Week 5

Lecture 1

Part 1: Public Key Cryptography: RSA Digital Signature
Part II: Security of RSA

Lecture 2

Revisiting Modes of Encryption and any left-over mathematics.

Workshop 3: Workshop based on Lectures in Week5

Quiz 5

Public Key Cryptography: RSA Digital Signature

COMP90043

Lecture 1

Public Key Cryptography: Diffie-Hellman and RSA

Lecture 1

Part I Public Key Cryptography: RSA Digital Signature

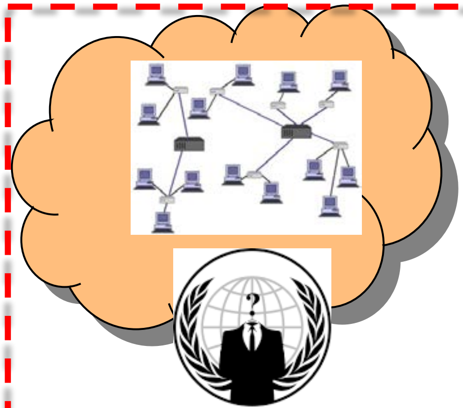
- 1.1 RSA Digital Signature
  - Digital Signature Introduction.
  - Different Versions RSA Signatures.
  - Signature Algorithm in Practice

- 1.2 Mathematical Attacks
  - Security of RSA
  - Elementary Attacks
  - Status of Factorization problem

Part II Security of RSA

- 1.3 Security Notions and Attack Models
  - Security Notions
  - CCA Attack
  - Timing Attacks

Recap:Story of Alice and Bob terms and notations



Alice (Sender)

- Uses an Encryption Function (E)

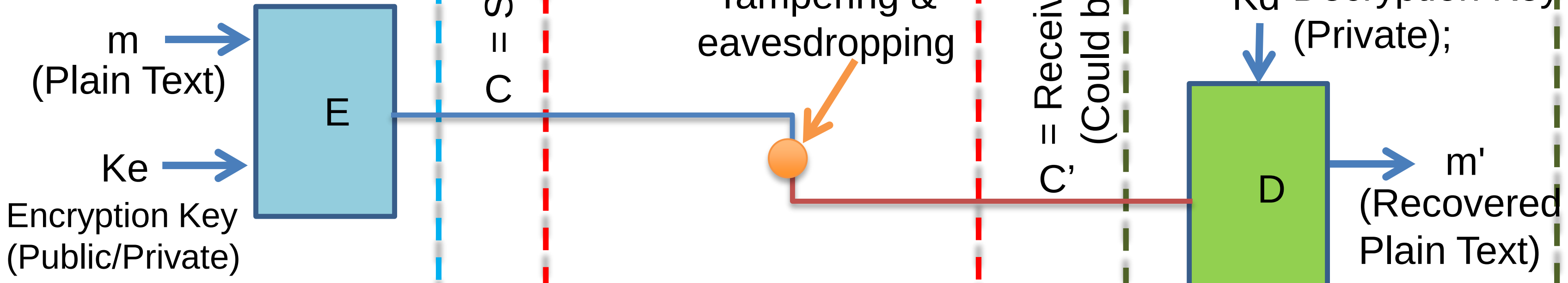
Eve (Adversary)

- Uses network probes to attack communication

Tampering & eavesdropping

Bob (Receiver)

- Uses a Decryption Function (D)



E, D are public; c is the ciphertext, c' is received ciphertext; ideally m=m';

Cryptography involves many conceptual ideas, we look at the basic functions

Images: General Internet resources

The Figure Illustrates the notations
And use of Public Key functions;
We will use this notation
throughout this semester.

Public key of B : Pu_b
Private key of B : PR_b

Encryption and Decryption by A

↓

$Y = E(PU_b, X)$

$X := D(PR_b, Y)$

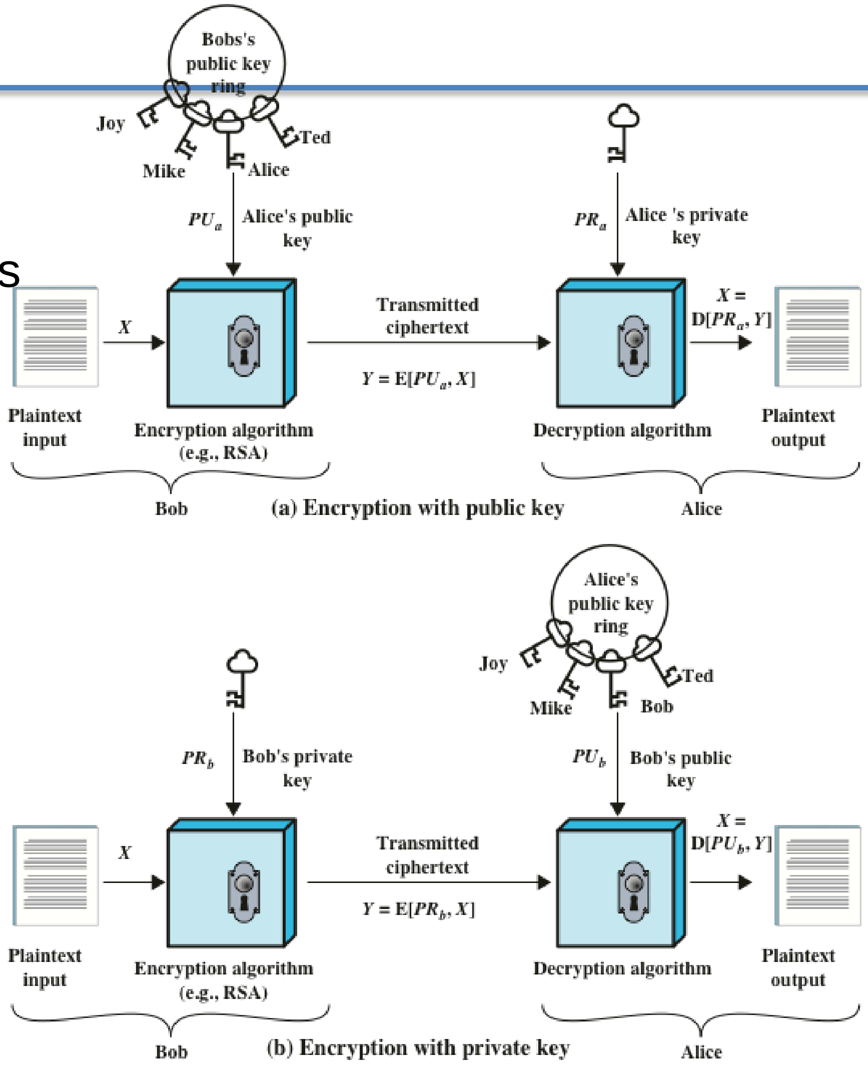


Figure 9.1 Public-Key Cryptography

Figure: © 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

1.1 RSA Digital Signature

COMP90043

Lecture 1

Digital Signature

- We studied Public Encryption last week. We saw how RSA encryption and decryptions work.

- What is a digital signature?
- An electronic tag as a sequence bits created by a sender on a message sequence, enabling a verifier to conclude that the tag was indeed created by the Sender for the message, providing authentication and assurance that the contents of the transmitted bits are not modified prior to the verification.
- Main goal for signature is to bind the message to the sender with assurance that the signature is not forged by anyone other than the sender.

- Can we use RSA to create signatures?
- We will tackle this problem this week.

RSA PKC (1978)

- Let $n = p \times q$; p, q are primes. Let the plain text and cipher text belong to integers modulo n and let (e, d) pair be computed such that

$$e \times d \equiv 1 \bmod \phi(n)$$

(ϕ :Euler's totient function)

- For the RSA key parameter set $K = (n, p, q, e, d)$, define

$$E_k(x) = x^e \bmod n$$

And

$$D_k(y) = y^d \bmod n,$$

where $(x,y \text{ in } Z_n)$. The values (n,e) are termed the public key, and the values p, q and d form the private key.

RSA Example

- Let $n = 91$; $p=13, q=7$ are primes. Let the plain text and cipher text belong to Z_{91} (residue Integers *modulo 91*). $\phi(n) = 12 \times 6 = 72$.

- For K = ($n=91, p=13, q=7, 5, 29$), define

$$E_k(x) = x^e \; mod \; n$$

And

$$D_k(y) = y^d \; mod \; n,$$

- Verify $5 \times 29 = 145 \bmod 72 = 1$
- Message $x = 11$

- $E_k(11) = C = 11^5 = 72$

- $D_k(72) = 72^{29} = 11$

RSA Signature

- The purpose of this discussion is to understand how RSA signature works and how it is different from RSA encryption we studied earlier.
- RSA signature is a public key digital signature scheme.
- In general, a signature is a means for a trusted third party (Network Security Manager) to bind the identity of a user to a public key.
- Why do you we need a trusted third party? It becomes clear when you try to answer the following?
- Can we conceive digital signature using symmetric key cryptosystem?
- Answer: Technically yes, but you require a trusted centre who provide key management service for creators and verifiers of the digital signature. We will not study this concept in this subject.
- Do we need trusted third party for Public key signatures?
- Yes, it is mainly to address the authentication of public addresses, we will look at this issue later.

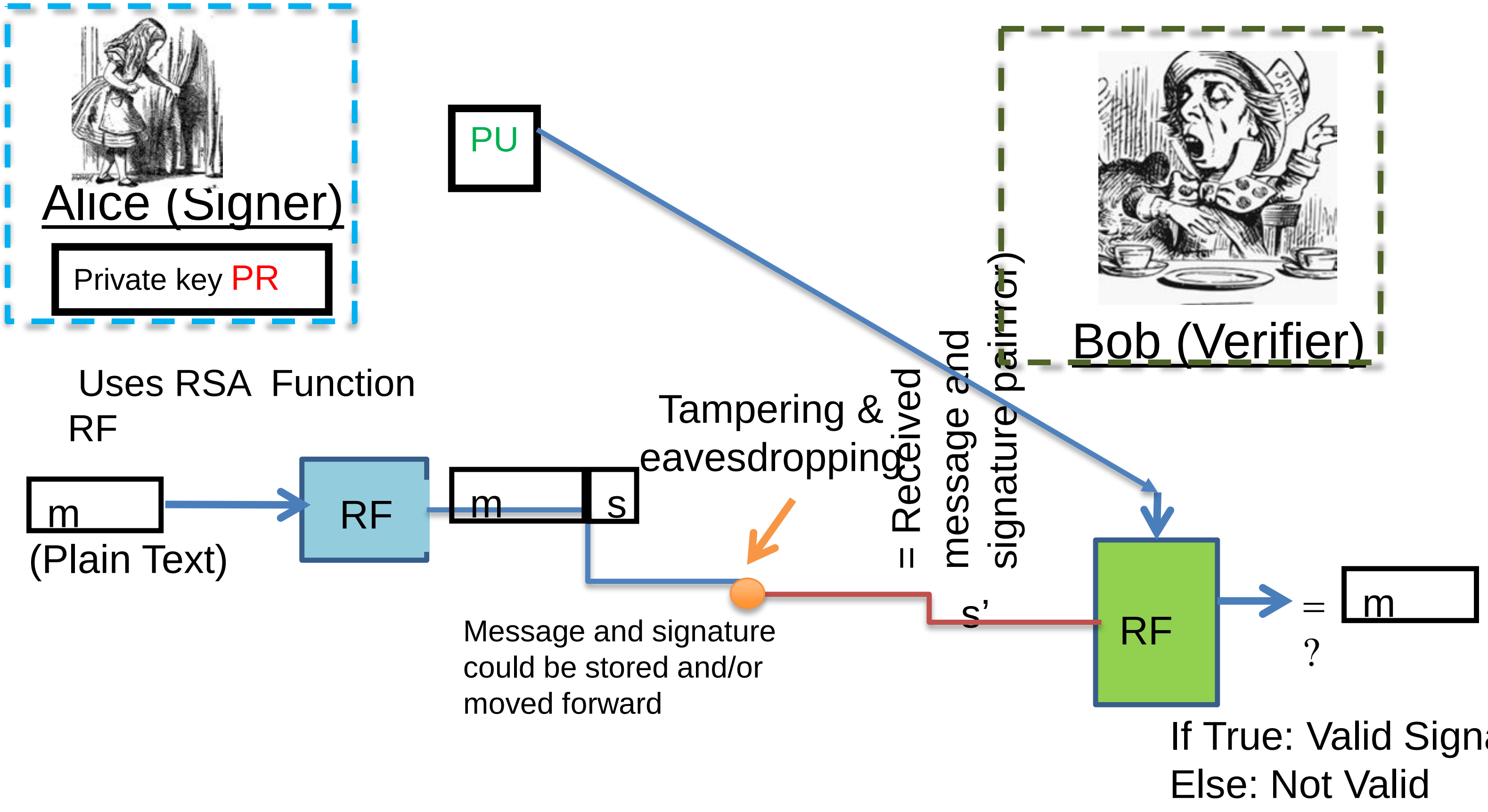
RSA signature

- RSA signature is a complement of RSA public encryption.
- In RSA Signature, the owner for the private key signs messages; Anyone with the public key can verify the signatures.
- Sig = RF(PR,Message); Verification is a RF(PU, Message, Sig), which outputs 1 if it is a True Signature and 0 if invalid.
- In RSA encryption, anyone with the public key can encrypt messages and the owner of the private key can decrypt the messages.
- Ciphertext= RF(PU, Message), Message can be decrypted by RF(PR,Ciphertext) which also gives additional output of 1 if correctly decrypted or 0 if erroneous.

What is the Signature Structure

- Let M be a message, PR and PU are private and Public keys of Alice.
- Signature is obtained by a
  - <M, Sig= RF(PR,M)>, where RF is RSA function,
- Verification is essentially checking
- RF(PU,Sig) is equal to M or not?
- Let us try to create Signature process using RSA construction.

RSA Digital Signature


Alice (Signer)
Private key PR
PU
Bob (Verifier)
Uses RSA Function RF
m
(Plain Text)
RF
m
s
Tampering & eavesdropping
Message and signature could be stored and/or moved forward
= Received message and signature pair (m', s')
s'
RF
= ?
m
If True: Valid Signature
Else: Not Valid

First Version of RSA Signature

- $N = P \times Q$; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]
- Signature Generation:
- Message $0 < M < N$;
- Compute: $s = M^d \bmod N$;
- Signature—[M,s]
- Verification – if $s^e \bmod N =?= M$ then “Signature Valid”
- Else “Signature Invalid”

First Version of RSA Signature

- N = P×Q; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]
- Signature Generation:
- Message $0 < M < N$;
- Compute: $s = M^d \bmod N$;
- Signature—[M,s]
- Verification – if $s^e \bmod N$ =?= M then "Signature Valid"
- Else "Signature Invalid"

Issues with the First Version of the Scheme

Multiplicative property of RSA signature

$$(M_1 \times M_2)^d = M_1^d \times M_2^d = (M_1 \times M_2)^d$$

i.e. if s_1 = Signature of M_1;
s_2 = Signature of M_2;

Then $(s_1 \times s_2)$ is the signature of $(M_1 \times M_2)$. Follows from exponential law.

This property leads to a possibility of forgery of signature!
This is in fact an example of existential forgery.
Further,

- What if Message is very long?
- You may need to split a long message into several messages of size less than N and sign the parts one by one.
- Also, a problem called blinding, which we will look in the next slide.

Blinding

- Alice’s Public address – [N, e]
- Alice’s Private address –[d]
- You want to get Alice sign a message M, which Alice may not be ready to do.
- Choose a random x – in the range [0..N-1]
- Form a blinded message -- $M_b = x^e M \mod N$
- Alice may agree to sign this blinded message M_b (assume),
- Alice then signs the message M_b as $s_b = M_b^d \mod N$
- This blinded signature can be used to compute the signature for M using the multiplicative property:
- Now you can compute signature for M as
- $s = s_b/x \mod N$
- This is true because, let us apply the verification rule, Note
- $s^e = s_b{}^e / x^e = (M_b)^{d\,e} / x^e = (M_b) / x^e = x^e M / x^e = M$
- Hence s is the signature of M. So we have produced a forgery!

The issues

- The multiplicative property makes it impossible to the textbook RSA in practical application because of possibility of blinding.
- Also there is a possibility of existential forgery!
- So, for use as digital signature we need to some way to break the multiplicative rule of the RSA function.
- But note that this multiplicative property itself will be useful in some other situation as creating electronic cash analogous to the printed cash!

Second Version of RSA Signature

- N = P×Q; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]

- Signature Generation:
- Message $0 < M < N$;
- Find $M_1 = R(M)$; where R is a redundancy function and $1 < R(M) < N$.

- Compute: $s = M_1^{d} \bmod N$;
- Signature—[M,s]
- Verification – $M_1 = R(M)$; if $s^e \bmod N$ =?= M_1 then “Signature Valid”
- Else “Signature Invalid”

RSA Signature in Practice

- A practical signature scheme should take care of the two problems discussed before.
- Messages are generally long
- RSA signature scheme needs a redundancy function to avoid existential forgery attacks.
- Also repeated messages carry same signature.
- In practice, generally a suitable cryptographic hash function is applied to the message (which could be arbitrarily large); and sign the hash.
- We will learn about Hash functions later.

Third (Better) Version of RSA Signature

- $N = P \times Q$; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]

- Signature Generation:
- Let M be a Message be of arbitrary length: M =[………]; Find H(M), where H is a Hash function
- Format M_1 = [H(M), Identity Information, Random Number], such that $1 < M_1 < N$.

- Compute: $s = M_1^d \bmod N$;
- Signature—[M,s]
- Verification – Extract M_1 by computing $s^e \bmod N$;
- If Any formatting violations – Reject the Signature.
- Further Verify H(M) = H(M) on M_1

Signature Algorithm in Practice

- The previous discussion is about the issues related to RSA signature.
- In practice, standard signature algorithm proposed by NIST Digital Signature Algorithm (DSA) and Elliptic Curve DSA(ECDSA) are generally used, which we will discuss later.
- The 2019 version of the standard (FIPS186) include a version based on RSA. Please read Section 13.6 for further details.

1.2 Mathematical Attacks

COMP90043

Lecture 1

Security of RSA

- Brute force Attack: (infeasible given size of numbers)
- Attack by making use of loopholes in Key distribution.
- Mathematical attacks (Factoring and RSA problem)
- Elementary attacks
- Advanced Factorization methods
- Network attacks

Mathematical attacks

- The RSA function is one way. This is an assumption and we do not have a proof, but it is considered as one-way by researchers.
- The problem
- Given n,e, $c=M^e$ (mod n),
  - Can we determine M?
  - Do we have an algorithm to find the e^{th} root of c mod n?
  - Can we find d such that de = 1 mod $\Phi(n)$?
- Can we factor n?

Factorization Problem

- In general the factorization is hard.
- Brute force algorithm is exponential in b, where b is number of bits in the representation of the number n to be factored.
- Complexity of the best known algorithm for factorization:

$$\exp((\, c+O(1)b^{1/3} \operatorname{Log}^{2/3}(b)),$$

for some integer $c < 2$

- It is not worth thinking of factoring.
- Maybe quantum computers come to our rescue; but not in immediate future!

Elementary attacks

- Can we use common modulus for more than one user?

- Facts:
- Knowing e and d such that

$$ed = 1 \bmod \Phi(n)$$

Is equivalent to factoring.

- Knowing n, $\Phi(n)$ is also equivalent to factoring.

Common modulus

- Every user chooses same modulus n=pq set up by a trusted central authority. But each user chooses their own private and public key pairs
- User i ------(e_i,d_i)
- So using the facts in previous slide, any user can factor common modulus n and can find the private information of other user by using only the public information.
-
- Hence it is extremely important that every entity chooses its own RSA modulus n.
- Again the modulus need to be create using random primes.

Broadcast Problem

A group of entities may all have a same encryption exponent but should have different modulus. Further, to improve the public encryption, let the public key be small, say e=3.

- A wishes to send a common message m to three entities with modulus n n_1, n_2 and n_3.
- The cipher text for three entities are given by
- $c_1 = m^3 \pmod{n_1}$
- $c_2 = m^3 \pmod{n_2}$
- $c_3 = m^3 \pmod{n_3}$.

Broadcast Problem

- Then, to recover the message m solve,
- $x = c_1 \pmod{n_1}$,
- $x = c_2 \pmod{n_2}$,
- $x = c_3 \pmod{n_3}$,
- You can use CRT and Then obtain an unique
- $x=m^3$ modulo $n_1\, n_2\, n_3$
- m can then be obtained by taking the cube root of x. Finding a cube root in integers is not a hard problem.

Status of Factorization

- The textbook gives a detailed account of development in the factorization algorithm for integers.
- Please revisit Table 9.5 and Fig. 9.9.
- Currently RSA modulus should be as long as 2000 to 4000 bits.

Week 5

Lecture 1

Part 1: Public Key Cryptography: RSA Digital Signature
Part II: Security of RSA

Lecture 2

Revisiting Modes of Encryption and any left-over mathematics.

Workshop 3: Workshop based on Lectures in Week5

Quiz 5