PennState

CSE543
Introduction to Computer and Network Security
Module: Authentication

Asst. Prof. Syed Rafiul Hussain

Authentication and Authorization

- Fundamental mechanisms to enforce security on a system
- Authentication: Identify the principal responsible for a "message"
  - Distinguish friend from foe
- Authorization: Control access to system resources based on the identity of a principal
  - Determine whether a principal has the permissions to perform a restricted operation
- Today, we discuss principles behind authentication

What is Authentication?

- Short answer: establishes identity
  - Answers the question: To whom am I speaking?
- Long answer: evaluates the authenticity of identity by proving credentials
  - Credential – is proof of identity
  - Evaluation – process that assesses the correctness of the association between credential and claimed identity
    - for some purpose
    - under some policy (what constitutes a good cred.?)

Why authentication?

- Well, we live in a world of rights, permissions, and duties
  - Authentication establishes our identity so that we can obtain the set of rights
  - E.g., we establish our identity with Tiffany's by providing a valid credit card which gives us rights to purchase goods ~ physical authentication system

- Q: How does this relate to security?

Why authentication (cont.)?

- Same in online world, just different constraints
  - Vendor/customer are not physically co-located, so we must find other ways of providing identity
    - e.g., by providing credit card *number* ~ electronic authentication system
  - Risks (for customer and vendor) are different
    - Q: How so?

- *Computer security is crucially dependent on the proper design, management, and application of authentication systems.*

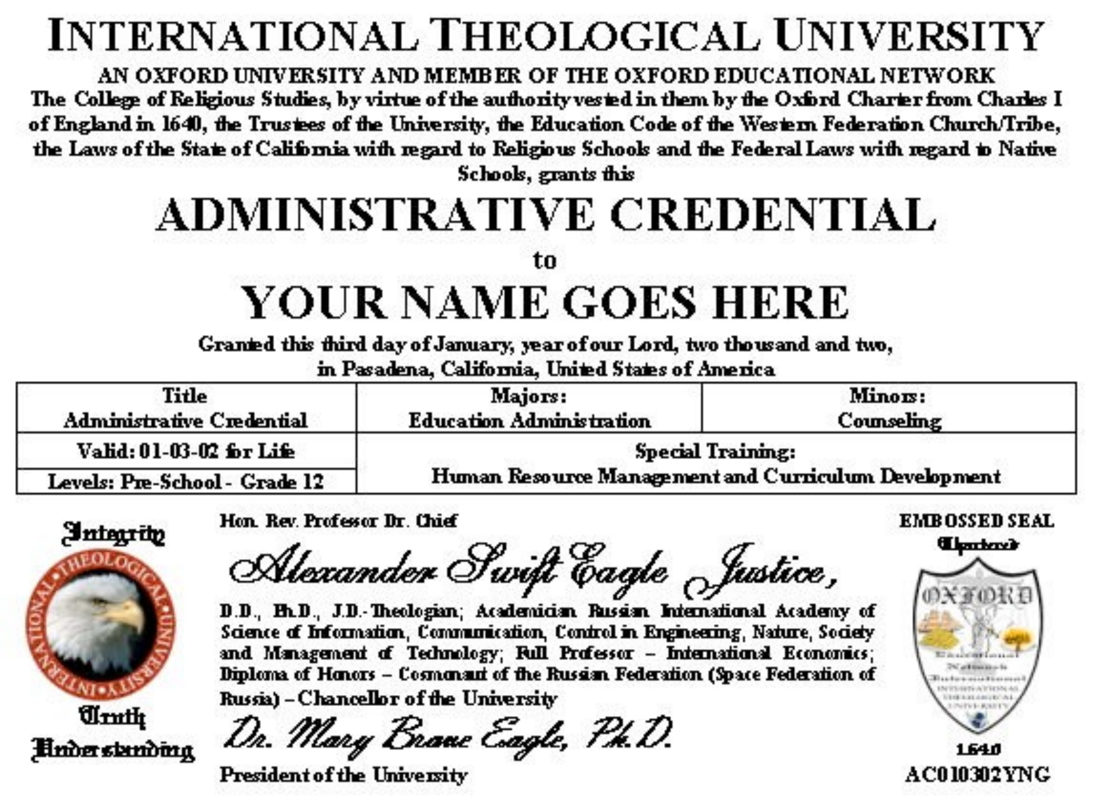
What is Identity?

- That which gives you access … which is largely determined by context
  - We all have lots of identities
  - Pseudo-identities
- Really, determined by who is evaluating credential
  - Driver's License, Passport, SSN prove …
  - Credit cards prove …
  - Signature proves …
  - Password proves …
  - Voice proves …

- Exercise: Give an example of bad mapping between ide purpose for which it was used.

Credentials

- … are evidence used to prove identity
- Credentials can be
  - Something I am
  - Something I have
  - Something I know



INTERNATIONAL THEOLOGICAL UNIVERSITY

AN OXFORD UNIVERSITY AND MEMBER OF THE OXFORD EDUCATIONAL NETWORK

The College of Religious Studies, by virtue of the authority vested in them by the Oxford Charter from Charles I of England in 1640, the Trustees of the University, the Education Code of the Western Federation Church/Tribe, the Laws of the State of California with regard to Religious Schools and the Federal Laws with regard to Native Schools, grants this

ADMINISTRATIVE CREDENTIAL

to

YOUR NAME GOES HERE

Granted this third day of January, year of our Lord, two thousand and two, in Pasadena, California, United States of America

Title Administrative Credential	Majors: Education Administration	Minors: Counseling
Valid: 01-03-02 for Life	Special Training: Human Resource Management and Curriculum Development	
Levels: Pre-School - Grade 12		

Integrity

Truth

Understanding

Hon. Rev. Professor Dr. Chief

Alexander Swift Eagle Justice,

D.D., Ph.D., J.D.-Theologian; Academician Russian International Academy of Science of Information, Communication, Control in Engineering, Nature, Society and Management of Technology; Full Professor – International Economics; Diploma of Honors – Cosmonaut of the Russian Federation (Space Federation of Russia) – Chancellor of the University

Dr. Mary Brave Eagle, Ph.D.

President of the University

EMBOSSED SEAL

1640

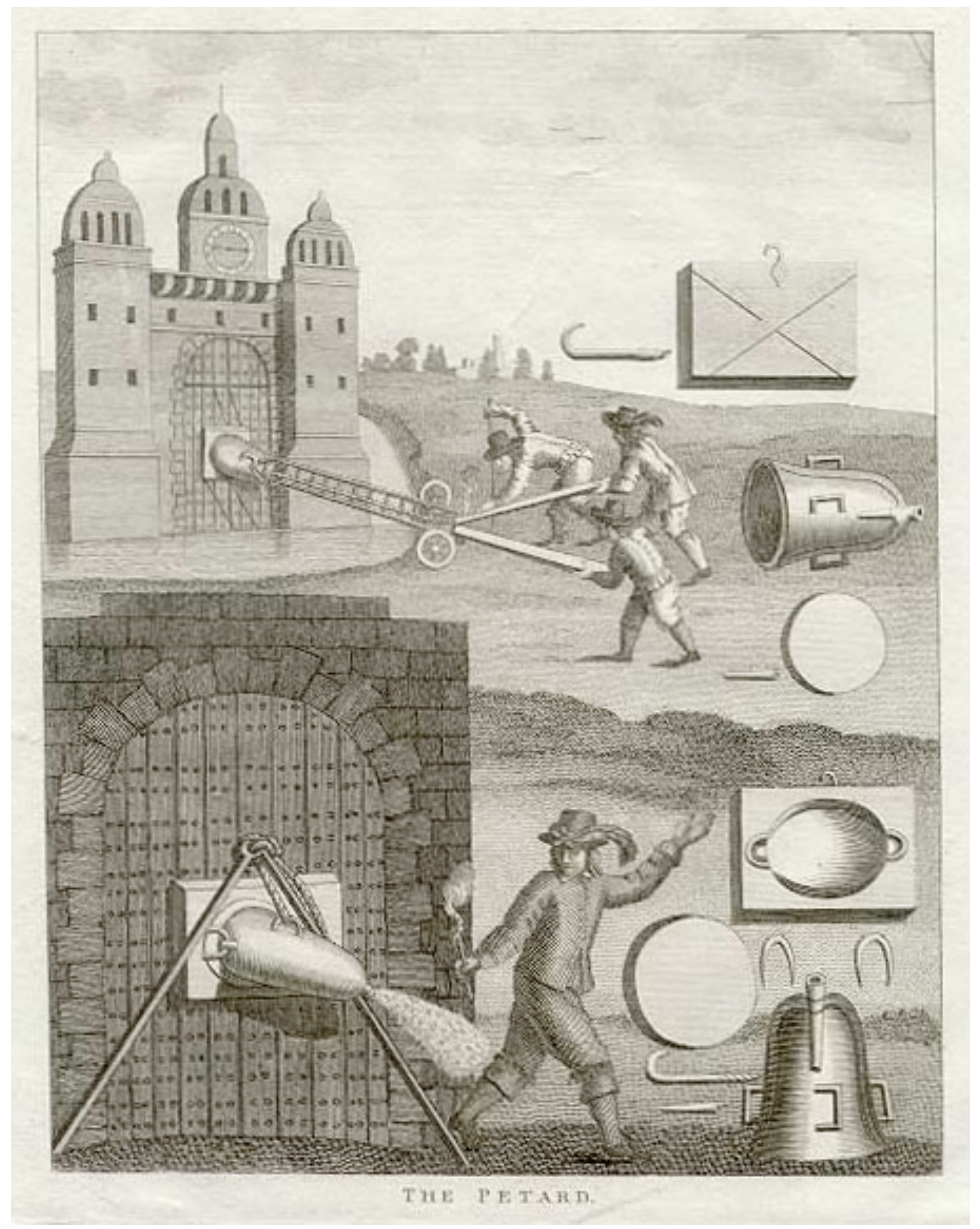
AC010302YNG

Something you know …

- Passport number, mothers maiden name, last 4 digits of your social security, credit card number
- Passwords and pass-phrases
  - Note: passwords have historically been pretty weak
    - Same bias with the context. E.g.?
    - Passwords used in more than one place
  - Not just because bad ones selected: If you can remember it, then a computer can guess it
    - Computers can often guess very quickly
    - Easy to mount offline attacks
    - Easy countermeasures for online attacks

"Hoist with his own petard"

- The rule of seven plus or minus two.
  - George Miller observed in 1956 that most humans can remember about 5-9 things more or less at once.
  - Thus is a kind of maximal entropy that one can hold in your head.
  - This limits the complexity of the passwords you can securely use, i.e., not write on a sheet of paper.
  - A perfectly random 8-char password has less entropy than a 56-bit key.



- Implication?

Password Use

- Naively: Retrieve password for ID from database and check against that supplied password

- Baravelli: ...you can't come in unless you give the password.
- Professor Wagstaff: Well, what is the password?
- Baravelli: Aw, no. You gotta tell me. Hey, I tell what I do. I give you three guesses. It's the name of a fish.
- …….
- [Slams door. Professor Wagstaff knocks again. Baravelli opens peephole again.] Hey, what's-a matter, you no understand English? You can't come in here unless you say, "Swordfish." Now I'll give you one more guess.
- Professor Wagstaff: ...swordfish, swordfish... I think I got it. Is it "swordfish"?
- Baravelli: Hah. That's-a it. You guess it.
- Professor Wagstaff: Pretty good, eh?

[Marx Brothers, *Horse Feathers*]

- How should you store passwords to protect them?
  - Just storing them in a file gives anyone with access to the file your password

Password Storage

- Store password as a "hash" of its value

- What properties must hash function satisfy for this purpose?
  - Should hash entries be invertible?
  - Could two passwords result in the same hash value?

Password Storage

- Store password as a "hash" of its value
  - Originally stored in /etc/passwd file (readable by all)
  - Now in /etc/shadow (readable only be *root*)

- What if an adversary can gain access to a password file?
  - How would you attack this?

“Salt”ing passwords

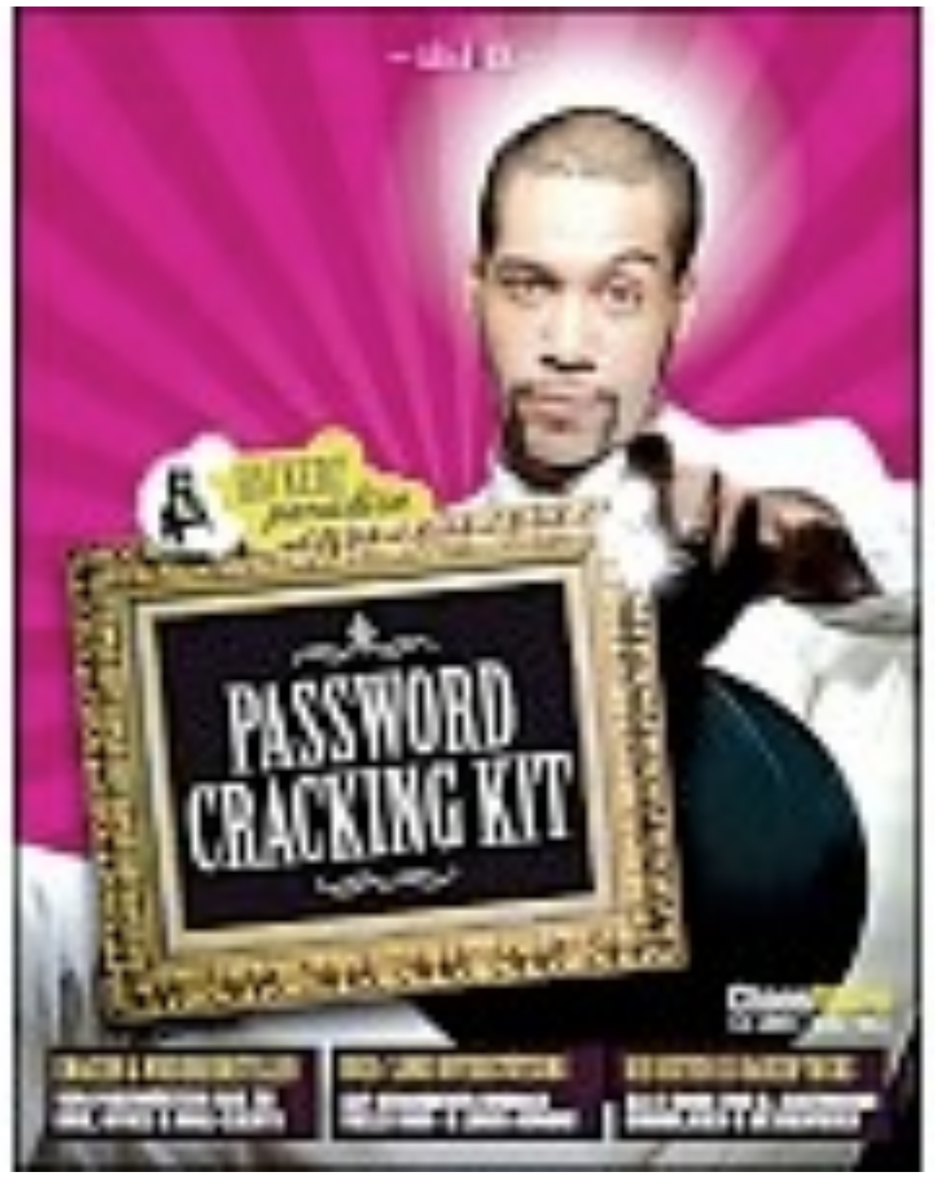
- Suppose you want to avoid a *offline dictionary attack*
  - bad guy precomputing popular passwords and looking at the password file
- A *salt* is a random number added to the password differentiate passwords when stored in `/etc/shadow`

$$\boxed{\begin{array}{c} salt_1, h(salt_1, pw_1) \\ salt_i, h(salt_2, pw_2) \\ salt_i, h(salt_3, pw_3) \\ \ldots \\ salt_n, h(salt_n, pw_n) \end{array}} \longleftarrow$$

- *consequence*: guesses each password independently

Password Cracking

- Attacker can access the hashed password
  - Can guess and test passwords offline
- Called "password cracking"
- Lots of help
  - John the Ripper
- How well do these work?



Cracking Passwords

- How hard are passwords to crack?
- How many 8-character passwords are there given that 128 characters are available?

Cracking Passwords

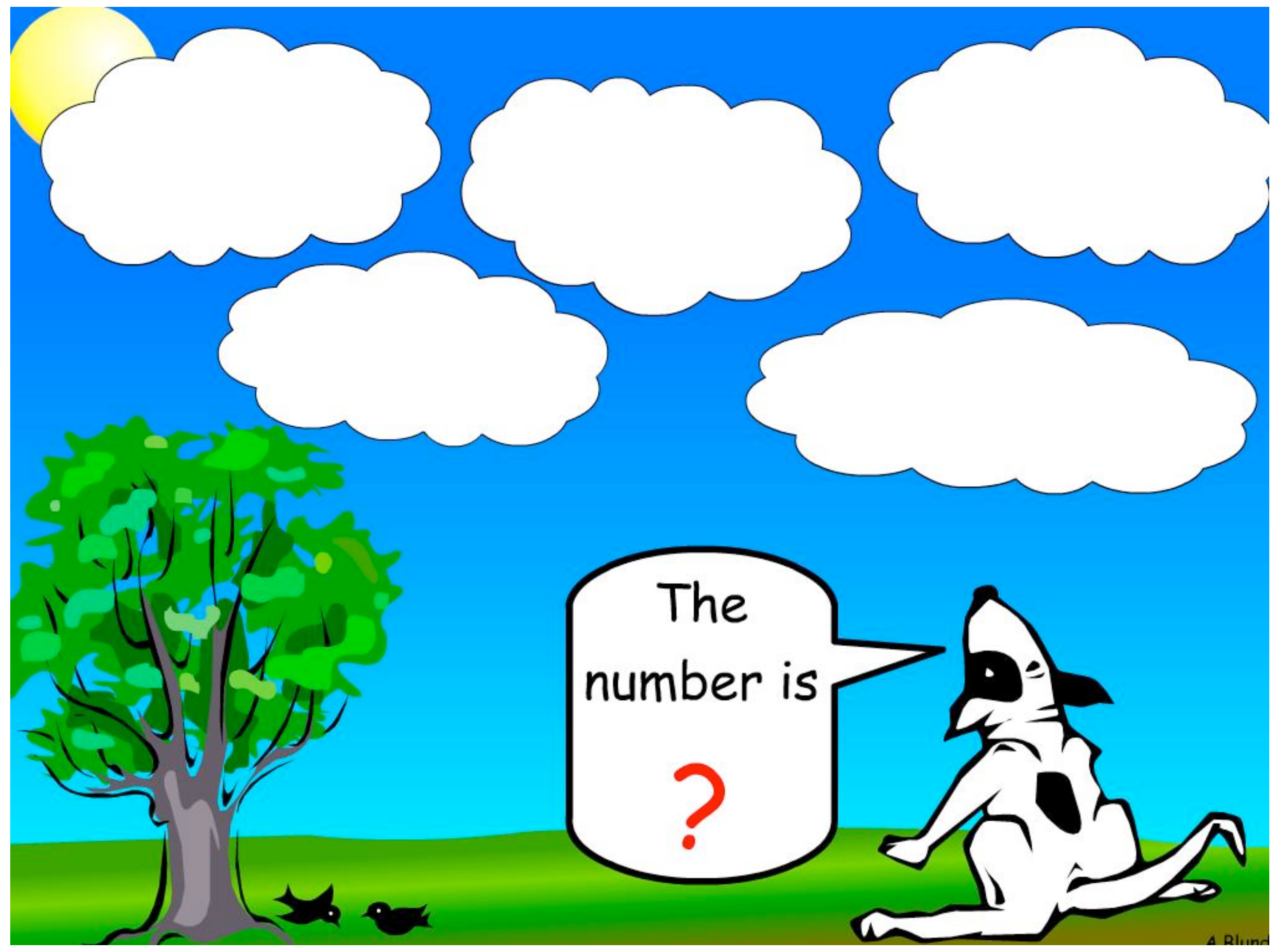
- How hard are passwords to crack?
- How many 8-character passwords given that 128 characters are available?
  - $128^8 = 2^{56}$
- How many guesses to find one specific user's password?
  - $2^{56}/2 = 2^{55}$

Guess Again...

- How do you know if your password will be guessed?
  - Follow password-composition policies
- Example properties
  - *Length*: 8 or12 or 16 chars?
  - *Requirements*: Password must contain at least one...
  - *Blacklist*: Password must not contain a dictionary word
- How do you know which policy to choose?
  - Studied in "Guess again ...: Measuring password strength by simulating password cracking algorithms," Gage Kelley, et al., IEEE Security and Privacy, 2012

Guess Number

- How do you predict how many guesses it will take to crack your password?
  - Try to crack it?
    - That can be time consuming
  - Compute number of guesses it would take?
    - How do we do that?



Guess Number

- Use specific cracking algorithm to compute number of guesses it would take to crack a specific password
  - Produce a deterministic guess ordering
- For "brute-force Markov" cracker
  - Uses frequencies of start chars and following chars
    - Most likely first, most likely to follow that, and so on...
  - Sum the number of guesses to find each character
    - In an N character alphabet and a password of length L:
      - The first character is the kth char tried in $(k-1)N^{L-1}$ guesses
      - The second character is the kth char tried in $(k-1)N^{L-2}$ guesses
      - Etc.

Guessing Passwords

- Suppose password is "CAC"
  - In character set {ABC}
- Start with highest probability start - A
  - Compute all passwords that start with A
  - In highest probability order - count so far - $k^n = 9$
- Then go to the next highest prob. start - say C
  - Next highest prob. for second char - A
  - Then A, B, C for third char
- For a guess number of 11

Guess Number

- Use specific cracking algorithm to compute number of guesses it would take to crack a specific password
  - Produce a deterministic guess ordering
- For "Weir" cracker
  - (Probabilistic Context-Free Grammar)
  - Uses probabilities of password structures
    - E.g., Small letter ^ N + Number ^ 1 + Capital letter ^ M …
- Computing guess number
  - Determine the guesses necessary to reach the "probability group" for that password
  - Add number of further guesses to reach exact password

Guessing Passwords

- Suppose highest password is "BA1"
  - In character set {AB1}
- Start with highest probability struct - $\{L^2D^1\}$
  - Search for most likely L^2 and most likely D^1
- For Markov, search from highest probability - A
  - $K^n = 2$
  - Next highest prob. - B
  - Then A
  - Then 1 for D^1
- For a guess number of 3

How Many Guesses For?

- By password-composition policy

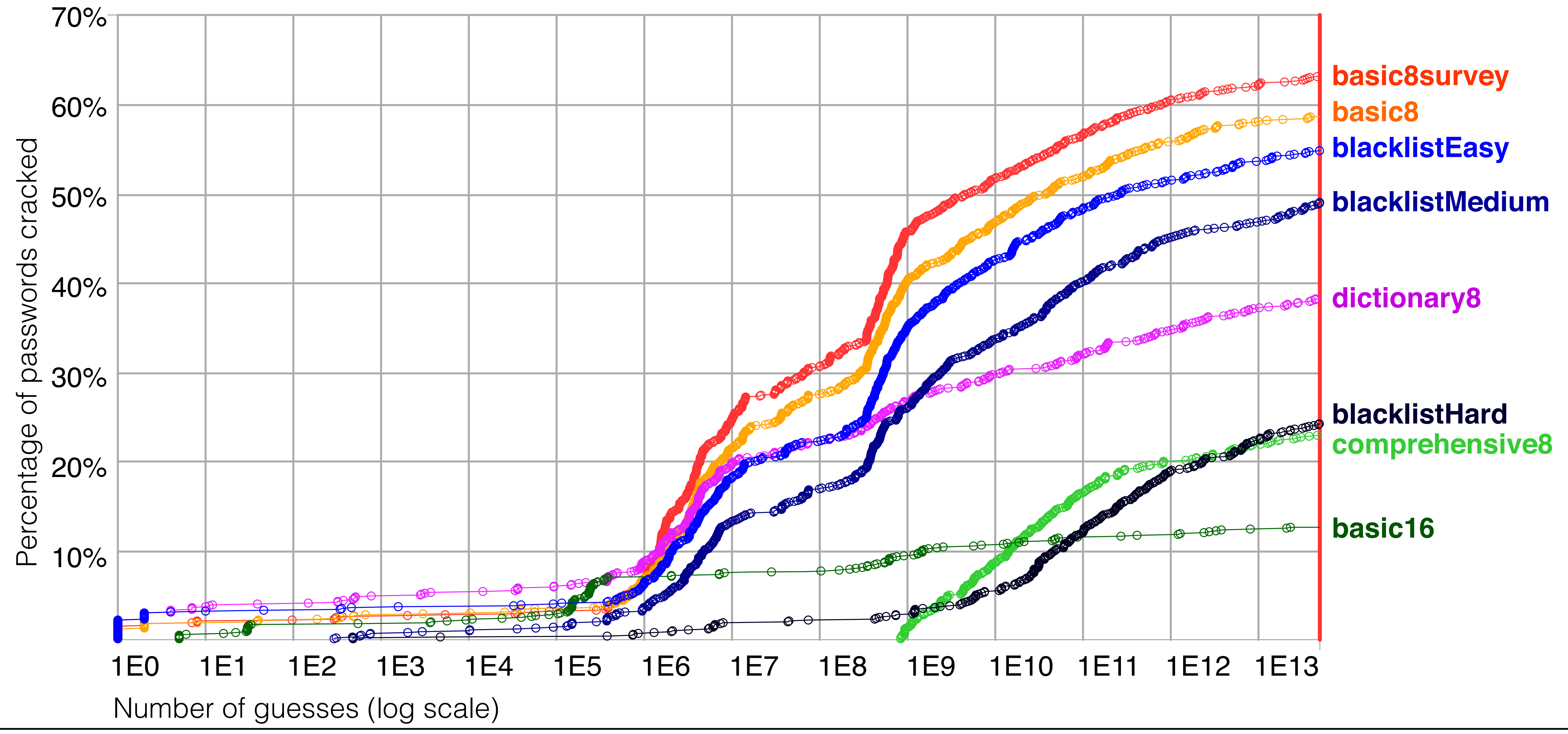


Figure 1. The number of passwords cracked vs. number of guesses, per condition, for experiment E. This experiment uses the Weir calculator and our most comprehensive training set, which combines our passwords with public data.

Something you have …

- Tokens (transponders, …)
  - Speedpass, EZ-pass
  - SecureID
- Smartcards
  - Unpowered processors
  - Small NV storage
  - Tamper resistant

- Digital Certificates (used by Websites to authenticate themselves to customers)
  - More on this later …

A (simplified) sample token device

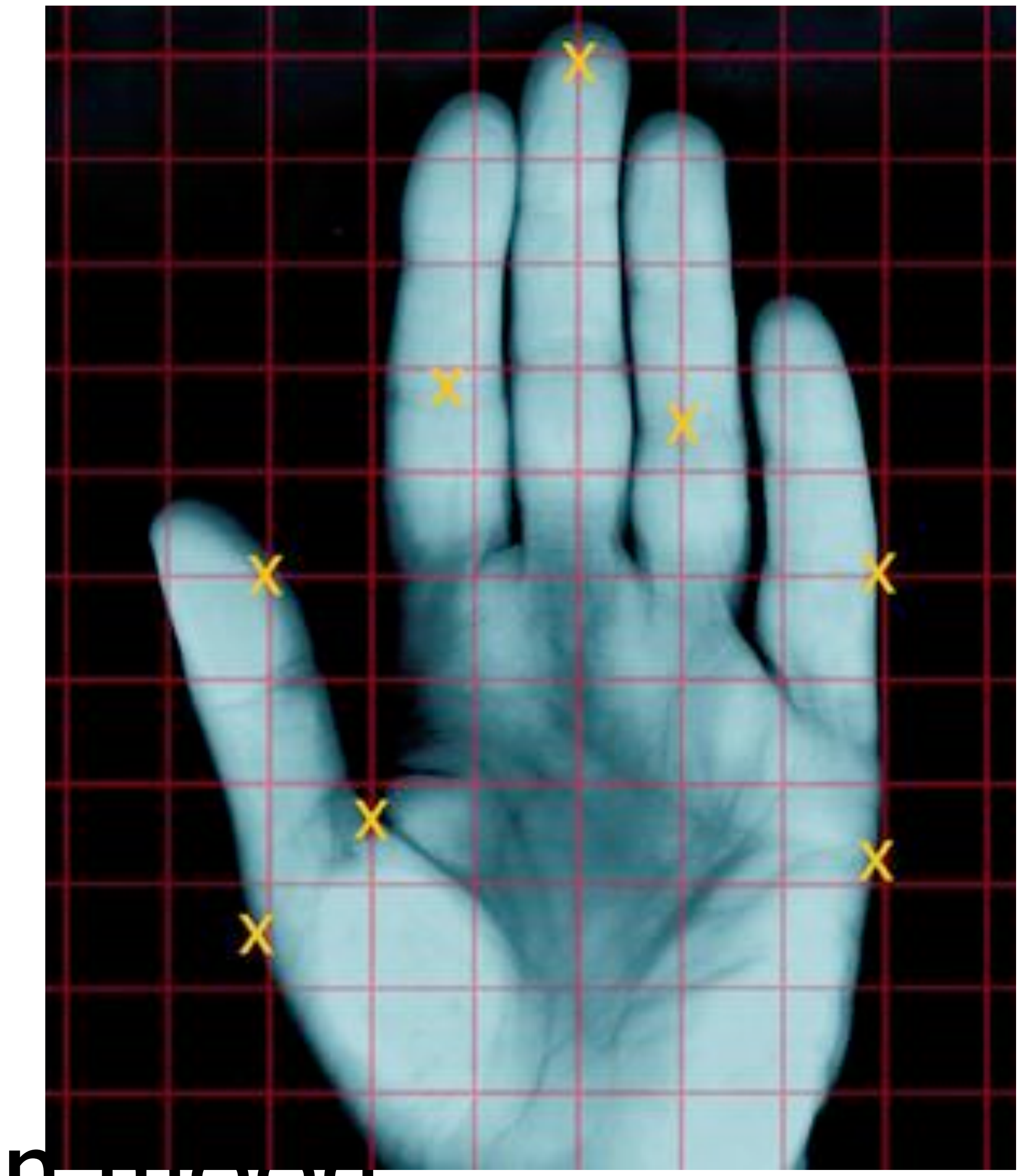
- A one-time password system that essentially uses a *hash chain* as authenticators.
  - For seed (S) and chain length (l)
  - Tamperproof token encodes S in firmware

$$\mathrm{pw}_i = h^{l-i}(S)$$

  - Device display shows password for epoch i
  - Time synchronization allows authentication server to know what i is expected, and authenticate the user.
- *Note*: somebody can see your token display at some time but learn nothing useful for later periods.

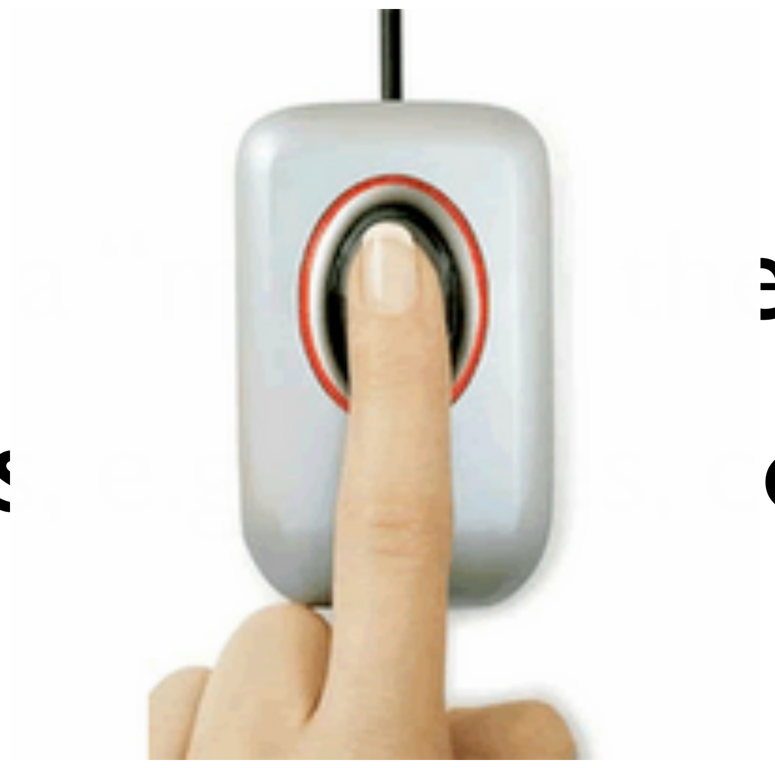
Something your are …

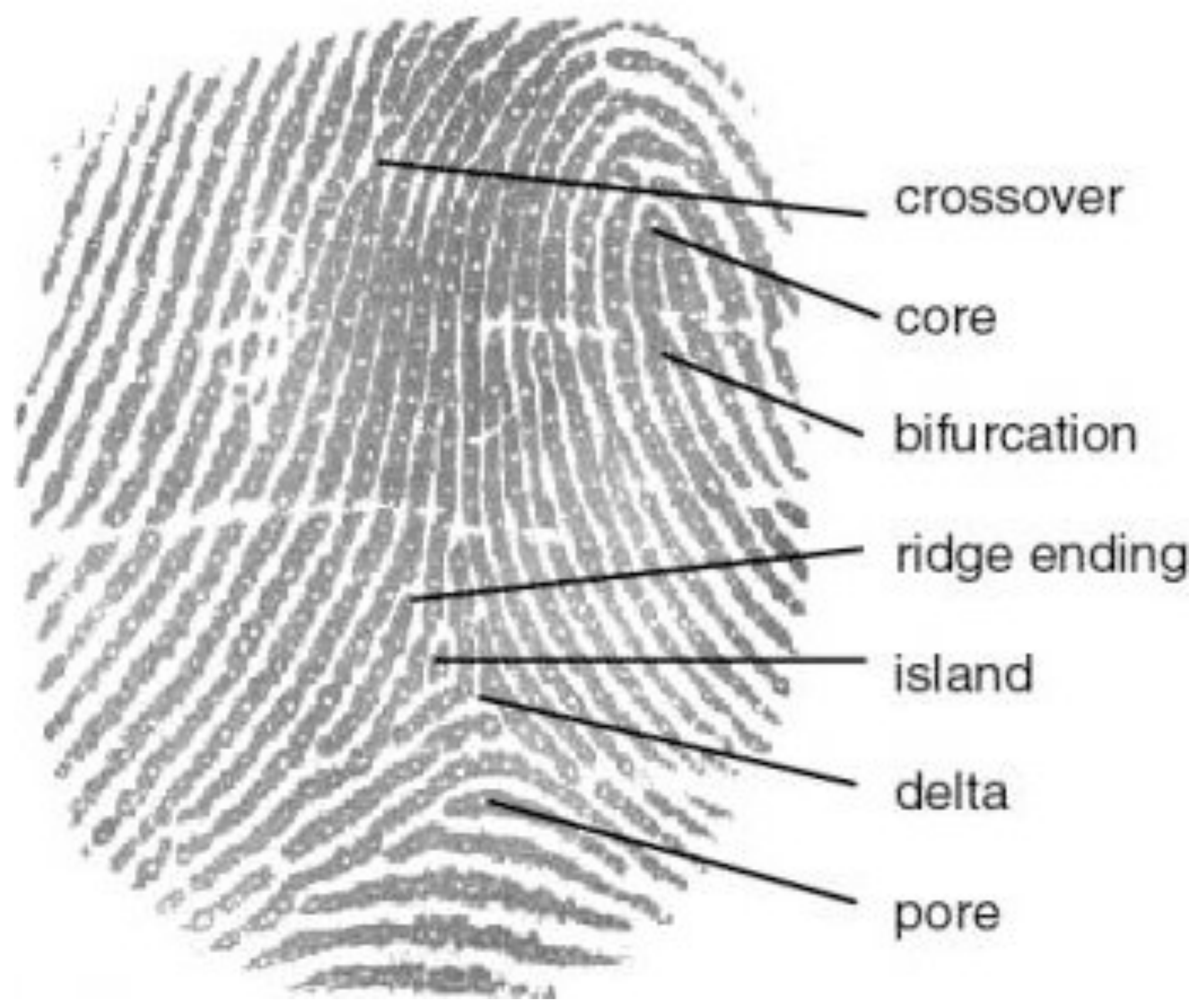
- Biometrics measure some physical characteristic
  - Fingerprint, face recognition, retina scanners, voice, signature, DNA
  - Can be extremely accurate and fast
  - Active biometrics authenticate
  - Passive biometrics recognize



- Issues with biometrics?
  - Revocation – lost fingerprint?
  - "fuzzy" credential, e.g., your face changes based on mood ...
  - Great for physical security, not feasible for on-line systems

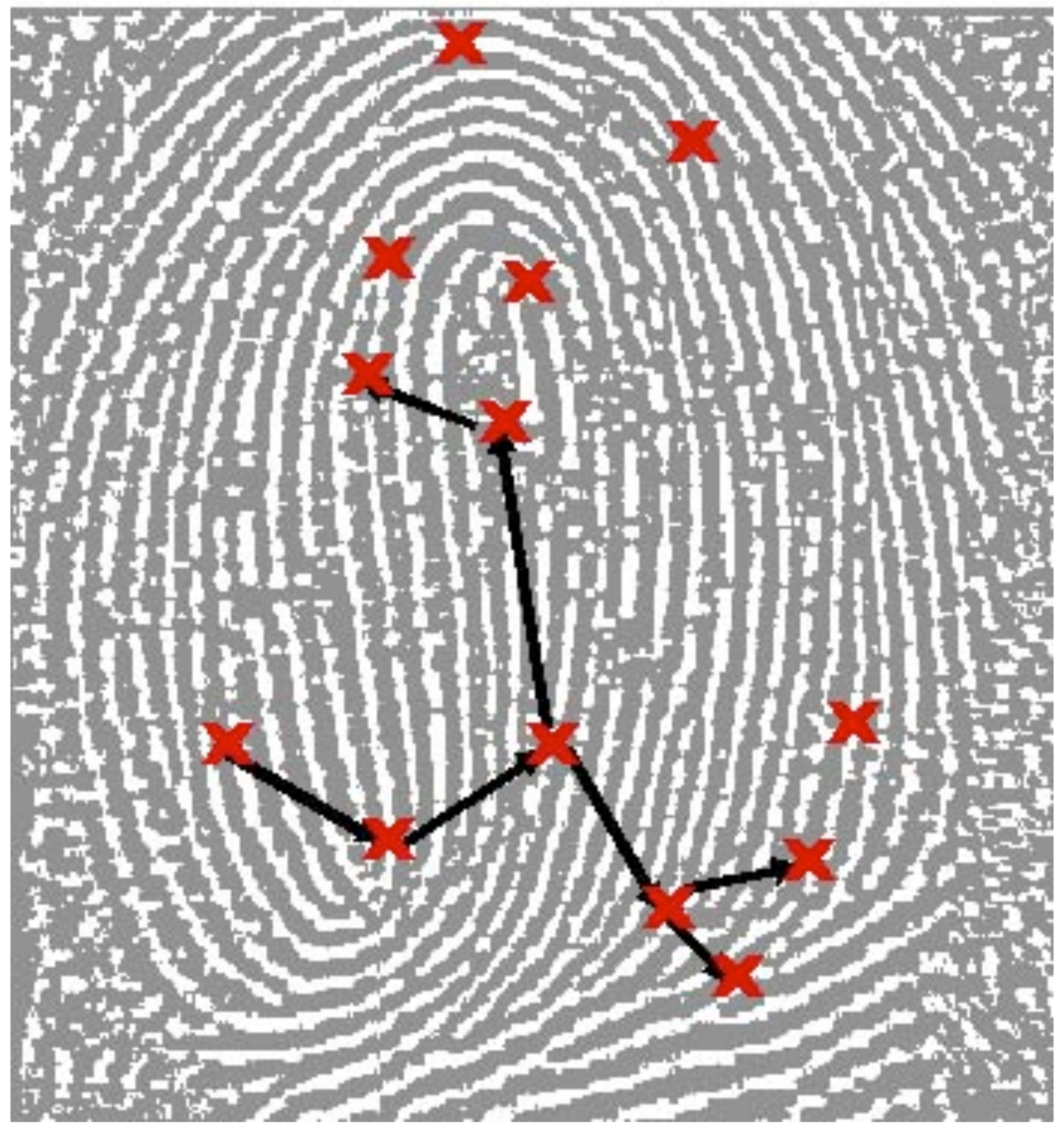
Biometrics Example

- A fingerprint biometric device (of several)
  - record the conductivity of the surface of your finger to build  e ridges
  - scanned map converted into a graph by looking for landmarks ores, ...



Fingerprint Biometrics (cont.)

- Graph is compared to database of authentic identities
- Graph is same, the person deemed "authentic"
  - This is a variant of the *graph isomorphism* problem
  - Problem: what does it mean to be the "same enough"
    - rotation
    - imperfect contact
    - finger damage



- *Fundamental Problem*: False accept vs. false reject rates?