4.2 Authentication

Name: ______________ Class: __________ Date: ____________ **Total: 48 marks**

KEY WORDS hash 散列值 • hashing 哈希 • cryptographic hash function 密码散列函数 • authentication factors 认证要素 • password attacks 密码攻击

Objective

Build the skills to answer exam questions on **hashing 哈希** for password storage, **password attacks 密码攻击**, **authentication factors 认证要素**, and secure login settings.

You must be able to:

- state the four properties of a cryptographic hash function and say why each matters
- explain why passwords are hashed and **salted 加盐** rather than stored in plaintext
- distinguish online from offline password attacks, and name spraying, stuffing and rainbow tables
- classify an authentication factor as knowledge, possession, biometric or location
- explain why **MFA 多因素认证** is stronger, and configure the five login settings

1 Worked examples

Study these first. Each one shows the method for a task used later.

■ What a hash is, and its four properties

A **cryptographic hash function** takes binary data of any length and produces a fixed-length output - the hash, digest, message digest or checksum.

Property	What it means	Why it matters
Collision resistant	hard to find two inputs with the same output	a forced collision would let a different password authenticate
Pre-image resistant	given a hash, infeasible to work out the input	a stolen hash file does not directly hand over passwords
Repeatable	the same input always gives the same output	this is what makes a login check - and a file integrity check - possible
Fixed length	the output length is constant whatever the input	a 20-character and a 20-MB input hash to the same size

Collisions must exist. An n-bit hash has 2^n possible outputs but infinitely many possible inputs, so two inputs somewhere share an output. The security claim is not that collisions are impossible - only that finding one is infeasible. If an efficient algorithm to force a collision is discovered, that hash function is considered broken.

Why hashing, and why salting on top

Storing plaintext passwords means one breach of the user:password directory hands the adversary every password. Storing hashes means pre-image resistance stands between them and the passwords.

But hashing alone leaks something: **two users with the same password produce identical hashes**, so the directory itself reveals who shares a password, and one cracked hash unlocks all of them. A **salt** - a few random bits added to each password before hashing - makes the two hashes different, so each password must be attacked separately and a precomputed table is useless.

Online or offline

	Online	**Offline**
Where	against **your** system	on the **adversary's** computer, using a captured hash
Detectable	yes - in the authentication log	**no**
Limited by	lockout, rate limiting, MFA	only the adversary's processing power

Named online attacks: **guessing** common passwords for one account; **password spraying** - one common password tried across many accounts, which evades a per-account lockout; **credential stuffing** - default `user:password` pairs, which works because switches, routers and IoT devices ship preconfigured with them.

Offline: **hash-cracking tools** hash candidate passwords and compare against the captured hash - hashes cannot be reversed, so the attack works forwards. A **rainbow table** precomputes each common password with its hash so the lookup is instant; salting is exactly what defeats it.

Password reuse is what turns one breach into many: when one organisation is breached, the recovered passwords are tried against the user's other services.

The four authentication factors

Factor	**The proof**	**Effective when**
Knowledge	password, PIN, challenge question	an adversary could not learn or guess it - so not a pet's name
Possession	access card, bank card, phone, token	the object is hard to duplicate
Biometric	fingerprint, palm, face, iris, retina, voice	the measurement is reliable
Location	Wi-Fi, GPS, time zone, IP address	rules can allow or deny by where the request came from

MFA uses **more than one factor** - and the factors must be *different kinds*. A password plus a security question is still one factor twice, and an adversary who has learned one has probably learned both.

■ **The five login settings**

- **Complexity** - a new password must include characters from several character sets.
- **Minimum length** - at least n characters.
- **Maximum password age** - the user is prompted to change it after n days.
- **Password history** - the system stores previous passwords so they cannot be reused.
- **Lockout** - after n invalid attempts, further attempts are blocked, which stops continuous online guessing.

Lockout is the one that directly defeats online guessing - and the one spraying is designed to slip under, by trying only one password per account.

2 Practice

2.1 State the four properties of a cryptographic hash function. [4]

2.2 Explain why collisions must exist for any hash function, and why the function can still be considered secure. [3]

2.3 Explain why a password-based service stores hashes rather than plaintext passwords. [2]

2.4 Two users choose the same password.

(a) State what their stored hashes would look like without salting. [1]

(b) Explain how a salt fixes this and what attack it defeats. [2]

2.5 State whether each attack is online or offline, and whether it can be detected.

(a) Password spraying. [2]

(b) A rainbow table attack on a captured hash file. [2]

2.6 Explain why credential stuffing works so often against routers and IoT devices. [2]

2.7 Classify each as a knowledge, possession, biometric or location factor.

(a) A fingerprint. [1]

(b) A code from an authentication app on a phone. [1]

(c) A PIN. [1]

(d) A rule denying logins from outside the country. [1]

2.8 Explain why a password plus a challenge question is **not** true multifactor authentication. [2]

2.9 Name the login setting that addresses each problem.

(a) A user keeps switching between the same two passwords. [1]

(b) An adversary is trying thousands of passwords against one account. [1]

(c) A user's password is "password". [1]

(d) A password has not changed in six years. [1]

3 Exam-style questions

3.1 An n-bit hash has how many possible outputs? [1]

A n
B $2n$
C 2^n
D infinitely many

3.2 Salting a password before hashing primarily prevents [1]

A an adversary reversing the hash directly
B identical passwords producing identical hashes
C the hash output varying in length
D an online lockout being triggered

3.3 Which attack cannot be detected in an organisation's logs? [1]

A password spraying
B credential stuffing
C online guessing of one account
D an offline hash-cracking attack

3.4 Password spraying is designed specifically to evade [1]

A B
C D

A password complexity requirements
B account lockout after invalid attempts
C biometric authentication
D maximum password age

3.5 A college stores staff passwords as unsalted hashes. Login settings require eight characters and nothing else: no complexity, no history, no maximum age, no lockout. An adversary copies the user:password directory. Two weeks later, forty staff accounts are accessed successfully; the authentication logs show each of those logins succeeding on the first attempt. Several of the compromised accounts are also used at an external journal service with the same password.

(a) Name the type of password attack the adversary most likely used on the copied file, and explain why the college's logs show no failed attempts. [3]

(b) Explain how the absence of salting made this attack faster, referring to rainbow tables. [3]

(c) The IT team argues that the eight-character minimum was "reasonable". Evaluate this, referring to keyspace and to processing power. [3]

(d) Explain how the external journal service is now at risk even though it was never breached. [2]

(e) Recommend **four** changes - at least one login setting, one storage change and one authentication change - and explain which single change would have done most to prevent this incident. [5]