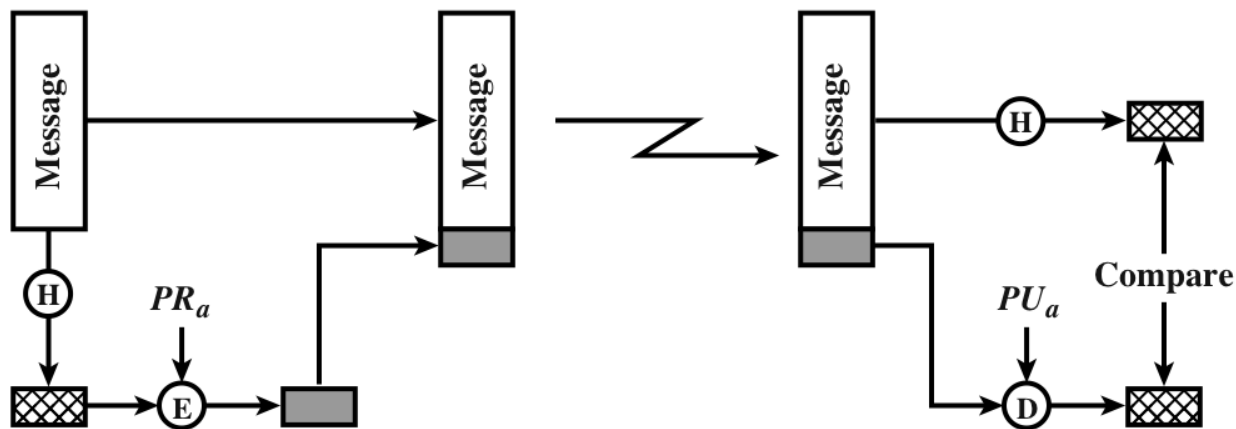


Digital Signatures

Digital Signatures

- Diagram illustrating how to sign a message
- Why do we use a one-way hash?
- How does a collision or second pre-image attack relate to this?



(b) Using public-key encryption

Digital Signatures

- What assurances do we get?
 - Authentication?
 - Confidentiality?
 - Integrity?
 - Non-repudiation?
- MAC vs. Digital Signature
 - What are the differences?
- What do we sign?

Digital Signatures

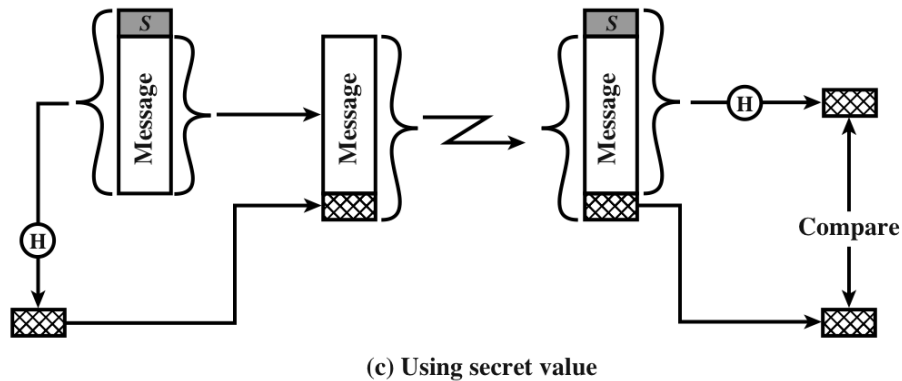
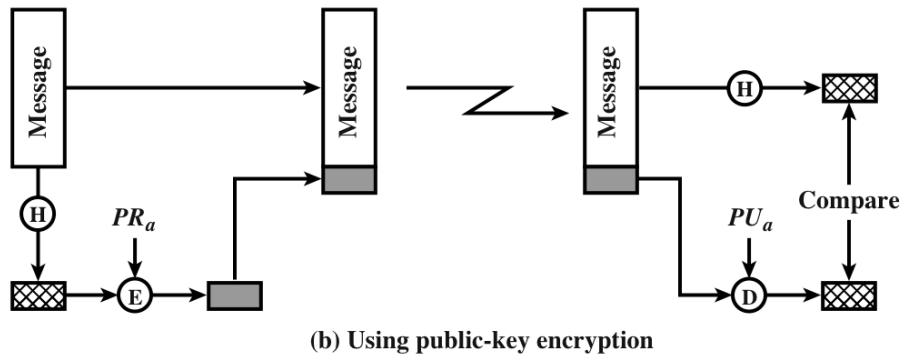
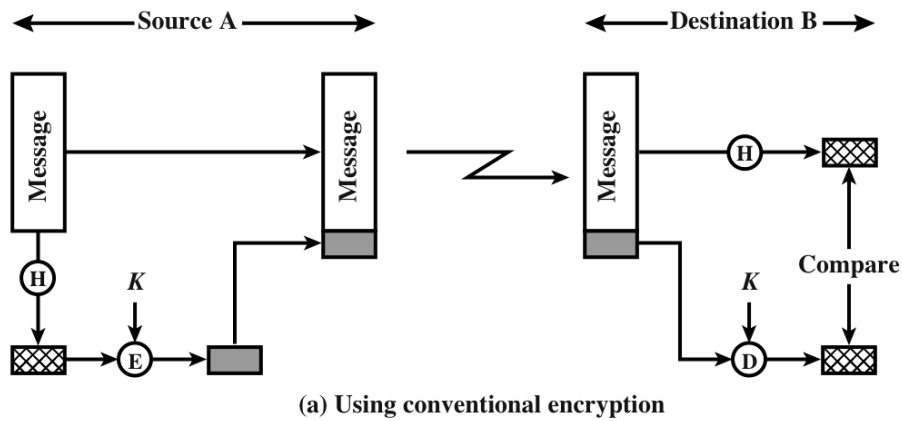
- When do we sign?
 - Sign-then-Encrypt (common)
 - Surreptitious forwarding attack
 - Encrypt-then-Sign
 - Authorship claim attack

Order of Authentication/ Encryption

- Encrypt then Authenticate (Sign/MAC)
 - More secure (in theory)
 - More efficient to discard bogus messages
- Authenticate (Sign/MAC) then Encrypt
 - Harder to attack the MAC, not visible
 - Disclosing data is less severe than accepting modified data
 - Horton Principle – authenticate what you mean, not what you say
- Encrypt and Authenticate
 - Process in parallel
 - MAC protect authenticity, not privacy
- All choices can be used securely and insecurely

Message Authentication

- Three approaches
 - Benefits of each?
 - Drawbacks?
 - Third option broken, use HMAC



Digital Certificates

- Who has a digital certificate?
- What is the most common use for certificates?

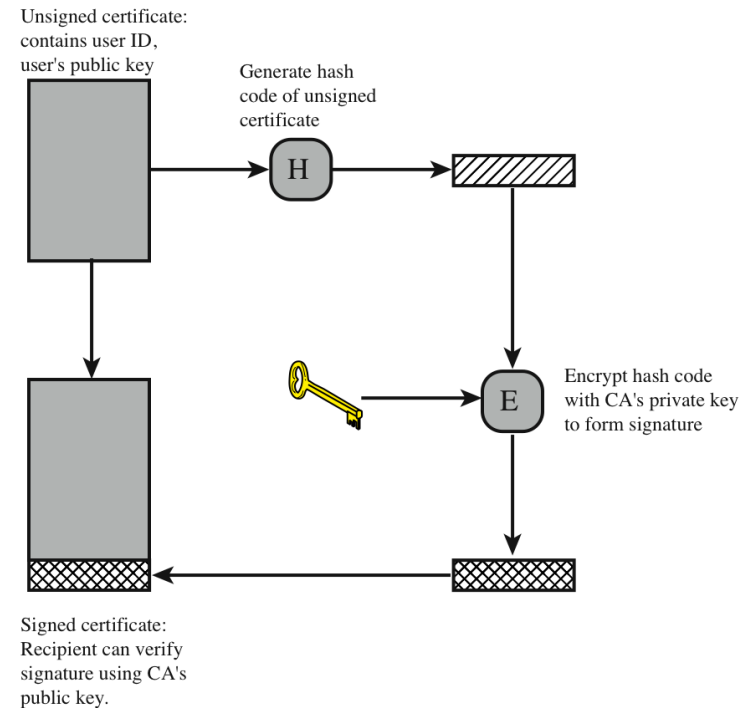


Figure 3.12 Public-Key Certificate Use

Performance Test

- Goal: figure out the relative computing time for algorithms (one block)
 - SHA-1
 - AES
 - RSA Encrypt
 - RSA Signature
- Order the above, fastest to slowest

Performance Test

- Average of 10, 000 iterations

- SHA-1 – 0.0017 ms

- AES – 0.0142 ms

8x slower than a hash

- RSA encrypt – 0.3647 ms

- 3515x slower than a hash

**215x slower than a hash
26x slower than AES**

- RSA signature – 5.9751 ms

**421x slower than AES
16x slower than RSA enc.**

-

•

RSA Padding

...

Or: “How RSA should really be used”

(See PKCS #1)

RSA Primitives

- RSA Encryption
 - $m^e = c \pmod{n}$
- RSA Decryption
 - $c^d = m \pmod{n}$
- Pitfalls of using RSA
 - What if m is very small? Take the e 'th root of c
 - Use RSA padding
 - Implementation flaws – timing attacks
 - Insert constant delays or binding
- RSA Labs publishes the standards for using RSA
 - Public Key Cryptography Standard #1 (PKCS #1)

Overview of PKCS #1

- Good cryptographic practice
 - Employ a key pair in only one scheme
 - One pair for signatures, one for encryption
- Provides 2 approaches for using the RSA primitives
 - Legacy
 - PKCS1-v1_5
 - Recommended
 - OAEP, PSS

PKCS1-v1_5 Encryption

PS is randomly generated, non-zero, bytes

• PS must be at least 8 bytes

• Why must PS have a minimum length?

- Encryption

- Message m becomes EM
 - $EM = 0x00 \parallel 0x02 \parallel PS \parallel 0x00 \parallel M$
- EM is then used with the encryption primitive

- Decryption

- Ensure that the decrypted message conforms to the expected structure above, remove padding, etc

- How does this scheme affect the size of the message to be encrypted?

- When does this approach produce identical ciphertexts?

•

•

PKCS1-v1_5 Signatures

- Signature

- Hash message m , pre-pend the digestID to create T
 - $EM = 0x00 \parallel 0x01 \parallel PS \parallel 0x00 \parallel T$
- EM is then used with the signature primitive

**PS is the byte
0xff repeated
over and over**

- Verification

- Use public key to obtain EM
- Ensure that EM conforms to the expected structure above, remove padding, etc

- How does this scheme affect the size of the message digests that can be used?

- When does this approach produce identical signatures?

-

-

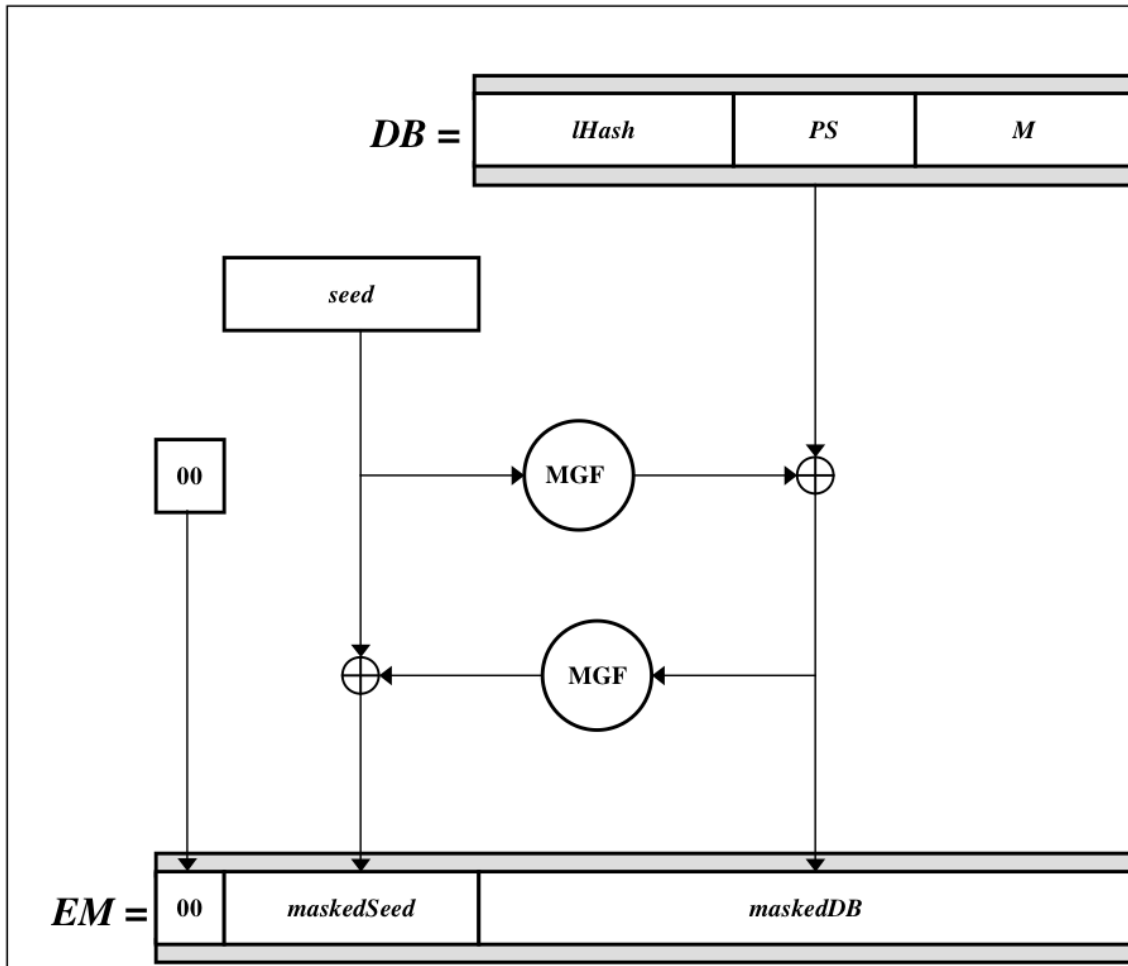
PKCS1-v1_5

- Encryption
 - $EM = 0x00 \parallel 0x02 \parallel PS \parallel 0x00 \parallel M$
- Signature
 - $EM = 0x00 \parallel 0x01 \parallel PS \parallel 0x00 \parallel T$
- Why do both start with 0x00?

RSA Encryption with OAEP

- OAEP
 - Optimal Asymmetric Encryption Padding
 - In addition to being more secure adds the ability to associate a label with the message
- $\text{Encrypt}(M, L \text{ (optional)}, n, e)$

RSA Encryption with OAEP



PS is the byte 0x00 repeated over and over

seed is a random series of bytes

How does this scheme affect the size of the message to be encrypted?

When does this approach produce identical ciphertexts?

RSA Signatures with PSS

- PSS
 - Probabilistic Signature Scheme
- Padding_1 and Padding_2 are bytes of 0x00
- Salt is random
- bc is 0xbc
- When does this approach produce identical signatures?

