

# ECE 443/518 – Computer Cyber Security

## Lecture 11 Digital Signatures and Authentication

Professor Jia Wang  
Department of Electrical and Computer Engineering  
Illinois Institute of Technology

September 28, 2022

# Outline

Digital Signatures

Authentication

# Reading Assignment

- ▶ This lecture: UC 10.1 – 10.3
- ▶ Next lecture: UC 13

# Midterm Exam

- ▶ Lecture 1 ~ Lecture 13
- ▶ Students registered for main campus section: Wed. 10/12, 11:25 AM – 12:40 PM, in class.
  - ▶ A physical calculator is allowed. Laptop or any other electronic device or calculator apps running on them are not allowed.
  - ▶ Closed book/notes. A letter-size page of cheat sheet is allowed.
- ▶ Students registered for online sections: contact Charles Scott, [scott@iit.edu](mailto:scott@iit.edu), from the Center of Learning Innovation to make arrangement, and confirm with me.
  - ▶ No make-up exam will be offered if you fail to do so.
  - ▶ You may confirm with me directly if you plan to take the exam with the main campus section as mentioned above.
- ▶ 100+20 points
  - ▶ See Homework 2 and 3 for sample questions.
  - ▶ You are required to show steps of calculations.
  - ▶ Points may be deducted if key steps are missing even if the answers are correct.
- ▶ Emergency/extraordinary reasons for make-up midterm exams are accepted only with documented proof like doctor's notes.

# Outline

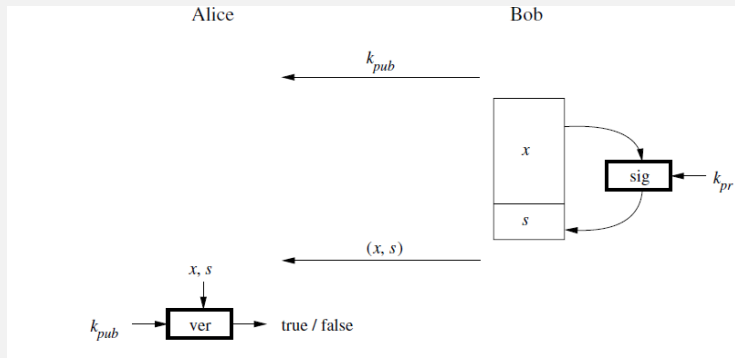
Digital Signatures

Authentication

# Security Services

- ▶ The services we are familiar with
  - ▶ Confidentiality
  - ▶ Integrity and message authentication
- ▶ Nonrepudiation: sender can not deny creation of message.
  - ▶ But who is the sender?
- ▶ Authentication: who are you?
  - ▶ A.k.a. entity/user authentication, or identification
  - ▶ Within the context of computer cyber security, shall be built on top of a nonrepudiation service (but usually is not!).
- ▶ Services enabled by authentication
  - ▶ Access control/authorization: decide who can do what.
  - ▶ Auditing: provide a proof of who did what.
- ▶ Anonymity/privacy: what if we don't want to be identified?
  - ▶ E.g. to guard against potential misuse of identity.
  - ▶ Can we authenticate an amonymous user?

# Principle of Digital Signatures

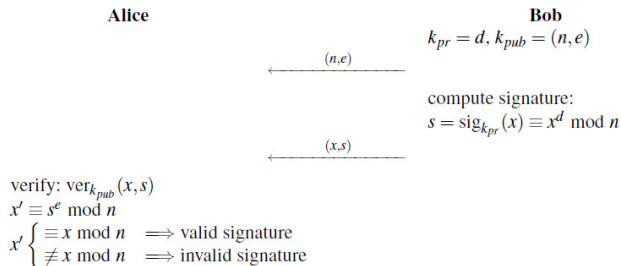


**Fig. 10.1** Principle of digital signatures which involves signing and verifying a message (Paar and Pelzl)

- ▶ Nonrepudiation: no shared secret
  - ▶ Bob signs with his private key  $k_{pr}$ .
  - ▶ Alice verifies with Bob's public key  $k_{pub}$ .
- ▶ Sign the hash if the message is too long.

# RSA Digital Signature

## Basic RSA Digital Signature Protocol



(page 265, Paar and Pelzl)

- ▶ Same key setup as RSA
- ▶ RSA digital signature works as inversed RSA encryption!
  - ▶  $\text{sig}()$  is  $d()$ ,  $\text{ver}()$  is essentially  $e()$ .
  - ▶ Time complexity is the same as RSA encryption.



# Example

- ▶  $k_{pub} = (n = 221, e = 5), k_{pr} = (p = 13, q = 17, d = 77)$
- ▶  $x = 35$
- ▶ Bob computes the signature:  
 $s = x^d \bmod n = 35^{77} \bmod 221 = 120.$
- ▶ Alice verifies the signature:  
 $x' = s^e \bmod n = 120^5 \bmod 221 = 35.$ 
  - ▶ So  $x == x'$  and  $x$  is indeed generated by Bob.
- ▶ What prevent Oscar to forge Bob's signature?

# Oscar's Attack

- ▶ To forge a signature  $s$  for  $x$ , Oscar need to
  - ▶ Either compute  $d$  and then  $s = x^d \bmod n$ .
  - ▶ Or solve  $s^e \equiv x \pmod{n}$
- ▶ Both are equivalent to break RSA.

# Elgamal Digital Signature

- ▶ Setup Bob's key pair as in DHKE and Elgamal
  - ▶ A well-known large prime  $p$  and an integer  $\alpha$ .
  - ▶  $k_{pr} = d \in \{2, 3, \dots, p-2\}$
  - ▶  $k_{pub} = \beta = \alpha^d \bmod p$
- ▶ To sign a message  $x \in \{0, 1, 2, \dots, p-1\}$  with  $(r, s)$ ,
  - ▶ Choose a random ephemeral key  $k_E \in \{0, 1, \dots, p-1\}$  such that  $\gcd(k_E, p-1) = 1$ .
  - ▶ Compute  $r = \alpha^{k_E} \bmod p$
  - ▶ Solve  $k_E s \equiv x - dr \pmod{p-1}$  for  $s$
- ▶ To validate the signature  $(r, s)$  for the message  $x$ ,
  - ▶ Compute  $t = \beta^r r^s \bmod p$
  - ▶ Apply Fermat's Little Theorem,  $r^s \equiv \alpha^{k_E s} \equiv \alpha^{x-dr} \pmod{p}$
  - ▶ So  $t \equiv \beta^r r^s \equiv \alpha^{dr} \alpha^{x-dr} \equiv \alpha^x \pmod{p}$  should hold.

# Oscar's Attack

- ▶ To forge a signature  $(r, s)$  for  $x$ , Oscar need to solve

$$\alpha^x \equiv \beta^r r^s \pmod{p}$$

- ▶ Oscar could first choose any  $k_E$  and  $r = \alpha^{k_E} \pmod{p}$ , then
  - ▶ Either solve  $r^s \equiv z \pmod{p}$  directly with some  $z$
  - ▶ Or find  $d$  first and then solve for  $s$  as the signature process.
- ▶ Both are equivalent to break DHKE.

# Practical Considerations

- ▶ For both RSA and Elgamal digital signature, padding is needed to prevent other attacks.
- ▶ Elgamal digital signature is rarely used in practice. Instead, a variant named DSA and an ECC generalization named ECDSA are widely used
  - ▶ Check FIPS PUB 186-4 (2013) for details.

# Digital Signature vs. MAC

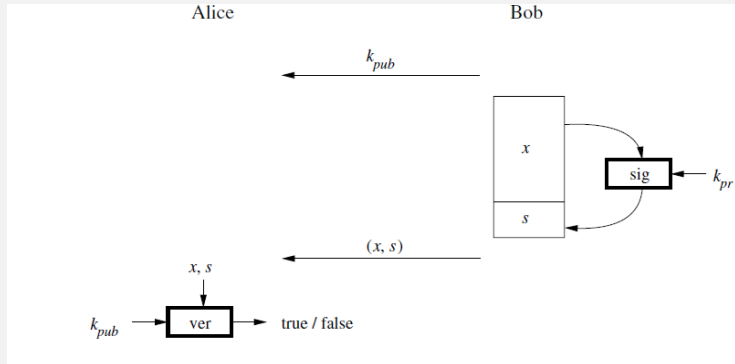
- ▶ Digital signatures provide stronger guarantees (nonrepudiation) than MAC (message authentication), and thus can replace MAC.
  - ▶ Assume no man-in-the-middle attack.
- ▶ Practically, MAC is more efficient.
  - ▶ MAC is almost as efficient as hash at both sides.
  - ▶ Digital signature need to compute exponentials at both sides in addition to hash.
  - ▶ Use MAC if nonrepudiation is not required.
- ▶ While we prefer to apply MAC to ciphertext for authenticated encryption, digital signatures are almost always applied to plaintexts if the messages need to be encrypted.

# Outline

Digital Signatures

Authentication

# Digital Signatures Revisited



**Fig. 10.1** Principle of digital signatures which involves signing and verifying a message (Paar and Pelzl)

- ▶ What should be sent over an authentic channel?
  - ▶  $k_{pub}$ , if Alice need a proof that  $k_{pub}$  is indeed Bob's public key.
- ▶ What if  $k_{pub}$  is sent over an insecure channel?
  - ▶ Nonrepudiation still works in some sense: Alice can confirm that  $x$  is created by someone who owns  $k_{pr}$ .



# Public Key and Identity

- ▶ Authentication: Bob need to decide if Alice is Alice.
  - ▶ For recurring activities.
  - ▶ Two steps: Alice first leaves Bob some information for her identity, and then everytime Bob uses such information to verify that Alice is Alice.
- ▶ Public key is identity.
  - ▶ Without an authentic channel: Bob receives a public key and names it Alice.
  - ▶ “Anonymous”: this identity associates to no real-world entity.
- ▶ Public key as a representation of identity.
  - ▶ With an authentic channel: Alice need to prove she is Alice to Bob, e.g. via a passport, before she can provide a public key for Bob to store.
  - ▶ The public key could be revoked, e.g. when Alice lost her private key.
- ▶ Which one is better? Depending on the application.

# Authetication with Digital Signatures

- ▶ With Alice's public key on file, Bob authenticates by asking whoever claims to be Alice to sign a message with Alice's private key.
- ▶ This seems to be very secure.
  - ▶ Assume Alice keeps her private key as a secret, and Bob stores Alice's public key in a way no one can modify it.
  - ▶ Oscar cannot forge digital signatures.
  - ▶ Even if Oscar steals Alice's public key from Bob, he/she cannot use it to prove he/she is Alice to another party.
- ▶ Replay attack: but Oscar may record the message with Alice's signature and replay it to Bob at a later time.
  - ▶ Bob need to ask Alice to sign a chosen message!
- ▶ Challenge-response authentication.
  - ▶ Challenge: Bob generates a nonce and sends it to Alice.
  - ▶ Response: Alice signs the nonce and replies to Bob.
  - ▶ Any possibility of man-in-the-middle attack?

# Mutual Authentication

- ▶ In many cases Alice also need to be sure that Bob is Bob.
- ▶ Alice may authenticate Bob by Bob's public key.
  - ▶ Same as how Bob setups Alice's public key.
- ▶ Complications
  - ▶ The two channels Alice-to-Bob and Bob-to-Alice could be different.
    - ▶ Both are authentic.
    - ▶ Both are not authentic.
    - ▶ One is authentic and the other is not authentic.
  - ▶ The need for confidentiality.

# Practical Applications and Considerations

- ▶ Common people used to have limited access to public-key cryptography.
  - ▶ Due to sophisticated/costly hardware/software, patents, business practices etc.
- ▶ Servers usually identify themselves via digital signatures.
  - ▶ Mostly via HTTPS.
  - ▶ Still, people with little knowledge about cyber security and digital signatures are subject to phishing scams.
- ▶ For professionals nowadays, adoption of Linux makes authentication with digital signatures widely available.
  - ▶ Mostly via SSH, e.g. GitHub.
  - ▶ Sometimes even enforced, e.g. AWS EC2.

# Summary

- ▶ RSA digital signature
  - ▶ Key generation: by Bob,  $k_{pub} = (n, e)$ ,  $k_{pr} = (p, q, d)$
  - ▶ Sign: Bob only,  $s = d_{k_{pr}}(x) = x^d \bmod pq$ .
  - ▶ Verify: everyone,  $x' = e_{k_{pub}}(s) = s^e \bmod n$ ,  $x == x'$ ?
  - ▶ Assumption: Oscar cannot factorize  $n$  into  $p$  and  $q$  in polynomial time.
- ▶ Other digital signature algorithms like DSA and ECDSA.
- ▶ Identification/authentication: solutions exist, but need to make trade-offs.