School of Electrical Engineering and Computing

SENG2250/6250
SYSTEM AND NETWORK SECURITY
(S2, 2020)

User Authentication

THE UNIVERSITY OF
NEWCASTLE
AUSTRALIA

Outline

- Digital User Authentication
- Authentication Mechanisms
    - *Password based authentication*
    - *Token based authentication*
    - *Biometric based authentication*
    - *Remote authentication*
- Authentication Protocols
    - *Challenge-response*
    - *Mutual authentication*
    - *Anonymous authentication*

User Authentication

- Verify the (claimed) identity of a user, process, or device.
- Basics of access control and user accountability.
- Two Steps
  - *Identification: obtains identity.*
  - *Verification: bind (check) the binding of user and the identity.*

Authentication

- Authentication is based on something you know, are or have.
- Something user knows
  - *Password and PIN number.*
- Something user is
  - *Physical characteristic, biometrics.*
- Something user has
  - *Identity badges, physical keys, driver’s licence.*

Authentication vs. Identification

- Identity
  - *Often to be well known, predictable or guessable*
  - *E.g., email address, account name*
- Identification
  - *Is the act of asserting who a person is.*
  - *Showing or claiming your identifier, but not necessarily to be proved.*
- Authentication
  - *Is the act of proving that asserted identity.*
  - *Confirming an identity via some information you know, is or have.*

Password Based Authentication

- Password
  - *A secret string of characters used in authentication process to confirm the claimed identity.*
- How?
  - *User chooses a password and associate it with the identity.*
  - *System checks the pair of (identity, password) to authenticate user.*
- Issues?
  - *How to generate a good password?*
  - *How does system verify password?*
  - *How the password to be stored?*

Password Vulnerabilities (1)

- Exhaustive Key Search (brute-force)
  - *Attempt every possible combination of password characters.*
  - *Increasing length of password will increase the expected time in exponentially.*
- Dictionary Attacks
  - *Using password dictionary*
    - People’s name, ordinary words, address
  - *Online*
  - *Offline*
    - Much faster than online.

Password Vulnerabilities (2)

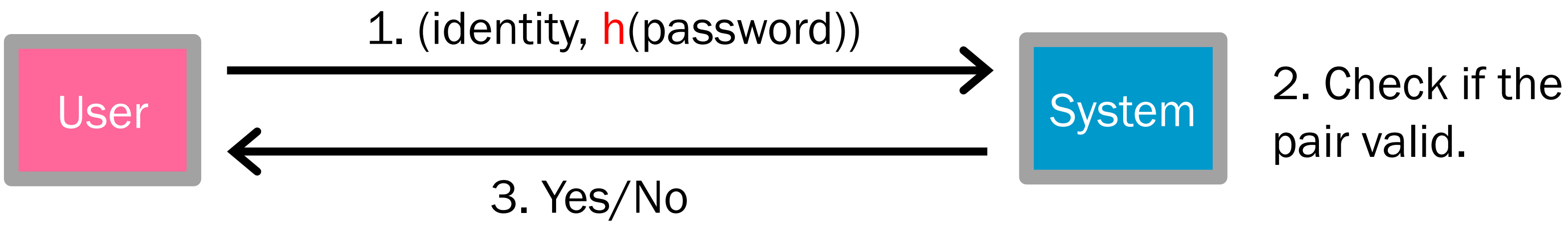
- Human chosen password is usually not completely at random.
- Meaningful information can be used to infer highly likely password.
- Attack could be succeeded in a second due to small amount of such candidates.
- Every password can be guessed.
- Password strength is determined by how many guessing attempts are needed.

Countermeasures

- Encrypted network links
- Prevent unauthorised access to password file
- Intrusion detection measures
- Password strength checking
- Training and enforcement of policies

Password Based Authentication Protocols

- Registration
  - *User creates a password.*
  - *System stores (identity, h(password)) pair.*
  - *How to store password?*
- Verification



Hashed Password

Plain password

Identity	Password
Alice	sdK.)L?9cD31
Bob	JIJoijf092fj
Coral	Kjasf2$3jf,09jf

Hashed password (MD5)

Identity	Password
Alice	511d61dce56e8dfbe8df2782a89d6798
Bob	4c340a777a6593f7d580a45291ae80d3
Coral	1ddd575045fd387fd356c1593c2ef73e

Hashed Password

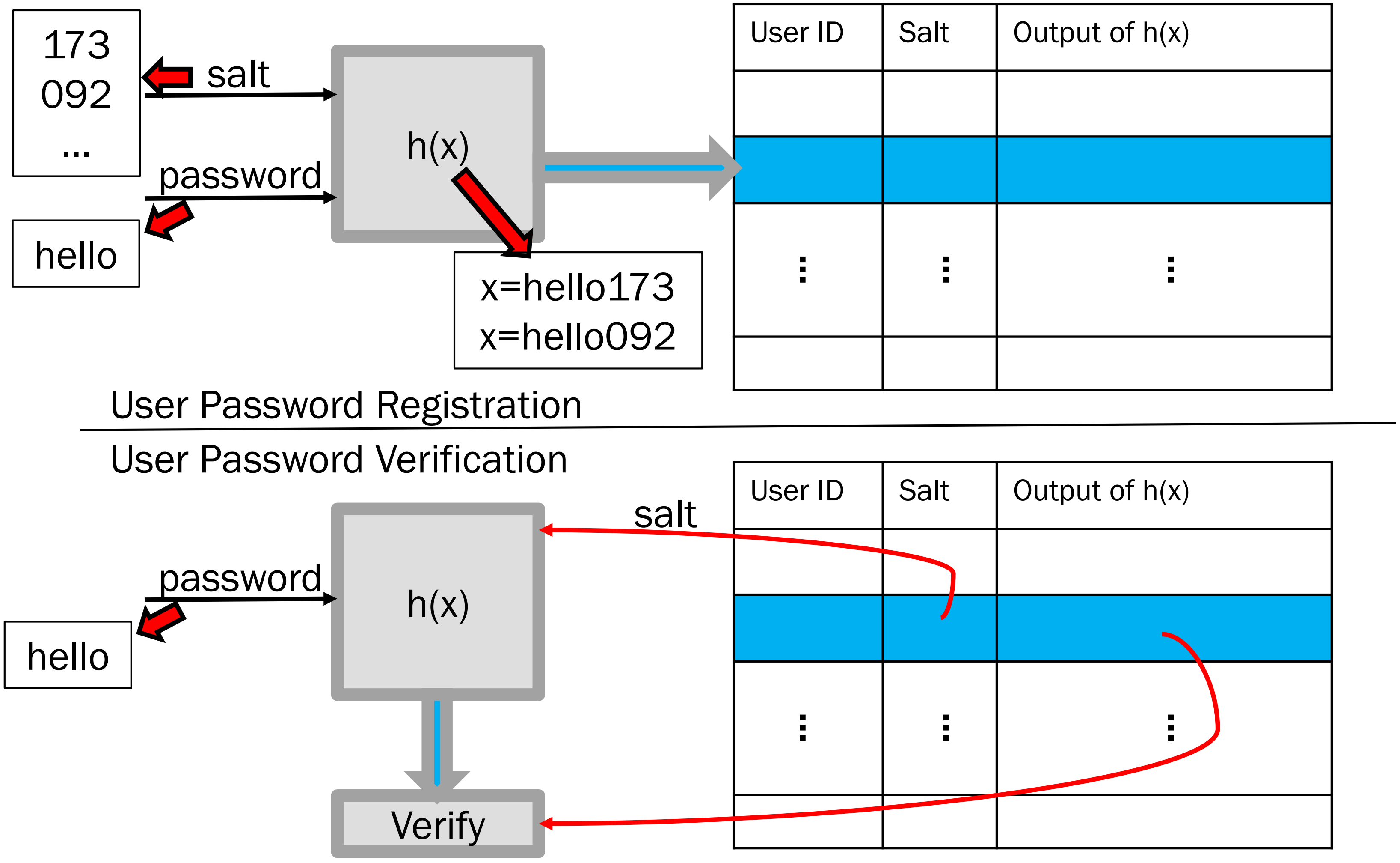
- The hash function must be one-way.
- Password is hidden even if insider adversary gains access to password file.
  - *You can only reset password but not retrieve.*
- Same password result in same hash value.
  - *Rainbow table (attack): precomputed list of popular values.*
- Vulnerable to offline attacks if the hashed password file and the underlying hash function were obtained.

Salt

- Salt is a user-specific (random) component joined to an hashed password to distinguish identical password.
- Why?
  - *Prevent duplicate password being shown in password file.*
  - *Mitigate offline dictionary attacks.*
- Can be any length, but typically 12-bit, 48-bit …

H(password||salt) = hash_pwd

Salt - Example



173
092
…

salt

password

hello

h(x)

x=hello173
x=hello092

User ID	Salt	Output of h(x)
⋮	⋮	⋮

User Password Registration

User Password Verification

password

hello

h(x)

salt

Verify

User ID	Salt	Output of h(x)
⋮	⋮	⋮

Good Password – Tips

- Choose long passwords, at least 8 characters.
- Use special characters to enlarge key space.
- Avoid actual names or words.
- Easy to remember, hard to guess.
- Change password regularly.
- Do not write it down.
- Do not disclose it to anyone.

Difficulties of Using Password

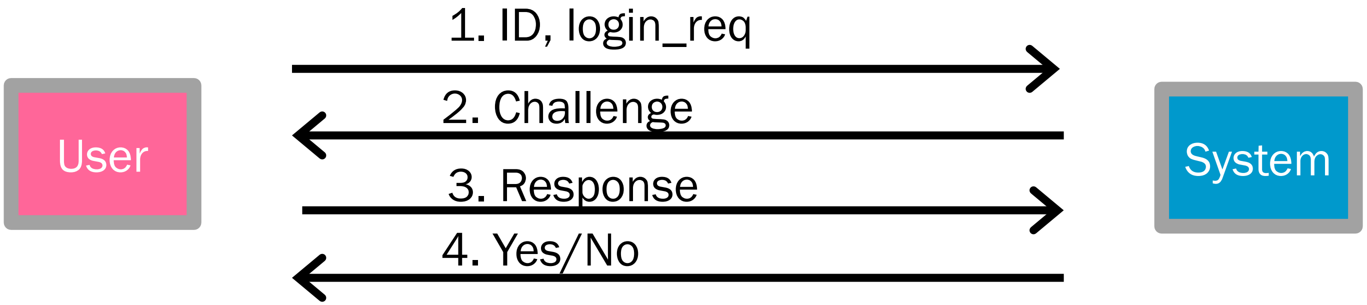
- Use
  - *Supplying a password for each access to an object can be inconvenient and time consuming.*
- Disclosure
  - *A disclosed (to unauthorised party) password will be able to use immediately until the password changed.*
  - *Changes to password must be notified to all users who uses the same account/identifier.*
- User Revocation
  - *Change to a new password, and notify all authorised users.*

One-Time Password

- Password could be disclosed due to variant reasons, such as attacks and accident.
- Disclosed password must be revoked regardless of its strength.
- Use a new password in different log in sessions and not reuse previous password.
- Two types of one-time password
  - *Challenge-response*
  - *Codebook*

One-Time Password

- Challenge-Response Authentication
  - *Something a person has*
  - *Something a person knows*
  - *Challenge: string is different for every login session of each user*
  - *Response: generated by using the predefined algorithms, it may need a secret input.*



One-Time Password

- Codebook
  - *A list of password to be used one at a time, and they are not to be reused.*
  - *User and server share the same codebook.*
  - *A codebook can be a list of password or be generated using the specified algorithm when needed.*
  - *Example*
    - S/KEY (uses hash chain)

Authentication Based on Hash Chains

- Step 1. Choose a cryptographic hash function $h: \{0,1\}^* \to \{0,1\}^\ell$ and a random seed $s \in \{0,1\}^*$.
- Step 2. Server/User computes n times of hash of s, such that

$$H_1 = h(s), H_2 = h(H_1), \dots, H_n = h(H_{n-1})$$

- Step 3. Server discards s (if known) and all hashes $H_1, \dots, H_{n-1}$, while keeps H_n.
- Step 4. User stores $H_n, \dots, H_1$ or H_1 only

Authentication Based on Hash Chains

- Verification
  - *If only H_1 is known to user*

    User → Server: $H_i = h^{i-1}(H_1)$

    Server: check if $H_{i+1} = h(H_i)$

  - *If full hash chain is stored, to verify the ith hash value, where $i = 1, \dots, n-1$.*

    User → Server: H_i

    Server: check if $H_{i+1} = h(H_i)$

Authentication Based on Hash Chains

- The maximum number of logins depends on the length of hash chain
- What are the threats to the hash chain based authentication?
  - *DoS attacks.*
  - *Security of hash functions.*

Token Based Authentication

- Token: Something user has
  - *A physical object which contains information bound to an identity for authentication purpose.*
  - *E.g., smart card, secure token.*
- Passive Token
- Active Token

Passive Token

- Stores data but not process it
- Cannot do actions.
- For example, memory card
- May need special reader
- Loss of token issues
- Some may with password enabled

Active Token

- Has own processor, memory etc.
- Authentication
    - *Static: internal state is static, e.g, store secret.*
    - *Dynamic: password created periodically.*
    - *Challenge-response: run interactive authentication protocol with system which checks the validity of token.*

Example

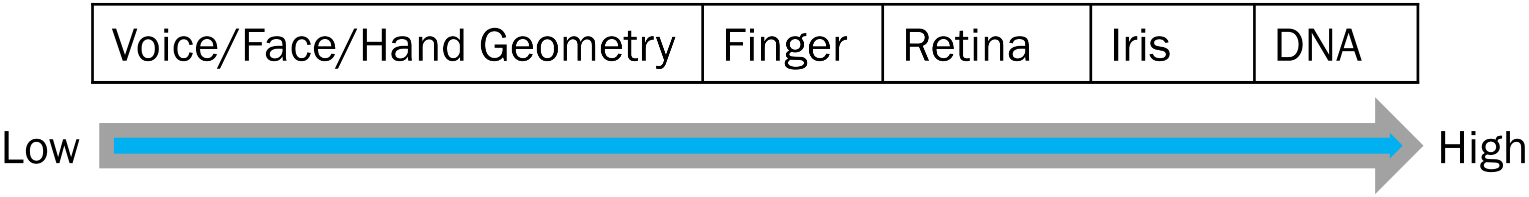
- Time-Based Token Authentication
- RSA SecureID Token

- Token value changes periodically, say 60 seconds.

https://en.wikipedia.org/wiki/RSA_SecurID

Biometric Authentication

- Biometrics: are biological properties, based on some physical characteristic of the human body.
  - *Unique, but may not be true in practice.* ☹
  - *Biometric matches are not exact, we are checking if they are close.*
- General Accuracy of biometrics

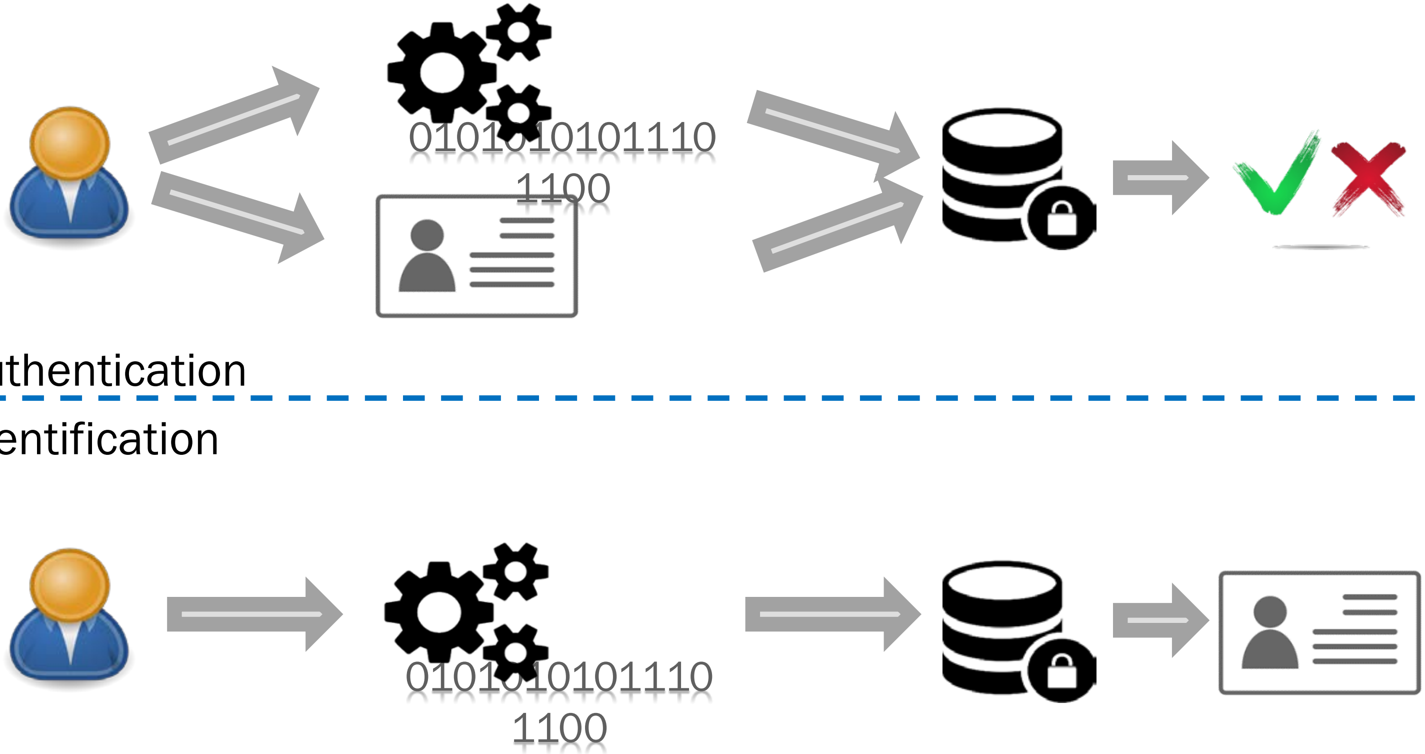


Voice/Face/Hand Geometry	Finger	Retina	Iris	DNA

Biometric Authentication

- Enrolment
  - *User provide his/her biometric information to server.*
  - *Server generates a binding between user's identity and the biometric.*
- Identification Mode
  - *Given an input of biometric information, system outputs the corresponding identity if the biometric is registered.*
- Verification Mode
  - *Given an input of biometric information and the claimed identity, system outputs Yes, if the (biometrics, identity) pair is valid, otherwise, outputs No.*

Biometric Authentication vs. Biometric Identification



Problems of Using Biometrics

- Accuracy
  - *False positive rate: incorrectly confirming an identity/biometric.*
  - *False negative rate: incorrectly rejecting an identity/biometric.*
  - *These rate should be in acceptable range*
- Biometric recognition devices are costly
- Biometric readers and comparisons can become a single point of failure.
  - *People's biometric is hard to change.*

Privacy Issues of Biometrics

- Where is my biometric information stored?
- How is it stored?
- Who can access my information?
- What if the stored data is compromised?

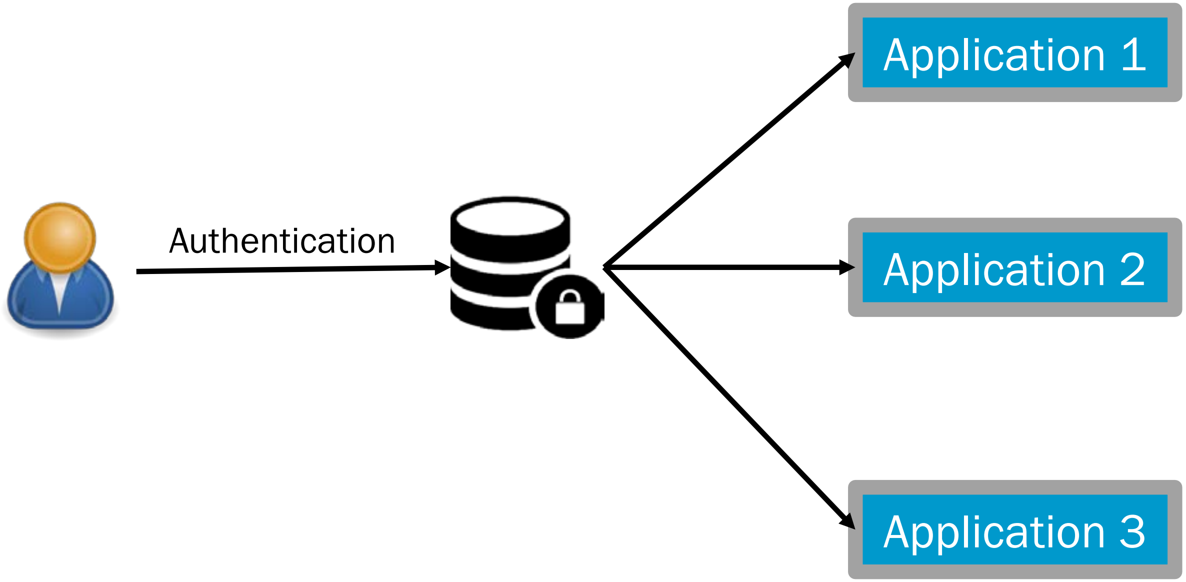
Remote User Authentication

- Authentication over network
    - *More complex issues, e.g eavesdropping, replay and MITM.*
- Normally use challenge-response mechanism.
- Needs to withstand a number of attacks
    - *DoS*
    - *MITM*
    - *…*

Multi-Factor Authentication

- One factor is not sufficient to achieve secure authentication in many applications.
  - *Banking, confidential documentation access.*
- Multi-factor authentication is to combine two or more forms of authentication.
  - *Password + token*
  - *Password + biometrics*
  - *Password + token + biometrics*
  - *…*
- Adversary needs to obtain secret of all factors.

Single Sign-On (SSO)

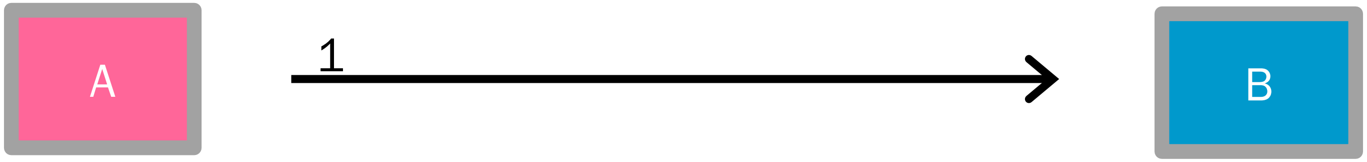


- Single log in authentication for multiple applications and systems.
- No need for authentication at multiple places.

Challenge-Response Authentication Protocols

- Typically used in remote authentication.
- A common technique to resist number of attacks.
  - *E.g, MITM, replay and eavesdropping attacks*
- Mutual Authentication
  - *Both user and server need to prove its identity each other.*

1-Way Authentication

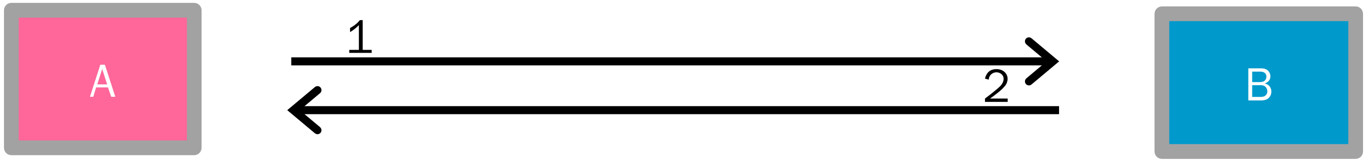


$A \rightarrow B$: nonce_A, timestamp_A, B,

$[K_{AB}]_{PKB}$, $\{[\text{nonce}_A, \text{timestamp}_A, B]_{KAB}\}_{SKA}$

- Authentication of A only
- Requires time synchronisation
- **Replay:** timestamp, nonce

2-Way Authentication

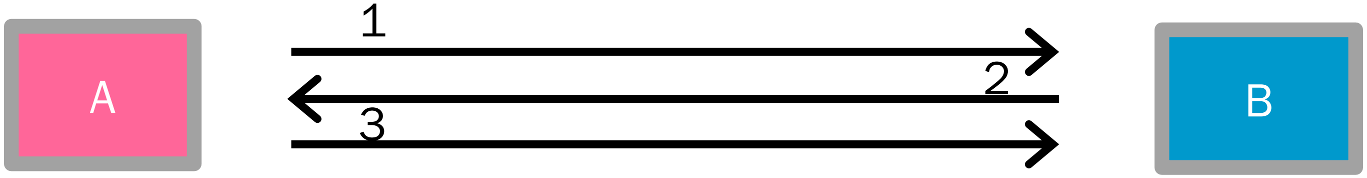


$A \rightarrow B$: nonce_A, timestamp_A, B,
$[K_{AB}]_{PKB}$, $\{[\text{nonce}_A, \text{timestamp}_A, B]_{KAB}\}_{SKA}$

$B \rightarrow A$: nonce_B, nonce_A, timestamp_B, A,
$[K_{BA}]_{PKA}$, $\{[\text{nonce}_B, \text{nonce}_A, \text{timestamp}_B, A]_{KBA}\}_{SKB}$

- Mutual authentication
- Requires time synchronisation

3-Way Authentication



A → B: nonce_A, timestamp_A, B,
$[K_{AB}]_{PKB}$, $\{[\text{nonce}_A, \text{timestamp}_A, B]_{KAB}\}_{SKA}$

B → A: nonce_B, nonce_A, timestamp_B, A,
$[K_{BA}]_{PKA}$, $\{[\text{nonce}_B, \text{nonce}_A, \text{timestamp}_B, A]_{KBA}\}_{SKB}$

A → B: $\{\text{nonce}_B, B\}_{SKA}$

- Mutual authentication
- Can remove time synchronisation

Security Issues on Authentication

- **Client attacks**: attacker attempts to achieve user authentication without access to the remote host
  - *Masquerade as a legitimate user (e.g., guess the password or try all passwords)*
  - *Countermeasure: strong passwords; limit number of attempts*
- **Host attacks**: attacker attacks the host where passwords/passcodes are stored
  - *Countermeasure: hashing, protect password databases*

Security Issues on Authentication

- **Eavesdropping**: attacker attempts to learn passwords by observing the user, finding written passwords, keylogging
  - *Countermeasures*
    - diligence to keep passwords
    - multifactor authentication
    - admin revoke compromised passwords

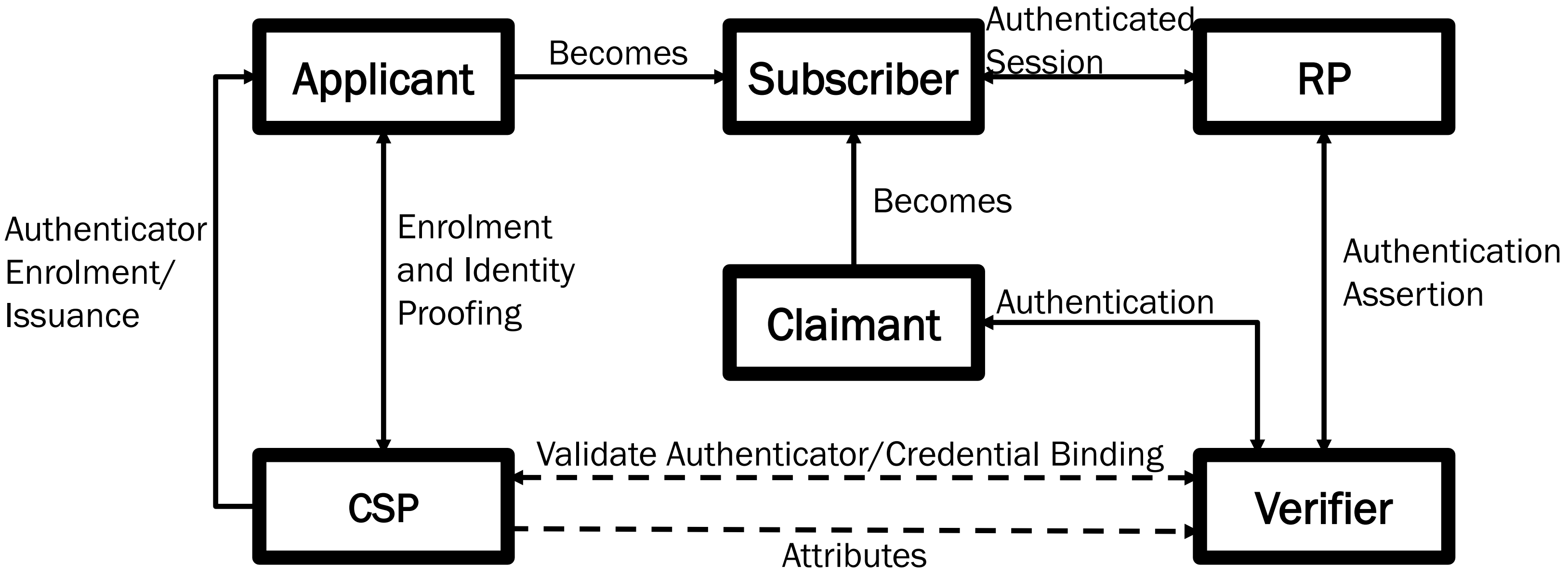
Security Issues on Authentication

- **Replay:** attacker repeats a previously captured user response
  - *Countermeasure*
    - Challenge-response
    - 1-time passcodes
- **Denial of service:** attacker attempts to disable a user authentication service (via flooding)
  - *Countermeasure: a multifactor authentication with a token*

A Model of Digital User Authentication: NIST SP 800-63-3

- Digital Authentication: The process of establishing confidence in user identities presented digitally to a system.
- Digital Identity Model consists of:
  - ***Applicant:*** *undergoing process of enrolment and identity proofing.*
  - ***Credential Service Provider (CSP)****: a trusted party registers/issues subscriber authenticators and credentials.*
  - ***Claimant****: a subject to be authenticated.*
  - ***Subscriber****: receives credentials/authenticators from CSP.*
  - ***Relying Party (RP)****: process assertion about subscriber.*
  - ***Verifier****: a party verifies.*

A Model of Digital User Authentication: NIST SP 800-63-3



References

- NIST SP 800-63. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-3.pdf