Experiment No. 11

Aim: To perform Digital Signature Scheme Experiment using Virtual lab.

Theory:

What Are Digital Signatures?

The objective of digital signatures is to authenticate and verify documents and data. This is necessary to avoid tampering and digital modification or forgery during the transmission of official documents.

With one exception, they work on the public key cryptography architecture. Typically, an asymmetric key system encrypts using a public key and decrypts with a private key. For digital signatures, however, the reverse is true. The signature is encrypted using the private key and decrypted with the public key. Because the keys are linked, decoding it with the public key verifies that the proper private key was used to sign the document, thereby verifying the signature's provenance.

RSA Signatures

The RSA public-key cryptosystem provides a digital signature scheme (sign + verify), based on the math of the modular exponentiations and discrete logarithms and the computational difficulty of the RSA problem (and its related integer factorization problem).

Key Generation

The RSA algorithm uses **keys** of size 1024, 2048, 4096, ..., 16384 bits. RSA supports also longer keys (e.g. 65536 bits), but the performance is too slow for practical use (some operations may take several minutes or even hours). For 128-bit security level, a 3072-bit key is required.

The **RSA key-pair** consists of:

- public key {$\boldsymbol{n}$, $\boldsymbol{e}$}
- private key {$\boldsymbol{n}$, $\boldsymbol{d}$}

The numbers ***n*** and ***d*** are typically big integers (e.g. 3072 bits), while ***e*** is small, typically 65537.

By definition, the RSA key-pairs has the following property:

$$(m^e)^d \equiv (m^d)^e \equiv m(mod n)$$

for all ***m*** in the range [0...***n***)

RSA Sign

Signing a message ***msg*** with the private key exponent ***d***:

1. Calculate the message hash: $\boldsymbol{h} \ = \ hash(\boldsymbol{msg})$
2. Encrypt ***h*** to calculate the signature: $s = h^d(mod n)$

The hash ***h*** should be in the range [0...***n***). The obtained **signature *s*** is an integer in the range [0...***n***).

RSA Verify Signature

Verifying a signature ***s*** for the message ***msg*** with the public key exponent ***e***:

Calculate the message hash: $\boldsymbol{h} \ = \ hash(\boldsymbol{msg})$

Decrypt the signature: $h' = \ s^e(mod n)$

Compare ***h*** with $\boldsymbol{h'}$ to find whether the signature is valid or not

If the signature is correct, then the following will be true:

$h' = \ s^e(mod n) = (h^d)^e(mod n) = h$

<u>Vlab output</u>:

Digitally sign the plaintext with Hashed RSA.

Plaintext (string):

testing [SHA-1]

Hash output(hex):

dc724af18fbdd4e59189f5fe768a5f8311527050

Input to RSA(hex):

dc724af18fbdd4e59189f5fe768a5f8311527050 [Apply RSA]

Digital Signature(hex):

```
4e29ebb817be5d8fc591a631626dcd19fd8ef343cc42e6b6e875397394a1b170
2ece10d16d571833be50b3cbc3dd785b8ea3c0fa36057e0637210a0e4f89f8b5
29eee4bd6943f30858cd6c6f64f8227926eb6dc8b4fc151d1380ae9640fabdda
f535d0b1f0766c2556619bdb0b3e9ab159f3b8d106f559c1a0ba9d8164ca6e78
```

Digital Signature(base64):


```
TinruBe+XY/FkaYxYm3NGf2O80PMQua26HU5c5ShsXAuzhDRbVcYM75Qs8vD3Xhb
jqPA+jYFfgY3IQoOT4n4tSnu5L1pQ/MIWM1sb2T4Inkm623ItPwVHROArpZA+r3a
9TXQsfB2bCVWYZvbCz6asVnzuNEG9VnBoLqdgWTKbng=
```


Status:

Time: 5ms

RSA public key

Public exponent (hex, F4=0x10001):

10001

Modulus (hex):

```
a5261939975948bb7a58dffe5ff54e65f0498f9175f5a09288810b8975871e99
af3b5dd94057b0fc07535f5f97444504fa35169d461d0d30cf0192e307727c06
5168c788771c561a9400fb49175e9e6aa4e23fe11af69e9412dd23b0cb6684c4
c2429bce139e848ab26d0829073351f4acd36074eafd036a5eb83359d2a698d3
```

[1024 bit] [1024 bit (e=3)] [512 bit] [512 bit (e=3)]