Theory of Blockchain

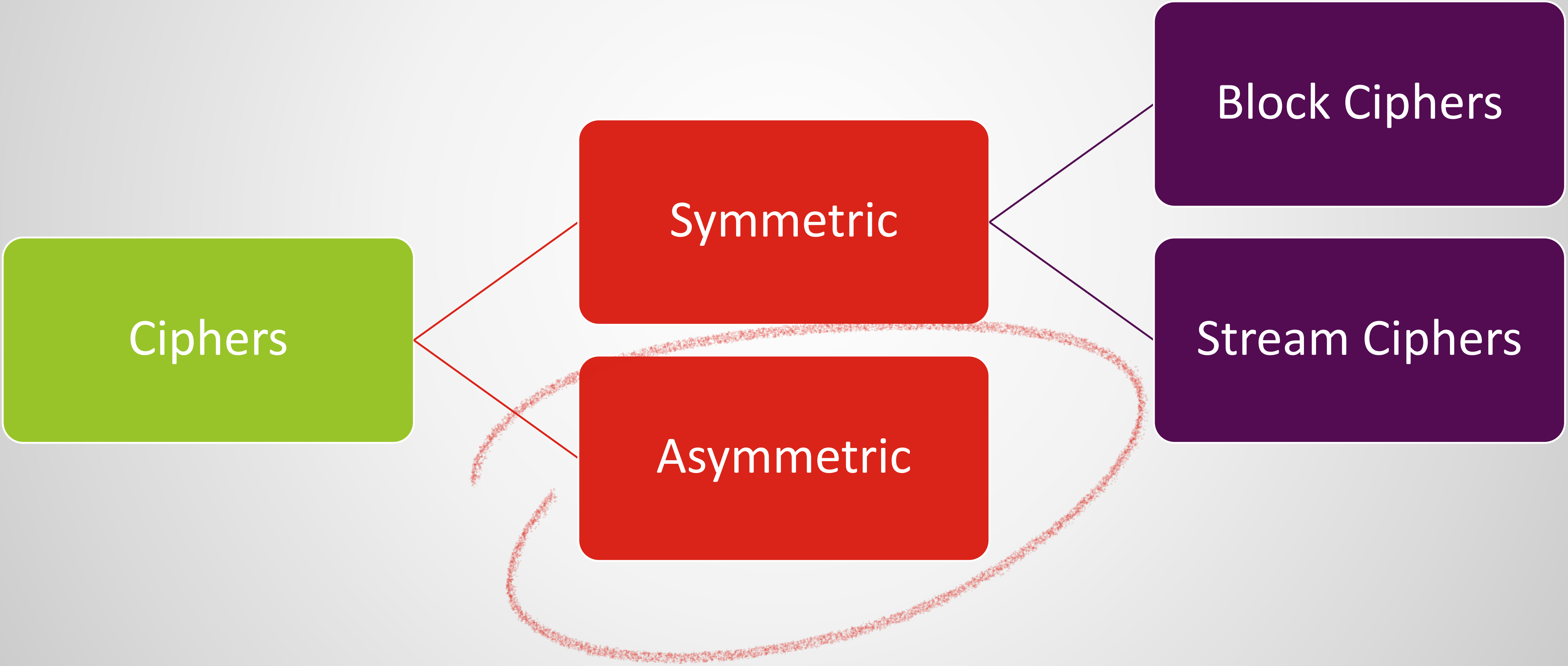
Session 3:

Asymmetric Cryptography - Part 1

Module 3 – Digital Signature (RSA and DSS)

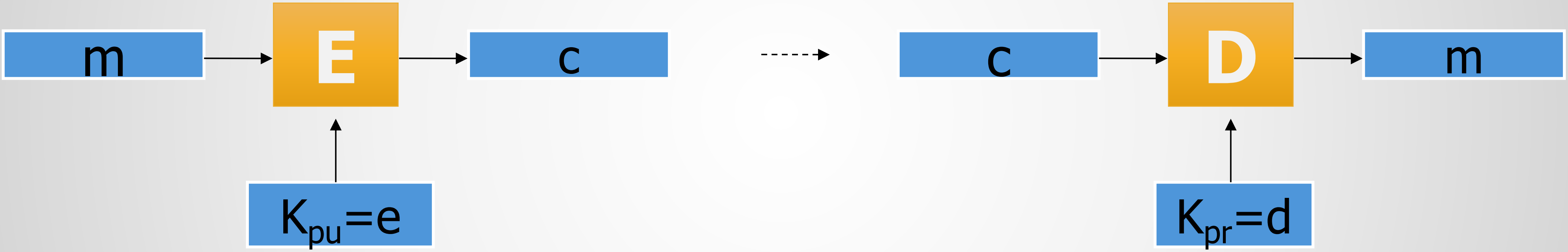
Classification of Ciphers

Asymmetric Cryptography (Public Key Cryptography)



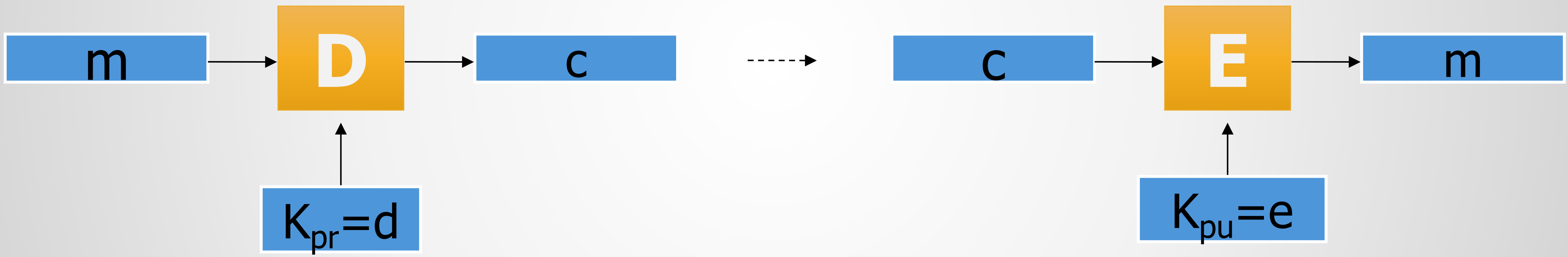
Recall

Asymmetric Cryptosystems:



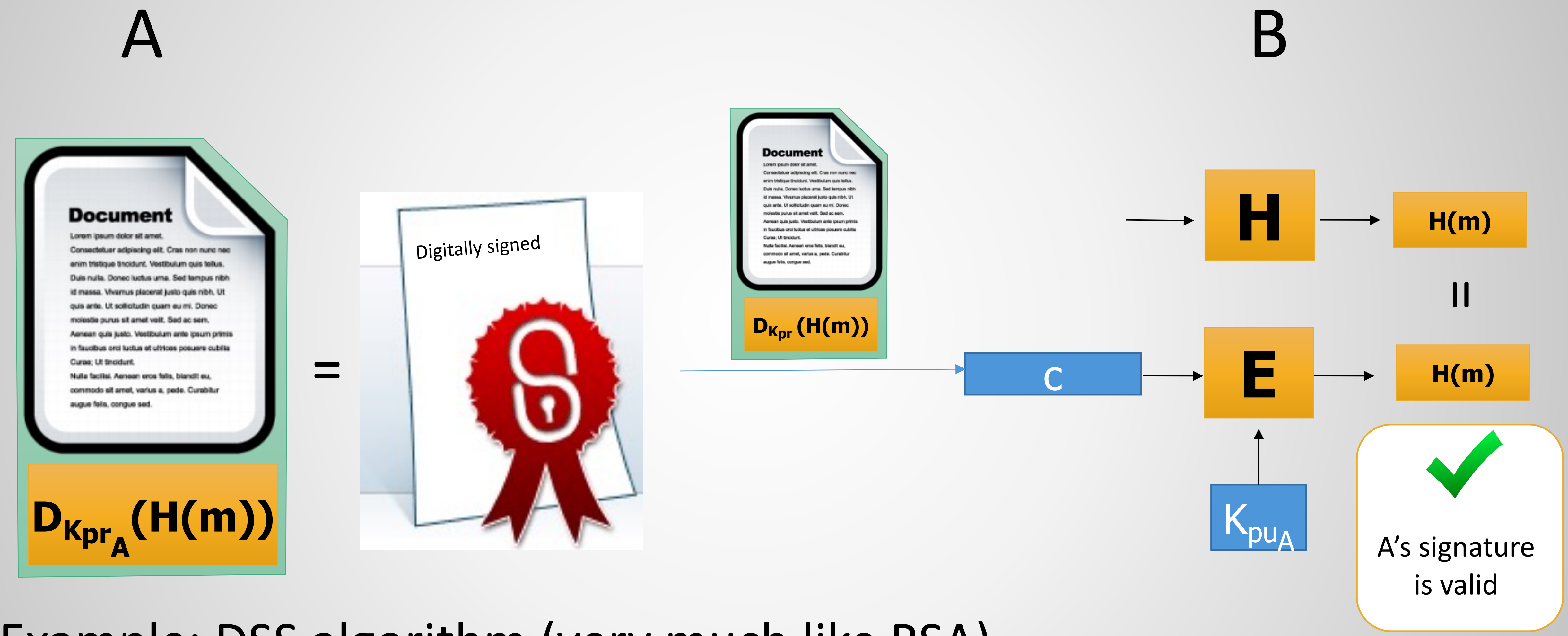
Encryption and Decryption in the Reverse Order

$$E=D^{-1}\ ,\ D=E^{-1}$$



We mentioned that this order is usually used in digital signatures.

Digital Signature

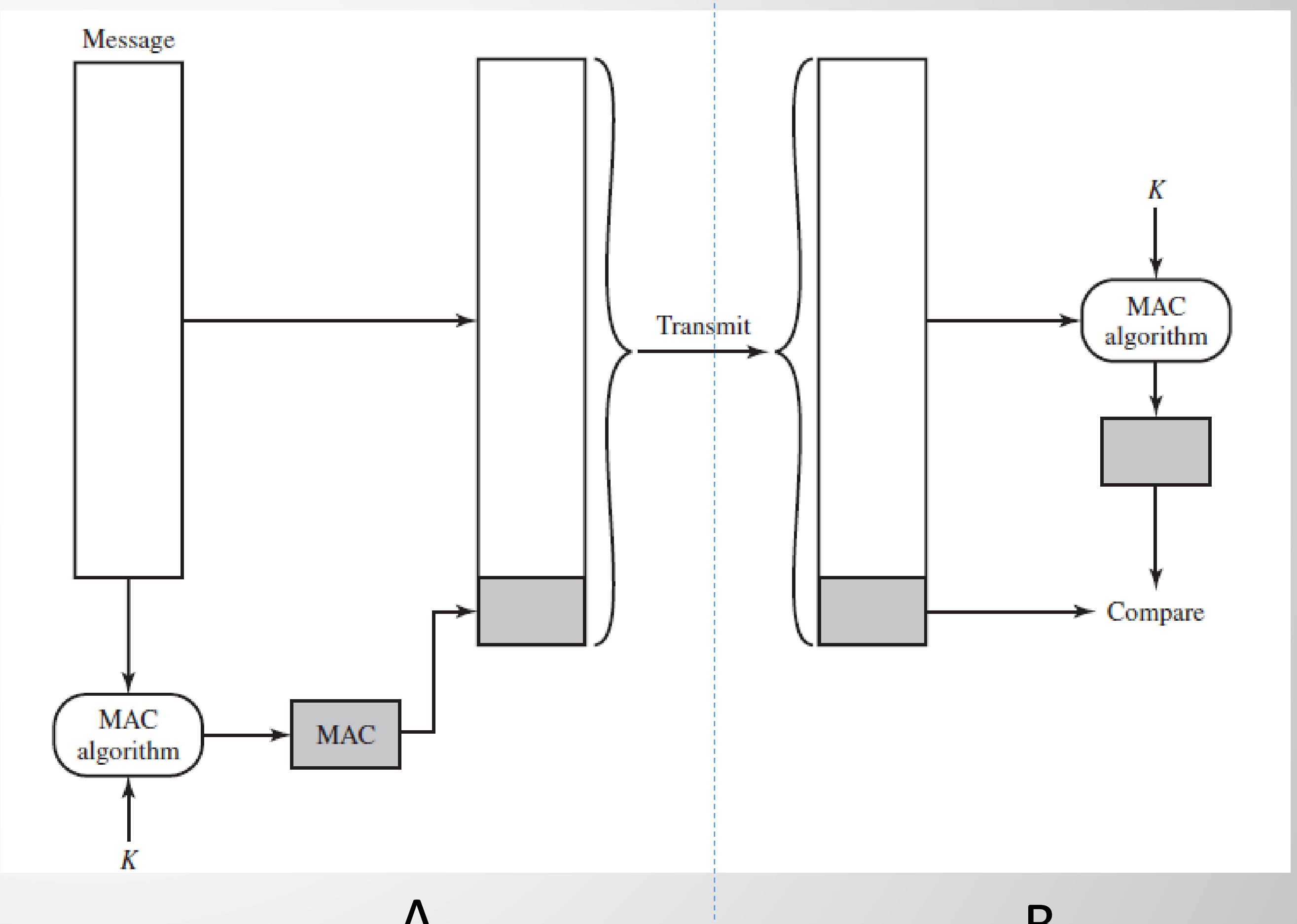


Example: DSS algorithm (very much like RSA)

Do You Remember MAC?

Digital signature is similar to a MAC, but the key that is used for creation of the MAC is the sender's private key.

$MAC_M = F(K, M)$



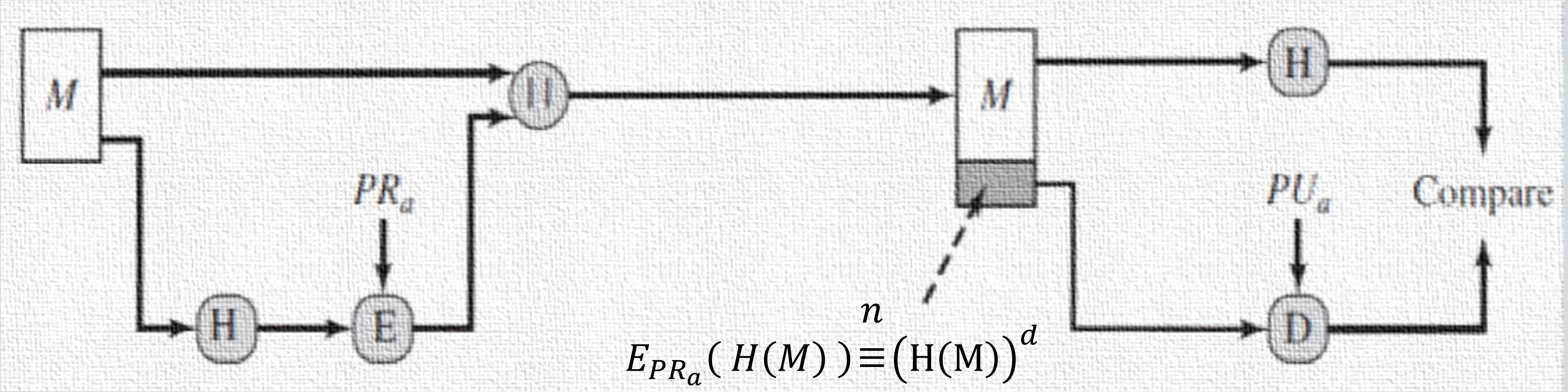
A

B

Major Points about Digital Signature

- Digital signatures (DSs) provide “integrity”, “authentication” and “non-repudiation” services.
- Forging a DS is not possible, since nobody has the private key but **A**.
    - Having the public key does not give a clue to what the private key is (Remember, it is computationally hard!).
- Any manipulation of the message on its way will cause mismatches of the hashes (i.e. H(m)) at the final stage and hence, will be detected.
- Since nobody else has **A**’s private key, whatever he signs cannot be denied later on. This is called non-repudiation. Everybody is free to check what **A** has signed.

RSA Digital Signature



$$E_{PR_a}(\,H(M)\,) \overset{n}{\equiv} \left(\mathrm{H(M)}\right)^d$$

A technical version is defined in NIST FIPS-186 (v4)

Digital Signature Standard/Algorithm (DSS/DSA)

DSS borrows some ideas from Elgamal cryptosystem. A technical version of it is defined in NIST FIPS-186 (v5).

It is used only for signing, and nothing else.

Key Generation:

1. Select two prime numbers (p,q) (such that $q \mid (p-1)$)
2. Choose g to be an element in Zp* with order q
   - Example of Generation: Let α be a generator of Z_p^*, and set $g = \alpha^{(p-1)/q} \bmod p$
3. Select $1 \le x \le q-1$, and Compute $y = g^x \bmod p$
4. Public key (PU) : (p, q, g, y)
5. Private key (PR) : x

DSS

Signature Generation for Message M:

1. Select a random integer k for each message, $0 < k < q$
2. Compute:

$$r = (g^k \bmod p) \bmod q$$

$$s = k^{-1} (H(M) + xr) \bmod q$$

Signature is: **(r, s)**

When q is 160 bits, signature will consist of two 160-bit numbers.

DSS

$r = (g^k \bmod p) \bmod q$

$s = k^{-1}(H(M) + xr) \bmod q$

Signature = (r, s)

Signature Verification for M:

1. Compute

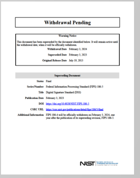
$$u_1 = H(M)s^{-1} \bmod q, \qquad y = g^x \bmod p$$

$$u_2 = rs^{-1} \bmod q$$

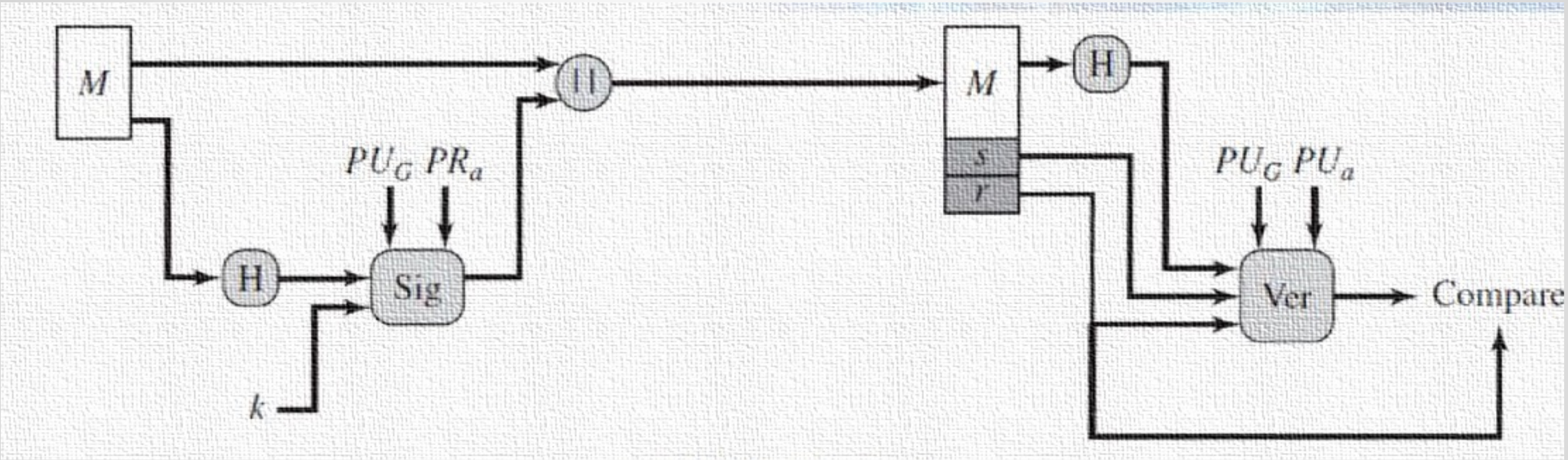
2. Signature is valid if $r = (g^{u_1} y^{u_2} \bmod p) \bmod q$

Why ? -> $g^{u_1} y^{u_2} = g^{H(M)s\wedge(-1)} \, g^{xr\, s\wedge(-1)}$

$$= g^{(H(M)+xr)s\wedge(-1)} = g^k = r$$



DSS Visually



DSS Security

- The value k must change for every message.
- DSS became an standard in 1991 but revised many times, including in 2023.
  - Some benefits over RSA:
    - Signature size (320 bits for $|q|=160$ b) is smaller than that of RSA.
    - One cannot use the implementation for encryption

Other Signatures

- There are other signature schemes, like Elgamal or Elliptic Curve Digital Signature Algorithm (ECDSA).

- ECDSA is used in Bitcoin and Ethereum (and may other cryptocurrencies).

- We will introduce ECDSA later.

What Comes Next …

- We learned how digital signatures are made.

- We learned two signature algorithms; one was based on RSA and the other was called DSS.

- In the next video, we explain the concept of elliptic curves and how they are used in key agreements and in creating signatures.

See you in the next video …