User Authentication

Reading: Ch. 3, Van Oorschot

Furkan Alaca

University of Toronto Mississauga

CSC347H5, Fall 2018

Authentication

- Authentication is the process that establishes the origin of information or determines an entity's identity
  - Fundamental building block for computer security, and primary line of defence
  - User authentication is the basis for most types of access control and for user accountability
  - Other types of authentication: Server authentication (see SSH and HTTPS), message authentication, API authentication (e.g., OAuth 2.0)
- **User authentication** consists of two steps:
  1. Identification: The entity presents an identifier (e.g., user name) to the security system
  2. Verification: The entity presents or generates authentication information (e.g., a password) that corroborates the binding between the entity and the identifier
- Systems can use the authenticated identity to determine if the authenticated individual is **authorized** to perform particular functions

Means of Authentication

- Something you **know**, e.g., password, personal identification number (PIN), or answers to a prearranged set of questions
- Something you **possess**, e.g., electronic keycards, smart cards, or physical keys (also referred to as a **token**)
- Something you **are (static biometrics)**, e.g., recognition by fingerprint, retina, or face
- Something you **do (dynamic biometrics)**, e.g., recognition by voice pattern, handwriting characteristics, or typing rhythm
- Each method has advantages and disadvantages:
  - A user may forget a password or lose a token
  - Biometrics involve dealing with false positives and false negatives, user acceptance, cost, and convenience
  - An attacker may guess or steal a password, or steal (and possibly duplicate) a token
- **Two-factor** or **multi-factor** authentication: Combination of two or more factors

Risk Assessment

- **Assurance level** describes the degree of confidence (1) in the authentication process and (2) that the entity presenting the credentials is the same entity to whom those credentials were issued
- A higher assurance level is required for accounts whose breach can lead to more severe security consequences
- Accounts can be classified by **potential impact** of a security breach (Florencio et al., 2014)
  - **Don't care:** Breach has no impact on the user, e.g., throwaway accounts created to read "free" articles
  - **Low consequence:** Breach has minimal or easily-repaired consequences, e.g., discussion board account
  - **Medium-consequence:** Breach would have non-trivial but limited consequences, e.g., secondary e-mail account
  - **High-consequence:** Related to finance or primary employment, compromise has major consequences, e.g., primary e-mail, online banking, access to corporate network
  - **Ultra-sensitive:** May cause life-altering or irreversible damage, e.g., storage of state secrets

Password-Based Authentication

- The most commonly-used method of user authentication
- Requires the user to present both a user ID and a password
- The user ID is used for:
  - Determining whether the user is authorized to gain access to the system
  - Determining the privileges accorded to that user
  - In Discretionary Access Control (see Ch. 5), the user may grant permissions to other users by specifying their user IDs
- The system must maintain a password file indexed by user ID
- To authenticate a user, the system must compare the password presented with the one stored for the specified user ID
  - Typically, it is a one-way hash of the password which is stored
- **Important:** We mostly focus on **remote** authentication, but also discuss **local** authentication. We will be drawing contrasts between them: do not get confused between the two!

Attacks on Passwords

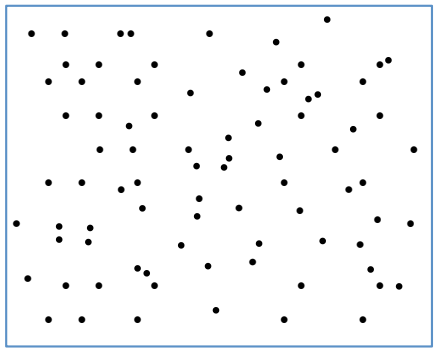
- **Password capture:**
  - Client-side malware, software/hardware keyloggers
  - Shoulder surfing
  - Passwords written down and stored in a non-secure location
  - Phishing
  - Code injection attacks
  - Eavesdropping/MITM
  - Side-channel attacks
- **Session/Workstation hijacking:**
  - Attacker may hijack existing session
    - Countermeasure: Channel binding
  - Attacker may misuse unattended workstation
    - Countermeasures: Continuous auth., screen lockout policy, IDS
  - Applies to other forms of user authentication as well
- **Exploiting user mistakes:**
  - Default passwords that have not been changed
  - Password reuse

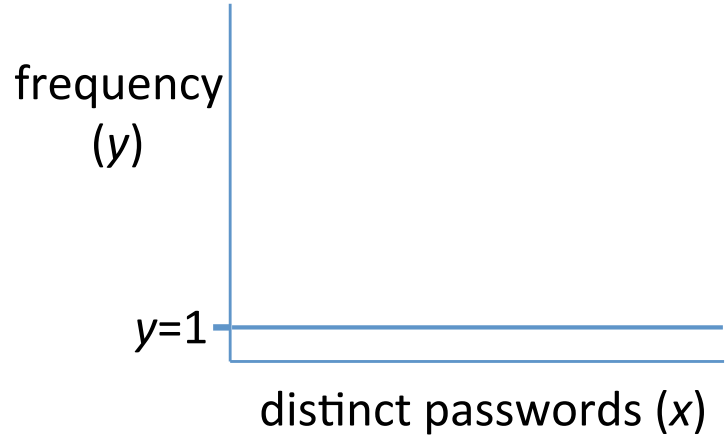
Attacks on Passwords (2): Guessing Attacks

- **Offline dictionary attack:**
  - If the attacker steals the password file, hashes of common passwords can be compared with values stored in the password file
  - Access control should be used to protect the system's password file
  - If a breach is detected, all passwords should be re-issued
- **Online guessing attack:**
  - Attacker picks a list of popular passwords and tries them against a wide range of user IDs
  - Countermeasures: Throttling, password composition policies/blacklists
- **Targeted online guessing attack:**
  - Attacker targets a specific account and submits password guesses until the correct password is discovered
  - Attacker may also try to use knowledge about the user (e.g., birth date, name) to guess the password – could be countered with a password policy
- Next slide: Let's discuss password **space** vs. **distribution**

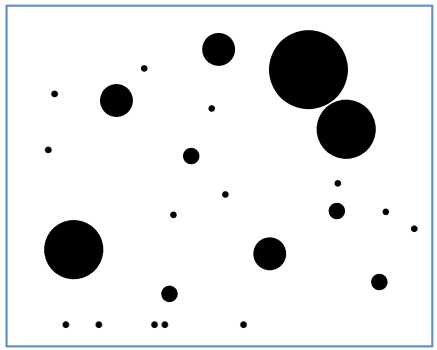
Password Distributions

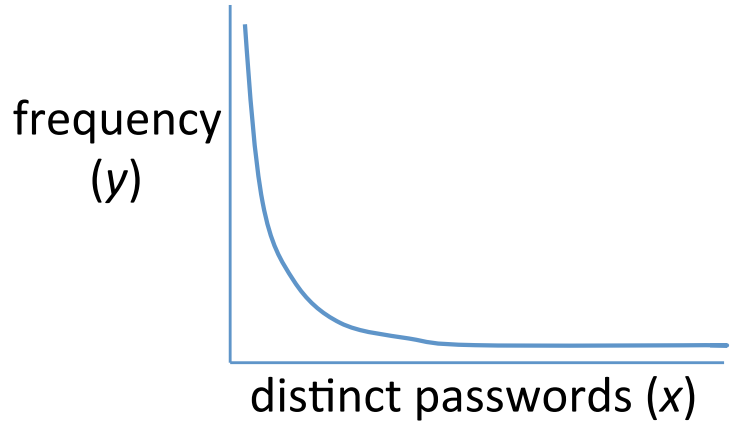
(a) What we want: randomly distributed passwords:





(b) What we get: predictable clustering, highly skewed distribution:





Source: Van Oorschot

Popular User-Chosen Passwords

Source: `http://lorrie.cranor.org/blog/2013/08/12/security-blanket/`

Password Cracking

- Dictionary attacks
  - Naive approach: Try every single possible password (brute-force)
  - Better approach: Develop a dictionary of likely passwords
  - For each guess, compute the hash of the password (with the salt, if applicable) and compare with the stored value
- Rainbow table attacks
  - Pre-compute tables of hash values for all salts (trade off storage for computation time)
  - Can be rendered infeasible by sufficiently large random salt values, since each password would need to be hashed with every possible salt value
- Intelligent dictionary attacks
  - Model the space of all passwords, search in the optimal order (i.e., from most likely to least likely) to maximize number of passwords cracked
  - Use leaked password databases and apply algorithms to try likely variations (Probabilistic context-free grammars, Markov models)
    - Idea is to exploit patterns, e.g., capital letters likely to occur at the beginning, numbers or punctuation likely to occur at the end

Password Cracking (3)

- Analysis of 25,000 student passwords at a university by Mazurek et al.
- Used RockYou leaked password database and Weir's algorithm

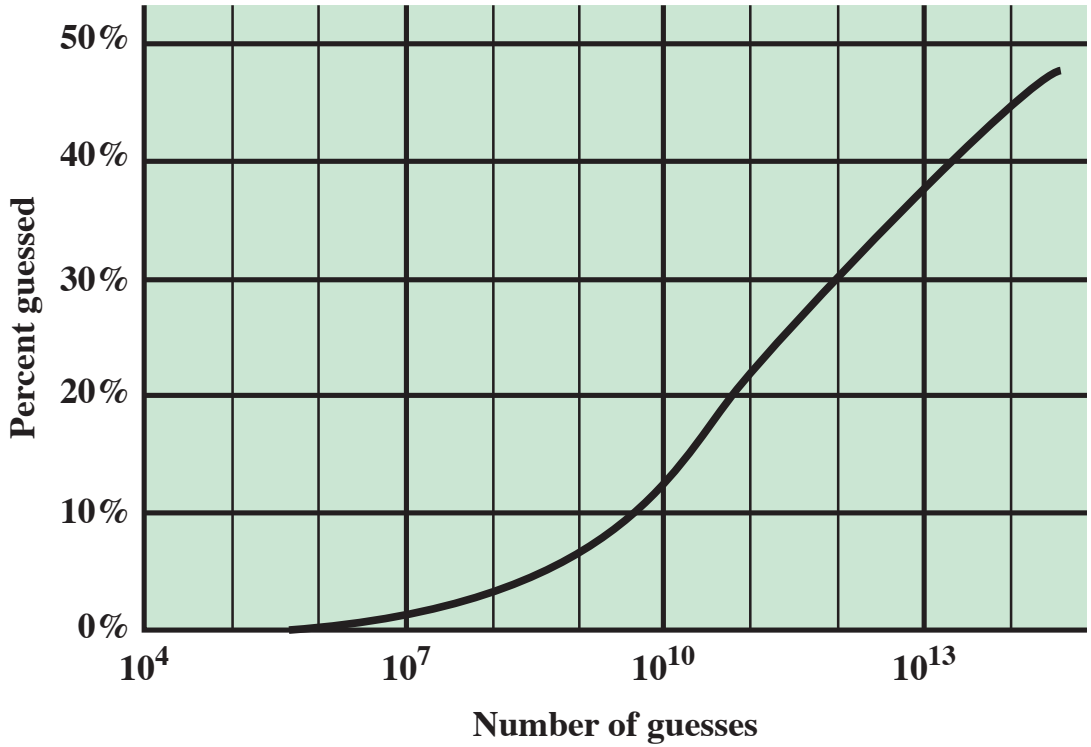


Figure 3.3 The Percentage of Passwords Guessed After a Given Number of Guesses

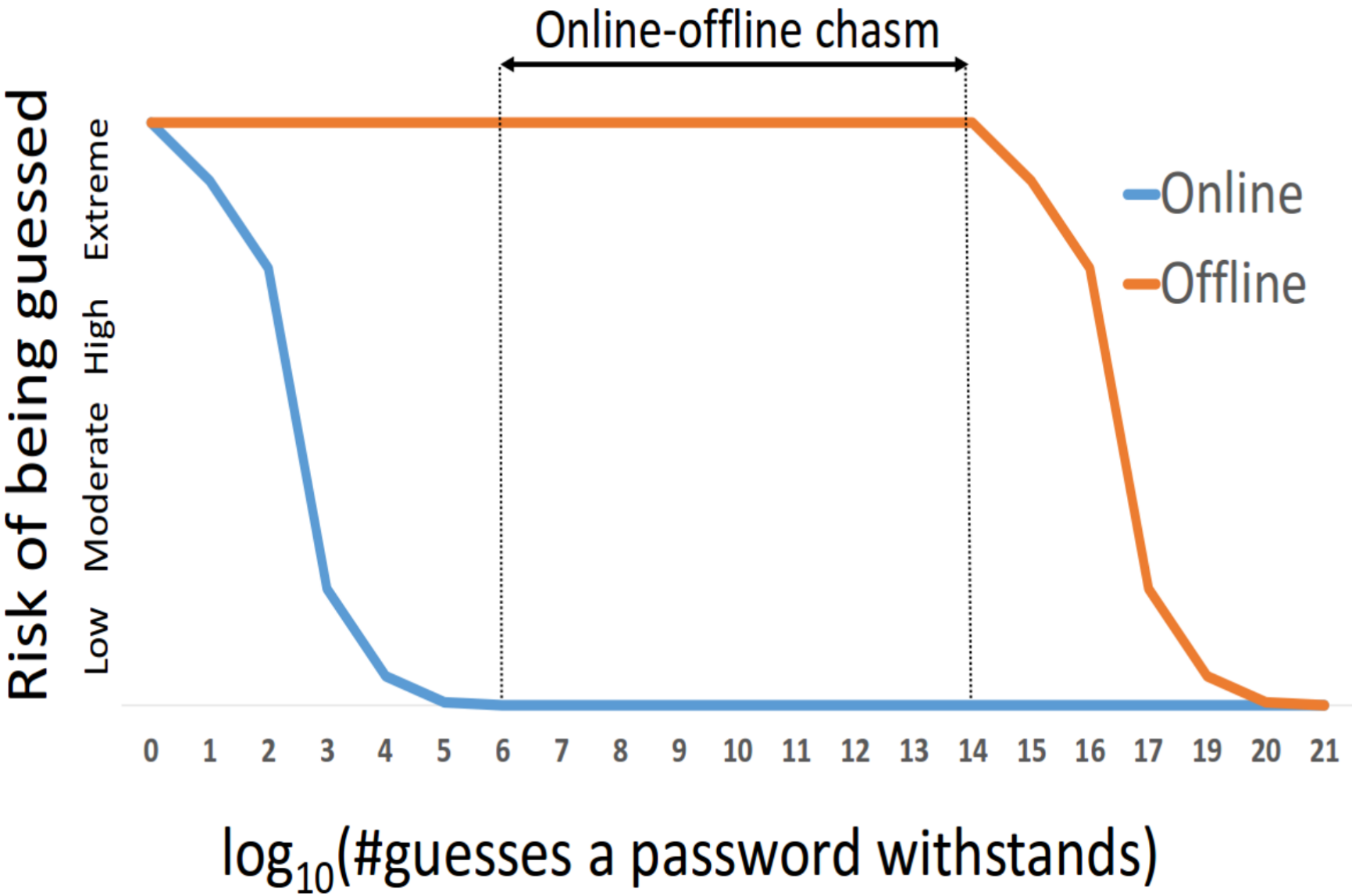
Hashing and Salting Passwords

- Naive approach: Hash the password and store the hashed value in the system's password file
  - Duplicate passwords are visible in the password file, and even across different systems
  - Attackers can pre-compute **rainbow tables** (space vs. time trade-off)
- Appending a **salt** value to a password before computing the hash solves the above problems
  - Even if two users have the same password, they will have different salts, so the hashed passwords will be different
  - Guessing difficulty increases by a factor of 2^b, where b is the size in bits of the salt
  - Constructing rainbow tables becomes infeasible
- The salt value is randomly generated, and it is stored alongside the hashed password and user ID in the password file
- Modern algorithms (e.g., Argon2, scrypt, bcrypt) typically involve multiple iterations to slow down attacks
- Is encryption a suitable alternative to hashing?

Password Selection

- Asymmetry: Attacker needs to break only **one** account to gain a foothold in the system, so the system administrator has the burden of ensuring that **all** users select strong passwords
- Password selection involves a security/usability trade-off:
  - If there is no password policy, users may choose a password that is too short or too easy to guess
  - System-assigned passwords which are random and sufficiently long are infeasible to crack, but also difficult to remember
- User education has not proven to be very effective
- Password policies can be used to force users to pick passwords that conform to a certain policy
  - How effective are they in reducing the predictability of user-chosen passwords?

Online-Offline Chasm



Source: D. Florencio, C. Herley, P. C. van Oorschot, "An Administrator's Guide to Internet Password Research", USENIX LISA 2014

Password Selection

- A **reactive password checker** installed on a system periodically runs a password cracker on its own database to find guessable passwords
  - Can notify users with cracked passwords that they should pick a new password
  - Resource-intensive if done right
  - Existing weak passwords may remain vulnerable until the reactive password checker finds them
- A **proactive password checker** allows the user to select their own password, but checks to see if the password is allowable before accepting it
  - Rule enforcement
  - Blacklisted passwords (computation/storage requirements can be reduced using Bloom filters)
  - Difficult to determine the "guessability" of a password
  - Password meters: Lots of recent research, e.g., see **zxcvbn**

Password meters

	qwER43@!	Tr0ub4dour&3	correcthorsebatterystaple
zxcvbn	Weak	So-so	Great!
Dropbox (old)	Great!	Great!	So-so
Citibank	Medium	Strong	1 number required
Bank of America	(not allowed)	(not allowed)	(not allowed)
Twitter	✓ Password is perfect!	✓ Password is perfect!	✓ Password is perfect!
PayPal	Weak	Strong	Weak
eBay	Strong	Strong	(not allowed)
Facebook	Password strength: Strong	Password strength: Strong	Password strength: Weak
Yahoo!	Very strong	Very strong	Weak
Gmail	Strong	Strong	Good

Source: Dan Wheeler blog post on zxcvbn,
`https://blogs.dropbox.com/tech/2012/04/zxcvbn-realistic-password-strength-estimation/`

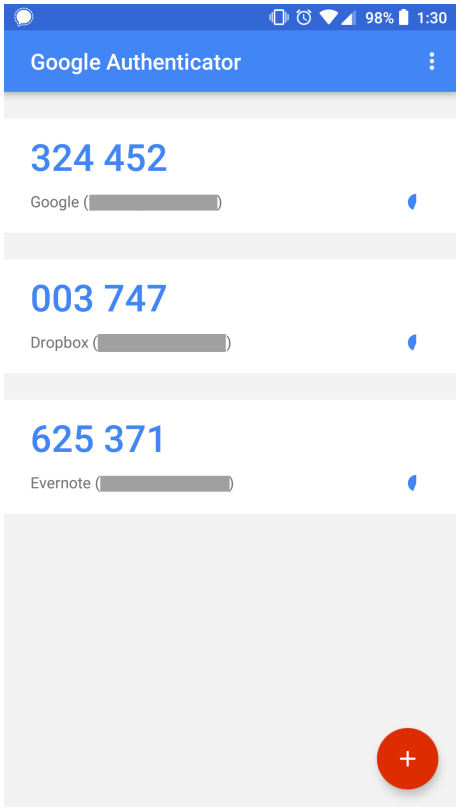
Token-Based Authentication

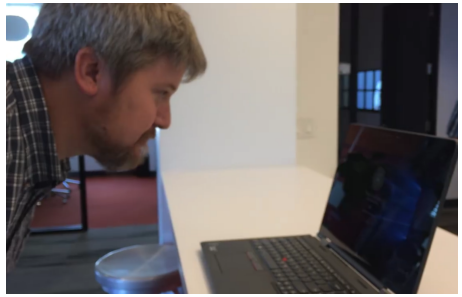
- Memory tokens: Can store but not process data (very outdated)
  - Commonly a magnetic stripe card (sometimes combined with a PIN)

- Smart tokens:
  - Hardware tokens: Include an embedded microprocessor, and may also have a user interface (e.g., keypad or display)
    - Common examples: Smart cards, USB tokens (e.g., Yubikey)
  - Software tokens: Most commonly a smartphone app
  - Use an authentication protocol
    - Two-stage authentication: User authenticates with token, token authenticates to server; e.g., W3C WebAuthn** (UAF protocol) implementations such as Windows Hello
    - One-time password (OTP) generator: Token generates single-use passcodes; e.g., SMS OTP*, HMAC-based OTP, time-based OTP (see Yubikey)
    - Challenge-response: Computer generates a random challenge string, token generates a signed response; e.g., W3C WebAuthn** (U2F protocol)

*SMS OTP is problematic, e.g., see recent news `https://arstechnica.com/information-technology/2018/08/password-breach-teaches-reddit-that-yes-phone-based-2fa-is-that-bad/` **Based on FIDO 2.0 UAF and U2F standards

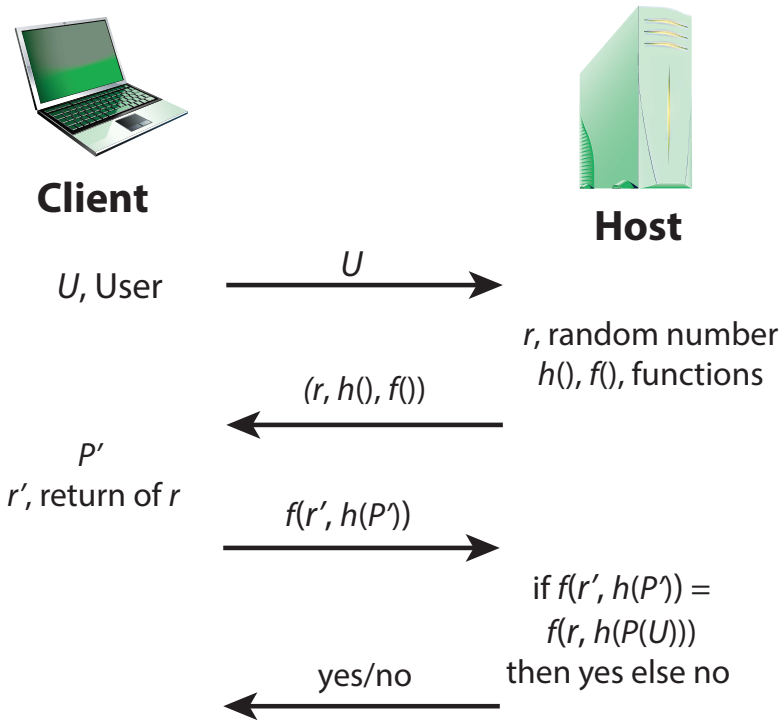
Token-Based Authentication (2)





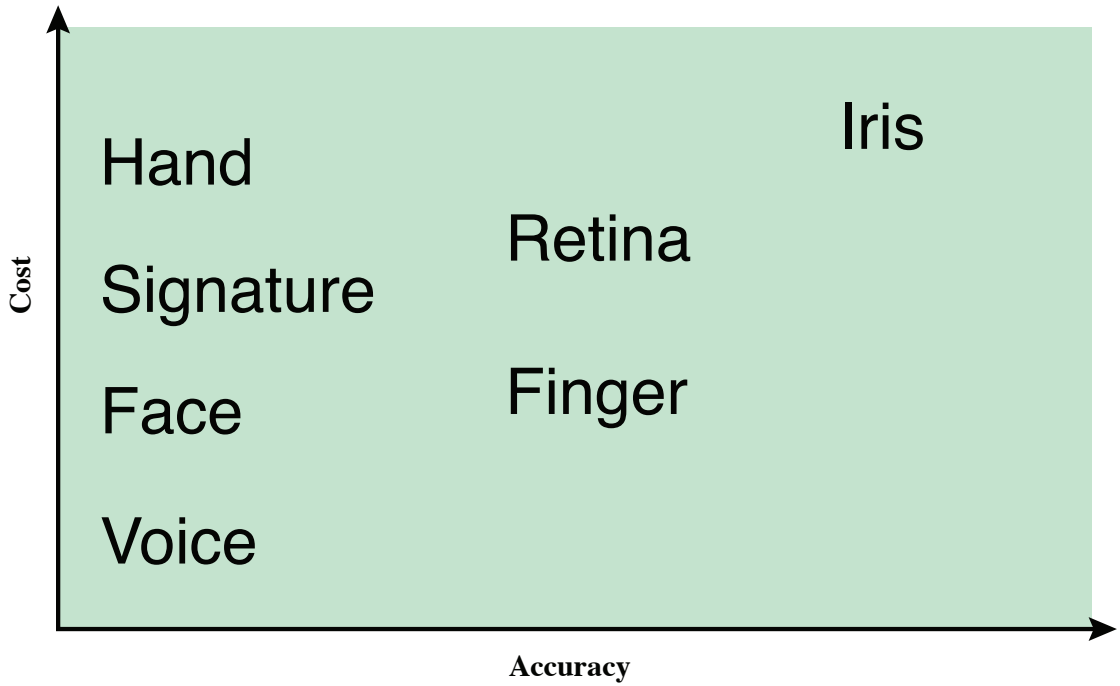
Remote User Authentication

- Authentication over a network is more complex than local authentication
- Additional threats: Eavesdropping or replay attacks
- A **challenge-response protocol** can be used to protect against replay attacks
  - Example on the right is a simple example for passwords
  - Similar technique can be applied for biometrics
  - Only a limited defense: If password or biometric is captured through other means (e.g., shoulder surfing, lifting fingerprint with gel), all bets are off

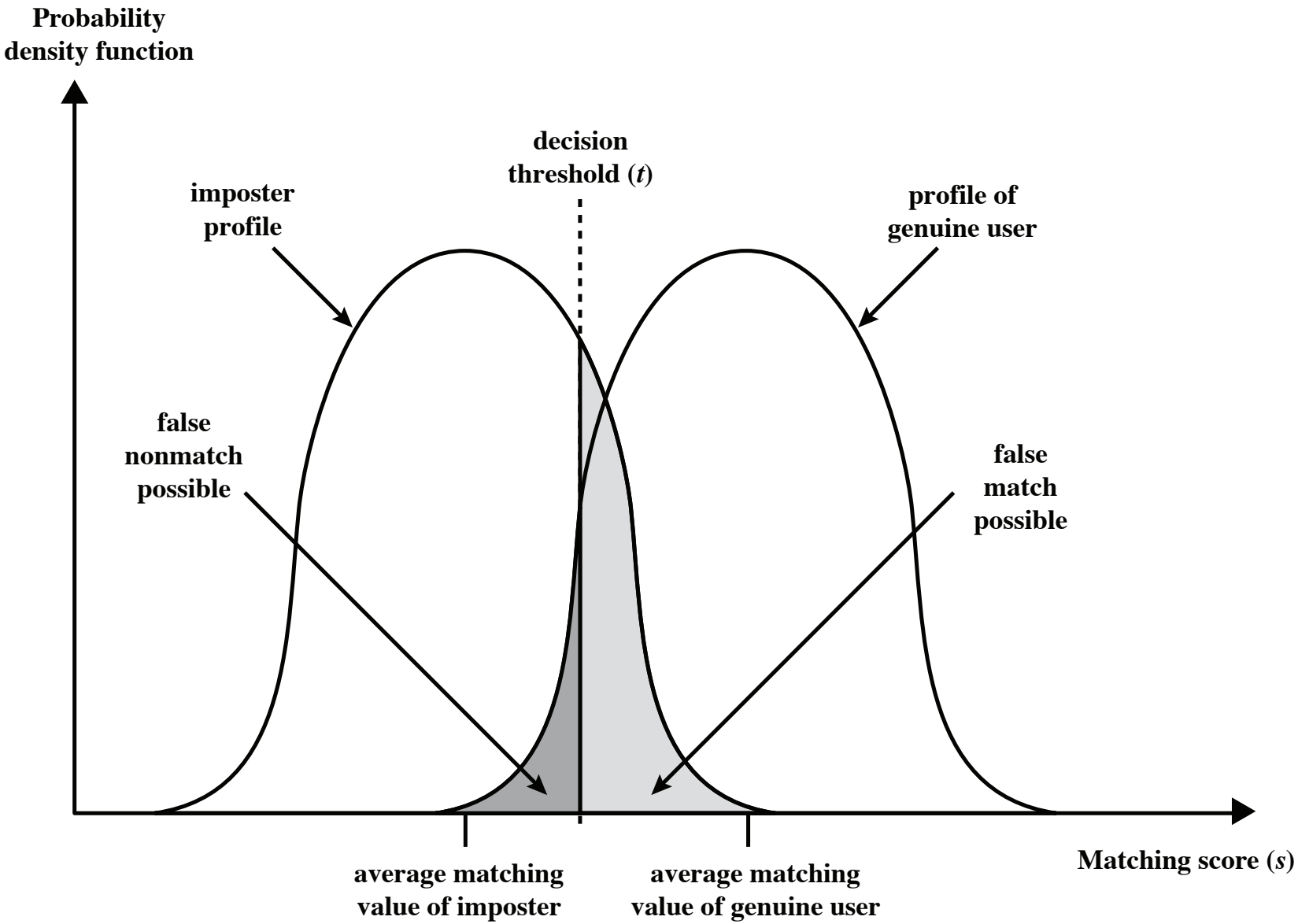


Biometric Authentication

- Attempts to authenticate individual based on unique physical characteristics
- Based on pattern recognition: possible false/positives negatives
  - e.g., fingerprint reader: sensor noise, changes in print due to swelling/dryness, finger placement
- Discussion: How suitable are biometrics for remote authentication?



Biometric Authentication (2)



Biometric Authentication (3)

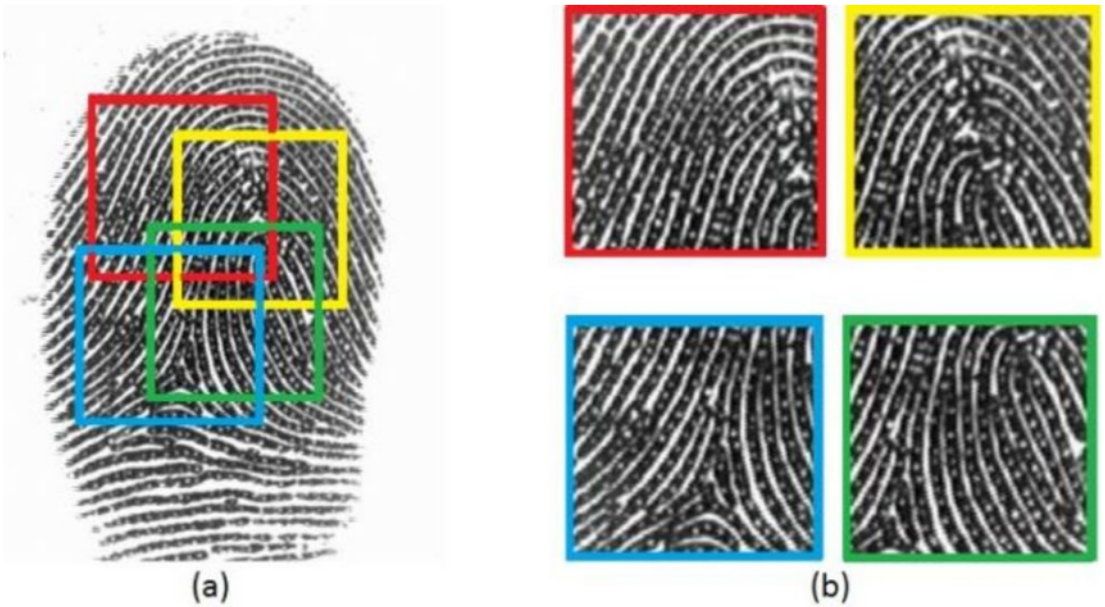
POPULAR SCIENCE

TECHNOLOGY

Computer scientists are developing a 'master' fingerprint that could unlock your phone

But questions remain

By Rob Verger *April 13, 2017*



Just how unique are partial finger prints?

Aditi Roy, Nasir Memon, and Arun Ross

Single Sign-On and Federated Identity Systems

- **Federated identity systems** (FIS) enable **single sign-on** (SSO), whereby a user logs in to their **Identity Provider** (IdP) and gains access to all services provided by **Relying Parties** (RP)
  - The user authenticates with the IdP, using the IdP's authentication mechanism of choice
  - The user presents an **identity assertion** to the RP, as proof that they have been authenticated by a trusted IdP
  - The RP verifies the identity assertion and initiates an authenticated session with the user
- **OpenID Connect** is one of the most popular FIS protocols on the web (e.g., used by Google, Microsoft)
- SSO may also be achieved by **password managers**
  - Most popular design: User's passwords are encrypted client-side using a key derived from a master password, and uploaded to a third-party synchronization service such as Firefox Sync, Google Sync, LastPass (master password is not revealed to the sync service)
- Advantages/disadvantages of both approaches?