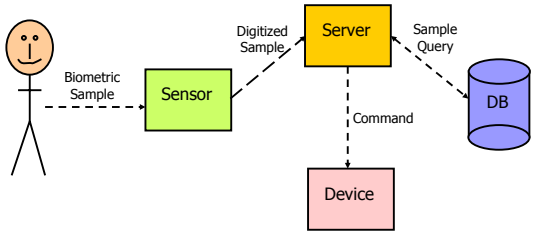
Biometrics (IX)

- DNA Recognition:
  - DNA matching against known patterns
  - Sample could be taken from external tissues

- Highly accurate
- Expensive (equipment)
- Conceived as invasive
- Not widely used (yet)

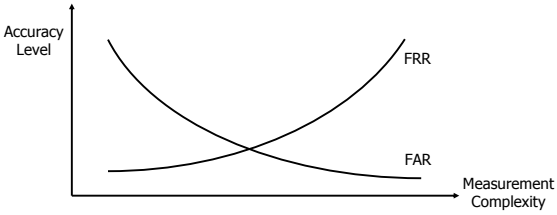
Biometrics (X)

- Biometric System Components:



Biometrics (XI)

- Accuracy levels:
  - FAR – False Accept Ratio
  - FRR – False Reject Ratio



Biometrics (XII)

- Biometrics – Summary:
  - Not totally accurate (positive & negative)
  - Private data (not secret data)
  - Easy to steal data
  - Costly to operate
  - Portable
  - Non-transferable
  - Non-revocable

Tokens (I)

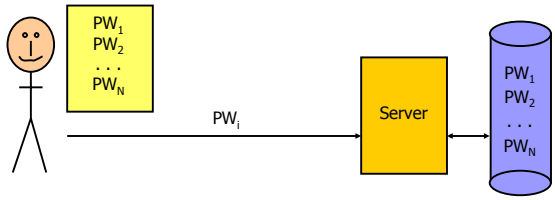
- Physical Tokens:
  - Physical keys (to physical locks)
  - Paper/plastic cards (ID, passwords, procedures, etc.)
  - Smart cards (crypto keys, algorithms, contact or contact-less, etc.)
- Assist in opening / remembering / computing

Tokens (II)

- Tokens – Summary:
  - Highly accurate (positive & negative)
  - Secret data / code
  - Some are hard to forge
  - Some are expensive
  - Simple but costly
  - Portable
  - Some are transferable
  - Some are revocable

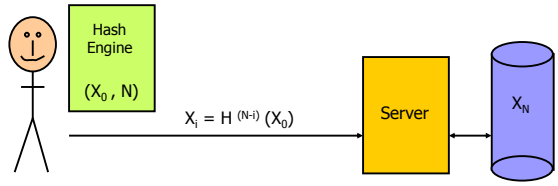
One-Time Passwords (I)

- Password Lists:
  - User has card with N one-time passwords
  - Server stores password list
  - Need to administer after N uses



One-Time Passwords (II)

- S/Key:
  - Card stores seed X_0 of a hash chain of length N
  - Server stores end of hash chain: $X_N = H^{(N)}(X_0)$
  - User submits password $X_i = H^{(N-i)}(X_0)$ at use i
  - Need to administer after N uses



Crypto Tokens (I)

- Token types:
  - Lists of one-time-passwords
  - Hash Engines for S/Key-like protocols
  - Crypto keys and algorithms
  - Clocks and time-synchronization algorithms
  - Password / PIN for token enabling / disabling
  - Password / PIN can be part of protocol

Crypto Tokens (II)

- Cryptographic Time-Challenge-Response Tokens:
- User enters PIN to enable token
- Token computes one-time-password based on:
  - secret key, current time, user PIN
- Server verifies one-time-password