Secure Software Development Design Principles – Part 1

Objectives

- Explain why data and code comingling can cause problems
- Compare and contrast asymmetric and symmetric cryptography design.
- Explain the concept of a certificate
- Explain the usage of a hash code to verify the integrity of data.
- Compare and contrast a salted and unsalted hash key.

- What makes for good design principles?

Interactive Time

Modularity
Separation of concerns
Reuse SE 2811
Notes

Discussion

- What is fundamentally insecure about e-mail?

Not designed for encryption

Easy to spoof

Easy to deliver programs as "Attachments"

MSOE

Don't mix code and data

- 1985 Lotus 1-2-3 added Macro functions to a spreadsheet
  - Now a spreadsheet holds both data and user written code

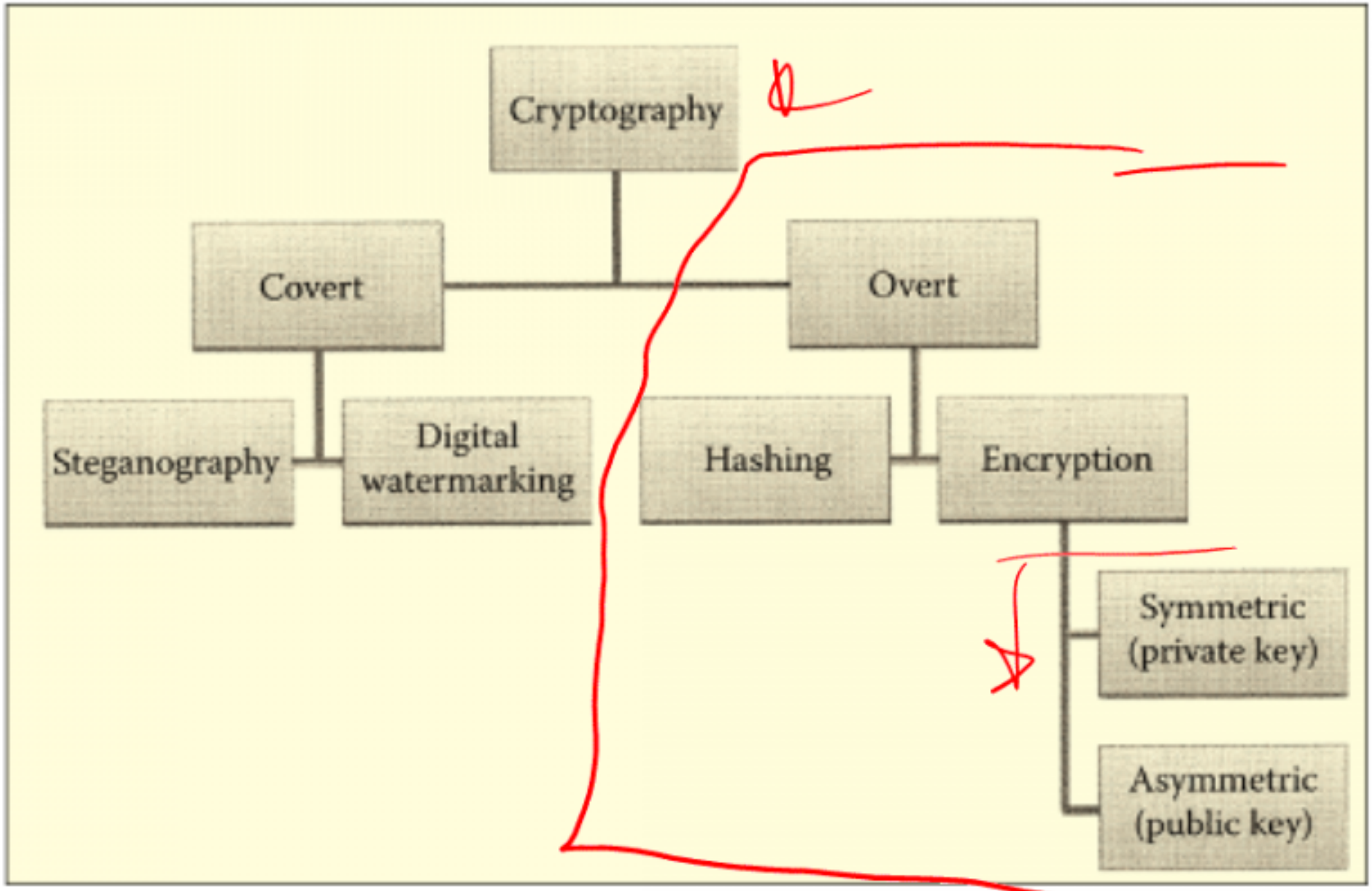
How do we protect?

- By default, DO NOT RUN CODE intermixed with data

⇒ Make the user do something

⇒ Warn them if they are doing something bad

Confidentiality Design



- Cryptanalysis
  - The science of finding vulnerabilities in cryptographic protection mechanisms

What is important to cryptography?

Not easily reversable

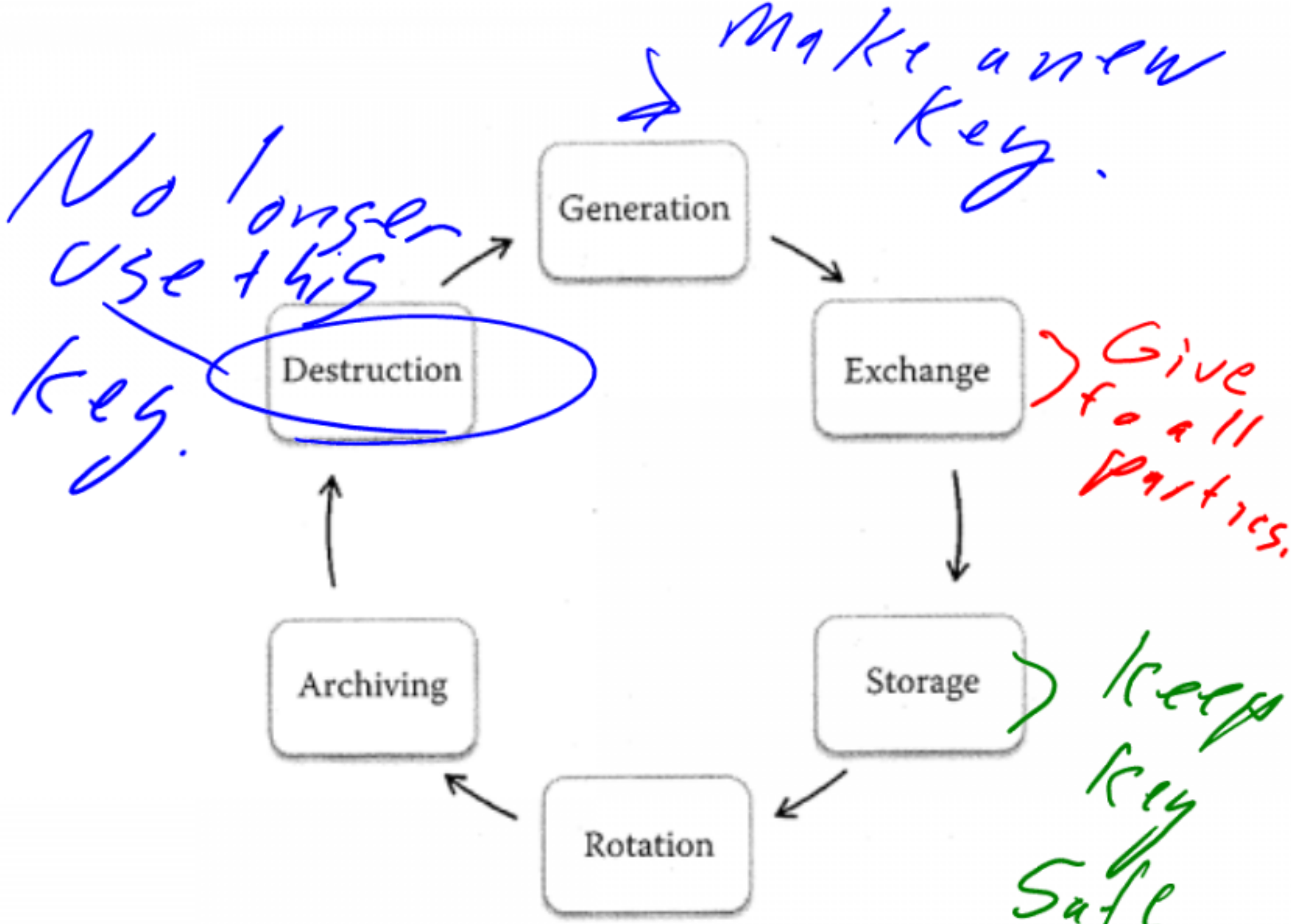
⇒ High performance

⇒ (and by CPU intensive)

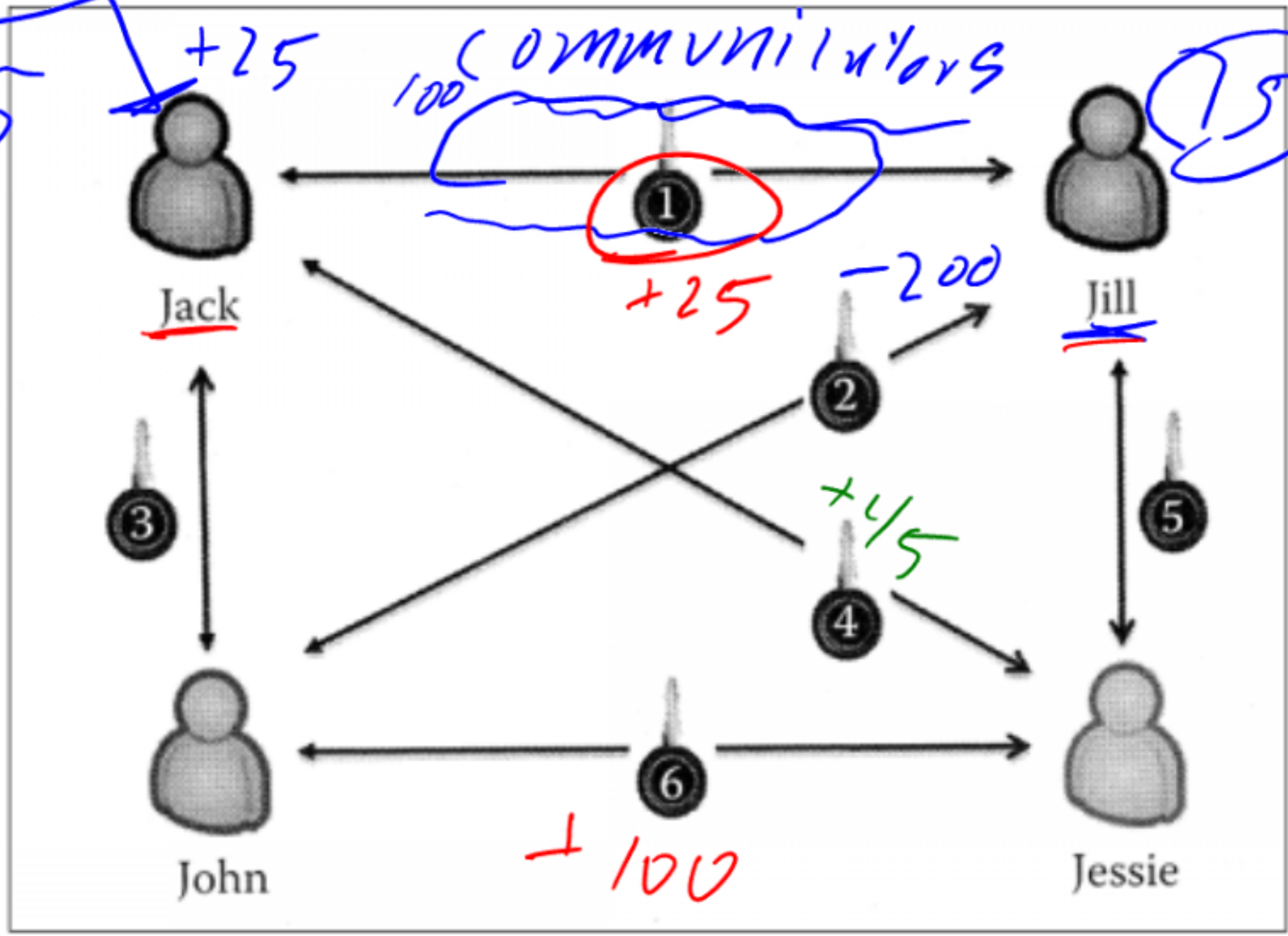
⇒ Computational difficulty to break

⇒ keeping the Key Safe

Key Management



Symmetric Algorithms



Add 25 to the number.

Symmetric Key Challenges

- Key Exchange and Management

⇒ Have to deliver the keys to all parties

- Scalability

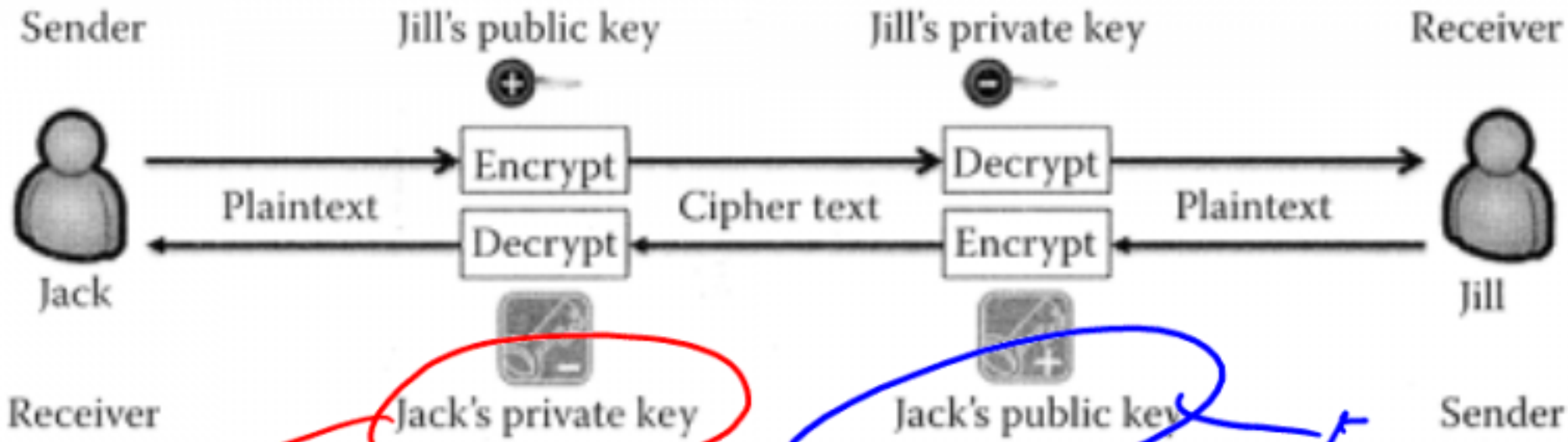
$\frac{n(n-1)}{2}$ keys to manage

- Nonrepudiation not addressed

If the data is changed midstream, we would not know.

MSOE

Asymmetric Encryption



very secure 2 keys so leaf everyone knows

⇒ one of which is private

⇒ one of which is public

Asymmetric Advantages

- Key exchange and management
- Scalability
- Nonrepudiation addressed

RSA

SSL

PGP

Mechanisms

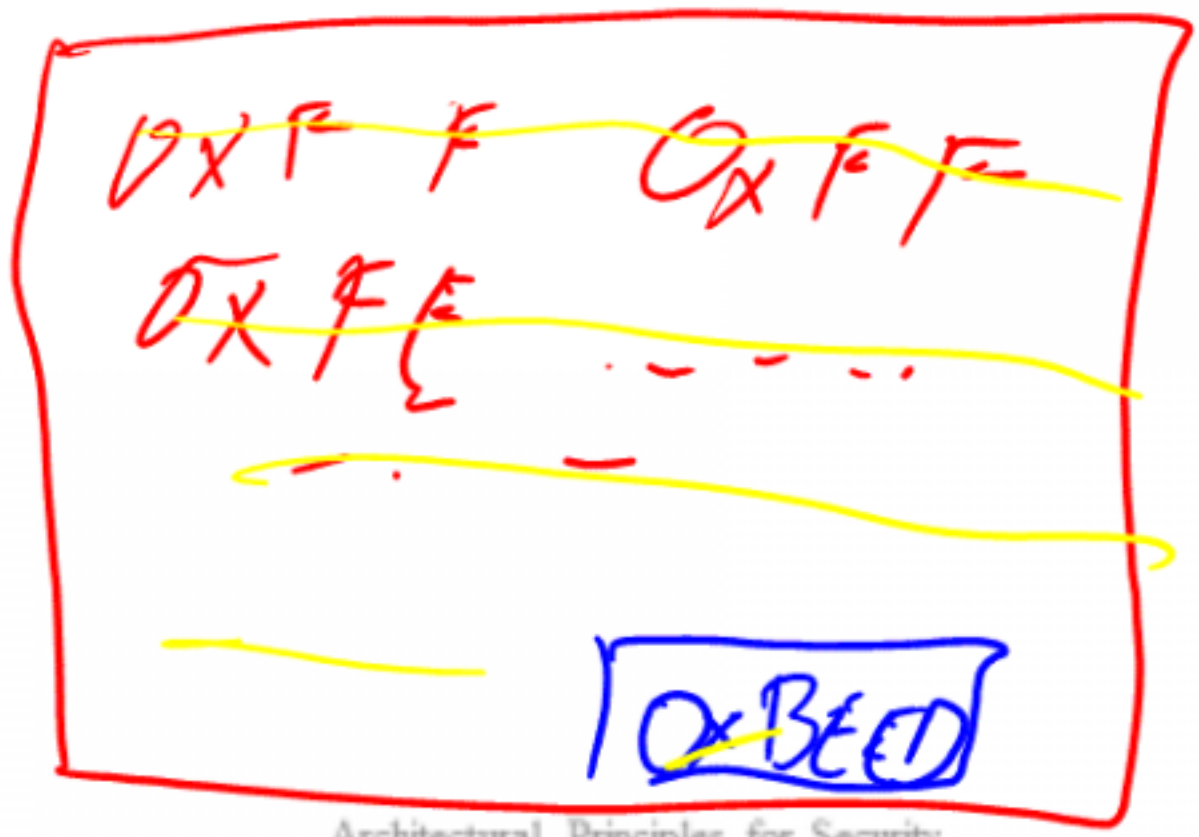
Certificates

Digital Certificates

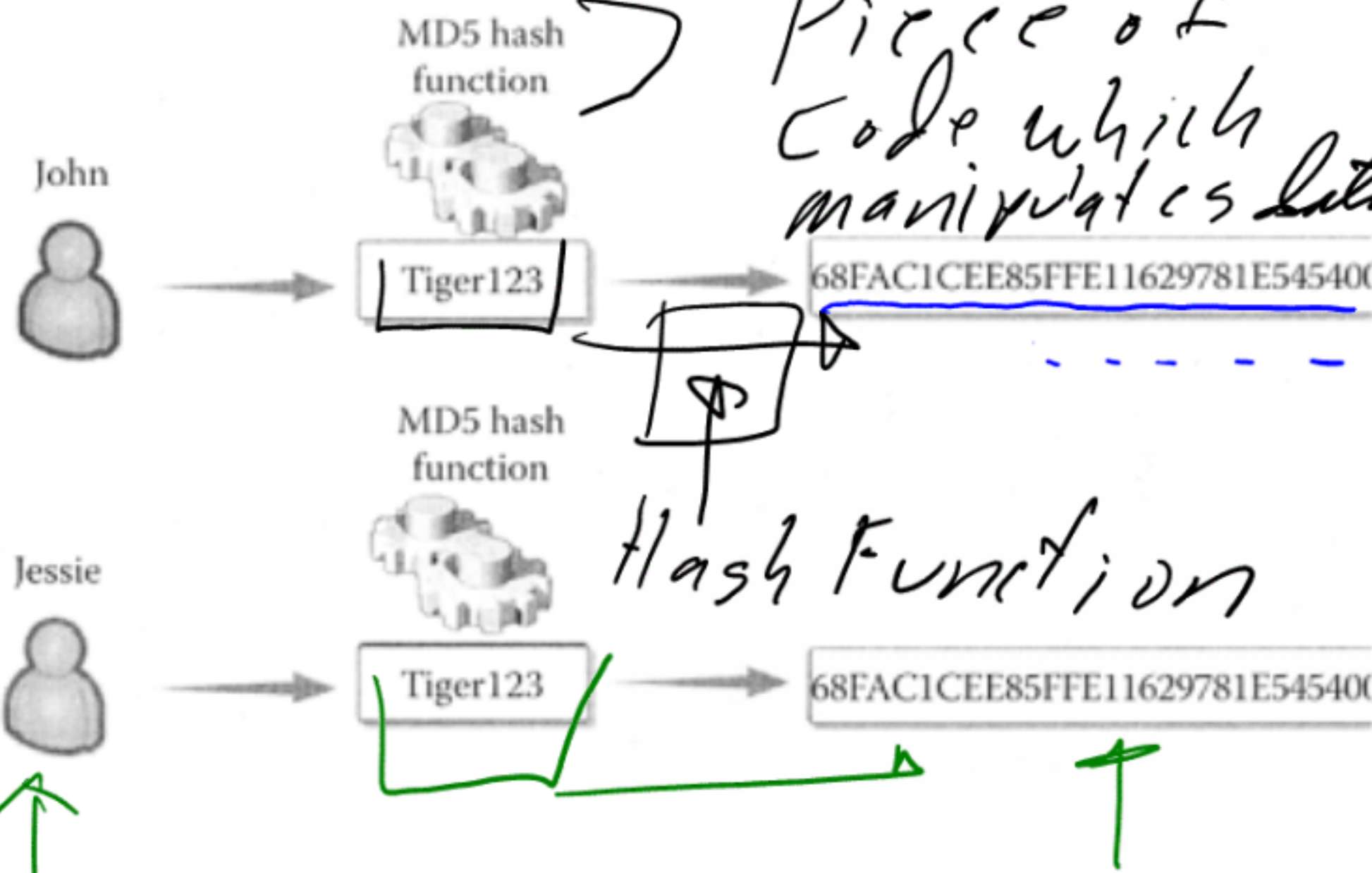
- Certificates carry
  - Public key
  - Algorithms information
  - Owner and subject data
  - Certification authority
  - Validity Period
- Types of certificates
  - Personal
  - Server
  - Software Publisher Certificates

- How do we determine if data has been changed?

checksum @ end of file



Hashing

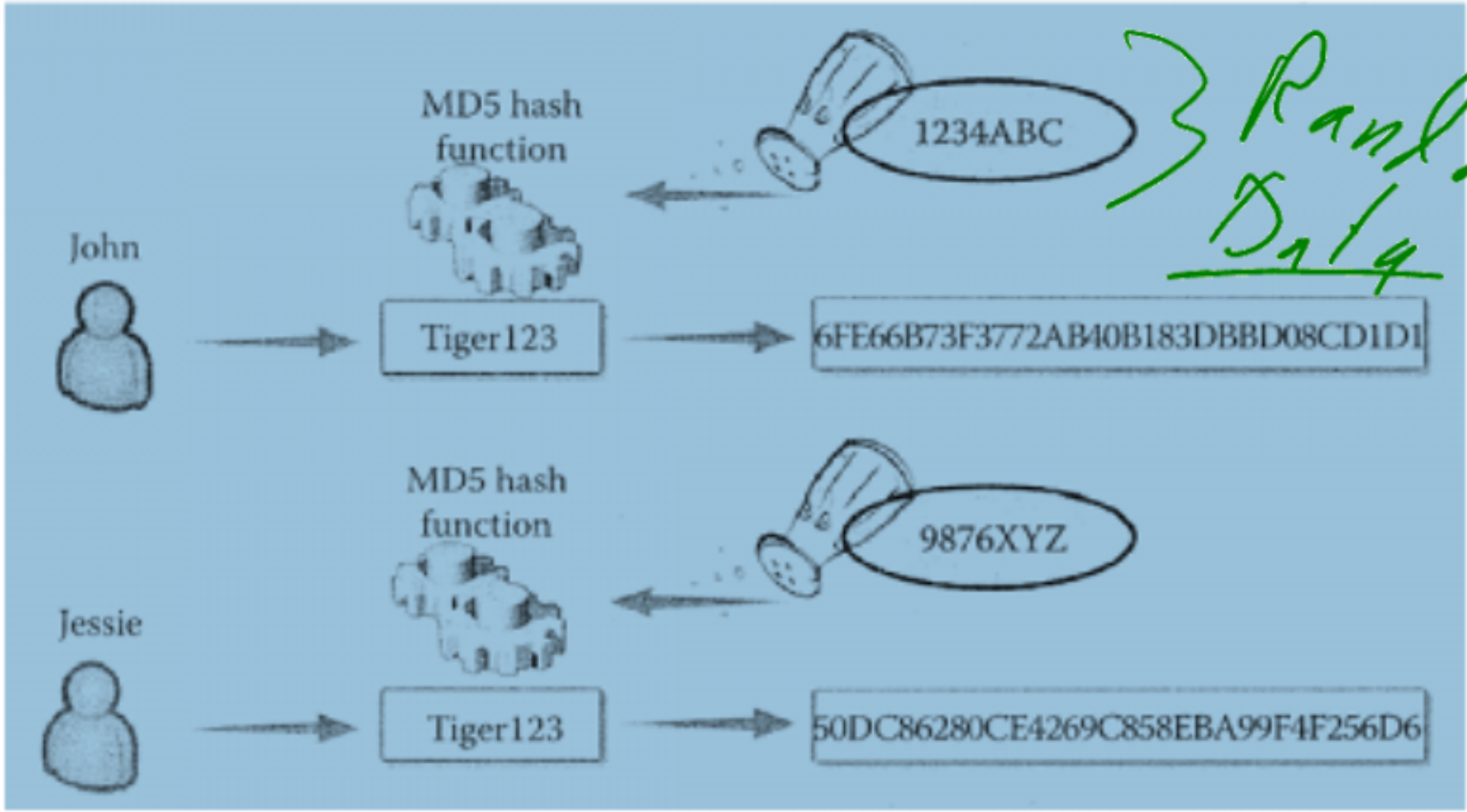


Piece of code which manipulates data.

Hash function

User ID's

Salting the hash



Random Salt

- How do we ensure that the code we use is correct?

Code Signing

MD5 Hash

⇒ File accidentally changing.

⇒ CodeSigning valid.

⇒ "Certificate for checking