

17 Asymmetric encryption and hashing – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary

Write each word three times.

English	Write it 3 times (English)		
Asymmetric encryption	_____	_____	_____
public key	_____	_____	_____
private key	_____	_____	_____
cryptographic hash	_____	_____	_____
digest	_____	_____	_____

Recall

Symmetric encryption struggles to share its key. Asymmetric encryption solves that with a pair of keys.

New ideas

Read these first. Each idea has a short worked example. Then do the Practice.

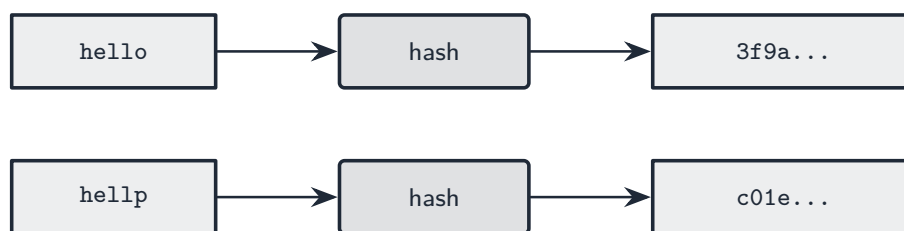
■ 1 Asymmetric encryption



Asymmetric encryption gives each user a **public key** (shared) and a **private key** (secret). Anyone can lock with the public key, but only the owner's private key unlocks – so no key needs sharing first.

Worked example. Alice publishes her public key for all to see. Bob locks a message with Alice's public key and sends it. Only Alice's private key can unlock it – so they never had to share a secret key beforehand.

■ 2 Hashing



one letter changed → a totally different digest (cannot be reversed)

A **cryptographic hash** turns any input into a fixed-size **digest**. It is one-way (you cannot get the input back) and a tiny change flips the digest completely.

Worked example. A website stores the *hash* of your password, not the password. At login it hashes what you typed and compares. Because hashing is one-way, a thief who steals the stored hashes still cannot read the passwords.

Watch out: a hash is *not* encryption – it has no key and cannot be reversed. And in asymmetric encryption the public key only *locks*, the private key only *unlocks* – don't swap their jobs.

Practice

Try these in order.

17.1 How many keys does each user have in *asymmetric* encryption? [1]

17.2 Which key is shared, and which is kept secret? [1]

17.3 If a message is locked with someone's *public* key, whose key can unlock it? [1]

17.4 Can a *hash* be reversed to get the input back? [1]

17.5 What happens to the digest if you change one letter of the input? [1]

17.6 Explain why a website stores the *hash* of your password rather than the password itself. [2]

17.7 Challenge. Asymmetric encryption removes the key-sharing problem, yet symmetric encryption is still widely used for bulk data. Suggest why. [2]
